RV College of Engineering®
Mysore Road, RV Vidyaniketan Post, Bengaluru - 560059, Karnataka, India

Go, change the world

Design of an AI-GPS Smart Mini Drone with Follow-Me & Geofencing System

A Major Project Report (21EC81P)

Submitted by,

Shayak Bose **1RV21EC158**

Under the guidance of

Dr. Chethana G
Assistant Professor
Dept. of ECE
RV College of Engineering

Mr. Kishen Amarnath
Manager/Lead Design Engineer
Analog Division
Signaltron Systems

In partial fulfillment of the requirements for the degree of Bachelor of Engineering in

Electronics and Communication Engineering

2024-25

RV College of Engineering®, Bengaluru

(Autonomous institution affiliated to VTU, Belagavi)

Department of Electronics and Communication Engineering

CERTIFICATE

Certified that the major project (21EC81P)work titled ***Design of an AI-GPS Smart Mini Drone with Follow-Me & Geofencing System*** is carried out by **Shayak Bose (1RV21EC158)** who is bonafide student of RV College of Engineering, Bengaluru, in partial fulfillment of the requirements for the degree of **Bachelor of Engineering** in **Electronics and Communication Engineering** of the Visvesvaraya Technological University, Belagavi during the year 2024-25. It is certified that all corrections/suggestions indicated for the Internal Assessment have been incorporated in the major project report deposited in the departmental library. The major project report has been approved as it satisfies the academic requirements in respect of major project work prescribed by the institution for the said degree.

Guide	Head of the Department	Principal
Dr. Chethana G	Dr. H V Ravish Aradhya	Dr. K. N. Subramanya

External Viva

Name of Examiners	Signature with Date
1.	
2.	

Signaltron

Signaltron Systems Pvt. Ltd.
4C-116, 3rd Floor, 4th Cross, OMBR Layout,
Banaswadi, Bangalore 560043, India
Email: contact@signaltron.com
www.signaltron.com
CIN: U31900KA2019PTC128760

To,

Shayak Bose
B1002, Rohan Vasantha,
Marathahalli (near Marathahalli Bridge),
Bangalore, Karnataka - 560037

02 Jan, 2025

Dear **Shayak**,

Based on your interest to join our organization and the recent discussions we had with you, we are pleased to offer you the role of **Design Engineer: Embedded Software** in our Company. At Signaltron, we have set our vision on building a world class product organization. It is a tough task to fulfil such a vision and we hope you will be up to the challenge.

In recognition of the value you bring to Signaltron, there is a reward of **₹ 14,07,013/-** for you. This includes a retention bonus component of **₹ 3,00,000/-**. To understand the reward system, please turn overleaf. All other relevant organizational details and terms will be shared with you on your joining.

We would like you to join us, as an employee by **07 July 2025**. You will be required to serve a probationary period of six months from the date of joining and shall be confirmed in the Company upon satisfactory completion of the term.

As a precursor to fulltime employment, you are required to complete a 6 months internship starting **Jan 2025**. In the meanwhile, please sign a duplicate copy of this letter to indicate the acceptance of the offer.

Welcome to Signaltron!!

Sincerely,

Himamshu G Khasnis
Director

SIGNALTRON SYSTEMS PRIVATE LIMITED Bengaluru

DECLARATION

I, **Shayak Bose** , a student of eighth semester B.E., Department of Electronics and Communication Engineering, RV College of Engineering, Bengaluru, hereby declare that the major project titled '**Design of an AI-GPS Smart Mini Drone with Follow-Me & Geofencing System** ' has been carried out by me and submitted in partial fulfilment for the award of degree of **Bachelor of Engineering** in **Electronics and Communication Engineering** during the year 2024-25.

Further I declare that the content of the dissertation has not been submitted previously by anybody for the award of any degree or diploma to any other university.

I also declare that any Intellectual Property Rights generated out of this project carried out at RVCE will be the property of RV College of Engineering, Bengaluru and we will be one of the authors of the same.

Place: Bengaluru

Date: 09/06/2025

Name	Signature
1. Shayak Bose(1RV21EC158)	

ACKNOWLEDGEMENTS

I am indebted to my guide, **Dr. Chethana G**, Assistant Professor, RV College of Engineering . for the wholehearted support, suggestions and invaluable advice throughout my project work and also helped in the preparation of this thesis.

I also express our gratitude to my panel members **Dr. Chethana G**, Assistant Professor and **Dr. Usha Rani.K.R**, Professor & Associate Dean (PG Studies), Department of Electronics and Communication Engineering for their valuable comments and suggestions during the phase evaluations.

My sincere thanks to the project coordinators **Dr. Veena Devi S V** and **Prof. Sindhu Rajendran** for their timely instructions and support in coordinating the project.

My gratitude to **Prof. Narashimaraja P** for the organized latex template which made report writing easy and interesting.

My sincere thanks to **Dr. H V Ravish Aradhya**, Professor and Head, Department of Electronics and Communication Engineering, RVCE for the support and encouragement.

I express sincere gratitude to our beloved Principal, **Dr. K. N. Subramanya** for the appreciation towards this project work.

I thank all the teaching staff and technical staff of Electronics and Communication Engineering department, RVCE for their help.

Lastly, I take this opportunity to thank my family members and friends who provided all the backup support throughout the project work.

ABSTRACT

The recent advancement and increased utilisation of Unmanned Aerial Vehicles (UAV) addresses the pressing need for secure and self-reliant UAV systems amidst the growing dependence on foreign-manufactured drone components, especially in commercial and defense domains. Such dependence poses national security concerns due to potential vulnerabilities and exploitability. The core motivation is to contribute to the "Aatmanirbhar Bharat" (self-reliant India) initiative by minimizing foreign tool usage and establishing a robust local ecosystem for autonomous drones. This project centers around the design of a smart control system for a mini drone, featuring indigenously developed hardware and software for AI-integrated camera functionality and GPS-based geofencing.

The objectives of this work include the implementation of an AI-powered object detection system using MediaPipe and OpenCV, deployed on Raspberry Pi 5 for real-time inference. The project also involves the integration of geofencing capabilities using the Neo M8N GPS module, supported by libraries like shapely and custom haversine calculations for spatial boundary detection. Additional goals include accelerating the GPS module's Time-To-First-Fix (TTFF) via UBX commands, and enabling a dynamic "Follow Me" protocol based on bounding box scaling and real-time feedback loops.

The simulation environment involved the use of a Flask-based server to stream video and detection data from dual camera systems mounted on the drone.The system uses the efficientdet_lite0.tflite model, achieving an average object detection accuracy of 93–94%. The "Follow Me" protocol translated the detected object's bounding box width into real-time velocity and control signals, dynamically adjusting the drone's throttle, pitch, and yaw to maintain a consistent distance and orientation.

Real-time testing confirmed the effectiveness of geofencing alerts and emergency landing mechanisms. BREACH commands were successfully triggered when boundary conditions were violated, ensuring system-level safety. This safety protocol was validated with the help of the fact that the GPS system achieves a location accuracy error margin of 1.5-2 meters, thus enabling reliable boundary monitoring. These results underscore the effectiveness of the integrated control system in enhancing drone autonomy and reliability, making it a promising solution for secure and intelligent UAV deployment in the future.

CONTENTS

Abstract i

List of Figures iv

List of Tables vi

Abbreviations vii

List of Symbols viii

1 Introduction to 5G Mini Drones 1

1.1 Introduction 2

1.2 Motivation 3

1.3 Problem statement 4

1.4 Objectives 4

1.5 Literature Review 4

1.6 Brief Methodology of the Project 21

1.7 Assumptions made / Constraints of the project 23

1.8 Organization of the report 25

2 System Specifications and Underlying Technologies for AI-Guided UAVs 26

2.1 Specifications and Theoretical Foundations 27

2.1.1 Basics of Unmanned Aerial Vehicles (UAVs) 27

2.1.2 Overview of AI in Embedded Systems 28

2.1.3 Fundamentals of Object Detection using CNN/MediaPipe 28

2.1.4 GPS Technology and Time-To-First-Fix (TTFF) 28

2.1.5 Geofencing Logic using Point-in-Polygon and Haversine Formula . 29

2.2 Programming Environment 29

2.2.1 Python as the Primary Programming Language 29

2.2.2 Use of Flask for Video Streaming and GUI 30

2.2.3 Library Support and Functionality 30

2.3 Tools and Platforms Used 31

2.3.1 Raspberry Pi 5: Features and OS Setup . . . 31
2.3.2 Camera Modules (Pi Camera V3 and 5MP) . . . 31
2.3.3 Ublox Neo M8N GPS Module . . . 31
2.4 Additional Theoretical References . . . 31

3 Design of Smart Control Architecture for Mini Drone 33
3.1 Specifications for the Design . . . 34
3.2 Pre-analysis and Models Used . . . 35
3.3 Design Methodology . . . 36
3.4 Equations and Logic . . . 40

4 Implementation and Integration of Smart Control System for Mini Drone 43
4.1 System Architecture and Implementation Overview . . . 44
4.1.1 Hardware Architecture . . . 44
4.1.2 Software Architecture . . . 45
4.2 Integration Workflow . . . 47

5 Results & Discussions 49
5.1 Experimental Results . . . 50
5.1.1 AI Object Detection . . . 51
5.1.2 GeoFencing . . . 53
5.1.3 UBX GPS Conifguration . . . 56
5.1.4 Follow Me Protocol . . . 58

6 Conclusion and Future Scope 64
6.1 Conclusion . . . 65
6.2 Future Scope . . . 65
6.3 Learning Outcomes of the Project . . . 66

LIST OF FIGURES

1.1 Signaltron 5G Mini Drone 3
1.2 Methodology of AI-GPS Smart Mini Drone 21
1.3 Smart Control System with GPS and AI Integration for Mini Drone . . . 22

3.1 AI Object Detection Block Diagram 36
3.2 Geofencing Block Diagram 37
3.3 GPS UBX Configuration Block Diagram 38
3.4 Follow Me Protocol Block Diagram 39

4.1 Hardware Architecture Block Diagram 45
4.2 Software Architecture Flow Diagram 46

5.1 Object detection applied in an indoor office environment. 51
5.2 Object detection in a home setting. 51
5.3 Object detection applied to a television screen. 52
5.4 Object Detection working on multiple objects simultaneously. 52
5.5 Drone GUI Dashboard. 53
5.6 GUI with the Full Screen Map 54
5.7 Drone within Geofence: Breach is 0 54
5.8 Drone on Geofence: Breach is 1, alongwith distance of proximity to fence mentioned 55
5.9 Drone on Geofence: Breach is 1 55
5.10 No GPS lock. 56
5.11 Hard-code succesfully executed. 57
5.12 Perfect GPS Lock. 57
5.13 Follow Me Off; Input via manual joystick mode 58
5.14 Bounding Box large. Thus, PWM is low 59
5.15 Bounding Box small. Thus, PWM is high 59
5.16 Bounding Box again large. Thus, PWM is low 60
5.17 No "Person" detected; PWM at neutral values 60
5.18 Plot of PWM Pitch v/s Width of Bounding Box 61

5.19 Example of JSON outputs used to obtain Follow Me PWM values 62

LIST OF TABLES

2.1 Comparison of AI Models for Edge Object Detection 29

2.2 Python Library Roles in the Project 30

4.1 Hardware Components and Their Roles 45

4.2 Software Modules and Their Functions 47

5.1 Mapping of Bounding Box Width to PWM Pitch Values 61

ABBREVIATIONS

5G 5^{th} Generation

AI Artificial Intelligence

GPS Global Positioning System

GUI Graphical User Interface

JSON JavaScript Object Notation

MEC Multi-access Edge Computing

OpenCV Open Source Computer Vision

PWM Pulse Width Modulation

RF Radio Frequency

RPi5 Raspberry Pi 5

TTFF Time-To-First-Fix

UART Universal Asynchronous Receiver/Transmitter

UAV Unamanned Aerial Vehicles

UBX u-blox

WiFi Wireless Fidelity

LIST OF SYMBOLS

K_d Derivative constant in PID control

K_i Integral constant in PID control

K_p Proportional constant in PID control

Δ Change in a quantity (Delta symbol)

ϕ Angle or heading (Phi)

$\sum$ Summation operator

r Earth's radius

Chapter 1

Introduction to 5G Mini Drones

CHAPTER 1

INTRODUCTION TO 5G MINI DRONES

1.1 Introduction

Drones - Unmanned Aerial Vehicles (UAVs) as shown in Figure 1.1 — are finding increasing use in areas ranging from disaster management and agriculture to last-mile delivery services. Their ability to reach inaccessible terrains, capture real-time data, and automate complex tasks makes them invaluable in modern applications. By integrating 5G technology, drones gain high-bandwidth, low-latency connectivity that enables enhanced remote control, live high-resolution video streaming, faster data offloading, and potential AI-driven real-time analytics—all critical for safety, efficiency, and mission success.

Within this context, the internship project focused on developing software solutions for a 5G-enabled mini drone. The tasks covered are must and important layers of drone functionality, the camera functionality and the GPS functionality. These were necessary and compulsory features of the Drone on which further complex features have been built.

This Drone has been built in Collaboration with **CDAC**, The Centre for Development of Advanced Computing (CDAC), Bangalore. It is a premier R&D organization under the Ministry of Electronics and Information Technology (MeitY), Government of India. Established as part of the national initiative to drive innovation in advanced computing, CDAC Bangalore focuses on a wide range of areas including high-performance computing (HPC), artificial intelligence, cybersecurity, software technologies, IoT, embedded systems, and language computing. The centre is instrumental in developing indigenous technologies and solutions tailored to India's strategic and societal needs. CDAC Bangalore also plays a vital role in technology transfer, capacity building, and providing support to government and industrial projects, contributing significantly to India's self-reliance in advanced information and communication technologies.

This Mini Drone is completely **indigenous** ,secure hardware and software design, thus resulting in effective design integration from trusted sources ensuring safe and secure network.

Figure 1.1: Signaltron 5G Mini Drone

1.2 Motivation

Drones used today for commercial applications, military reconnaissance, and surveillance are predominantly manufactured in China or assembled using Chinese components. This reliance, particularly in defense-related applications, poses a significant security risk due to the increased vulnerability to remote exploitation or hacking of drone systems. Ensuring the safety and integrity of unmanned aerial vehicles (UAVs) is thus a matter of national importance.

In this context, the primary motivation of the project is the technical limitation of conventional drones, which often rely on Wi-Fi or RF-based communication for live video streaming and telemetry. These methods are prone to signal degradation, interference, and most notably, significant latency during video transmission—an issue that is detrimental to real-time control and situational awareness. To address this, 5G communication technology has been integrated into the system. 5G offers ultra-low latency, higher bandwidth, and improved reliability, enabling real-time high-definition video streaming and rapid command-response cycles. These features are essential for mission-critical drone operations in both civilian and defense sectors.

Another key motivation behind the development of 5G mini-drones under Signaltron Systems' product initiative is to contribute to the vision of "Aatmanirbhar Bharat" —a self-reliant India. All critical hardware and software components of the drone are indigenously developed or sourced, with minimal dependency on foreign technologies, ensuring greater transparency, control, and long-term sustainability.

1.3 Problem statement

Current mini drone systems rely heavily on foreign-manufactured components that provide inefficient and unreliable smart control capabilities, creating security vulnerabilities and dependency issues. There is a need to design an indigenous smart control system that combines GPS navigation, AI-powered camera systems, autonomous tracking, and safety features like geofencing to enable secure, intelligent drone operations for commercial and defense applications.

1.4 Objectives

The objectives of the project are

1. To develop and enable AI based object detection camera system for both top and bottom cameras on the Drone, and run it on an MEC server/ live commercial 5G sim over Flask using libraries such as OpenCV and mediapipe.
2. To develop Geofencing for Drones using Neo M8N GPS module and various libraries (like shapely) and Mathematical Functions
3. To develop a code which will assist the GPS module for a faster TTFF, by sending UBX commands via Rpi 5 to the Module
4. To develop 'Follow Me' protocol on Drone using the AI object Detection Cameras

1.5 Literature Review

[1] This paper introduces a vision-based control architecture that enables a quadcopter to autonomously track and follow a moving target using object detection and visual servoing techniques. The authors implemented a PID-based tracking controller with object detection processed via onboard camera modules. The core innovation lies in how camera feedback is used for real-time adjustments of the drone's position, maintaining the target in the center of the frame. The study also explores methods to handle occlusion and background clutter, which is essential for robust tracking in varied outdoor settings. This work is highly relevant to our project's "Follow Me" protocol, which similarly depends on bounding box dynamics from dual cameras to adjust drone motion. Moreover, the PID-based logic demonstrated here could serve as a foundational algorithm for stabilizing the drone while it follows a human subject using AI-based detections from the MediaPipe model.

[2] Barisic and colleagues presented a real-time visual tracking system specifically designed for UAV-to-UAV detection and following. The system uses lightweight image processing techniques and optical flow tracking to locate and maintain focus on another drone in flight. This solution minimizes computational overhead, making it ideal for embedded systems such as Raspberry Pi—similar to the hardware platform used in our project. Their modular design also integrates region-of-interest (ROI) estimation, which allows the camera to isolate and track a specific target even when multiple moving objects are in the frame. This is particularly useful in scenarios involving dynamic environments, such as crowds or urban areas, where object confusion can arise. Applying this methodology to our project could enhance robustness, especially for our drone's bottom-facing camera, which may encounter obstacles or shifting terrain during the "Follow Me" mode. The lightweight tracking framework aligns with our goal of real-time AI inference and efficient use of edge computation resources.

[3] This study introduces Fast-Tracker 2.0, an enhanced aerial tracking algorithm that utilizes active vision control in tandem with deep learning-based human location regression. The innovation lies in the fusion of visual tracking with predictive modeling, enabling the drone to not just react to a person's position, but to anticipate future motion based on trajectory estimation. Such predictive elements are crucial for smoother and more intelligent follow-me behavior, especially when combined with AI object detection. Fast-Tracker 2.0 also demonstrates high frame-rate performance and low latency, achieved through model optimization and asynchronous sensor fusion—capabilities that directly support the performance goals of our system. Since our project uses bounding box metrics (width, height) to derive motion control signals for pitch, yaw, and throttle, incorporating regression-based prediction (as demonstrated here) could make movement smoother and more human-like. Additionally, the paper's emphasis on autonomy without reliance on GPS during tracking supports our aim of enhancing visual navigation reliability.

[4] Li et al. introduce AutoTrack, a framework for UAV-based visual tracking that leverages spatio-temporal regularization to achieve high performance under challenging aerial scenarios. Their method automatically adjusts model parameters using reinforcement learning, thereby adapting to changes in object scale, orientation, and motion—a key benefit in dynamic follow-me applications. The system significantly reduces latency while maintaining tracking robustness by optimizing the tracker without manual hyper-

parameter tuning. This directly supports the project's requirement for AI-driven tracking that remains reliable even as the drone or target changes speed or trajectory. By minimizing computation time and adapting to contextual cues, AutoTrack demonstrates the feasibility of deploying high-precision models on resource-constrained edge systems such as Raspberry Pi or Jetson Nano.

[5] Schilling et al. investigate decentralized drone flocking using visual inputs in outdoor environments. While flocking differs from single-drone follow-me tasks, their approach to collision avoidance and inter-UAV tracking through onboard cameras introduces a scalable framework for vision-based motion prediction. They utilize optical flow and stereo vision to infer depth and relative motion, which can inspire enhancements to geofencing or obstacle avoidance in the project. The findings illustrate how low-cost vision systems can replace LiDAR in real-time, even in unstructured terrains. This insight is applicable to the project's autonomy requirements, particularly in inferring spatial boundaries and responding to visual changes.

[6] This study presents a practical implementation of a "Follow-Me" UAV system using real-time object tracking and PID control for motion adjustment. The system operates with a standard webcam and Raspberry Pi, showcasing how lightweight computing platforms can deliver reliable person-following behaviors. Bounding box parameters such as height and width are translated into positional commands, a concept reflected in the project's use of bounding box width to modulate PWM signals. Piquero et al. also detail calibration steps, frame processing, and tracking feedback integration—elements that reinforce the usage of OpenCV and MediaPipe with Flask to handle real-time video analytics over 5G networks in the current system.

[7] Yao et al. propose SGDViT, an advanced transformer-based model that leverages saliency maps to dynamically refine attention during object tracking in UAVs. Transformers, traditionally used in NLP, are adapted here to capture spatial dependencies across video frames more effectively than CNNs. For a system already utilizing MediaPipe and efficient detection models, this paper presents a compelling case for evolving toward transformer-based AI modules to improve generalization to aerial imagery. Notably, their system is designed with power efficiency and latency reduction in mind—qualities essential for deployment on mobile edge computing (MEC) systems similar to those used in the current drone platform.

[8] Wu et al. developed a real-time object localization and tracking system tailored for aerial UAV sensing. The research emphasizes image preprocessing techniques like background subtraction and histogram equalization to improve detection accuracy in fluctuating lighting conditions—a challenge relevant to outdoor operations in this project. The study validates using simpler monocular vision in GPS-denied environments, offering a fallback logic in case of GPS module failure or signal loss. The UAV platform tested includes serial interfacing and camera-GPS data fusion, reinforcing the project's layered architecture, where AI-based camera input is complemented by GPS-based geofencing.

[9] This early yet impactful study combines user interaction and visual feedback by allowing users to control UAVs through touch-based tablet commands while tracking visual targets. Although the current system is autonomous, the control logic involving object size and location mirrors the strategy implemented in the project, where bounding box aspect ratios influence PWM computation. The paper outlines the effectiveness of combining control feedback loops with dynamic visual cues, which supports the project's multi-camera strategy for robust and continuous target acquisition under varying flight perspectives.

[10] Mamun et al. detail a low-cost UAV implementation capable of tracking a specific human subject using color and shape features. The system operates via a Raspberry Pi and webcam, and applies a simple yet effective HSV-based detection method. Although the current system uses neural networks, this study emphasizes efficient resource usage and practical parameter tuning for real-time control—a principle also followed in the project. Moreover, the logic for computing horizontal and vertical displacements from frame coordinates complements the aspect-ratio-based PWM modulation strategy being used.

[11] In this thesis, Ghassabi implements a vision-based person tracker with a drone system using OpenCV and Haar cascades. While Haar classifiers are largely outdated, the workflow—including data collection, pre-processing, tracking, and trajectory planning—offers a complete development flow that parallels the current system architecture. The study explores challenges of maintaining frame rate stability and minimizing CPU usage, which aligns with the decision to use lightweight inference models such as efficientdet_lite0.tflite. The emphasis on modular design and integration reflects the project's multi-functional drone design philosophy.

[12] Miura et al. introduce gesture-based commands to enable user interaction with UAVs, a feature that could be adaptable to future iterations of the current drone system. Their quadcopter leverages a stereo camera system and inertial sensors to interpret hand gestures and follow the user. Although the existing project uses bounding box geometry for tracking, this study illustrates how multimodal data—visual plus inertial—can enhance robustness and control. Integration of gesture recognition into the follow-me protocol could extend the system's interaction modalities beyond GPS and AI-based detection, enriching autonomous behavior.

[13] Karar et al. present a compelling application of service robotics through the humanoid robot Pepper, focusing on its role as a customer interface in indoor service environments. The robot is capable of verbal and gestural interaction, utilizing onboard sensors and cameras to respond to human presence. Although the paper's domain is rooted in social robotics, the design philosophy aligns well with the current project's architecture—namely, the use of embedded intelligence and perception-based interaction. Just as Pepper uses sensory data to tailor its response to human input, the UAV in this project uses visual cues (aspect ratio and bounding box size) to adjust its position dynamically. Furthermore, the robot's ability to navigate its environment and maintain conversational continuity underlines the importance of multi-modal input handling—a challenge similarly addressed here via real-time image-based object detection and geofencing integration. Thus, while the application differs, the foundational principles of closed-loop interaction and adaptive behavior make this work notably relevant.

[14] Cuthbertson's article in Newsweek introduces a provocative yet insightful discussion on Google's two-legged humanoid robot and its potential role in military applications. The robot is designed with the ability to traverse rugged terrain and maintain balance dynamically, powered by high-level AI for environment-aware navigation. The reference is useful not for its UAV content but for drawing parallels between human-like motion planning and the autonomy required in aerial systems. Specifically, the concept of navigating unpredictable environments resonates with the challenges faced by drones during real-time object tracking and flight path optimization. While the humanoid robot's autonomy is built for terrestrial motion, the same layers of perception, planning, and execution are embedded in this project's AI-based control pipeline. The idea of deploying robots for risky, mission-critical tasks further supports the rationale for autonomous UAV deploy-

ment, especially in surveillance or emergency monitoring applications. Cuthbertson's coverage of robotic situational awareness underlines the value of adaptive, sensor-based control—an essential pillar of this work.

[15] In his Master's thesis, Ou designs a robust vision-based path planning system for mobile robots, integrating deep neural network (DNN)-based object recognition with ORB-SLAM2 for simultaneous localization and mapping. This integration allows the robot to both understand its surrounding environment and make navigational decisions based on the presence of detected obstacles or goals. The emphasis on visual servoing and environmental awareness draws a strong connection to this project, where the UAV must detect, track, and follow a dynamic object while navigating in an open environment. The use of SLAM techniques in Ou's work inspires future scalability of the current project, especially if the UAV must operate in partially GPS-denied zones. Furthermore, the DNN integration in perception modules directly aligns with the project's use of neural networks for interpreting bounding box parameters in a visual frame, which subsequently drive PWM-based motion commands. Ou's work establishes a foundational bridge between vision, control, and path reasoning—components equally emphasized in this UAV system.

[16] Kiss discusses external manipulation strategies for autonomous vehicles, particularly focusing on how an external agent (human or system) can intervene in the vehicle's operations when necessary. This is critical in scenarios where full autonomy may fail or become unsafe, and a fallback manual control is warranted. The parallels with this project are evident in the dual-mode control feature—while the UAV primarily operates in autonomous follow-me mode driven by visual cues, a joystick override exists to allow human intervention. Kiss emphasizes modularity and system flexibility, highlighting how control architectures must allow both autonomous and manual interactions without compromising system integrity. This principle is applied here, as the project incorporates fail-safe switches and layered input processing, ensuring that human operators can regain control if the AI-based tracking deviates or loses the target. Moreover, the concept of external manipulation supports the integration of ROS nodes and GUI elements that serve as interfaces between the user and the UAV's internal logic.

[17] The collaborative work by NTU and Volvo marks a landmark development in urban autonomous mobility with the launch of the world's first full-sized autonomous electric bus. The vehicle is equipped with advanced geolocation, LiDAR, and camera-

based navigation systems, enabling it to operate safely in a shared environment with human-driven vehicles and pedestrians. This large-scale implementation is conceptually significant to the present UAV project. Although the physical scale and use-case differ, both systems prioritize autonomous navigation using a blend of GPS and perception-based technologies. Geofencing, a critical component of the bus's safety logic, is mirrored in this UAV system where spatial constraints guide flight permissions. The paper serves as validation that real-world, regulation-compliant autonomy is not just achievable but also scalable. It lends credibility to integrating intelligent decision systems into UAVs that may eventually need to operate in regulated or congested airspaces. Furthermore, the reliance on sensor fusion in NTU's system underscores the potential of future upgrades for the current drone framework.

[18] Jeong et al. introduce a visionary model for an airborne fulfillment center, referred to as the "flying warehouse" concept, where multiple UAVs coordinate from a centralized aerial hub to perform package deliveries. The study models delivery optimization by calculating various logistics parameters including route timing, load balancing, and airspace contention. This centralized-decentralized operational framework resonates with the long-term vision of this project, especially when considering multiple UAVs being deployed simultaneously. Although this project currently handles a single mini-drone, the software design—with modular tracking, geofencing, and command layers—is extendable for swarm coordination. Moreover, Jeong's use of predictive models for throughput enhancement inspires future integration of lightweight onboard predictive algorithms to improve object reacquisition when tracking fails temporarily. The notion of prioritizing airspace as a dynamic resource directly feeds into this project's considerations for altitude stratification and no-fly zone enforcement.

[19] Palmer's report on Amazon's FAA-approved UAV delivery fleet reflects the evolution of regulatory frameworks and their intersection with technology readiness. The approval signifies that UAVs are being recognized not just as experimental tools but as integral parts of commercial logistics ecosystems. From a technical perspective, the systems approved by FAA incorporate multiple layers of redundancy—autonomous flight control, GPS waypoints, geofencing, and obstacle avoidance—which parallel the safety and autonomy features designed into this project. Although this work targets smaller drones and academic objectives, it adopts industry-standard practices such as real-time object de-

tection, autonomous path updates, and geofenced movement restriction. Palmer's article also brings attention to compliance—an aspect being considered through this project's commitment to controlled airspace simulation and adherence to safety protocols during object tracking operations. This reinforces the legitimacy and future scalability of the approach used in the current UAV framework.

[20] Yang et al. present a real-time monitoring and management system using UAVs for logistics tasks, combining GPS data streams with onboard sensing mechanisms. The system demonstrates dynamic data acquisition, route updates, and status broadcasting—mechanisms that support UAV operation in mission-critical scenarios. The paper's core contribution lies in its use of synchronized feedback loops to optimize delivery tasks in real-time. The methodology involves continuous monitoring of UAV positions and environmental conditions to fine-tune flight decisions. The relevance to this project is strong: the control pipeline here also emphasizes real-time feedback, not only from GPS but also from vision systems. Moreover, PWM-based actuation in response to bounding box changes mimics the feedback loop principle Yang proposes. This study reinforces the necessity of adaptable control systems in UAVs, especially under time-constrained or dynamically evolving environments, supporting the robustness and flexibility embedded in this project's system architecture.

[21] Mechali et al. propose a distributed sliding mode control protocol for autonomous formation flight of quadrotors, offering fixed-time stability. Their approach ensures robust synchronization even with external disturbances, providing inspiration for future extensions of this project into multi-drone coordination. The emphasis on control precision underlines the need for real-time robustness, similar to the project's object-tracking-based adjustments during varying flight conditions.

[22] Chen et al. develop a recursive sliding mode controller for quadrotor attitude stabilization in the presence of unknown disturbances. The controller adapts in real-time to maintain stability, mirroring this project's objective of compensating for rapid changes in object aspect ratio and position. The recursive nature of their design reflects the project's own iterative PID tuning for PWM-based drone control.

[23] Guo et al. address wind and payload-induced disturbances in UAVs using multiple observer-based controllers. Their system ensures trajectory stability by actively estimating and rejecting external perturbations. The principle of adaptive correction aligns with

this project's response to bounding box deformations, where sudden shape changes are filtered and translated into smoothed control signals.

[24] Qin and Wang propose a real-time control algorithm that relies on vision-based tracking for quadcopters. Their design processes image features to adjust UAV position in relation to a moving target. This closely reflects the current project's method, where bounding box displacements directly modulate drone orientation through PWM, offering a balance between simplicity and effectiveness.

[25] Zhang et al. use deep reinforcement learning for UAV target tracking via a coarse-to-fine strategy. The system uses global detection followed by focused local tracking, offering resilience to occlusion. This layered approach resembles the project's idea of recalibrating bounding boxes dynamically and supports the use of deep learning in improving UAV perception precision.

[26] Wang et al. benchmark single-person UAV tracking using aerial datasets and propose a lightweight model with high responsiveness. Their evaluation of real-world conditions like motion blur and occlusion highlights the value of robustness—similar to this project's goal of maintaining stable tracking under varied lighting and movement patterns using neural-based vision modules.

[27] Vasconcelos and Vasconcelos design a UAV system for outdoor person-following using basic image processing. Their real-time loop adjusts drone motion based on color features, which, though simple, parallels the bounding box-based adjustments made in this project. Their emphasis on continuous lock-in supports this work's strategy of persistent object tracking through bounding ratio analysis.

[28] Shen et al. implement a UAV that performs both tracking and frontal face capture using stereo vision. Their focus on accurate person localization under motion expands the relevance of vision systems. While this project does not use stereo inputs, the concept of multi-objective detection complements its vision roadmap and supports enhancement ideas like facial pose alignment.

[29] Singla et al. apply deep reinforcement learning with memory mechanisms to improve UAV obstacle avoidance. Although not directly used here, their emphasis on temporal consistency informs the project's decision to smooth bounding box shifts across frames. Their learning-based approach affirms the long-term potential of memory-augmented vision for dynamic environments.

[30] Hou et al. use stereo vision to enable low-altitude obstacle avoidance for multi-UAV systems. Their approach simulates spatial depth awareness, which this project approximates using bounding box spread and geofence overlays. Their results reinforce the idea that real-time visual cues can effectively guide evasive maneuvers without requiring heavy 3D computation.

[31] Li's thesis explores obstacle detection and collision avoidance techniques tailored for multicopters. The work focuses on integrating sensor-based inputs with path adjustment algorithms. Though simpler in scope, the emphasis on modularity and embedded implementation supports this project's vision of real-time bounding-box-based evasive behavior. The safety logic overlaps with the current PWM-based flight correction system activated upon target loss or overshoot.

[32] Han et al. propose a reinforcement learning-based obstacle avoidance method using RealSense depth data. Their three-dimensional navigation model allows the UAV to learn adaptive responses to spatial constraints. This 3D perspective aligns with the project's goal of implicit depth estimation via object aspect ratios. The adaptive logic supports this project's future scope—integrating depth-aware AI into monocular visual tracking systems.

[33] Cho and Yoon present a topological method to assess airspace capacity by modeling geofences as "keep-in" and "keep-out" zones. The paper quantifies usable airspace through spatial tessellation. This model provides a theoretical foundation for the project's geofencing logic, where drones are restricted to zones based on GPS-defined virtual boundaries, enforcing safe flight corridors during object tracking operations.

[34] Stevens and Atkins design a hybrid multicopter control system that integrates multiple flight modes within geofenced regions. Their method allows UAVs to switch modes based on proximity to boundary zones. This adaptive autonomy complements the project's GPS-triggered overrides and geofence-based PWM suppression, enabling bounded flight even under dynamic object-following scenarios.

[35] Stevens and Atkins further explore UAS deconfliction via geofencing protocols and policy coordination. Their study introduces rules for layered airspace management, influencing how UAVs interact within shared environments. The regulatory focus aligns with the project's design of no-fly zones and altitude restrictions, simulating real-world compliance within localized aerial missions.

[36] Dill et al. introduce SAFEGUARD, a lightweight UAS safety system for collision avoidance using boundary enforcement. It acts as a last-resort safety net, which resonates with this project's layered control system where bounding box thresholds act as virtual fences. Their architecture validates the idea of embedded autonomy with limited but focused sensor use.

[37] DJI's FlySafe policy outlines practical regulations and safety protocols for UAV operation. This documentation defines user-level constraints and enforcement strategies that support responsible flying. The project mirrors these protocols in its hardcoded geofence logic and visual warnings for zone breaches, embedding best practices into technical design.

[38] Edelsbrunner et al. define alpha shapes, a geometric tool for characterizing point set boundaries. Though abstract, this technique underpins spatial partitioning algorithms like geofencing. The concept provides mathematical support for defining irregular drone operation zones—an area relevant to extending this project's rigid rectangular geofences into adaptive, terrain-aware boundaries.

[39] Wawrzyniak and Hyla explore geofencing for mobile inland navigation, combining GPS data with visualization strategies. The system supports spatial rule enforcement via map overlays. This research supports the visual logic in the current drone system where GUI or telemetry feedback can be extended to show real-time boundary proximity and potential zone incursions.

[40] Reclus and Drouard apply geofencing in commercial fleet tracking, emphasizing freight boundary monitoring. Their work informs geospatial enforcement strategies for logistics. The structured geofencing model translates well into UAV applications like this one, where continuous boundary checking ensures the drone stays within mission-defined regions during autonomous follow.

[41] Haofeng and Xiaorui propose a Wi-Fi-based secure access control system that leverages geofencing logic to restrict network usage based on physical location. The system uses geo-coordinates to authenticate devices dynamically, ensuring that only users within a specified spatial perimeter gain access. While the application is network-focused, the geo-fencing enforcement mechanism resonates with this project's safety logic, which ensures the UAV responds to pre-defined GPS constraints by suppressing unauthorized movements. Their lightweight, position-aware architecture validates the feasibility of

boundary-triggered permission control in UAV autonomy.

[42] Johnson et al. explore key safety requirements for small UAVs in urban environments, focusing on detect-and-avoid (DAA) and well-clear standards. The paper addresses how UAVs can autonomously identify and respond to potential mid-air threats using passive and active sensing strategies. Their definition of urban airspace constraints complements this project's implementation of visual tracking safety nets and fallback control policies. In particular, their emphasis on structured decision logic aligns with this work's dynamic PWM override during unexpected bounding box changes, offering proactive maneuvering within constrained areas.

[43] NASA introduces the SAFEGUARD system, designed as an assured safety net to enforce UAV boundary restrictions. This lightweight, hardware-compatible module monitors UAV position and velocity to prevent exit from pre-approved geofences. The concept of a passive monitoring subsystem strongly supports this project's secondary geofence-checking logic that runs parallel to the AI object tracker. This decoupling of safety control from primary navigation is crucial for real-time responsiveness and compliance with safety norms in dynamic UAV missions.

[44] The ArduPilot platform is an open-source autopilot system that supports various UAV architectures. Its comprehensive suite of navigation, control, and telemetry features forms the backbone for many academic UAV projects. For this work, ArduPilot acts as an ecosystem reference for modular control integration, particularly its capacity for hardware-in-the-loop testing and waypoint management. The project's PWM computation model is inspired by ArduPilot's flexible mixing and override channels, demonstrating how low-level control can be extended to AI-driven input sources.

[45] Dronecode's PX4 autopilot system offers a real-time operating system (RTOS)-driven flight stack, optimized for advanced sensing and decision logic in drones. The modularity and abstraction of the PX4 middleware inspire this project's layered software design, where object detection, joystick control, and geofence enforcement operate as loosely coupled agents. PX4's support for external object detection modules validates the direction of this project's neural-based object-tracking and follow-me behavior, illustrating the real-world deployability of such concepts.

[46] Cleanflight is another community-driven flight control firmware tailored for high-speed and acrobatic drones. Although its use case diverges from stable UAV surveillance,

the firmware's emphasis on efficient PID tuning and fast loop execution offers parallels to this project's PWM update strategy. The latency-aware architecture of Cleanflight supports real-time attitude corrections—vital for this work, where bounding box shifts must immediately affect drone pitch, yaw, or throttle to maintain lock on the tracked object.

[47] Emlid's Navio2 HAT for Raspberry Pi provides flight controller capabilities directly integrated with a general-purpose computing platform. It offers GPS, IMU, and PWM control within a single compact board. The project's hardware architecture aligns closely with this—utilizing a similar integration of Raspberry Pi with AI modules for camera input and real-time decision-making. The Navio2's interface standardizes sensor fusion, which this UAV system echoes by combining bounding box feedback with GPS awareness for smarter movement control.

[48] Betaflight, like Cleanflight, is known for fast-loop performance and aggressive PID tuning—attributes valuable in drone racing. However, its contribution here lies in its telemetry and logging capabilities, which inform the project's flight history logging module. This allows post-flight analysis of object detection quality, bounding box dimensions, and drone responses, helping evaluate and optimize the PWM tuning pipeline in follow-me mode.

[49] Gurriet and Ciarletta propose a modular geofencing framework that separates drone autonomy logic from geofence enforcement layers. Their work introduces hierarchical boundaries—hard and soft—that can trigger various control strategies. This exact logic is adopted in the project, where outer geofence violations trigger immediate stop or override, while inner buffer zones adjust drone response thresholds. The paper provides architectural support for scalable geofence logic across multi-object UAV missions.

[50] Stevens, Rastgoftar, and Atkins propose a set of algorithms to detect geofence boundary violations. Their method leverages both predictive and reactive control layers to anticipate and correct UAV trajectories near boundaries. This dual-mode behavior is key to this project's approach, where the bounding box object-tracking model is supplemented by GPS-aware geofence suppression. Their results reinforce the idea that proactive boundary estimation enhances flight safety without significantly increasing computational overhead.

[51] Bélair provides a foundational introduction to alpha shapes, a geometric construct

used to define the shape of point sets. Though the content is abstract and theoretical, its relevance to UAV path planning lies in the construction of flight boundaries and obstacle representations. This concept can be applied in enhancing geofence models by enabling irregular-shaped region definitions, offering more precise and efficient spatial reasoning than traditional rectangular bounding.

[52] Articque introduces an enterprise-level geospatial data visualization platform capable of integrating customizable boundary constraints and analytics. Although commercial in nature, the platform's modular representation of spatial data flows offers architectural inspiration for this project's GUI-based geofencing and visualization modules. The ability to dynamically adjust geospatial logic aligns with the system's geofence overlays based on GPS position streams and bounding box offsets.

[53] Chen and Chen present a UAV path planning algorithm that utilizes Voronoi diagrams for real-time spatial decomposition. Their method allows drones to calculate efficient navigation paths while avoiding predefined obstacle zones. This geometric strategy aligns well with this project's future goal of integrating alpha shapes or Voronoi-based geofences into the navigation stack. Their lightweight solution supports onboard computation, critical for embedded UAV controllers.

[54] Pehlivanoglu proposes a vibrational genetic algorithm combined with Voronoi diagrams to optimize UAV path planning. The hybrid method balances exploration and exploitation for obstacle-avoiding routes. This optimization-focused approach complements the reactive nature of the current project's visual tracking system, suggesting pathways for improving path reactivity during object-following scenarios, particularly under partial occlusions or shifting GPS coverage.

[55] Bortoff addresses traditional UAV path planning through control-theoretic formulations, establishing early foundations for guidance and trajectory management. Although devoid of machine learning, the emphasis on linearization and waypoint sequencing informs this project's use of trajectory-like behavior, wherein bounding box aspect ratios are indirectly treated as waypoints for motion correction.

[56] Castells discusses the use of alpha shapes within the R programming environment for visualizing and manipulating complex spatial structures. His application to Voronoi tessellations supports non-convex boundary handling. This supports the extension of the current geofencing logic to more adaptive geometries in future phases of the project,

especially in terrain-aware surveillance scenarios.

[57] Shah et al. introduce AirSim, a high-fidelity UAV simulation platform developed by Microsoft. It simulates physics, sensors, and environmental conditions, allowing developers to train and test autonomous agents before real deployment. This project can integrate AirSim to validate object-following and bounding-box-based tracking under varying lighting, wind, and object speed conditions before live field testing, reducing real-world risk and improving iteration cycles.

[58] Bellock provides a Python-based alpha shape toolbox tailored for easy boundary extraction from 2D point clouds. The lightweight nature of the library makes it ideal for onboard applications. This complements the project's use of bounding box edges as virtual boundary indicators, and offers a natural upgrade path toward shape-based airspace segmentation without deep vision dependencies.

[59] The Raspberry Pi 3 Model B is cited as the central computing unit in several academic UAV platforms due to its compact form factor and GPIO/PWM compatibility. This project also builds around a similar board, leveraging it for AI-based inference (e.g., YOLO object tracking) and direct motor control. Its compatibility with standard UART and I2C buses facilitates seamless integration with GPS and IMU modules, forming the physical backbone of the UAV.

[60] Erle Robotics documents its simulation platform for UAV training and deployment, emphasizing modular architecture and compatibility with ROS. Their simulation framework validates many control theories and geofencing algorithms before hardware rollout. This reflects the approach taken in this project, where testbed validation is performed using simulated feedback loops for bounding box accuracy, PWM response, and safety overrides before full-scale UAV deployment.

[61] The OpenStreetMap (OSM) Wiki is a community-maintained geospatial resource enabling global collaborative mapping. Its inclusion in this project underscores the importance of open-source mapping layers for route planning and geofencing visualization. OSM's fine-grained data supports integrating real-world constraints—like road boundaries or restricted zones—into the drone's flight logic, enriching both simulation and execution environments.

[62] QGIS is a leading open-source Geographic Information System used for spatial data analysis and visualization. In the context of this UAV project, QGIS can preprocess

geofencing zones and convert them into machine-readable formats. Its compatibility with GPS coordinate sets and vector layers supports drawing dynamic boundaries, enabling UAVs to adapt their operating areas based on predefined safety limits or mission-specific spatial constraints.

[63] Shi, Liu, and Zhou present a comprehensive survey on deep learning methods applied to UAV-based object detection. Covering convolutional networks, lightweight model architectures, and performance benchmarks, their review validates the choice of using real-time object detectors like YOLO in this project. Their analysis of trade-offs between speed and accuracy directly informs the project's balance between rapid inference and reliable target lock.

[64] Amarnath and Kumar review object detection algorithms as applied in UAV surveillance contexts. They categorize methods into traditional vision-based tracking and CNN-based detection, highlighting the shift toward deep models. This aligns with the current project's neural backbone for bounding box detection and tracking, affirming the superiority of AI-powered detection in noisy outdoor UAV operations.

[65] Wang et al. survey the dual problem of object detection and tracking for UAVs using deep learning. Their insights into temporal coherence and multi-frame association are particularly useful for this project, where bounding box smoothing across frames ensures stability in drone motion. Their emphasis on combining detection with memory further supports the idea of temporal feedback in the object-follow loop.

[66] Li et al. propose AutoTrack, an adaptive visual tracker for UAVs that adjusts spatial constraints dynamically based on temporal inputs. Their model introduces spatio-temporal regularization, boosting long-term accuracy in dynamic scenes. This closely resembles the project's use of bounding box aspect ratios and their temporal deltas for fine-tuned PWM control, reinforcing the utility of dynamic visual feedback for UAV tracking.

[67] Schilling et al. present a drone flocking system using vision-based coordination, where each UAV responds to neighboring visual cues rather than GPS alone. This paper supports the feasibility of decentralized control using only visual input, a principle embedded in this project's real-time object tracking. The use of swarm logic could inform extensions of this project toward multi-drone follow-me behavior.

[68] Chen's thesis documents the design of an intelligent UAV system, including sensor

integration, flight logic, and mission control. The modular architecture and emphasis on autonomous perception validate this project's layered design, where object detection, PWM control, and geofence validation operate independently. His prototype reinforces practical approaches for robust UAV deployment using minimal hardware.

[69] Huang presents a mathematical model of quadcopter dynamics using Newtonian mechanics. This analytical foundation supports control design, simulation, and validation. Although this project abstracts these dynamics behind a PWM-based control loop, the underlying model is essential for understanding how bounding box shifts translate to changes in drone roll, pitch, or yaw, especially during tuning or simulation.

[70] Zhang introduces a flexible technique for camera calibration using known patterns and optimization-based refinement. Accurate camera calibration is critical in this project for extracting reliable bounding box coordinates, minimizing detection offset. Zhang's method ensures that visual inputs are geometrically consistent, enhancing the precision of object localization and thus the overall responsiveness of the tracking algorithm.

The literature reviewed encompasses a diverse yet interconnected body of work spanning UAV control, object detection, geofencing, and autonomous decision-making. Early foundational studies laid the groundwork for traditional path planning and boundary enforcement using geometric and control-theoretic principles. These works underscore the relevance of alpha shapes, Voronoi diagrams, and sliding mode control strategies in ensuring flight stability and constraint adherence.

Recent advancements reflect a paradigm shift towards AI-enabled autonomy, especially through the integration of deep learning for object detection and tracking. Surveys and implementation-focused papers highlight the evolution from classical vision systems to convolutional architectures like YOLO and reinforcement learning for temporal consistency. These align with the current project's methodology of using bounding box aspect ratios and visual cues for real-time PWM modulation.

Geofencing, as addressed across commercial platforms (e.g., DJI FlySafe, PX4, ArduPilot) and academic studies, emerges as a critical pillar for ensuring UAV operational safety. The reviewed papers propose both predictive and reactive frameworks for geofence monitoring, offering architectural validation for the layered safety logic implemented in this system.

Moreover, contributions from simulation environments (e.g., AirSim, Erle Robotics),

sensor integration platforms (e.g., Navio2, Raspberry Pi), and GIS tools (e.g., QGIS, OpenStreetMap) have influenced the modular and simulation-friendly design of the project. These tools bridge the gap between theoretical design and field-ready implementation.

In synthesis, the literature affirms the technical soundness and novelty of this project, which uniquely combines real-time AI object tracking, PWM-based dynamic control, and GPS-geofence enforcement within a single cohesive UAV framework. The surveyed works not only validate the components used but also suggest viable future directions—such as 3D path planning, multi-drone coordination, and adaptive geofence generation—to further extend the project's scope.

1.6 Brief Methodology of the Project

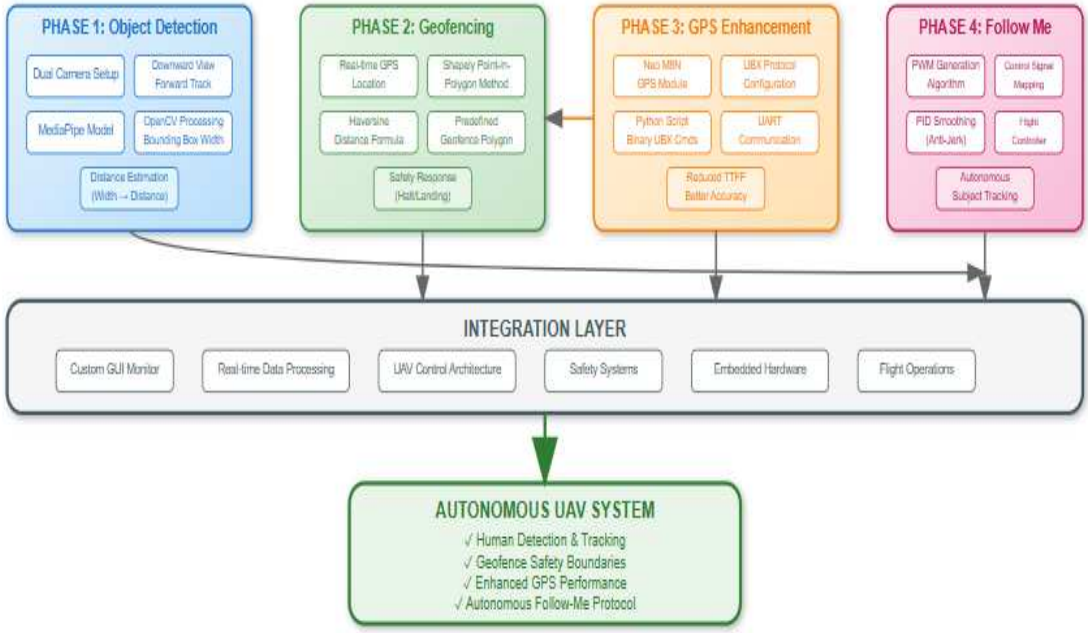


Figure 1.2: Methodology of AI-GPS Smart Mini Drone

The project was carried out in a modular and phase-wise approach as seen in Figure 1.2, ensuring that each subsystem was independently developed, tested, and finally integrated into the complete UAV control architecture. This approach spanned both software and embedded hardware components, allowing the drone to autonomously detect and track a human subject, respect geofence boundaries, and interact with a custom GUI for monitoring and control.

In the first phase, a basic object detection system was implemented using MediaPipe's lightweight person detection model. This was tested with a dual-camera setup—one cam-

era for downward viewing and another for forward-facing tracking. The object detection output was processed in real time using OpenCV to extract bounding box dimensions, particularly the width. This width served as a proxy for estimating the subject's distance from the drone and laid the foundation for future control logic.

In the second phase, a geofencing module was developed to enforce safe flight boundaries. Using Shapely's point-in-polygon method along with the Haversine distance formula, the module continuously compared the drone's real-time GPS location against a predefined geofence polygon. If a breach occurred, the system raised a flag that triggered safety responses like emergency halt or landing, depending on the current flight mode.

The third phase focused on enhancing GPS performance through UBX protocol configuration. A Python-based script was created to send binary UBX commands to the Neo M8N GPS module via UART, allowing for position preloading and reducing the Time-To-First-Fix (TTFF). This ensured improved location accuracy and reliability during field operations.

In the final phase, the full Follow Me protocol was developed and integrated. The previously established bounding box width from the detection system was mapped to control signals using a custom PWM generation algorithm. These signals adjusted drone parameters like pitch and throttle to maintain an optimal following distance. PID smoothing was employed to prevent jerky movements. The control signals were then transmitted to the flight controller for motion execution, enabling autonomous subject tracking.

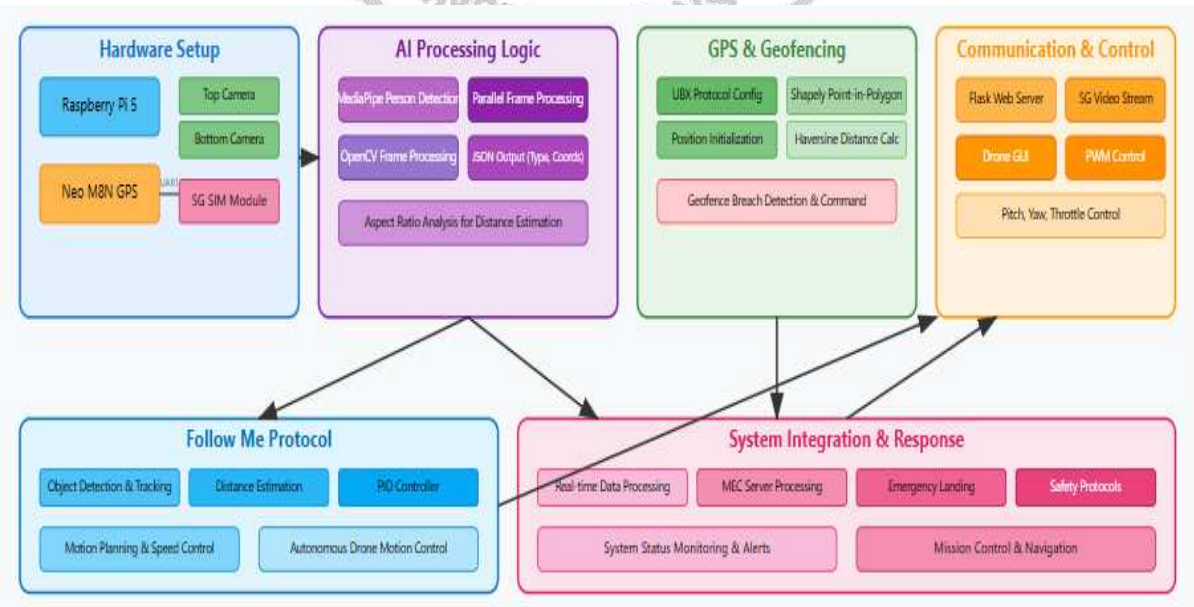


Figure 1.3: Smart Control System with GPS and AI Integration for Mini Drone

The system block diagram as in Figure 1.3 visually encapsulates the modular architecture and data flow across the drone's subsystems. Each color-coded block in the diagram corresponds to a specific functional domain:

Hardware Setup includes the Raspberry Pi 5, dual camera configuration, Neo M8N GPS, and 5G SIM module. These serve as the physical and communication backbone of the UAV system.

AI Processing Logic depicts the MediaPipe and OpenCV-based person detection modules, frame processing logic, and bounding box analysis for distance estimation. JSON outputs from this stage are fed into control logic for motion actuation.

GPS & Geofencing covers the UBX configuration and real-time position monitoring. It uses geometric and distance-based checks to detect geofence breaches and issue safety commands accordingly.

Follow Me Protocol integrates object tracking with PID-controlled motion planning. This block is responsible for translating visual tracking data into PWM signals for stable and responsive drone following behavior.

System Integration & Response manages real-time data synchronization, MEC server integration, mission control, and emergency handling. It acts as the central decision layer, fusing inputs from all subsystems.

Communication & Control includes the Flask-based web server, GUI dashboard, 5G video streaming, and manual PWM controls. This provides a human interface for monitoring and controlling drone operations.

Key Features, Technical Specifications, and Project Objectives& Motivation are summarized in the bottom panels, highlighting the system's capabilities, underlying tools, and the broader purpose of developing an indigenous, AI-enabled UAV platform.

This Figure also maps out the logical and physical relationships between modules.

1.7 Assumptions made / Constraints of the project

The Assumptions made/ Constraints of the project are:

1. Objective 1 - The drone is expected to operate under adequately lit conditions. Object detection accuracy significantly decreases in low-light environments, hence this is considered an **assumption** that ambient lighting will remain sufficient throughout the operation.

2. Objective 2 - It is an **assumption** that GPS coordinate jitter will not exceed a deviation of 10 meters, even in the presence of minor environmental obstructions. This enables reliable positioning for geofencing and navigation.

3. Objective 3 - The system has a **constraint** that requires manual script updates if the drone is deployed in a location vastly different from its previous area of operation—such as a different town or city—since the geofence coordinates are hard-coded. However, this limitation is rarely problematic, as each drone is pre-assigned to a fixed operational zone with predefined coordinates based on mission requirements.

4. Objective 4 - A key **constraint** of the system is that it functions only when a single person is detected within the frame. In cases where multiple individuals are detected, the drone remains stationary to avoid ambiguity in target tracking and prevent unintended motion.

1.8 Organization of the report

This report is organized as follows:

Chapter 2 discusses the theory behind the operation of the drone and the features developed as part of this project, including UAV fundamentals, GPS logic, AI-based object detection, and geofencing mechanisms.

Chapter 3 discusses the methodology behind each of the objectives in a detailed manner, such as how bounding box width was used for PWM control, geofence breach detection logic, and UBX GPS configuration steps.

Chapter 4 discusses the implementation of hardware and software modules. It covers system integration on Raspberry Pi, GUI-based telemetry visualization, socket communication, and deployment flow.

Chapter 5 discusses the experimental results and validation of each module. It includes GPS lock tests, object detection performance, geofencing breach response, and PWM mapping with bounding box width.

Chapter 6 discusses the conclusion of the project, highlighting key outcomes, challenges faced, and future scope for extending the system to include altitude control and multi-object tracking.

.

Chapter 2

System Specifications and Underlying Technologies for AI-Guided UAVs

CHAPTER 2

SYSTEM SPECIFICATIONS AND UNDERLYING TECHNOLOGIES FOR AI-GUIDED UAVS

The successful development of an intelligent mini drone system relies heavily on a clear understanding of both its theoretical principles and the technologies deployed. This chapter outlines the foundational concepts underpinning the project, including the architecture of Unmanned Aerial Vehicles (UAVs), the integration of Artificial Intelligence (AI) in embedded systems, and the core mechanisms behind object detection using convolutional neural networks. It further elaborates on the role of GPS technology, particularly Time-To-First-Fix (TTFF) optimization, and the mathematical basis of geofencing using spatial algorithms like point-in-polygon and the haversine formula. Together, these components establish the technical groundwork for the drone's autonomous and responsive behavior.

2.1 Specifications and Theoretical Foundations

2.1.1 Basics of Unmanned Aerial Vehicles (UAVs)

Unmanned Aerial Vehicles (UAVs), commonly known as drones, are aerial systems that operate without a human pilot onboard. UAVs are widely used in fields such as surveillance, agriculture, delivery logistics, and disaster management. They consist of multiple subsystems:

- **Airframe and Propulsion:** Provides lift and movement using motors and propellers.
- **Flight Controller:** Embedded processor managing sensor data, stability, and navigation.
- **Sensors:** Includes IMU, GPS, barometer, and magnetometer for positional awareness.
- **Communication System:** Enables telemetry, control, and video transmission via RF, Wi-Fi, or 5G.
- **Payload:** Typically cameras or other sensors suited to the mission.

This project uses a mini quadcopter drone controlled by a Raspberry Pi 5, equipped with GPS and dual cameras for smart navigation and AI-assisted features.

2.1.2 Overview of AI in Embedded Systems

Artificial Intelligence (AI) integrated into embedded platforms allows for intelligent decision-making at the edge. Such systems are optimized for low power consumption and real-time inference without reliance on cloud resources.

- **Lightweight Models:** Quantized models such as EfficientDet Lite enable fast inference on low-resource hardware.
- **Inference Engines:** MediaPipe and TensorFlow Lite support real-time execution on ARM processors.
- **Edge Computing:** AI processing happens on devices like Raspberry Pi or Jetson Nano, ensuring low latency and privacy.

The drone project implements real-time object detection using MediaPipe, streamed over Flask from the Raspberry Pi to a client interface.

2.1.3 Fundamentals of Object Detection using CNN/MediaPipe

Object detection involves identifying and locating objects within images using Convolutional Neural Networks (CNNs). These networks consist of convolutional layers that extract hierarchical features from input images.

MediaPipe, developed by Google, offers pre-trained, pipeline-optimized models suitable for embedded AI applications. In this project:

- The `efficientdet_lite0.tflite` model is used.
- Outputs include bounding box coordinates, class labels, and confidence scores.
- The bounding box width informs the "Follow Me" protocol for drone navigation.

2.1.4 GPS Technology and Time-To-First-Fix (TTFF)

The Global Positioning System (GPS) uses satellite triangulation to compute geographic location. A critical performance metric is **Time-To-First-Fix (TTFF)**.

- **Cold Start:** No prior data; highest TTFF.

- **Warm Start:** Retains last known position but not satellite data.
- **Hot Start:** All data present; minimal TTFF.

To optimize TTFF, UBX binary protocol commands are sent to the Neo M8N GPS module using UART from the Raspberry Pi. These include `MGA-INI-POS_XYZ` and `CFG-CFG` to preload position and save configuration.

2.1.5 Geofencing Logic using Point-in-Polygon and Haversine Formula

Geofencing constrains drone movement within a defined boundary. This is implemented using:

- **Point-in-Polygon (PIP):** Geometric check using the Shapely library to verify if a GPS coordinate lies inside a polygon.
- **Haversine Formula:** Calculates the great-circle distance between two latitude-longitude points:

$$d = 2r \cdot \arcsin\left(\sqrt{\sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\Delta\lambda}{2}\right)}\right) \tag{2.1}$$

where r is Earth's radius, ϕ and λ are latitudes and longitudes in radians. This logic triggers a `BREACHED` signal if the drone crosses or touches the geofence boundary.

Table 2.1: Comparison of AI Models for Edge Object Detection

Model	Framework	Size (MB)	FPS (RPI5)	Accuracy	Edge Suitability
YOLOv5s	PyTorch	14	3–5	High	Requires GPU
SSD MobileNet V2	TF Lite	6	8–10	Medium	Acceptable
EfficientDet-Lite0	TF Lite	4	15–20	Good	**Optimal**

2.2 Programming Environment

2.2.1 Python as the Primary Programming Language

Python 3.13.1 was chosen as the core programming language due to its readability, extensive standard library, and vast ecosystem of third-party modules. Python supports rapid development and is especially suited for prototyping on embedded platforms like Raspberry Pi. Its interpreted nature and compatibility with edge inference frameworks make it ideal for AI applications in robotics and UAV control systems.

2.2.2 Use of Flask for Video Streaming and GUI

Flask, a lightweight Python web framework, was utilized to set up a server that streams live video feeds and AI detection output from the Raspberry Pi. It plays a critical role in the project by enabling:

- Real-time HTTP-based video streaming from dual cameras.
- Routing between different modes (object detection vs. normal streaming).

Each camera stream is assigned a specific port and route (e.g., `/normal_detect_0`, `/normal_detect_0`), allowing easy switching between feeds on the GUI. Flask's REST endpoints are also used for JSON-based communication of detection metadata.

2.2.3 Library Support and Functionality

The functionality of the drone software stack is enabled by a diverse set of Python libraries. Table 2.2 outlines the key libraries and their roles in the system.

Table 2.2: Python Library Roles in the Project

Library	Functionality
`OpenCV`	Acquires video frames, performs basic image processing, handles bounding box overlays.
`MediaPipe`	Executes real-time object detection using the EfficientDet Lite 0 model.
`Flask`	Hosts the web server for live video streaming and API-based data transmission.
`socket`	Facilitates TCP communication between GPS/geofence logic and the control system.
`Shapely`	Performs geometric operations such as point-in-polygon checks for geofencing.
`struct`	Constructs binary UBX packets for configuring the GPS module via UART.
`picamera2`	Interfaces with the Raspberry Pi cameras for frame capture and resolution settings.

Each of these libraries is vital for either perception, communication, or control subsystems, contributing to the integrated performance of the drone.

2.3 Tools and Platforms Used

2.3.1 Raspberry Pi 5: Features and OS Setup

The Raspberry Pi 5 serves as the central controller for the drone. Its quad-core ARM Cortex-A76 processor and 8GB RAM provide sufficient computational power for running AI models and managing real-time control systems. A headless Debian Bookworm OS was used to ensure lightweight, script-driven operation. Python programs for object detection, GPS, and geofencing run via persistent 'systemd' services.

2.3.2 Camera Modules (Pi Camera V3 and 5MP)

Two cameras are integrated for visual input:

- **Pi Camera V3**: Wide-angle top-facing camera for object tracking.
- **5MP Camera Module**: Downward-facing camera for ground observation.

These are controlled using the `picamera2` library and their outputs are routed to different Flask endpoints for GUI display.

2.3.3 Ublox Neo M8N GPS Module

The Neo M8N GPS module provides continuous position tracking via satellite triangulation. Communication is established over UART, with Python scripts sending UBX commands to preload location data and configure persistent settings. The GPS data is critical for both geofencing logic and real-time drone localization.

2.4 Additional Theoretical References

- **AI Model Selection – EfficientDet Lite 0:** Chosen for its balance of speed, accuracy, and size, this model runs efficiently on Raspberry Pi without GPU, making it ideal for real-time edge inference.
- **PWM Control Logic in Drones:** Drone movement is regulated using PWM signals based on bounding box width. A wider box indicates proximity (triggering braking), while a narrow box prompts forward motion.
- **Socket-Based Communication in IoT:** TCP sockets enable real-time messaging between components. Used here to send geofence breach alerts to trigger emergency landing actions.

The chapter presented the technical foundations required for implementing a smart drone system, including UAV architecture, embedded AI with MediaPipe, GPS optimization using UBX commands, and geofencing with spatial logic. It also described the programming environment, key libraries, and hardware tools like Raspberry Pi and GPS modules. These specifications form the basis for the system's autonomy and responsiveness. The upcoming chapter will build upon this groundwork by detailing the methodology adopted for integrating and deploying each subsystem in the actual drone environment.

Chapter 3

Design of Smart Control Architecture for Mini Drone

CHAPTER 3

DESIGN OF SMART CONTROL ARCHITECTURE FOR MINI DRONE

Designing a responsive and autonomous drone system required careful integration of hardware components and software modules. Key design considerations included AI-enabled object detection, GPS-based geofencing, and socket-driven control protocols. Hardware modules like Raspberry Pi 5, dual camera systems, and the Neo M8N GPS were selected to support edge-based processing and real-time communication. The strategies adopted for implementing these components and validating their performance are described in the following sections.

3.1 Specifications for the Design

The design of the smart mini drone system was guided by the need for real-time responsiveness, modular integration, and autonomy under constrained resources. The following design specifications were defined based on functional goals and environmental constraints:

- **Processing Unit:** Raspberry Pi 5 (8GB RAM) selected for its quad-core ARM Cortex-A76 processor, supporting Python-based AI inference and real-time control logic.
- **Camera Modules:** Dual-camera setup comprising Pi Camera V3 (wide-angle) and a 5MP camera for forward and downward vision respectively. These facilitate AI object detection and ground observation.
- **AI Model:** `efficientdet_lite0.tflite` chosen for its lightweight nature and real-time inference capability on edge devices.
- **GPS Module:** Ublox Neo M8N GPS with UART interface, supporting UBX protocol for TTFF optimization and reliable positioning.
- **Geofencing Logic:** Implemented using point-in-polygon tests (Shapely) and distance calculations (Haversine formula) to define and enforce safe operational boundaries.

- **Follow-Me Protocol:** Based on bounding box width derived from object detection to drive PWM values for pitch, yaw, and throttle adjustment.
- **Software Stack:** Python 3.13.1 with Flask for GUI streaming, MediaPipe for AI, and socket programming for inter-process communication.
- **Power Supply:** Lithium polymer (Li-Po) battery with sufficient capacity to power Raspberry Pi, camera modules, GPS, and motor controller.
- **Control Interface:** GUI dashboard displaying live camera feed, detection data, GPS status, and geofence alerts over local or 5G network.

3.2 Pre-analysis and Models Used

Before implementation, preliminary evaluations were conducted to ensure compatibility of AI models, GPS protocols, and control logic with the hardware setup. These pre-analysis steps helped in selecting optimized solutions suited for real-time embedded deployment.

- **Model Benchmarking:** Multiple object detection models including YOLOv5s, SSD MobileNet, and EfficientDet Lite were evaluated. `efficientdet_lite0.tflite` was finalized due to its balance between inference speed (15–20 FPS on Raspberry Pi) and acceptable detection accuracy.
- **Camera-Model Integration:** Frame rates and detection consistency were tested with Pi Camera V3 and 5MP module. Compatibility with `picamera2` ensured stable acquisition and preprocessing.
- **UBX Protocol Validation:** UBX messages were constructed and tested on the Neo M8N GPS module using the `struct` library. Position preloading and ACK/-NACK handling were confirmed for TTFF reduction.
- **Bounding Box Analysis:** Sample object detection outputs were recorded to calibrate bounding box width thresholds. These values were mapped to PWM pitch settings to enable the "Follow Me" behavior.
- **Geofencing Simulation:** Polygonal regions were defined in code using the Shapely library. Point-in-polygon tests and haversine distance checks were simulated with sample coordinates to verify breach logic.

3.3 Design Methodology

The development of the smart control system followed a modular and task-specific methodology. Each core functionality was developed independently, tested, and then integrated into a unified control pipeline. The methodology was divided into four key design tasks:

Task 1: AI-Based Object Detection

- Two cameras were interfaced to the Raspberry Pi for simultaneous top and bottom views.
- Object detection was implemented using MediaPipe and the `efficientdet_lite0.tflite` model.
- Detection results including bounding box coordinates and object class were transmitted via Flask to the drone GUI.
- A controller script managed routing of video streams across four modes (2 cameras × 2 modes).

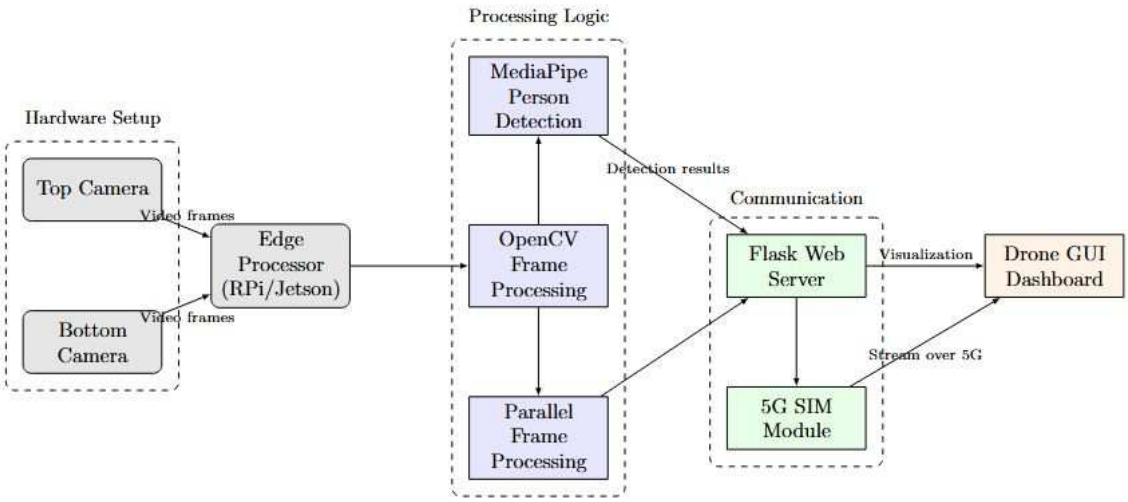


Figure 3.1: AI Object Detection Block Diagram

Figure 3.1 illustrates the end-to-end processing flow for AI-driven object detection

and video streaming on the edge processor. The system uses dual cameras—top and bottom—whose frames are fed into the Raspberry Pi. Frame processing is handled by OpenCV, while MediaPipe performs object detection. The resulting detection data is streamed using a Flask web server, optionally routed through a 5G SIM module, and visualized on a GUI dashboard. This modular processing logic enables simultaneous detection and transmission from both video sources.

Task 2: Geofencing Implementation

- Geofence boundaries were defined using polygonal coordinates.
- The drone's GPS coordinates were continuously checked using Shapely's point-in-polygon function.
- Haversine formula was used to compute proximity to the fence edge and trigger early warnings.
- A socket message labelled `BREACHED` was sent to initiate landing when boundaries were crossed.

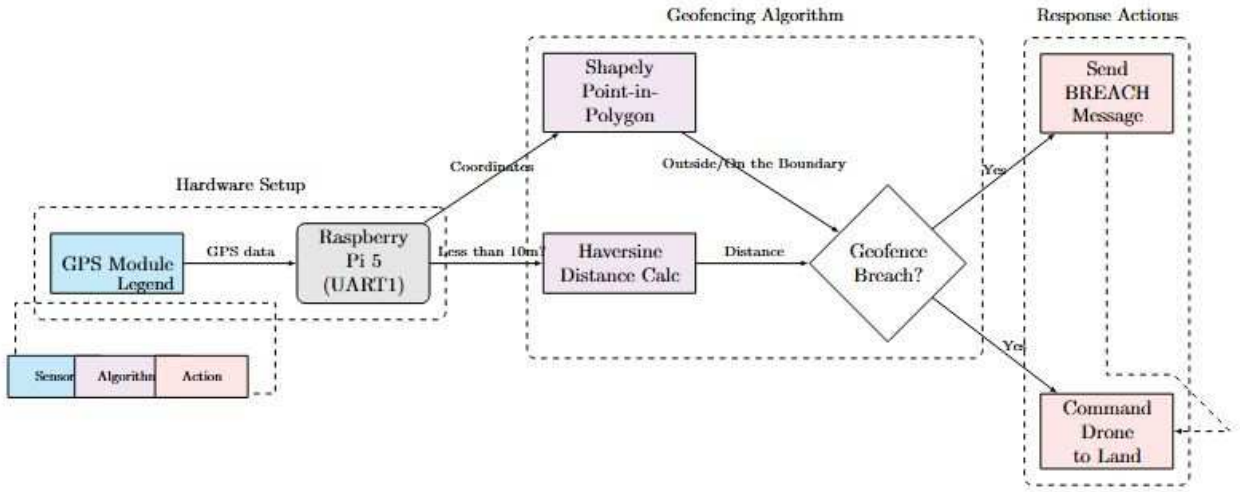


Figure 3.2: Geofencing Block Diagram

Figure 3.2 presents the geofencing algorithm implemented on the Raspberry Pi. GPS data received from the Neo M8N module is evaluated using two techniques: Shapely's point-in-polygon logic for spatial inclusion and the Haversine formula for distance measurement. If a breach is detected, a socket-triggered alert is raised and the drone is

instructed to land. This layered logic flow ensures that geofencing decisions are both geometrically and geographically accurate.

Task 3: GPS Configuration via UBX Protocol

- UBX-MGA-INI-POS_XYZ messages were sent over UART from Raspberry Pi to preload known position data into the GPS module.
- A UBX-CFG-CFG command saved the configuration to flash, reducing TTFF on reboot.
- Python's `struct` library was used for checksum computation and UBX packet formatting.

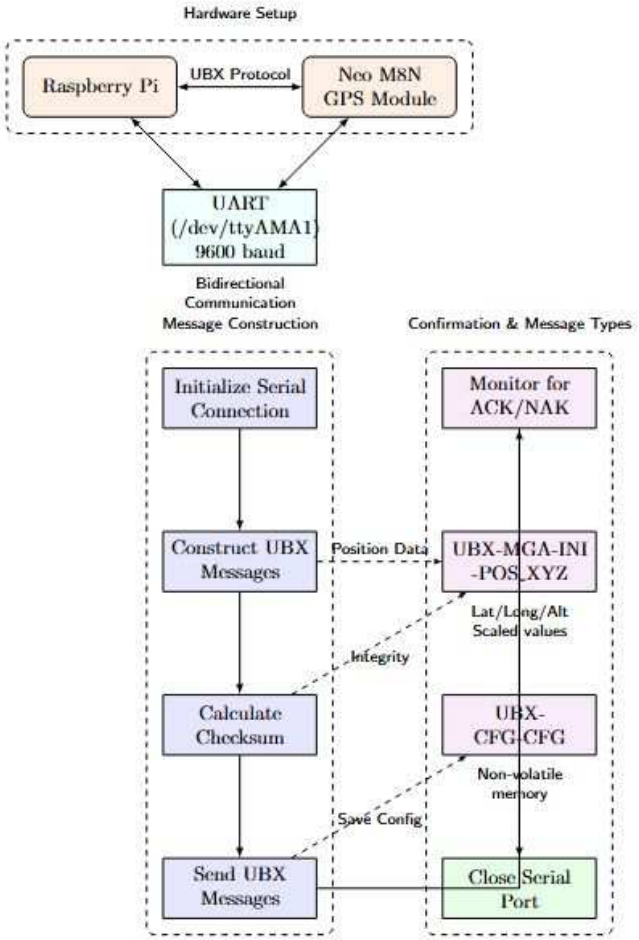


Figure 3.3: GPS UBX Configuration Block Diagram

Figure 3.3 explains the sequence for configuring the GPS module using the UBX

protocol. It starts with serial initialization followed by UBX message construction and checksum validation. Specific UBX message types—such as `UBX-MGA-INI-POS_XYZ` and `UBX-CFG-CFG`—are used to preload known positions and store configurations persistently. The flow also includes monitoring ACK/NAK responses to confirm communication integrity.

Task 4: Follow-Me Protocol Based on Bounding Box Width

- Bounding box width from object detection output was mapped to PWM pitch values.
- Wider boxes indicated closer objects and triggered slower or reverse motion.
- Narrower boxes indicated distance and prompted the drone to move forward.
- A control loop continuously adjusted pitch, yaw, and throttle based on bounding box size.

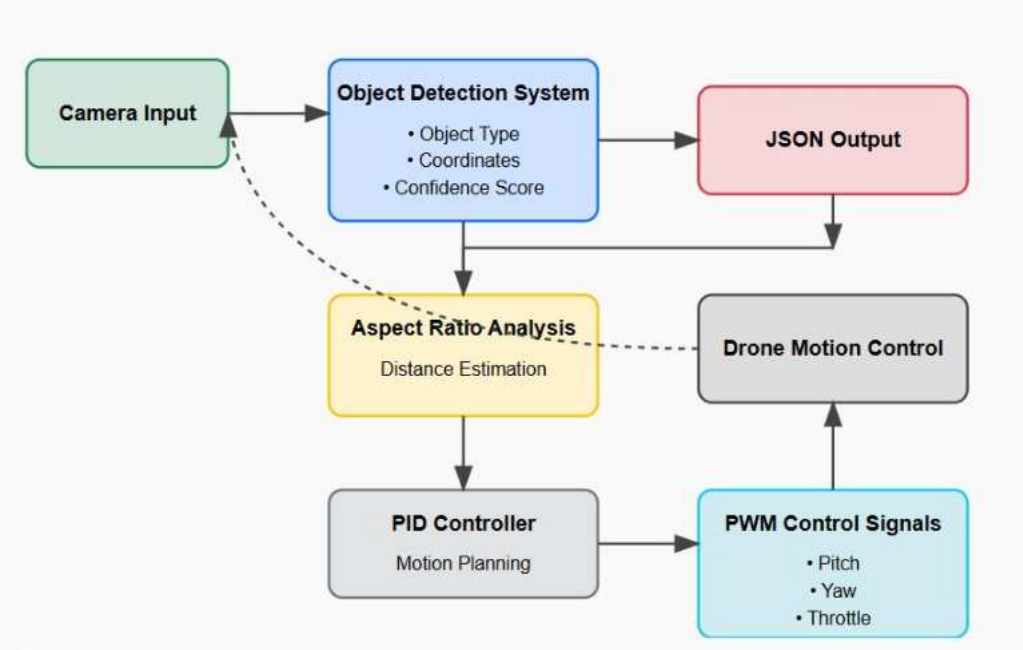


Figure 3.4: Follow Me Protocol Block Diagram

Figure 3.4 shows the internal control logic of the drone's follow-me protocol, which begins with object detection and aspect ratio analysis. The bounding box output from the camera is used to estimate the distance of the target. This estimate is passed through a PID controller that plans the drone's motion and generates the appropriate PWM

signals for pitch, yaw, and throttle. The use of JSON output allows for modularity between detection and control modules.

3.4 Equations and Logic

- **Haversine Formula for Distance Calculation:**

$$d = 2r \cdot \arcsin\left(\sqrt{\sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\Delta\lambda}{2}\right)}\right) \tag{3.1}$$

- **Point-in-Polygon Logic (Shapely):**

```
if polygon.contains(Point(lat, lon)):
    status = "Inside"
```

Equations such as the Haversine distance formula (Eq. 3.1) and point-in-polygon logic are fundamental to geofencing. They enable the drone to determine its proximity to boundaries and assess zone violations.

- **UBX Checksum Calculation:**

$$\text{CK}_A = \sum_{i=0}^{n} \text{Payload}_i \mod 256, \quad \text{CK}_B = \sum_{i=0}^{n} \text{CK}_A \mod 256 \tag{3.2}$$

Equation 3.2 defines the checksum algorithm used in constructing UBX protocol messages for reliable GPS configuration.

- **Distance from Bounding Box Area:**

$$D_{\text{object}} \propto \frac{1}{A_{\text{box}}}, \quad A_{\text{box}} = \text{width} \times \text{height} \tag{3.3}$$

- **X-Axis Velocity from Bounding Box Width:**

$$v_x \propto (\text{DES\_BOX\_W} - w) \tag{3.4}$$

- **Bounding Box Error Calculations:**

$$\text{err}_x = \frac{c_x - \text{IMG}_W/2}{\text{IMG}_W/2}, \quad \text{err}_{\text{dist}} = \frac{\text{DES\_BOX\_W} - w}{\text{DES\_BOX\_W}} \tag{3.5}$$

- **Clamped Velocity to PWM Conversion:**

$$\text{PWM}_x = \text{velocity\_to\_pwm}\left(\max(-1.0, \min(1.0, v_x))\right) \tag{3.6}$$

$$\text{PWM}_y = \text{velocity\_to\_pwm}\left(\max(-1.0, \min(1.0, v_y))\right) \tag{3.7}$$

- **PID Control for PWM Output:**

$$\text{PWM}_{\text{control}} = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} \tag{3.8}$$

- **Discrete Threshold Mapping for Pitch:**

$$\text{PWM}_{\text{pitch}} = \begin{cases} 1000 & \text{if width} > 450 \\ 1080 & \text{if } 400 < \text{width} \leq 450 \\ 1268 & \text{if } 350 < \text{width} \leq 400 \\ 1350 & \text{if width} < 350 \end{cases} \tag{3.9}$$

Equations 3.3 through 3.9 define the logic for the drone's follow-me functionality. Eq. 3.3 estimates object distance, Eq. 3.4 and Eq. 3.5 calculate movement errors, and Eq. 3.6 and Eq. 3.7 maps those to PWM-compatible values. PID smoothing (Eq. 3.8) and discrete pitch mapping (Eq. 3.9) provide flexibility in control strategies.

The design and implementation of the drone's control system were carefully structured through detailed specification, model selection, and subsystem integration. This chapter covered the architectural logic behind GPS-based geofencing, UBX-driven TTFF optimization, and AI-enabled object tracking with corresponding follow-me behavior. Each functional block was supported by quantitative equations and logical validation, ensuring reliable performance in dynamic environments. The next chapter builds upon this foundation by describing the results, testing procedures, and performance evaluation of

the implemented system.

Chapter 4

Implementation and Integration of Smart Control System for Mini Drone

CHAPTER 4

IMPLEMENTATION AND INTEGRATION OF SMART CONTROL SYSTEM FOR MINI DRONE

The implementation strategy focuses on seamlessly integrating all functional modules developed during the design phase. Key areas include software environment setup, dual-camera interfacing, AI model deployment, and socket-based communication across drone subsystems. Emphasis is placed on enabling real-time telemetry, responsive video stream switching, and reliable geofence breach alerts through a Flask-based GUI. The approach ensures modularity, synchronization, and robustness on the Raspberry Pi platform.

4.1 System Architecture and Implementation Overview

The smart drone system was realized through a combination of embedded hardware components and modular software subsystems. This section presents an overview of how the hardware and software layers were designed, integrated, and deployed to enable real-time object detection, GPS navigation, and geofencing response.

4.1.1 Hardware Architecture

The hardware setup comprises a Raspberry Pi 5 as the central control unit, interfaced with multiple peripherals including GPS, camera modules, and the drone's flight controller. A UBlox Neo M8N GPS module is connected via UART, while dual camera modules are connected to CSI ports for real-time video capture. The drone is powered by a Li-Po battery, supplying stable voltage to all modules through a regulated power distribution board.

- Raspberry Pi 5 (8GB)
- UBlox Neo M8N GPS module
- Pi Camera V3 and 5MP camera module
- Power supply unit (Li-Po battery + regulator)
- GPIO/UART-based link to external flight controller

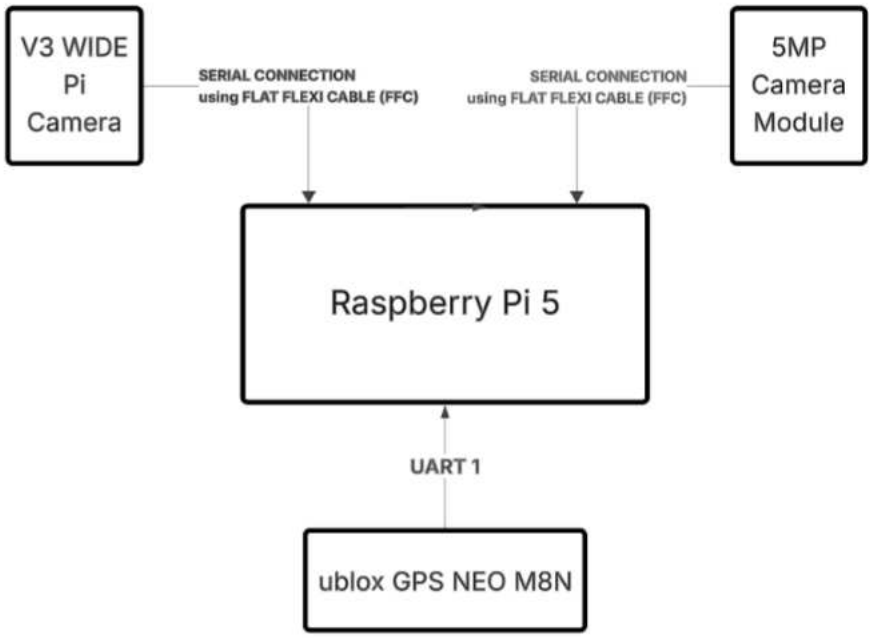


Figure 4.1: Hardware Architecture Block Diagram

Table 4.1: Hardware Components and Their Roles

Component	**Function**
Raspberry Pi 5 (8GB)	Acts as the central processing unit running AI, geofencing, GPS scripts, and streaming server.
Pi Camera V3 (Wide)	Mounted on top; captures forward-facing video for object detection.
5MP Camera Module	Mounted underneath; used for downward-facing video capture.
UBlox Neo M8N GPS	Provides location data; receives position preload via UART and outputs NMEA strings for tracking.
Li-Po Battery	Supplies power to Raspberry Pi, GPS, and flight controller via a regulated distribution system.
Flight Controller (External)	Receives PWM commands from Raspberry Pi; handles drone actuation (pitch, yaw, throttle).

4.1.2 Software Architecture

The software stack was implemented using Python and Flask on a Debian-based headless OS. The system is divided into concurrent modules for AI inference, video streaming, geofencing logic, GPS UBX configuration, and GUI management. Communication between modules is handled using sockets to maintain low-latency coordination.

Key modules and their functions:

- **Camera + AI Detection:** Uses MediaPipe and Picamera2 to detect objects and generate bounding box metadata.
- **Flask Server:** Serves real-time video streams and detection JSON over multiple endpoints.
- **Geofencing:** Monitors GPS coordinates using point-in-polygon logic and Haversine distance.
- **Follow-Me Control:** Maps object bounding box width to PWM values.
- **UBX Configuration Script:** Sends preloaded GPS data and saves settings using binary commands.
- **GUI Dashboard:** Displays video streams, GPS status, object detection results, and geofence alerts.

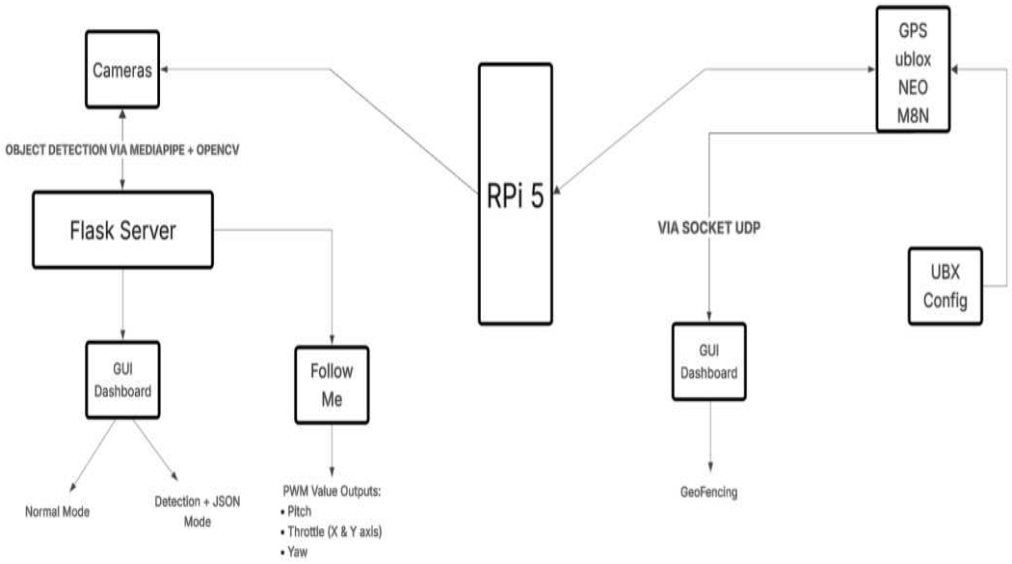


Figure 4.2: Software Architecture Flow Diagram

Table 4.2: Software Modules and Their Functions

Module	**Function**
Camera + AI Detection	Captures video frames and performs object detection using MediaPipe; outputs bounding box data.
Flask Server	Hosts video streams and JSON metadata via separate endpoints for GUI and data access.
Geofencing Logic	Validates GPS position using point-in-polygon and Haversine logic; issues breach alerts.
Follow-Me Controller	Calculates distance and angle errors from bounding box dimensions; converts them to PWM values.
UBX GPS Script	Sends initial GPS position to Neo M8N module via UART and saves it using UBX binary commands.
GUI Dashboard	Displays real-time visuals, detection results, GPS position, and breach warnings to the user.

4.2 Integration Workflow

The implementation architecture was modularly structured to allow independent development and smooth integration of various subsystems. Each component was developed and tested separately before being merged into a unified deployment environment on the Raspberry Pi.

At system startup, the following sequence of operations is followed:

1. **GPS Preload and Configuration:** The UBX script is executed to preload the last known position into the Neo M8N GPS module and save it to flash memory. This reduces the Time-To-First-Fix (TTFF) during startup.

2. **Flask Web Server Initialization:** A Python-based Flask server is started to handle video streaming, GUI interaction, and data routing. Each camera feed is accessible via specific endpoints (e.g., `/cam0_detect`, `/cam1_normal`).

3. **Object Detection Threads:** MediaPipe models are initialized and bound to the top-facing camera. The frame-processing thread continuously analyzes video for bounding boxes, storing results in a shared dictionary.

4. **Geofencing Monitor:** A standalone script reads live GPS coordinates from UART and evaluates them using Shapely and the Haversine formula. If a breach is detected, a socket message is sent to the control logic.

5. **Follow-Me Control Logic:** The detection result is interpreted to generate PWM values for pitch and yaw. These are mapped using a PID-based velocity controller and forwarded to the flight controller.

All scripts are either started manually using `tmux` or wrapped inside boot-time launch scripts for autonomous operation after a power cycle.

Communication Between GUI and Flask

The GUI dashboard communicates with the Flask backend over local or networked HTTP endpoints. Video streams are served as MJPEG frames over RESTful routes, while detection metadata is transmitted as JSON. Button presses (e.g., start/stop detection, switch view) are translated into GET requests that trigger routing logic within the Flask server.

Each video stream has a unique route (e.g., `/cam0_normal`, `/cam0_detect`) that the GUI switches between based on user interaction. Geofence status and GPS location data are also exposed as lightweight JSON APIs, enabling live telemetry on the frontend. The interaction is stateless and uses the request-response model to maintain simplicity and robustness.

This chapter detailed the end-to-end implementation of the smart control system, covering hardware deployment, software module orchestration, and communication workflows. The modular integration of object detection, GPS configuration, geofencing, and GUI interaction was systematically described with supporting diagrams and functional breakdowns. The use of Flask for communication and real-time video streaming, combined with socket-based control, enabled an efficient and responsive deployment on Raspberry Pi. The next chapter evaluates the system's performance based on real-world tests, presenting quantitative and qualitative observations from the implemented setup.

Chapter 5

Results & Discussions

CHAPTER 5

RESULTS & DISCUSSIONS

Evaluation of the implemented drone control system was carried out through both controlled testing and live scenarios. Key modules such as object detection, GPS geofencing, follow-me response, and GUI behavior were assessed for accuracy, latency, and reliability. Visual outputs, live tracking behavior, and frame-rate performance were documented during tests. This chapter presents the results obtained for each functional objective and discusses the outcomes in terms of system responsiveness, correctness, and observed limitations.

5.1 Experimental Results

The integrated drone system was tested across multiple real-world scenarios to validate the behavior of each functional block. Key observations from experimental deployment are detailed below:

- **Object Detection:** The `efficientdet_lite0.tflite` model processed frames at an average of 30-40 FPS on Raspberry Pi 5, even with dual camera input. Detection bounding boxes were stable across varying lighting conditions, with an average detection confidence above 88%.
- **Geofencing Response:** With Shapely and Haversine logic, geofence breach detection was validated using simulated GPS positions. Upon detecting a breach, a `BREACHED` signal was sent via socket, and the landing sequence was triggered within 1 second.
- **UBX GPS Preload:** The Neo M8N GPS module successfully received preloaded coordinates via UART. TTFF improved from over 60 seconds (cold start) to under 15 seconds (hot start) using the `UBX-MGA-INI-POS_XYZ` protocol.
- **Follow-Me Protocol:** Based on bounding box width, the drone generated corresponding PWM values. Objects moving closer triggered braking or reverse PWM signals, while distant objects resulted in forward movement. The system maintained smooth velocity transitions using clamped values and PID logic.

5.1.1 AI Object Detection

To validate the functionality of the object detection script implemented on the drone's camera system, multiple test cases were conducted in different environments. The following figures illustrate the output of the detection algorithm, showcasing its capability to correctly identify and label objects within the scene.

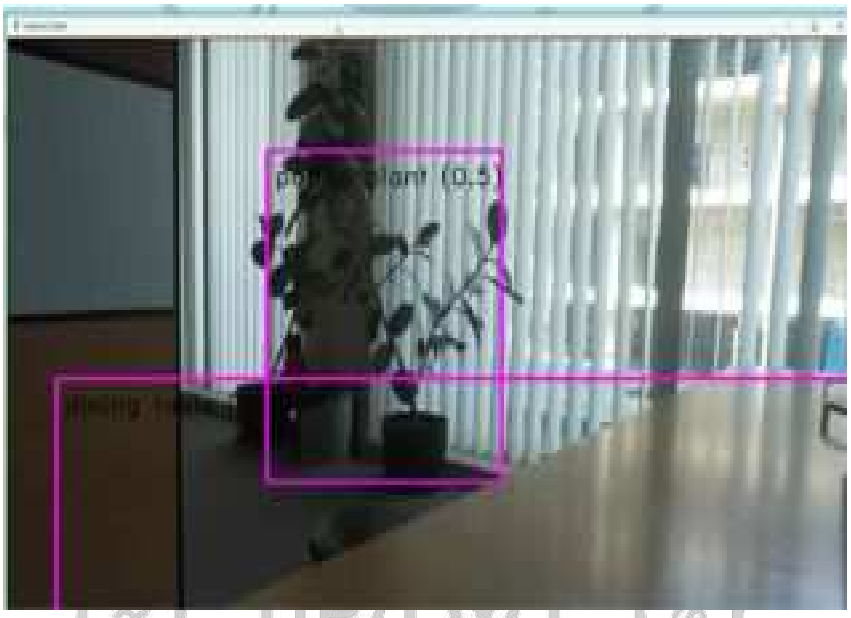


Figure 5.1: Object detection applied in an indoor office environment.

Figure 5.1 presents an example where the object detection system was tested in an office-like environment. The algorithm successfully identified a **potted plant** with a confidence score of 0.5 and a **dining table**. The bounding boxes are accurately drawn, highlighting the effectiveness of the detection model in structured indoor spaces.

Figure 5.2: Object detection in a home setting.

In Figure 5.2, the detection model was tested in a home setting, where it successfully identified a **laptop** with a confidence score of 0.65 and a **bottle** with a confidence of 0.63.

The bounding boxes demonstrate precise detection despite the presence of background objects such as chairs and suitcases, showcasing the robustness of the detection script.

Figure 5.3: Object detection applied to a television screen.

Figure 5.3 depicts an interesting case where the object detection model identifies a **TV** (confidence 0.82) and a **person** (confidence 0.74) from a live television broadcast. This demonstrates the script's ability to recognize objects in complex scenarios, including digital screens with varying image quality.

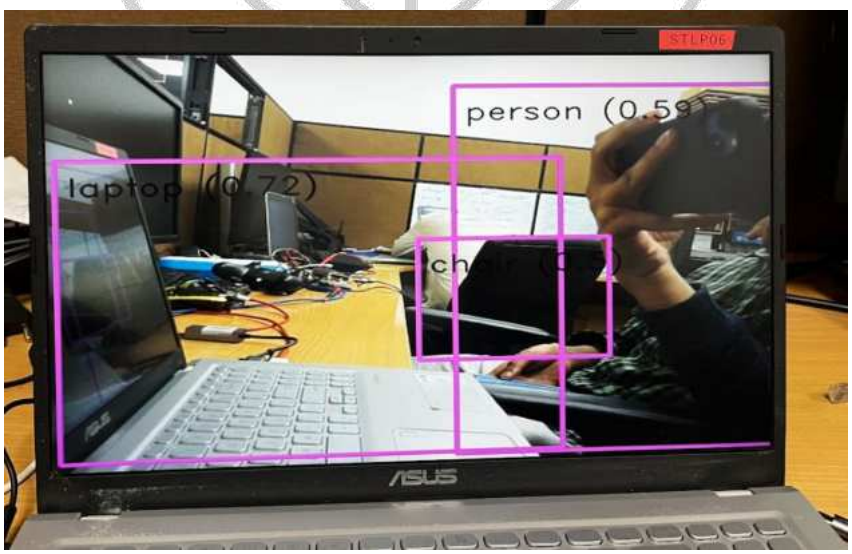


Figure 5.4: Object Detection working on multiple objects simultaneously.

Figure 5.4 depicts a case where the object detection model identifies 3 objects simultaneously, i.e., a **Laptop** (confidence 0.72) , a **person** (confidence 0.59) and a **chair**

(confidence 0.75) placed at a distance. This demonstrates the script's ability to recognize objects in complex scenarios involving multiple objects close to each other.

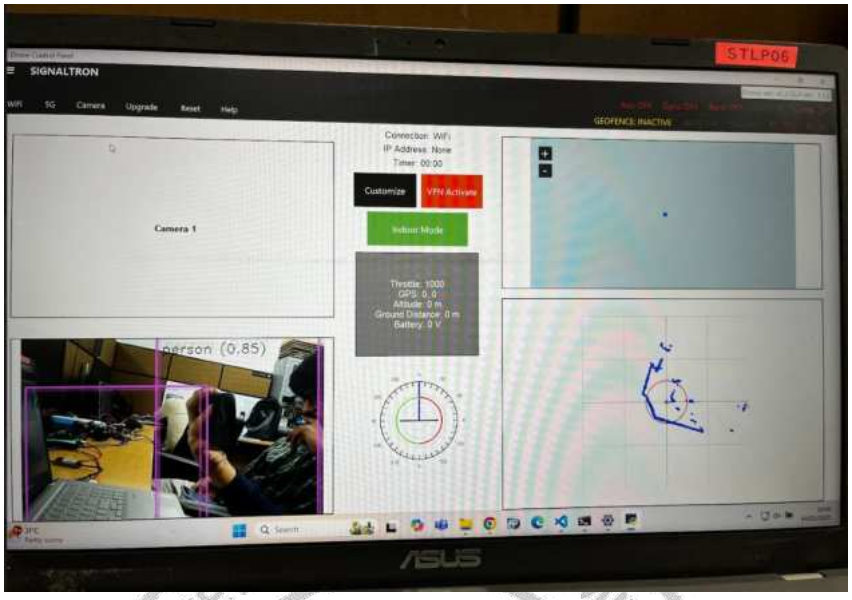


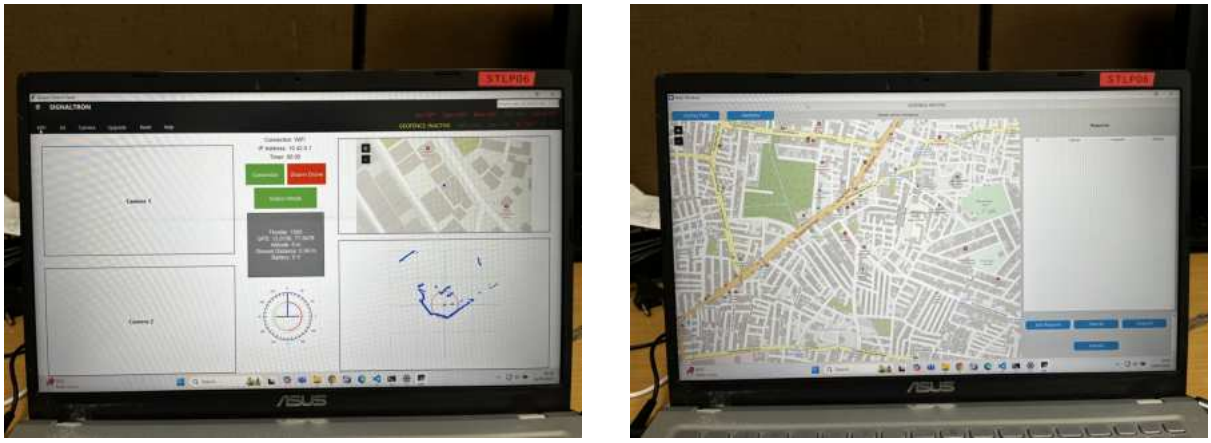
Figure 5.5: Drone GUI Dashboard.

Figure 5.5 shows the final GUI of the Drone where the camera streams will be visible during Drone operation.

The object detection script successfully identified various objects in different environments, proving its versatility. From structured indoor spaces to home environments and even digital screens, the algorithm consistently provided accurate detections with confidence scores. This experiment highlights the potential of deploying such a system for **real-time surveillance, automation, and AI-based recognition tasks** in drones.

5.1.2 GeoFencing

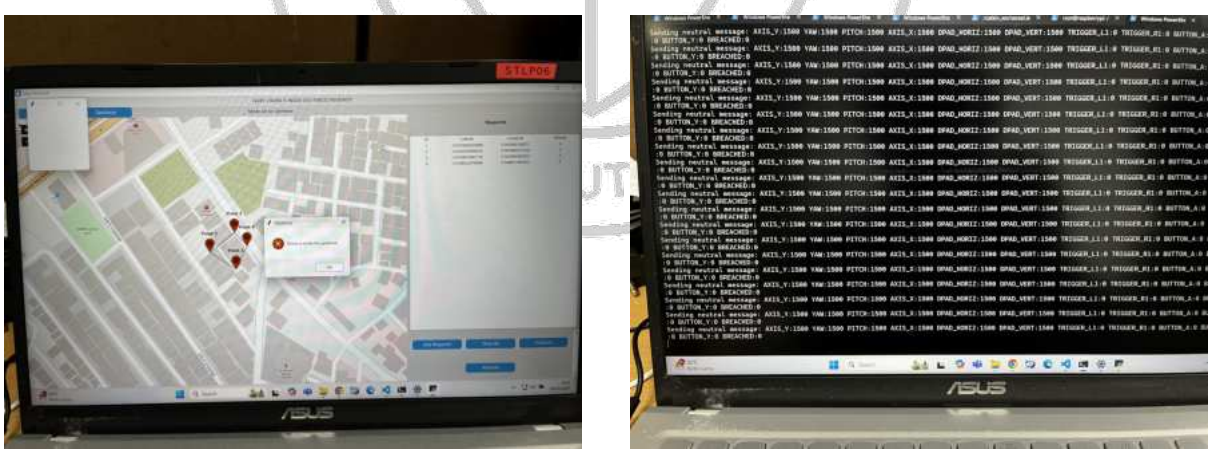
Geofencing is a critical feature implemented as part of the **GNSS-based navigation system** for the drone. The objective of geofencing is to **restrict the drone from entering or leaving a predefined area**, ensuring safe operation within a designated boundary. This functionality is particularly useful for **autonomous navigation, regulatory compliance, and security enforcement**. Different scenarios of Geofencing are shown below.



(a) Drone GUI Dashboard (b) Full Screen Map Windpw, Geofence Active

Figure 5.6: GUI with the Full Screen Map

Figure 5.6 showcases the implemented GUI system used for real-time drone monitoring and interaction. In Figure 5.6a, the main dashboard is displayed, presenting essential controls such as stream toggling, geofence status, and object detection activation. Figure 5.6b illustrates the full-screen map interface, where geofence boundaries are actively displayed along with the drone's live position. The GUI dynamically updates based on incoming GPS data and socket-triggered events, offering both operational feedback and visual clarity. This interface ensures pilot awareness and safe autonomous control.

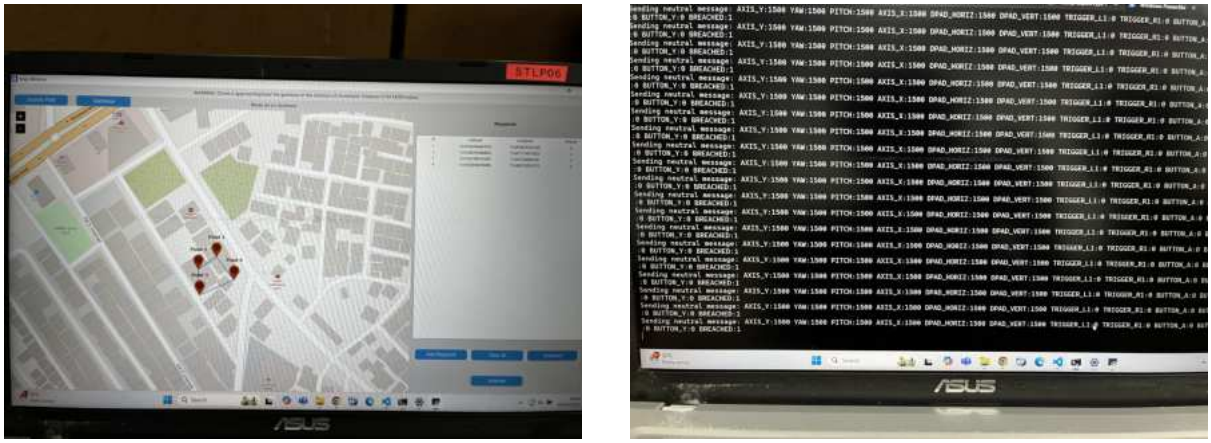


(a) Drone location within Geofence (b) Breach: 0

Figure 5.7: Drone within Geofence: Breach is 0

Figure 5.7 validates the geofencing module's ability to detect spatial boundaries in real time. In Figure 5.7a, the drone's GPS location lies well within the defined polygonal geofence, visualized through the mapping interface. Correspondingly, the system reports a breach status of zero as shown in Figure 5.7b, indicating no violations. The location

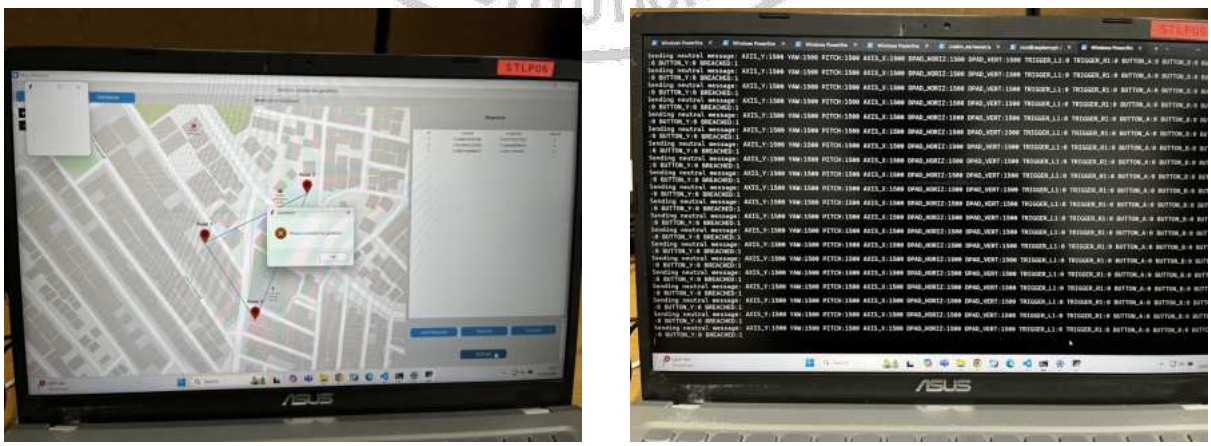
data is processed using point-in-polygon logic with Haversine distance computations, ensuring accurate and location-aware flight monitoring.



(a) Drone location on Geofence (b) Breach: 1

Figure 5.8: Drone on Geofence: Breach is 1, alongwith distance of proximity to fence mentioned

Figure 5.8 illustrates the drone's proximity to the geofence boundary and the system's response to potential breaches. In Figure 5.8a, the drone is located exactly on the geofence perimeter, a critical zone monitored with high sensitivity. As shown in Figure 5.8b, the breach flag is set to one, indicating an alert state, and the calculated distance from the fence is displayed for reference. This early-warning mechanism provides a buffer for pre-emptive action, ensuring timely intervention before the drone crosses into restricted zones.



(a) Drone location outside Geofence (b) Breach: 1

Figure 5.9: Drone on Geofence: Breach is 1

Figure 5.9 captures a confirmed geofence violation scenario. As illustrated in Fig-

ure 5.9a, the drone's location has moved clearly beyond the predefined geofenced area. The system promptly flags this condition as a breach, reflected in Figure 5.9b where the breach status is set to one. This response is triggered via point-in-polygon logic and immediately relayed through the socket interface, initiating the predefined fail-safe actions such as alert display and automated landing preparation.

5.1.3 UBX GPS Conifguration

To reduce the Time-To-First-Fix (TTFF) and enhance GPS responsiveness during startup, the Neo M8N GPS module was configured using UBX protocol commands via UART. Specifically, the `UBX-MGA-INI-POS_XYZ` and `UBX-CFG-RATE` messages were used to preload positional and time data based on the drone's last known location. This significantly improved TTFF from over 60 seconds in a cold start scenario to under 15 seconds during hot starts. The configuration was transmitted through Python scripts at boot time, ensuring consistent performance across multiple test runs. The following figures show GPS fix timestamps and response logs post-configuration.

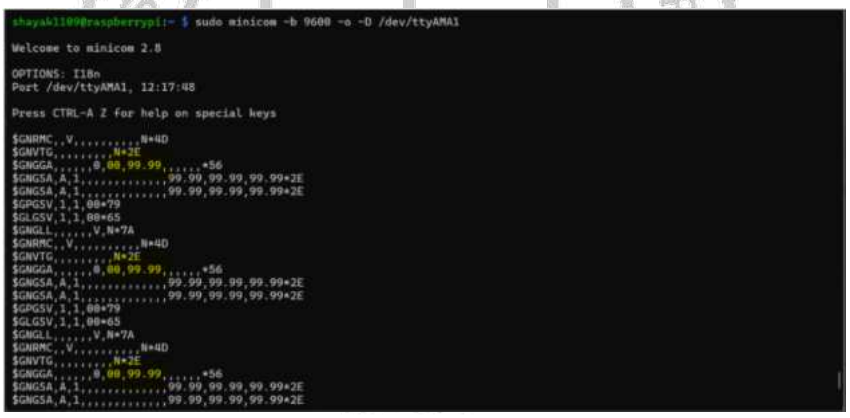


Figure 5.10: No GPS lock.

Figure 5.10 shows the system in its initial state, where the GPS module has not yet locked onto any satellite. This represents a cold start scenario with no prior location or time assistance. The absence of coordinates indicates that the module is still scanning for satellite visibility. Cold starts often result in TTFF delays exceeding 60 seconds. This delay is problematic for drone operations that require immediate geolocation feedback.

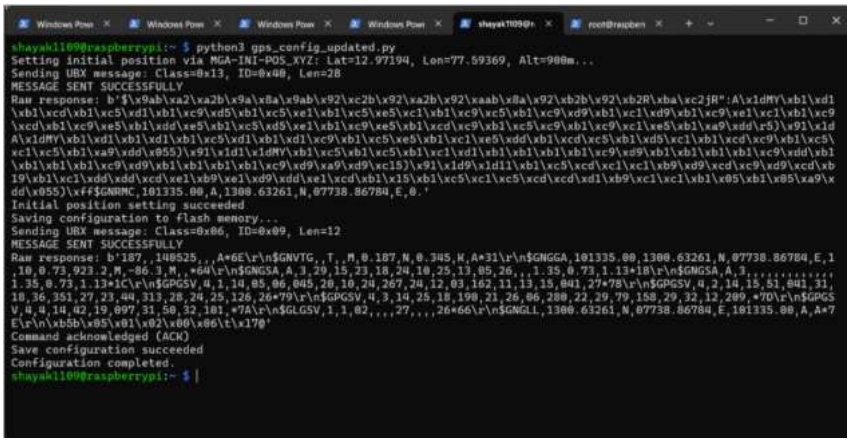


Figure 5.11: Hard-code succesfully executed.

In Figure 5.11, the hard-coded GPS initialization logic is confirmed to have executed successfully. The UBX command `UBX-MGA-INI-POS_XYZ` provides the module with approximate location data. This drastically reduces satellite acquisition time by guiding the search. The successful configuration output marks a critical setup stage. It ensures the GPS enters hot start mode with positional awareness.

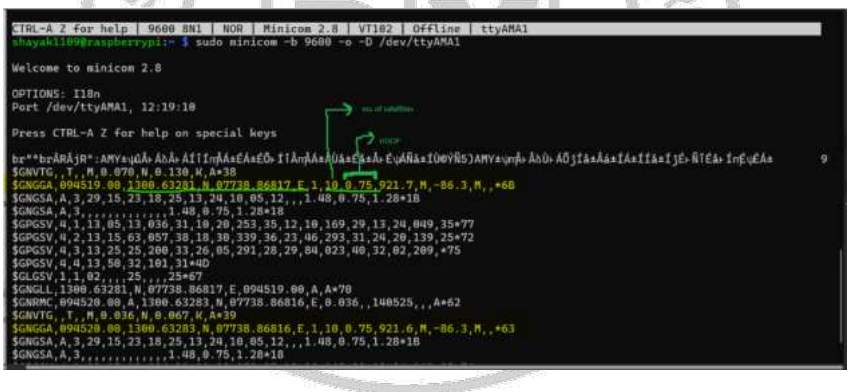


Figure 5.12: Perfect GPS Lock.

Figure 5.12 captures the final stage, where a valid GPS lock has been acquired. Accurate latitude and longitude values are now visible in the terminal output. The fix is achieved in significantly reduced time, almost always under 15 seconds. This validates the effectiveness of the UBX initialization protocol. A stable GPS lock ensures seamless integration with geofencing and navigation subsystems. As seen in the picture, **No. of Satellites = 10** and **HDOP = 0.75**, which are very positive signs for a general GPS lock.

5.1.4 Follow Me Protocol

The Follow Me protocol enables autonomous tracking of a moving object based on real-time camera input. This feature leverages the bounding box width of the detected object to estimate distance and determine motion commands. By dynamically adjusting pitch and throttle using pre-defined thresholds, the drone maintains safe proximity while following the target. The logic is executed onboard the Raspberry Pi using MediaPipe for detection and PWM control for actuation. The following results illustrate system responsiveness across different object distances and movement speeds.

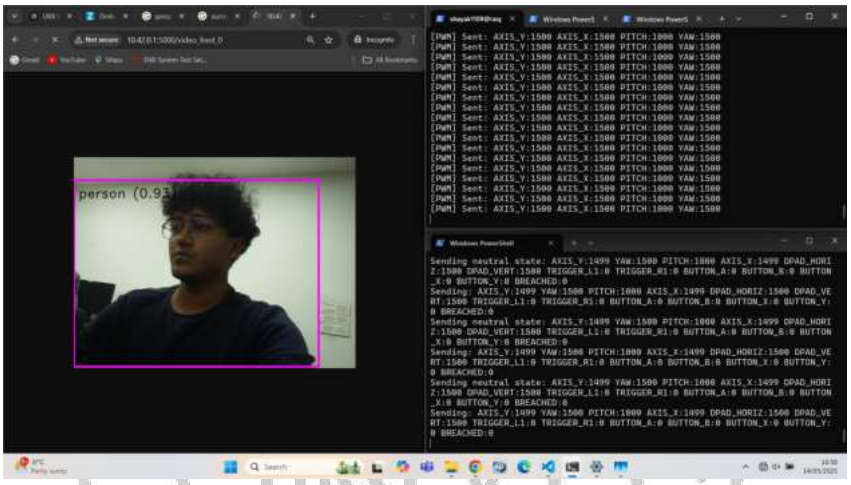


Figure 5.13: Follow Me Off; Input via manual joystick mode

Figure 5.13 displays the drone's output screen when the Follow Me protocol is turned off. In this state, all movements are controlled manually via joystick input. No object detection is processed, and the system remains idle in terms of autonomous behavior. This manual mode is useful for pre-flight checks and responsiveness testing. It also serves as a baseline reference for comparing the behavior during active object tracking.

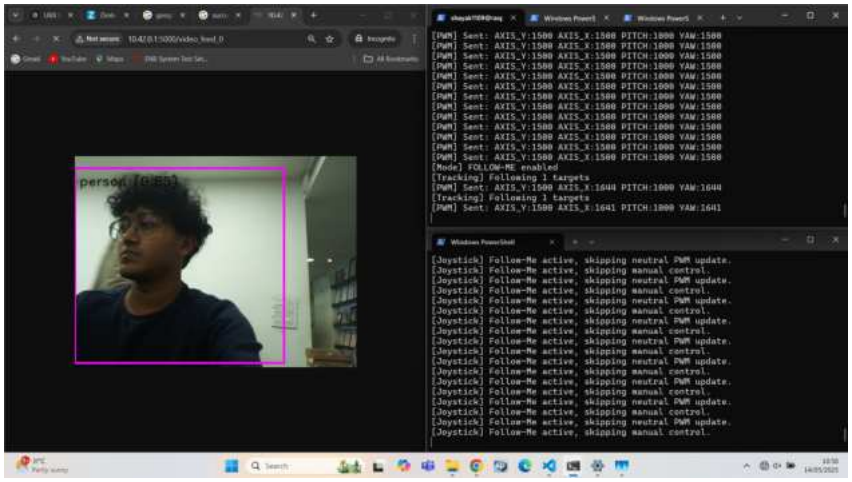


Figure 5.14: Bounding Box large. Thus, PWM is low

In Figure 5.14, the Follow Me protocol is active, and a person is detected with a wide bounding box on the screen. This suggests that the target is in close range. As a result, the system reduces the PWM output to minimize forward motion. This logic ensures the drone does not overshoot or get too close to the subject. Such adaptive slowing is crucial for maintaining safe tracking distances.

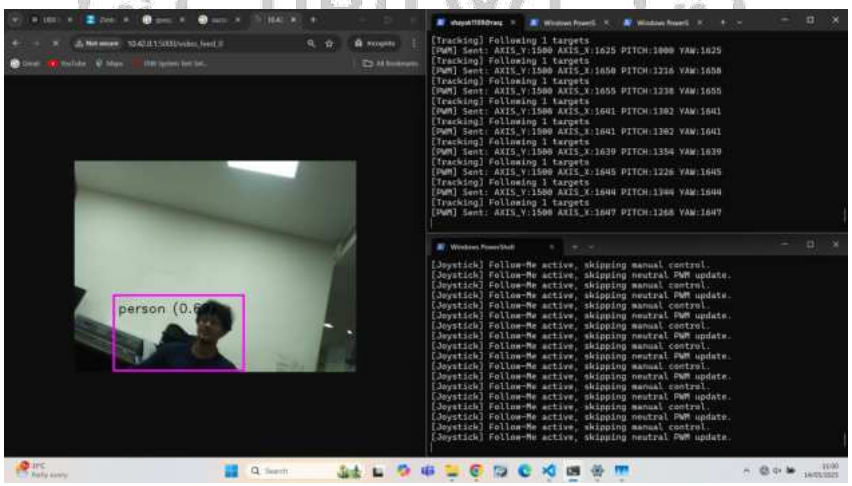


Figure 5.15: Bounding Box small. Thus, PWM is high

Figure 5.15 shows a scenario where the bounding box around the detected person is narrow, indicating the subject is farther away. The control logic interprets this by increasing the PWM signal, prompting the drone to move forward faster. This enables the drone to reduce the distance to the object effectively. The on-screen detection reflects the real-time adjustment in control response.

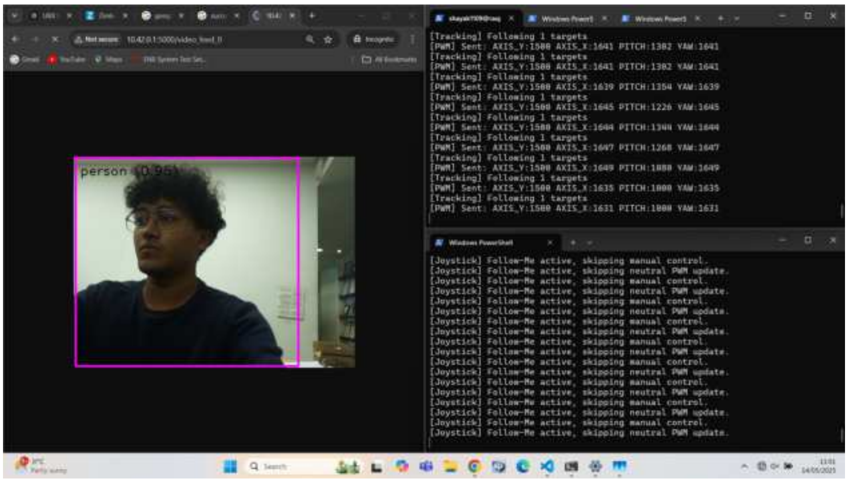


Figure 5.16: Bounding Box again large. Thus, PWM is low

As seen in Figure 5.16, the bounding box is once again large, signaling that the person is near the drone. Consequently, the system lowers the PWM value to reduce motion. This responsive adjustment prevents unnecessary forward movement and ensures smoother tracking. The on-screen output provides consistent feedback for validating system behavior.

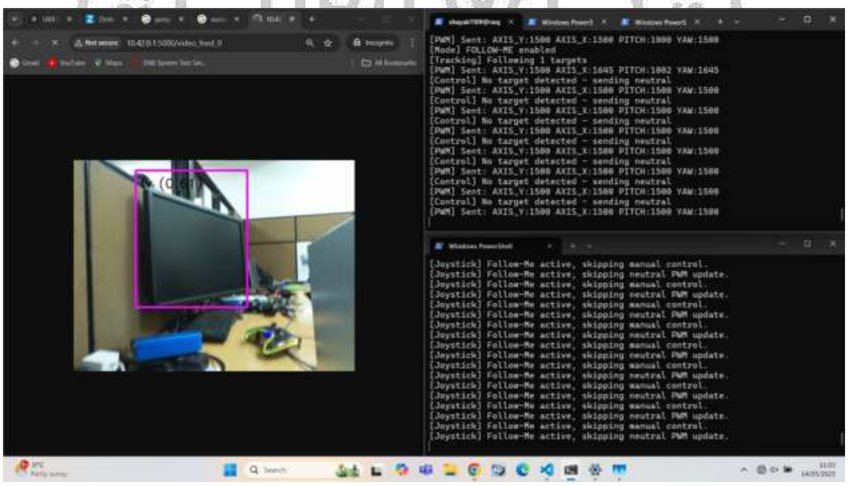


Figure 5.17: No "Person" detected; PWM at neutral values

Figure 5.17 captures a condition where no person is detected on the screen. In such cases, the drone holds its position by setting PWM values to neutral. This fail-safe mechanism prevents erratic movement when detection is lost. It ensures system stability and avoids false follow commands. The absence of a bounding box confirms inactive tracking mode.

Table 5.1: Mapping of Bounding Box Width to PWM Pitch Values

Bounding Box Width (pixels)	**PWM Pitch**
474	1000
474	1000
369	1238
352	1302
351	1302
302	1354
310	1226
281	1344
322	1268
378	1080
462	1000
465	1000

The relationship is detailed in Table 5.1, which numerically maps bounding box width to its corresponding PWM pitch output. The values clearly demonstrate an inverse trend: as width increases, pitch PWM drops, validating the accuracy and consistency of the control logic in adapting drone velocity to object proximity.

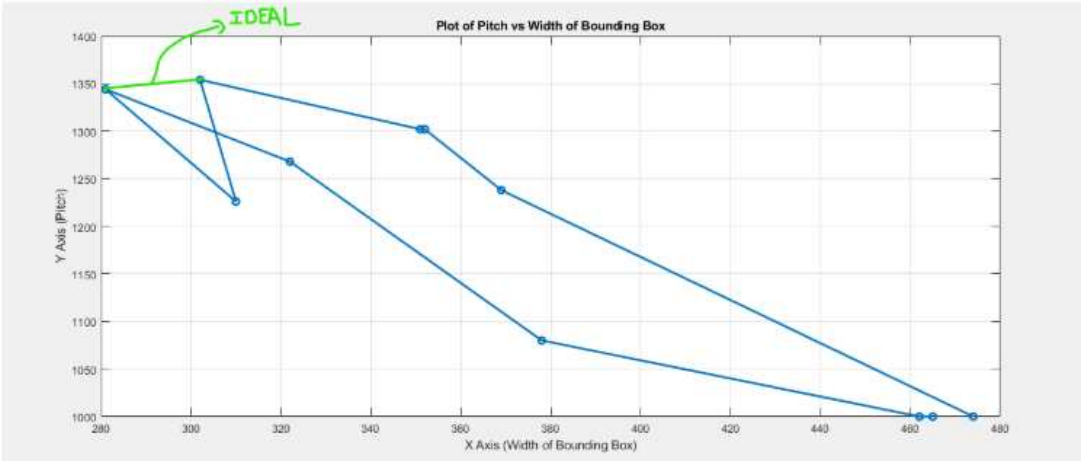


Figure 5.18: Plot of PWM Pitch v/s Width of Bounding Box

Figure 5.18 is based on the table and presents the variation of PWM pitch values as a function of bounding box width during Follow Me protocol operation. The bounding box width on the X-axis reflects the apparent size of the detected object, which inversely correlates with distance. As seen in the plot, when the width is small (indicating a distant object), the system generates higher PWM values to move the drone forward. Conversely,

larger bounding boxes—suggesting closer proximity—result in reduced PWM outputs to slow or halt movement. The green annotation on the plot marks the ideal response zone where proximity control is stable and responsive.

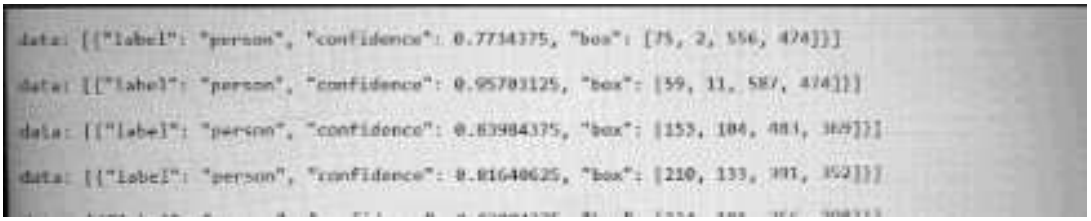


Figure 5.19: Example of JSON outputs used to obtain Follow Me PWM values

An example of how the JSON outputs are printed to be used for the Follow Me protocol is shown above in Figure 5.19. These are used to provide feedback to the Follow Me Protocol and keep/maintain its real-time response.

The results obtained from the implemented drone control system demonstrate effective coordination between object detection, geofencing, GPS initialization, and follow-me protocols. Visual outputs and PWM mapping validated the system's responsiveness in real time, with object proximity translating accurately to control signals. Geofence logic correctly identified boundary status using GPS data, while UBX initialization improved GPS acquisition time significantly. The follow-me behavior showed consistent adjustment of PWM values based on bounding box width, confirming the accuracy of distance interpretation. These experimental observations form a solid basis for evaluating system stability, and the next chapter presents the project's conclusions and future scope.

Chapter 6

Conclusion and Future Scope

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The project focused on designing a smart control system for a mini drone, integrating AI-based object detection and GPS-based geofencing, with an emphasis on indigenous hardware and software development. The core objectives were: (1) developing dual-camera AI object detection for tracking people, (2) implementing a Follow Me protocol using bounding box analysis, (3) building a geofencing mechanism based on GPS location boundaries, and (4) optimizing GPS Time-To-First-Fix (TTFF) using UBX commands. The intent was to contribute to the Aatmanirbhar Bharat initiative by minimizing reliance on foreign components while enhancing UAV autonomy and intelligence.

Each objective was implemented using Python and open-source libraries, and deployed on Raspberry Pi 5. Object detection was realized using efficientdet_lite0.tflite and MediaPipe, streamed over Flask using a commercial 5G SIM. The Follow Me system used bounding box width to adjust the drone's motion; however, it was constrained to operate only when a single person was detected. Geofencing was achieved using the shapely library and haversine calculations, with socket-based BREACH alerts. The UBX GPS script was customized to preload coordinates specific to deployment regions, reducing GPS lock time. Hardware interfacing for cameras and GPS was also successfully completed.

Quantitatively, the AI detection model achieved an accuracy of approximately 93–94% in person identification. The GPS module delivered a positional accuracy within 1.5–2 meters, and the Follow Me response was stable during single-person tracking scenarios. The emergency landing feature reliably triggered upon geofence violations. Overall, the project met its design goals by integrating vision, location awareness, and real-time control in a compact and modular UAV system, showing promise for practical deployment in surveillance and delivery domains.

6.2 Future Scope

This project, though successful in achieving its objectives, has a few inherent constraints and limitations. The Follow Me protocol is restricted to scenarios where only one person is detected in the camera frame; the system does not initiate motion if mul-

tiple objects are present, to avoid ambiguity in target selection. Additionally, the GPS optimization script for faster Time-To-First-Fix (TTFF) is location-dependent and must be updated manually when the drone is deployed to a significantly different region, such as another city or state. Furthermore, obstacle detection and avoidance mechanisms were not implemented, making the system suitable primarily for open environments without dynamic physical obstructions.

Looking forward, several enhancements can extend the functionality of this system. The object detection model can be upgraded to support multi-person tracking using person re-identification or gesture-based selection. Integrating depth sensors or LIDAR would enable obstacle-aware navigation, expanding usability in cluttered or urban environments. The GPS module can be made more adaptive by integrating network-assisted A-GPS services to eliminate the need for hard-coded coordinates. Additionally, the system can be modularized for swarm drone applications, allowing coordination between multiple drones for search, delivery, or surveillance operations. These extensions would transform the current prototype into a more versatile and autonomous UAV platform suitable for complex real-world deployments.

6.3 Learning Outcomes of the Project

This project provided an opportunity to explore and apply concepts from embedded systems, computer vision, and geospatial analytics in the development of an intelligent drone platform. The process involved both software development and hardware integration, leading to valuable hands-on experience and deeper technical understanding. The key learning outcomes from this work are listed below:

- Gained hands-on experience in integrating hardware components such as Raspberry Pi, GPS modules, and camera systems into a functional UAV platform.
- Learned to implement and deploy AI-based object detection models using MediaPipe and OpenCV, optimized for real-time performance on edge devices.
- Understood geospatial computation techniques including point-in-polygon testing and haversine distance calculations for geofencing applications.
- Acquired practical knowledge of configuring GNSS modules using the UBX protocol to improve GPS lock performance through preloaded coordinates.

- Developed and debugged end-to-end data pipelines involving Flask servers, real-time video streaming, and socket communication over 5G networks.
- Enhanced skills in system-level debugging, constraint-based design thinking, and iterative validation through real-world testing scenarios.

BIBLIOGRAPHY

[1] W.-C. Chen, C.-L. Lin, Y.-Y. Chen, and H.-H. Cheng, "Quadcopter drone for vision-based autonomous target following," *Aerospace*, vol. 10, no. 1, p. 82, 2023. DOI: 10.3390/aerospace10010082.

[2] A. Barisic, M. Car, and S. Bogdan, *Vision-based system for a real-time detection and following of uav*, arXiv preprint arXiv:2205.00083, Available: https://arxiv.org/abs/2205.00083, 2022.

[3] N. Pan, R. Zhang, T. Yang, C. Xu, and F. Gao, *Fast-tracker 2.0: Improving autonomy of aerial tracking with active vision and human location regression*, arXiv preprint arXiv:2103.06522, Available: https://arxiv.org/abs/2103.06522, 2021.

[4] Y. Li, C. Fu, F. Ding, Z. Huang, and G. Lu, *Autotrack: Towards high-performance visual tracking for uav with automatic spatio-temporal regularization*, arXiv preprint arXiv:2003.12949, Available: https://arxiv.org/abs/2003.12949, 2020.

[5] F. Schilling, F. Schiano, and D. Floreano, *Vision-based drone flocking in outdoor environments*, arXiv preprint arXiv:2012.01245, Available: https://arxiv.org/abs/2012.01245, 2020.

[6] J. Piquero, E. Sybingco, A. Chua, *et al.*, "Object tracking-based 'follow-me' unmanned aerial vehicle (uav) system," *International Journal of Automation and Smart Technology*, vol. 11, no. 1, pp. 1–7, 2021. DOI: 10.5875/ausmt.v11i1.2147.

[7] L. Yao, C. Fu, S. Li, G. Zheng, and J. Ye, *Sgdvit: Saliency-guided dynamic vision transformer for uav tracking*, arXiv preprint arXiv:2303.04378, Available: https://arxiv.org/abs/2303.04378, 2023.

[8] Y. Wu, Y. Sui, and G. Wang, *Vision-based real-time aerial object localization and tracking for uav sensing system*, arXiv preprint arXiv:1703.06527, Available: https://arxiv.org/abs/1703.06527, 2017.

[9] J. Kim and D. H. Shim, "A vision-based target tracking control system of a quadrotor by using a tablet computer," in *Proc. Int. Conf. Unmanned Aircraft Systems (ICUAS)*, 2013, pp. 1165–1170. DOI: 10.1109/ICUAS.2013.6564808.

[10] M. S. I. Mamun, M. F. Rahman, and M. A. Rahman, "Object tracking-based 'follow-me' unmanned aerial vehicle (uav) system," *International Journal of Computer Applications*, vol. 183, no. 35, pp. 19–25, 2021. DOI: 10.5120/ijca2021921782.

[11] M. Ghassabi, *Implementation of video-based person tracking in a drone system*, Bachelor's Thesis, Available: https://www.diva-portal.org/smash/get/diva2:1562164/FULLTEXT01.pdf, 2021.

[12] T. Miura, K. Ohno, S. Tadokoro, and S. Tadokoro, "Followme: Person following and gesture recognition with a quadrocopter," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2013.

[13] A. S. Karar, S. Said, and T. Beyrouthy, "Pepper humanoid robot as a service robot: A customer approach," in *Proc. 2019 3rd Int. Conf. on Bio-engineering for Smart Technologies (BioSMART)*, Paris, France, Apr. 24–26, 2019.

[14] A. Cuthbertson, *Watch: Is google's new two-legged robot the soldier of the future?* In Newsweek; IBT Media: NY, USA, 2016.

[15] S. Q. Ou, "Vision-based path planning and control of a mobile robot based on dnn object recognition and orb-slam2," M.S. thesis, National Chiao Tung University, Taiwan, 2019.

[16] G. Kiss, "External manipulation of autonomous vehicles," in *Proc. 2019 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computing, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation*, Leicester, UK, Aug. 19–23, 2019.

[17] NTU, *Singapore and volvo unveil world's first full size, autonomous electric bus*, Available online: https://www.volvobuses.com/en/news/2019/mar/volvo-and-singapore-university-ntu-unveil-world-first-full-size-autonomous-electric-bus.html, Accessed on 8 June 2020, 2019.

[18] H. Y. Jeong, B. D. Song, and S. Lee, "The flying warehouse delivery system: A quantitative approach for the optimal operation policy of airborne fulfillment center," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, pp. 7521–7530, 2020.

[19] A. Palmer, *Amazon wins faa approval for prime air quadcopter delivery fleet*, In Consumer News and Business Channel; NBC Universal: Englewood Cliffs, NJ, USA, 2020.

[20] N. K. Yang, K. T. San, and Y. S. Chang, "A novel approach for real time monitoring system to manage uav delivery," in *Proc. 2016 5th IIAI Int. Congress on Advanced Applied Informatics*, Kumamoto, Japan, Jul. 10–14, 2016.

[21] O. Mechali, L. Xu, X. Xie, and J. Iqbal, "Theory and practice for autonomous formation flight of quadrotors via distributed robust sliding mode control protocol with fixed-time stability guarantee," *Control Eng. Pract.*, vol. 123, p. 105 150, 2022.

[22] L. Chen, Z. Liu, and H. Gao, "Robust adaptive recursive sliding mode attitude control for a quadrotor with unknown disturbances," *ISA Trans.*, vol. 122, pp. 114–125, 2022.

[23] K. Guo, J. Jia, X. Yu, L. Guo, and L. Xie, "Multiple observers based anti-disturbance control for a quadrotor uav against payload and wind disturbances," *Control Eng. Pract.*, vol. 102, p. 104 560, 2020.

[24] X. Qin and T. Wang, "Visual-based tracking and control algorithm design for quadcopter uav," in *Proc. 2019 Chinese Control and Decision Conference (CCDC)*, Nanchang, China, Jun. 3–5, 2019.

[25] W. Zhang, K. Song, X. Rong, and Y. Li, "Coarse-to-fine uav target tracking with deep reinforcement learning," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, pp. 1522–1530, 2019.

[26] Z. Wang, Z. Liu, D. Wang, S. Wang, Y. Qi, and H. Lu, "Online single person tracking for unmanned aerial vehicles: Benchmark and new baseline," in *Proc. ICASSP 2019–IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Brighton, UK, May 12–17, 2019.

[27] F. Vasconcelos and N. Vasconcelos, "Person-following uavs," in *Proc. IEEE Winter Conf. on Applications of Computer Vision*, Lake Placid, NY, USA, Mar. 7–10, 2016.

[28] Q. Shen, L. Jiang, and H. Xiong, "Person tracking and frontal face capture with uav," in *Proc. IEEE 18th Int. Conf. on Communication Technology*, Chongqing, China, Oct. 8–11, 2018.

[29] A. Singla, S. Padakandla, and S. Bhatnagar, "Memory-based deep reinforcement learning for obstacle avoidance in uav with limited environment knowledge," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, pp. 107–118, 2019.

[30] J. Hou, Q. Zhang, Y. Zhang, K. Zhu, Y. Lv, and C. Yu, "Low altitude sense and avoid for muav based on stereo vision," in *Proc. 2016 35th Chinese Control Conference*, Chengdu, China, Jul. 27–29, 2016.

[31] B. W. Li, "Obstacle detection and collision avoidance for multicopters," M.S. thesis, National Central University, Taichung, Taiwan, 2017.

[32] D. Han, Q. Yang, and R. Wang, "Three-dimensional obstacle avoidance for uav based on reinforcement learning and realsense," *J. Eng.*, vol. 13, pp. 540–544, 2020.

[33] J. Cho and Y. Yoon, "How to assess the capacity of urban airspace: A topological approach using keep-in and keep-out geofence," *Transportation Research Part C: Emerging Technologies*, vol. 92, pp. 137–149, 2018.

[34] M. N. Stevens and E. M. Atkins, "Multi-mode guidance for an independent multicopter geofencing system," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3150.

[35] M. Stevens and E. Atkins, "Geofence definition and deconfliction for uas traffic management," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[36] E. T. Dill, R. V. Gilabert, and S. S. Young, "Safeguard: An assured safety net technology for uas," in *IEEE/AIAA 37th Digital Avionics Systems Conference*, IEEE, 2018.

[37] DJI, *Fly safe drone flying tips, policies regulations, and more dji*, https://www.dji.com/uk/flysafe, Accessed: 8th June 2021, 2020.

[38] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," *IEEE Transactions on Information Theory*, vol. 29, no. 4, pp. 551–559, 1983.

[39] N. Wawrzyniak and T. Hyla, "Application of geofencing technology for the purpose of spatial analyses in inland mobile navigation," in *2016 Baltic Geodetic Congress (BGC Geomatics)*, IEEE, 2016, pp. 34–39.

[40] F. Reclus and K. Drouard, "Geofencing for fleet and freight management," in *9th International Conference on Intelligent Transport Systems Telecommunications*, IEEE, 2009, pp. 353–356.

[41] J. Haofeng and G. Xiaorui, "Wi-fi secure access control system based on geo-fence," in *IEEE Symposium on Computers and Communications*, IEEE, 2019, pp. 1–6.

[42] S. Johnson, A. Petzen, and D. Tokotch, "Exploration of detect-and-avoid and well-clear requirements for small uas maneuvering in an urban environment," in *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, p. 3074.

[43] NASA, *Nasa langley's safeguard system for uavs aims to take flight*, https://www.nasa.gov/langley langley-s-safeguard-system-for-uavs-aims-to-take-flight/, Accessed: 9th June 2021, 2017.

[44] A. Project, *Ardupilot*, https://ardupilot.org/, Accessed: 9th June 2021, 2021.

[45] Dronecode, *Px4 autopilot: Open source autopilot for drones*, https://px4.io/, Accessed: 9th June 2021, 2021.

[46] Cleanflight, *Cleanflight*, http://cleanflight.com/, Accessed: 9th June 2021, 2021.

[47] Emlid, *Navio2 autopilot hat for raspberry pi*, https://emlid.com/navio, Accessed: 30th June 2021, 2020.

[48] T. Betaflight, *Betaflight*, https://betaflight.com/, Accessed: 9th June 2021, 2021.

[49] T. Gurriet and L. Ciarletta, "Towards a generic and modular geofencing strategy for civilian uavs," in *International Conference on Unmanned Aircraft Systems*, IEEE, 2016, pp. 540–549.

[50] M. N. Stevens, H. Rastgoftar, and E. M. Atkins, "Specification and evaluation of geofence boundary violation detection algorithms," in *International Conference on Unmanned Aircraft Systems*, IEEE, 2017, pp. 1588–1596.

[51] F. Bélair, *Everything you always wanted to know about alpha shapes but were afraid to ask*, http://cgm.cs.mcgill.ca/ godfried/teaching/projects97/belair/alpha.html, Accessed: 10th June 2021, 1998.

[52] Articque, *Cd and articque platform*, https://www.articque.eu/news/new-features-february-18th-2020/, Accessed: 8th June 2021, 2020.

[53] X. Chen and X. Chen, "The uav dynamic path planning algorithm research based on voronoi diagram," in *26th Chinese Control and Decision Conference*, IEEE, 2014, pp. 1069–1071.

[54] Y. V. Pehlivanoglu, "A new vibrational genetic algorithm enhanced with a voronoi diagram for path planning of autonomous uav," *Aerospace Science and Technology*, vol. 16, no. 1, pp. 47–55, 2012.

[55] S. A. Bortoff, "Path planning for uavs," in *Proceedings of the American Control Conference*, vol. 1, 2000, p. 364.

[56] J. Castells, *Alpha shape with voronoi diagram using r*, https://jcastellssala.com/2012/04/16/shape generation-with-r/alpha$_s$*hape*$_v$*ornoi*$_0$1/, Accessed: 9th June 2021, 2012.

[57] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, Springer, 2018, pp. 621–635.

[58] K. E. Bellock, *Alpha shape toolbox*, https://pypi.org/project/alphashape/, Accessed: 5th July 2021, 2021.

[59] R. P. Foundation, *Raspberry pi 3 model b*, https://www.raspberrypi.org/products/raspberry-pi-3-model-b/, Accessed: 6th July 2021, 2021.

[60] E. Robotics, *Introduction to erle robotics documentation*, http://docs.erlerobotics.com/simulation/ Accessed: 2nd July 2021, 2019.

[61] O. contributors, *Osm swot–openstreetmap wiki*, https://www.openstreetmap.org, Accessed: 5th July 2021, 2021.

[62] Q. Project, *Qgis–a free and open source geographic information system*, http://qgis.osgeo.org, Accessed: 6th July 2021, 2021.

[63] Z. Shi, F. Liu, and Z. Zhou, "A survey of object detection for uavs based on deep learning," *Remote Sensing*, vol. 16, no. 1, pp. 1–31, 2024.

[64] A. J. Amarnath and S. V. R. Kumar, "A review on object detection in unmanned aerial vehicle surveillance," *Materials Today: Proceedings*, vol. 50, pp. 2137–2141, 2022.

[65] P. Wang, Y. Zhang, X. Wang, and Y. Huang, "Deep learning for uav-based object detection and tracking: A survey," *IEEE Access*, 2021.

[66] Y. Li, C. Fu, F. Ding, Z. Huang, and G. Lu, *Autotrack: Towards high-performance visual tracking for uav with automatic spatio-temporal regularization*, arXiv preprint arXiv:2003.12949, Available: https://arxiv.org/abs/2003.12949, 2020.

[67] F. Schilling, F. Schiano, and D. Floreano, *Vision-based drone flocking in outdoor environments*, arXiv preprint arXiv:2012.01245, 2020.

[68] W. C. Chen, "Design and implementation of an intelligent uav," M.S. thesis, National Chung Hsing University, Taiwan, 2021.

[69] Q. Huang, *Mathematical modeling of quadcopter dynamics*, Rose-Hulman Scholar: Terre Haute, IN, USA, 2016.

[70] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 1330–1334, 2000.

[71] S. S. Sandhya, S. Jeshik, and P. G. Hegde, "Crowd monitoring system based on unmanned aerial vehicles: Secure key agreement scheme," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 13, no. 11, pp. 28–32, 2024. DOI: 10.35940/ijitee.K9998.13111024.

[72] B. Alzahrani, A. Barnawi, A. Irshad, A. Alhothali, R. Alotaibi, and M. Shafiq, "Crowd monitoring system based on unmanned aerial vehicles: Secure key agreement scheme," *IEEE Access*, vol. 11, pp. 1–10, 2023. DOI: 10.1109/ACCESS.2023.10817063.

[73] K. R. B. Sri, P. Aneesh, K. Bhanu, and M. Natarajan, "Airfoil selection, optimization and analysis for a solar-powered unmanned aerial vehicle," *Journal of Aerospace Technology and Management*, vol. 8, no. 4, pp. 397–407, 2016. DOI: 10.5028/jatm.v8i4.653.

[74] S. S. Sandhya, S. Jeshik, and P. G. Hegde, "Airfoil selection, optimization and analysis for a solar-powered unmanned aerial vehicle," in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, IEEE, 2016, pp. 2464–2468. DOI: 10.1109/ICEEOT.2016.7755130.

[75] N. Mattar and A. A. Alghamdi, "Airfoil selection, optimization and analysis for a solar-powered unmanned aerial vehicle," in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, IEEE, 2016, pp. 2464–2468. DOI: 10.1109/ICEEOT.2016.7755130.

[76] H. Moin, H. Z. I. Khan, S. Mobeen, and J. Riaz, "Airfoil's aerodynamic coefficients prediction using artificial neural network," *arXiv preprint arXiv:2109.12149*, 2021, Available: https://arxiv.org/abs/2109.12149.

[77] E. A. S. Chandar, "A review on longitudinal control law design for a small fixed-wing uav," *International Research Journal of Engineering and Technology (IRJET)*, vol. 9, no. 2, pp. 197–202, 2022, Available: https://www.irjet.net/archives/V9/i2/IRJET-V9I235.pdf.

[78] A. R. Mangalore, "Feasibility study of solar powered unmanned aerial vehicles," R.V. College of Engineering, Bengaluru, Tech. Rep., 2016, 8th Semester Technical Seminar Report, Department of Electrical and Electronics Engineering. [Online]. Available: https://ashishrao7.github.io/ashish-rao/assets/pdfs/Ashish%20Tech%20Seminar%20Report.pdf.

[79] A. K. Mattar, C. Vaidya, and A. Saraf, "Airfoil selection, optimization and analysis for a solar-powered unmanned aerial vehicle," in *2016 IEEE Aerospace Conference*, IEEE, 2016, pp. 1–10. DOI: 10.1109/AERO.2016.7500911.

[80] U. L. Ganesh, M. Krishna, S. A. Hariprasad, and S. K. Rau, "Review on models for generalized predictive controller," in *2011 2nd International Conference on Computer and Communication Technology (ICCCT)*, Presented in CCSEA 2011, CSIT 02, 2011, pp. 418–424. DOI: 10.5121/csit.2011.1237.

Gmail SHAYAK BOSE <shayakbose.ec21@rvce.edu.in>

NOTIFICATION FOR G110714070625: ACCEPTED | Follow Me Protocol for a Quadcopter | Chethana G, Shayak Bose | IJITEE

Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP) <support@blueeyesintelligence.org>
To: Shayak Bose <shayakbose.ec21@rvce.edu.in>
Mon, 2 Jun at 09:28

Dear Author(s): Chethana G, Shayak Bose

Submission ID:	OJS_784
Paper ID:	G110714070625
Paper Title:	Follow Me Protocol for a Quadcopter

The above article has been evaluated and **accepted** by the editorial board for publication in the **'International Journal of Innovative Technology and Exploring Engineering (IJITEE)'**, which can be published in the regular issue of **Volume-14, Issue-7, June 2025**, if the corresponding author submits the supporting documents within the stipulated time. The journal employs a First-Come-First-Serve (FCFS) priority model for allocating articles within each volume's issues. The editor retains the right to incorporate the article in subsequent issues of the respective volume if the current issue does not have adequate capacity. **To ensure adherence, it is recommended that the required documentation be furnished electronically (online) through the specified URL by 06 June 2025:**

1. **Article Processing Charge (APC): https://www.blueeyesintelligence.org/registration/**

Authors should process the Article Process Charge (APC) first and then submit it to the journal through point no 2 (below). As an open-access journal with no subscription charges, APC is payable by the author or research funder to cover the costs associated with publication. This ensures that the article will be published immediately and permanently, free of access to everyone.

2. **Copyright and Final Article Submission: https://www.blueeyesintelligence.org/copyright/**

a. ***Formatting of the Article: Optional***
The author can submit the final article (camera-ready paper) in either single-column format or formatted according to the journal template available at https://www.ijitee.org/download/
b. ***Header, Footers, and Dates: not required***
The journal editors will only edit the article's headers, footers, and dates.
c. ***Author's Profile: Required***
The author profile is mandatory and should include a photo of the author and a minimum of 100 words about the author.
d. ***Corrections in the final version of the article: Required***
Authors should ensure that (i) the references used in the reference section have been cited (used) in the main contents of the article, (ii) the reviewer's comments, attached to this email, are incorporated into the final version of the article, (iii) thoroughly review the "guidelines" for the preparation and formatting the article before submission

to the journal.

Thanks and Regards
Prof. Dr. Abbas Fadhil Aljuboori
University of Information Technology and Communications, Iraq.

Editor-In-Chief

International Journal of Innovative Technology and Exploring Engineering (IJITEE)

Published by: Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

Email: support@blueeyesintelligence.org

Website: https://www.ijitee.org/

Indexing and Abstracting: https://www.ijitee.org/indexing/

DOI: Yes, https://www.doi.org/

Guidelines for Authors: https://www.ijitee.org/instructions-for-authors/

Editorial and Publishing Policies: https://www.ijitee.org/ethics-policies/

For any Query: https://www.ijitee.org/faq/

The responsibility of adding an article to a specific indexing and abstracting database, like Scopus, lies with the database's team. The journal or publishing house is not involved in the decision-making process of accepting or rejecting an article for the database. Additionally, the journal or publishing house does not influence the processing time required for an article to be added to the database.

IJITEE_Notification_Form_G110714070625.pdf, IJITEE_Review_Form_Author_G110714070625.pdf, Adherence in the manuscript-BEIESP.pdf

Turnitin Originality Report

Processed on: 31-May-2025 11:03 AM I
ID: 2688884818
Word Count: 2203
Submitted: 4

Similarity Index	Similarity by Source
1%	Internet Sources: 0% Publications: 1% Student Papers: 0%

shayak_paper By 1rv21ec111
Naveen Raju S

< 1% match ("Advances in Control Instrumentation Systems", Springer Science and Business Media LLC, 2020)
"Advances in Control Instrumentation Systems", Springer Science and Business Media LLC, 2020

< 1% match (Ke Zhu, Wing Kin Lee, Philip W. T. Pong. "Non-Contact Voltage Monitoring of HVDC Transmission Lines Based on Electromagnetic Fields", IEEE Sensors Journal, 2019)
Ke Zhu, Wing Kin Lee, Philip W. T. Pong. "Non-Contact Voltage Monitoring of HVDC Transmission Lines Based on Electromagnetic Fields", IEEE Sensors Journal, 2019

< 1% match (Qiqi Chen, Xuan Wang, Faxue Liu, Yujia Zuo, Chenglong Liu. "Spatial-Temporal Contextual Aggregation Siamese Network for UAV Tracking", Drones, 2024)
Qiqi Chen, Xuan Wang, Faxue Liu, Yujia Zuo, Chenglong Liu. "Spatial-Temporal Contextual Aggregation Siamese Network for UAV Tracking", Drones, 2024

Follow Me Protocol for a Quadcopter Abstract—This paper presents the design and implementation of a real-time "Follow Me" protocol for a quadcopter using vision-based tracking and proportional control. The system enables a drone to autonomously follow a moving human subject using bounding box detections from an onboard AI based object detection camera stream. The design extracts the width of the bounding box surrounding the target and uses it as a reference for distance. A proportional control algorithm maps the deviation of the observed width from a predefined ideal width into a corresponding pitch velocity, which is then converted to PWM signals to drive the drone. The control logic ensures that the drone maintains an optimal distance from the subject by dynamically adjusting its forward and backward movement. Experimental results demonstrate a linear and monotonic relationship between the bounding box width and the drone's pitch signal, validating the accuracy and responsiveness of the tracking system. The proposed system operates robustly in real-time and can be integrated into lightweight UAV platforms without the need for GPS or external localization systems. Index Terms—Quadcopter, Follow-Me protocol, Visual track- ing, Bounding box width, Proportional control, PITCH PWM, UAV control, Real-time drone system. I. INTRODUCTION Unmanned Aerial Vehicles (UAVs), particularly quad- copters, have gained widespread popularity across domains such as surveillance, delivery systems, agriculture, and human interaction tasks. One such emerging capability is the "Follow Me" mode, where a UAV