

ACOUSTIC-BASED DRONE DETECTION: IDENTIFYING UAVS THROUGH SOUND SIGNATURES

Capstone Project Report

END SEMESTER EVALUATION

Submitted By:

- (102203012) Miet Pamecha
- (102203061) Gautam Dhawan
- (102203408) Lipsita Devgan
- (102203413) Tamanna Bajaj
- (102203449) Devansh Dhir

BE Fourth Year – COE

CPG No. 179

Under the Mentorship of

Dr. Sharad Saxena

Associate Professor



Computer Science and Engineering Department

Thapar Institute of Engineering and Technology, Patiala

December, 2025

ABSTRACT

Unmanned Aerial Vehicles (UAVs), better known as drones, have experienced explosive growth in both commercial and recreational use. However, their accessibility is posing serious threats to privacy, critical infrastructure, and public safety. Traditional detection methods, such as radar, are inefficient at detecting little plastic drones because of their low radar cross-section while visual systems struggle in low-light conditions. To help fill this gap, "Our Drone Detection Project", a cost-effective passive acoustic detection system for real-time monitoring of the airspace, is presented in this project.

The proposed solution is an edge computing device that is based on a Raspberry Pi 5 as the computing unit, coupled with ReSpeaker 4-Microphone Array as the assembly unit. Instead of using heavy computer resources, the system uses an efficient Machine Learning pipeline. It applies Mel-frequency cepstral coefficients (MFCCs), Delta features to extract the unique "velocity" of drone motor signatures and classifies them with a Random Forest algorithm. A key innovation of this project is the use of Automatic Gain Control (AGC) to improve detection range as well as Direction of Arrival (DOA) estimation to triangulate the drone's angle of approach.

To be usable for practical use, the system includes a "Stealth Mode" web dashboard hosted locally using Flask. This enables security personnel to visualize threats and track the target on a radar-like interface from any smartphone or laptop. Experimental results show that the prototype has high classification accuracy with a latency of less than 100 milliseconds, providing a scalable and robust solution for affordable drone defence.

DECLARATION

We hereby declare that the design principles and working prototype model of the project entitled “Acoustic-based drone detection: Identifying UAVS through sound signatures” is an authentic record of our own work carried out in the Computer Science and Engineering Department, TIET, Patiala, under the guidance of Dr. Sharad Saxena during 7th semester (2025).

Date: December 23, 2025

Roll No.	Name	Signature
102203012	Miet Pamecha	
102203061	Gautam Dhawan	
102203408	Iipsita Devgan	
102203413	Tamanna Bajaj	
102203449	Devansh Dhir	

Counter Signed By:

Faculty Mentor:

Signature:



Dr. Sharad Saxena

Associate Professor

CSED, TIET, Patiala

ACKNOWLEDGEMENT

We would like to express our thanks to our mentor Dr. Sharad Saxena. He has been of great help in our venture and an indispensable resource of technical knowledge. He is truly an amazing mentor to have.

We are also thankful to Dr. Neeraj Kumar, Head, Computer Science and Engineering Department, the staff of the Computer Science and Engineering Department, and also our friends who devoted their valuable time and helped us in all possible ways towards successful completion of this project. We thank all those who have contributed either directly or indirectly towards this project.

Lastly, we would also like to thank our families for their unyielding love and encouragement. They always wanted the best for us and we admire their determination and sacrifice.

Date: December 23, 2025

Roll No.	Name	Signature
102203012	Miet Pamecha	
102203061	Gautam Dhawan	
102203408	Iipsita Devgan	
102203413	Tamanna Bajaj	
102203449	Devansh Dhir	

TABLE OF CONTENTS

ABSTRACT.....	i
DECLARATION.....	ii
ACKNOWLEDGEMENT.....	iii
TABLE OF CONTENTS.....	iv
LIST OF TABLES.....	vii
LIST OF FIGURES	viii
1. INTRODUCTION.....	1
1.1 Project Overview	1
1.2 Need Analysis	4
1.3 Research Gaps.....	7
1.4 Problem Definition and Scope	9
1.5 Assumptions and Constraints.....	10
1.6 Standards.....	12
1.7 Approved Objectives	12
1.8 Methodology.....	13
1.9 Project Outcomes and Deliverables	14
1.10 Novelty of Work.....	15
2. Requirement Analysis.....	17
2.1 Literature Survey	17
2.1.1 Theory Associated with Problem Area	17
2.1.2 Existing Systems and Solutions	18
2.1.3 Research Findings for Existing Literature	18
2.1.4 Problem Identified	20
2.1.5 Survey of Tools and Technologies Used.....	21
2.1.6 Summary	21

2.2	Software Requirement Specification	22
2.2.1	Introduction.....	22
2.2.2	Overall Description	23
2.2.3	External Interface Requirements.....	24
2.2.4	Other Non-functional Requirements.....	26
2.3	Cost Analysis	28
2.4	Risk Analysis	29
3.	Methodology Adopted.....	31
3.1	Investigative Techniques.....	31
3.2	Proposed Solution	34
3.3	Work Breakdown Structure.....	36
3.4	Tools and Technology	39
4.	Design Specifications	42
4.1	System Architecture	42
4.2	Swimlane Diagram.....	43
4.3	Use Case Diagram.....	44
4.4	Snapshots of Working Prototype.....	45
5.	Implementation and Experimental Results	48
5.1	Experimental Setup	48
5.2	Experimental Analysis	49
5.2.1	Data	49
5.2.2	Performance Parameters	49
5.3	Working Project	50
5.3.1	Procedural Workflow	50
5.3.2	Algorithmic Approaches Used.....	50
5.3.3	Project Deployment	51
5.4	Testing Process.....	51

5.4.1	Test Plan.....	51
5.4.2	Features to be Tested.....	52
5.4.3	Test Strategy.....	53
5.4.4	Test Techniques	53
5.4.5	Test Cases.....	54
5.4.6	Test Results	55
5.5	Results and Discussions.....	56
5.6	Inferences Drawn.....	56
5.7	Validation of Objectives.....	57
6.	Conclusions and Future Scope.....	58
6.1	Work Accomplished.....	58
6.2	Conclusions.....	58
6.3	Environmental / Economic / Social Benefits	59
6.4	Future Work Plan	59
7.	Project Metrics	61
7.1	Challenges Faced	61
7.2	Relevant Subjects.....	61
7.3	Interdisciplinary Knowledge Sharing	62
7.4	Peer Assessment Matrix	62
7.5	Student Outcomes Description and Performance Indicators (A–K Mapping)	63
7.6	Brief Analytical Assessment.....	65
REFERENCES.....		66
PLAIGAIRSM REPORT.....		68

LIST OF TABLES

Table 1: Key Research Gaps and Proposed Directions.....	8
Table 2: Risk Analysis according to categories	29
Table 3: Phase 1 (Prototyping).....	39
Table 4: Phase 2 (Deployment).....	41
Table 5: Summary of System Test Cases and Expected Results.....	54
Table 6: Intra-Group Peer Assessment Scores	62
Table 7: Student Outcomes (A–K) and Project Performance Mapping.....	63

LIST OF FIGURES

Figure 1: Project schedule and activity timeline.....	38
Figure 2: High-level block diagram of Phase 1 & 2 of Drone Detection System	42
Figure 3: Swimlane Diagram of Drone Detection System	43
Figure 4: Use Case Diagram of Drone Detection System	44
Figure 5: Centralized Dashboard for Drone Detection	45
Figure 6: Drone detected and direction estimated using acoustic signals	46
Figure 7: Real time monitoring dashboard showing no drone presence detected	47

1. INTRODUCTION

1.1 Project Overview

Unmanned Aerial Vehicles (UAVs), popularly referred to as drones, are a game-changing technology with applications in many different areas. They are increasingly used in commercial applications like aerial photography, agriculture and logistics, recreational applications and critical security applications. While the sudden proliferation of drones has led to innovation and efficiency in various contexts, we have also opened new challenges. Unauthorized drone flights are dangerous to sensitive air spaces like airports and military bases and government facilities. In addition, drones can also be abused for surveillance, smuggling or even hostility. These dangers have spurred the urgent need for robust and scalable monitoring systems that can detect drones in real time in a variety of environmental conditions.

Traditional drone detection methods, such as radar and vision-based methods, have shown good performances in some cases and have limits. Radar systems need costly infrastructure and are not very good at detecting small, low altitude consumer drones because these plastic devices have a very small radar cross-section. Vision-based systems on the other hand are severely constrained by lighting and visibility conditions. They don't work in low light, fog or bad weather - all of which are often used for unauthorized flights. As a result, researchers and engineers have turned to acoustic detection as an alternative in recent years. Drones have some characteristic sound signatures, usually a high-frequency whine produced by their high-speed rotors and motors, that can be distinguished from other sounds in the environment. This acoustic approach offers many benefits: it is cost-effective, passive (non-intrusive) and can work regardless of weather or visibility.

This capstone project aims at developing an innovative drone detection system from the analysis of these sound signatures. The system takes advantage of the unique functionalities of a 4-microphone array to capture ambient audio, which is then followed by sophisticated digital signal processing techniques and machine learning algorithms to separate the drone noises from the other ambient sounds. This solution by detecting drones based on their unique acoustic footprint promises to offer an accessible

and reliable approach to drone monitoring in different environments. Unlike huge radar installations or extremely costly optical systems, an acoustic-based solution is lightweight and can be easily transported, making it an ideal choice for implementation in urban areas, large public gatherings, and in sensitive installations where it may be necessary to quickly deploy a system.

The methodology applied to this project is a departure from the heavy and more theory-heavy deep learning models and towards efficient, real-time edge computing. The system is based on the Raspberry Pi 5 platform, which was chosen because it's a good balance of processing power and portability. For the data collection and sensing, Seeed ReSpeaker 4-Microphone Array is used. This is an important choice of hardware because it goes beyond simply recording audio to capturing spatial audio. The data collection process consisted of creating a custom, real world dataset made up of the live recordings of commercial quadcopters in a variety of flight scenarios: hovering at close range (1-3 meters), flying at a distance (10+ meters), and performing high-speed flybys. Crucially, a separate "Noise" category was also captured which includes silence, human speech, and wind noise to make sure that the system can effectively ignore false alarms.

In terms of processing, the project uses a very efficient pipeline, which is suitable for edge devices. Instead of using computationally expensive deep neural networks such as CNN14 which requires powerful GPUs, this system uses Mel-Frequency Cepstral Coefficients (MFCCs) for feature extraction. Specifically, the system computes both the standard MFCCs and the Delta-MFCCs. The Delta features are especially important in that they record the rate of change in the audio signal and can therefore identify the specific Doppler effect and motor pitch changes as a drone manoeuvres. These are the features that we enter in Random Forest Classifier. This machine learning model was chosen for its rapid speed of inference and high accuracy on structured audio data, which is optimal for a limited resource environment such as a Raspberry Pi.

Beyond mere classification, the project represents a substantial step forward in the state of the art by implementing the real-time Direction of Arrival (DOA) estimation. By taking advantage of the array of four microphones, the system uses techniques of time-delay localization to constrain the angle of the sound source relative to the device. This enables the system to identify the position of the drone (0 degrees to 360 degrees) in an instant. Furthermore, in order to solve the physical limitation of sound dropping off

over distance, digital Automatic Gain Control (AGC) algorithm was implemented. This dynamic system automatically increases the sensitivity of the microphone in a quiet environment for detection of distant drones and decreases the sensitivity in a loud environment to avoid distortion, which effectively increases the range of detectability of the prototype.

The proposed system also includes a dedicated real-time graphical user interface (GUI) which has been purely developed in Python. Instead of depending on internet and web browsers, the system opens a stand-alone application window directly on the device's display. This high-contrast dashboard functions like a "Tactical Radar" to give the operator immediate visual feedback. When a drone is detected, the radar interface changes dynamically, displaying a red target indicator showing the exact angle of arrival - calculated by the microphones. To increase the usability, the software has "Temporal Persistence," a logic that keeps the target dots on the screen for a short duration. This provides a visual trace of the moving drone that stabilizes the display and prevents jitter that is often found in raw acoustic data. This direct desktop integration ensures zero latency and allows the system to function securely in offline environments without network dependence.

The possibilities of this work are broad. In security and defense, it can be used to monitor restricted airspace around airports, governmental facilities and military installations where traditional radar blind spots exist. In civil administration, the system can be used to ensure the safety of crowds at large public events by alerting to unauthorized flights of drones that could be a physical threat. Moreover, its adaption capability makes it suitable to integrate with IoT-based smart city infrastructure, where airspace real-time monitoring becomes increasingly critical.

Looking to the future, there are a number of directions for future development. Expanding the dataset with recordings from other drone models and in different environmental conditions will further improve the robustness of the model. Integration with PTZ (Pan-Tilt-Zoom) cameras could enable the system to automatically pan a camera to the sound location in case of doubt (visual verification) to create a multi-modal defence system. Furthermore, multiple Raspberry Pi units could be connected in a mesh network in order to triangulate precise GPS coordinates instead of just the angular direction.

In conclusion, this capstone project presents a comprehensive, working solution for acoustic drone detection, combining the strengths of signal processing, machine learning, and embedded deployment. By exploiting the distinctive sound signatures of drones and leveraging efficient algorithms on the Raspberry Pi 5, the system provides a practical and scalable approach to drone monitoring. Its ability to not only detect but also localize drones via a real-time radar interface marks an important step toward enhancing situational awareness and improving airspace security. The successful implementation of this project demonstrates that effective defense systems do not necessarily require massive budgets, but rather intelligent application of accessible technology.

1.2 Need Analysis

The fact that Unmanned Aerial Vehicles (UAVs), or drones, are now being rapidly integrated into modern society has created a paradox. While these devices have revolutionized industries from logistics and agriculture to filmmaking, they have also created a new layer of vulnerability to public safety, privacy, and national security. Drones are no longer merely military expensive tools, they are cheap, accessible and increasingly capable. This accessibility means that anybody - from a hobbyist flying carelessly near an airport to a malicious individual who wants to sneeze in on a prison to smuggle contraband in - can disrupt critical operations. As skies become more crowded and the need for a robust detection system more accessible and effective, the need to have a robust and effective system has never been more important.

The main driving factor behind this project is the high failure of the traditional surveillance methods to cope with this particular threat. Current defense infrastructures were built to detect large aircraft, not little plastic quadcopters. When we break down the current market solutions, there are obvious gaps that we need to address in order to take a new approach.

For the first, take Radar systems. While effective in detecting airplanes, traditional radar often struggles at detecting small consumer drones. These drones have a very small "radar cross-section," which means that they reflect very little energy and they basically become invisible to older radar technology. Furthermore, radar systems are prohibitively expensive and require massive infrastructure, rendering them impractical

for protecting smaller sites, such as private homes or public events or small office buildings. They also have problems distinguishing between a drone and a bird, resulting in confusing false alarms.

Second, Visual detection systems (cameras) is the most intuitive solution, but they are very limited by the environmental conditions. A camera is only as good as the light that is available. In low light situations, fog, or bad weather, visual systems are useless. Even in perfect daylight, visual detection is harmed because of "line-of-sight" problems; if a drone is flying behind a tree or a building in an urban setting, it will be invisible to a camera. Additionally, high-resolution cameras with the ability to zoom-in on a small moving target are costly and computationally intensive to operate.

Third, the Radio Frequency (RF) scanners monitor the communication link between the drone and its remote controller. While this is a popular low-cost method, it has a fatal flaw: it only works if the drone is transmitting. Modern autonomous drones are capable of flying on pre-programmed paths using GPS and, on some occasions, do not even transmit any signals back to a pilot making detecting them via RF totally ineffective.

This landscape of limitations equates to a definite operational need for an alternative solution: Acoustic Detection.

The need for this particular project stems from the fact that sound is the one "unhideable" attribute of a drone. However, no matter how small or how large and however it flies (autonomous or manual), a drone needs to use motors and propellers to be able to stay in the air. These components produce a distinctive, high-frequency Acoustic Signature that can't be masked.

The particular shortcomings of the other methods are addressed by an acoustic based approach. Unlike cameras, acoustic sensors are completely passive and they work perfectly in total darkness, fog or rain. Unlike radar, they are low cost and easily detect small plastic devices. Unlike RF scanners, they can hear a drone even if the drone is flying autonomously in "radio silence." This makes acoustic detection a critical layer of defense, especially in complex urban environments, where buildings may prevent a camera from seeing, but sound may still travel.

In addition, there is an urgent need for protection of privacy in society. As drones become capable of carrying high-resolution cameras, this capability for individuals and organisations to understand if they are being watched is very important. A passive, low-cost acoustic system helps to empower users-whether they're stadium security managers or homeowners-to take charge of their airspace without military-grade budgets and/or special licenses.

In conclusion, the development of this acoustic drone detection system is not only a technical exercise, but a response to a real void in our security capabilities. It meets the demand for a portable, affordable, day-night capable, next-generation autonomous aerial threats defense system.

1.3 Research Gaps

A review of current literature reveals several key research gaps that this project is strategically positioned to address through a single, cohesive system. While acoustic detection is promising, existing solutions often suffer from environmental vulnerability, limited spatial awareness, and high computational requirements that hinder practical deployment.

- 1. Noise vulnerability in real-world environments** Current acoustic drone detection systems often perform well in quiet laboratory settings but fail in urban or outdoor environments. Heavy background noise from traffic, wind, and human activity can drown out the specific frequencies of drone motors. These uncontrolled conditions produce high false positive rates and reduce reliability, highlighting the need for more robust preprocessing and feature extraction methods that focus on the specific "texture" and velocity of sound rather than just raw volume.
- 2. Limited detection range and localization** Many existing acoustic solutions act as simple "tripwires"—they can tell you a drone is nearby but cannot tell you where it is. Without localization, detection alone is insufficient for security operations, as responders cannot determine the direction of the threat or track its movement. Most low-cost systems lack the hardware or algorithms to perform real-time Direction of Arrival (DOA) estimation, leaving a critical gap in situational awareness.
- 3. Binary-only classification limits (Oversimplification)** Much of the existing research reduces the problem to a simple "Drone vs. Silence" task. This ignores the complexity of real-world audio landscapes where "Noise" is not just silence but includes complex sounds like talking, walking, or wind. Systems trained on simple binary datasets often fail when introduced to these dynamic background noises, leading to poor real-world reliability.
- 4. High computational demand vs. Real-time performance** Many academic models achieve high accuracy by using deep learning architectures (like heavy CNNs) that require powerful GPUs to run. This makes them unsuitable for portable, battery-operated edge devices. For practical security deployment, there is a significant gap in lightweight, efficient models that can run on low-power hardware like a Raspberry Pi while maintaining low latency (under 100ms).

5. High cost of conventional radar-based solutions Traditional radar and Radio Frequency (RF) systems for drone detection are prohibitively expensive (\$50,000+), require complex infrastructure, and are often restricted by regulatory barriers. This creates a market gap for affordable, scalable, and portable acoustic solutions that can be deployed in sensitive or budget-constrained environments such as private properties, public events, or small security units.

To address the above gaps, Table 1 outlines the core problem areas and suggests practical research directions to bridge them:

Table 1: Key Research Gaps and Proposed Directions

Serial No.	Research Gap	Proposed Research Direction
I.	Noise vulnerability in real-world environments	Develop a robust pipeline using MFCCs and Delta features to capture sound texture and velocity, combined with Digital Automatic Gain Control (AGC) to filter noise and boost weak signals.
II.	Limited detection range and localization	Integrate a 4-Microphone Array to enable Direction of Arrival (DOA) estimation, allowing the system to triangulate the exact angle (0°-360°) of the drone source in real-time.
III.	Binary-only classification limits	Utilize efficient machine learning algorithms (Random Forest) instead of heavy Deep Learning models, enabling sub-100ms inference times directly on a Raspberry Pi 5 CPU.
IV.	High cost of radar-based systems	Engineer a low-cost (<\$150) , passive acoustic radar system using off-the-shelf components (ReSpeaker, Raspberry Pi) that requires no specialized infrastructure.

1.4 Problem Definition and Scope

Problem Definition:

The exponential growth in drone technology has created an important security paradox: even as drones have become cheaper and more capable, the systems built up to detect them remain prohibitively costly and technically limited. Traditional defence mechanisms come with huge blind spots in the real world. Radar systems also have trouble detecting small, plastic, consumer drones because of their low radar cross-section. Optical cameras are useless in fog, darkness or in the presence of obstructions, and Radio Frequency (RF) scanners are useless against autonomous drones that do not transmit communication signals. Furthermore, the high infrastructure cost of these conventional systems makes them inaccessible for protection of smaller perimeters such as private properties, public events or small office compounds. There is an urgent and unmet need for a passive, affordable and stealthy detection solution that can be operated reliably 24*7 without requiring a military-grade budget.

Scope:

- The scope of this project is the end-to-end design and deployment of a fully functional acoustic drone detection prototype. The system is specially designed to be a stand-alone edge device. The scope includes:
 - Spatial Audio Acquisition: Use of a Seed ReSpeaker 4-Microphone Array to acquire multi-channel audio data to not only detect but to spatially be aware.
 - Signal Processing & Feature Extraction: Implementing Digital Signal Processing (DSP) techniques, such as Automatic Gain Control (AGC), as well as the extraction of Mel-Frequency Cepstral Coefficients (MFCCs) and Delta features to identify unique motor signatures.
 - Edge-Based Machine Learning: Training and optimizing a lightweight Random Forest Classifier for distinguishing between drone signatures and complex environmental noises (wind, speech, traffic) at the device.
 - Real-Time Localization: Developing algorithms for Direction of Arrival (DOA) estimation in the triangulation of the sound source with respect to the sensor.
 - Tactical Visualization: Designing a custom Python-based graphical user interface (GUI) to act as a real-time radar screen that gives real-time visual alerts and target tracking.

- Hardware Deployment: Completing the software stack on a Raspberry Pi 5 to achieve a portable, low-power, and offline-able security solution.

1.5 Assumptions and Constraints

Assumptions:

- Distinct Drone Sound Signatures It is assumed that all standard commercial drones produce a unique, identifiable sound - mostly a high-pitched whine produced by the high-speed rotors and motor vibrations - that is distinct enough to be separated from common background sounds such as the sound of human speech or traffic.
- Manageable Background Noise: Although real-world environments contain different noises, it is assumed that these background noises don't entirely overlap with the specific frequency range of the drone's motors, and the system is then able to separate the target signal.
- Representative Custom Dataset: It is assumed that the custom dataset noted for this project - which consists of various scenarios such as 'hovering', 'fly-by', 'distant flight' and a diverse 'noise' class - is enough to teach the model to recognize drones in general, even if the specific drone model changes.
- Hardware Reliability: It is assumed that the Raspberry Pi 5 and the ReSpeaker 4-Microphone Array will perform consistently without significant hardware-induced latency or electrical interference, assuming that they are powered correctly.
- Unobstructed Microphone Placement: It is assumed that the device will be implemented in a place where there is a relatively clear line-of-hearing, without physical coverings or obstructions that would significantly muffle the sound as the sound waves travel to the microphones.
- Model Efficiency: It is presumed that the lightweight Random Forest algorithm is optimized enough to make inference cycles take the form of milliseconds, to ensure the software can keep up with the incoming live audio stream without crashing the system.

Constraints

- Environmental Noise Limits: In very loud environments, such as right next to a highway, a construction site, or during a storm, the amount of background noise can be so much that even the sensitivity of the microphone is overwhelmed and the detection accuracy is reduced.
- Physics of Sound (Range): The system is limited by the physical laws of sound propagation (the square inverse law). As a drone flies away, its sound energy decreases dramatically. Even with Automatic Gain Control, there is a hard physical limit of how far the microphones can "hear."
- Dataset Variety: As the machine learning model is trained on a certain set of recordings, it may not be able to detect custom-built drones or very large industrial UAVs if their acoustic signature is significantly different from the quadcopters in the training data.
- Hardware Resources: The system has a limited resource pool of the computing power of the Raspberry Pi 5. While powerful for its size, it is unable to run the enormous, deep neural networks that are used by supercomputers. The software needs to be put under strict optimization so that it does not overheat or cause a memory overflow.
- Weather and Wind: Acoustic sensors are physically sensitive to wind striking the microphone diaphragm. Strong wind gusts can cause "clipping" or distortion in the audio signal, which could potentially confuse the classifier or cause false negatives for a short period of time.
- Indoor vs. Outdoor Acoustics: The system is mostly designed for outdoor use. If tested indoors in a small room, sound reflections (echoes) bouncing off walls might interfere with the Direction of Arrival (DOA) estimation to make the radar tracking less precise.

1.6 Standards

The project will be developed in accordance with established industry and academic standards to ensure quality, reliability and maintainability:

- **Software Engineering Standards:** The development process follows the Agile Software Development Life Cycle (SDLC) in order to enable iterative improvements. Best practices include modular Python programming (PEP 8 style guide), the use of version control (Git) to track the changes, and unit testing to ensure the logic of the machine learning pipeline (informed by coursework in Software Engineering (UCS503)).
- **Acoustic & Data Standards:** The recording audio data follows standard digital audio protocols (16kHz sample rate, 16-bit depth, mono/multi-channel WAV format) for compatibility with the machine learning libraries. Data handling follows some basic principles of balancing datasets to avoid the bias between "Drone" and "Noise" classes.
- **Hardware and Safety Compliance:** The use and installation on the Raspberry Pi 5 and ReSpeaker Array adheres to standard operating limits for voltage and thermal management. The system is designed to work passively with no radio interference so that it meets the general regulations of passive surveillance and electronic safety (CE/FCC guidelines) for edge devices.

1.7 Approved Objectives

The following objectives for this project were formally approved during the evaluation of the proposal by the panel:

- **Develop an edge-based acoustic detection system:** Design a portable, off-the-shelf system with the Raspberry Pi 5 and ReSpeaker 4-Microphone Array to capture and analyze drone sound signatures in real-time without external cloud processing.
- **Apply efficient signal processing and machine learning:** Apply Mel-Frequency Cepstral Coefficients (MFCCs) for feature extraction and lightweight classifiers (Random Forest) in order to accurately distinguish drone motor sounds from complex background noises.

- **Estimate direction by using acoustic localization:** Implement Direction of Arrival (DOA) algorithms using the microphone array to estimate the exact angle of the sound source and determine the location of the drone with respect to the sensor.
- **Approximate Distance by Signal Strength:** Analyze the signal strength of the detected audio to approximate the relative proximity of the drone, differentiating between close-range threats and distant target.
- **Evaluate the robustness of the system in real-world scenarios:** Test the integrated prototype in different environmental conditions, such as varying distances and noise levels, to validate its robustness and detection latency in real-world field operations.

1.8 Methodology

The methodology for this project is aimed at developing, training and deploying a light weight acoustic UAV detection system that can run on edge hardware in a systematic way. It involves the following important stages:

- **Data Acquisition:** Rather than using only data available on the internet, we created our own, real-world dataset using the Seed ReSpeaker 4-Microphone Array. Audio samples were recorded in different operational scenarios, such as the drones hovering at close range (1-3 meters), flying at a distance (10+ meters) and performing high-speed flybys. A separate "Noise" class was also recorded, which included environmental sounds such as the wind, people speaking and silence to make sure the model learns to filter out false alarms.
- **Signal Preprocessing:** The original audio signals are first acquired and resampled to 16kHz to conform to the standard requirement of voice processing. We implemented a digital Automatic Gain Control (AGC) system used to dynamically change the input volume. This step is a critical one for normalizing the audio so that the system can amplify faint signals coming from remote drones without distorting the audio due to nearby loud noises.
- **Feature Extraction (MFCCs):** In order to make the system efficient enough for a Raspberry Pi, we made a shift in the embeddings, to heavy deep learning. Instead, we extract features called Mel-Frequency Cepstral Coefficients (or

MFCCs) from the sound frames (or audio frames). We also calculate "Delta" features, which are the rate of change in the sound. These features specifically pick up the high frequency whine and Doppler effect (change in pitch) of spinning drone motors to form a unique digital fingerprint for the target.

- **Model Development and Training:** A Random Forest Classifier was used as the core machine learning engine. Unlike heavy Neural Networks which need GPUs Random Forest is very efficient with CPUs and gives fast decision making. The model was trained to perform the binary classification task (Drone vs. Noise), finding the best "decision trees" for separating the drone signatures from the background noise respectively based on the MFCC extracted data.
- **Localization and Visualization:** Not only does the methodology include simple detection, it also includes spatial awareness. We use the hardware capabilities of the 4-microphone array system to carry out Direction of Arrival (DOA) estimation, that is to calculate the angle of the sound source location (0deg-360deg). Distance is approximated by analyzing the relative signal intensity (volume decay) of an object detected. Finally, these outputs are passed into a custom Python-based Graphical User Interface (GUI), which displays the data in the form of a real-time tactical radar screen to the user.

1.9 Project Outcomes and Deliverables

- **Accurate Edge-Based Detection:** The system is able to accurately detect signs of drones in the air by the Raspberry Pi 5. By using the efficient Random Forest algorithms and MFCC feature extraction it is effectively distinguishing drones from different types of background noises without relying on heavy external computers or cloud processing.
- **Real-Time Radar Visualization:** One of the main deliverables is the tailored Python-based Graphical User Interface (GUI) acting as a live tactical radar. Unlike simple text logs, this standalone window gives immediate feedback with visual data that plots detected drones as discrete targets on a polar grid to ensure rapid interpretation by the operator.
- **Directional Tracking (DOA):** Taking advantage of the hardware capabilities of the 4-microphone array, this system gives precise Direction of Arrival

(DOA) estimation. It triangulates the sound source to provide specific (angle in degrees 0-360) information on the drone position in respect to the device, turning simple detection into active tracking.

- **Standalone Offline Operation:** The final prototype is delivered as a totally portable unit, which functions completely offline. This ensures zero-latency and maximum privacy, which is why it is a good system to deploy in a remote location or a secure facility where internet dependency is considered a vulnerability.
- **Adaptive Noise Handling:** The system uses digital Automatic Gain Control (AGC) software. This makes it possible for the device to dynamically adapt the sensitivity of its microphone according to the surrounding volume, so that in a quiet environment in the open field as well as in a noisy environment the detection performance is consistent.
- **Cost-Effective Security Prototype:** The project provides a replicable low-cost hardware-software blueprint. It proves that airspace monitoring can be effectively created using accessible components and it will provide a scalable solution to civilian and commercial safety problem instead of expensive military-grade radar systems.

1.10 Novelty of Work

The novelty of this project lies in engineering a practical, "edge-native" security solution that moves acoustic drone detection from theoretical research to real-world application. Unlike traditional radar or optical systems that are expensive, weather-dependent, or limited by line-of-sight, this project introduces a low-cost, passive defense mechanism that listens for the "unhideable" sound signatures of drones.

The innovation of this work is demonstrated in three specific areas:

Efficient Edge Intelligence: Instead of relying on heavy cloud computing or massive deep learning models (like CNNs) that require expensive GPUs, this system achieves high-accuracy detection using optimized **Random Forest algorithms** and **MFCC feature extraction**. This allows the entire pipeline to run locally on a Raspberry Pi 5 with sub-100ms latency.

Spatial Awareness via Audio: The system goes beyond simple "presence detection" (binary classification) by integrating a 4-microphone array to perform real-time **Direction of Arrival (DOA) estimation.** This effectively turns a simple sensor into an "Acoustic Radar" that can pinpoint the angle of a threat.

Tactical Offline Visualization: A key novelty is the development of a standalone **Python-based Radar GUI.** By visualizing targets on a polar grid with temporal persistence tracking, the system offers professional-grade situational awareness in a completely offline, portable package that requires no internet connection or technical expertise to operate.

In summary, this project bridges the gap between complex signal processing and accessible security, proving that robust airspace defence can be achieved with affordable, off-the-shelf IoT hardware.

2. Requirement Analysis

This chapter lays down the necessary groundwork and requirements that make the project. It starts with a review of existing literature to both set the theoretical background and practical context. This is followed by a detailed Software Requirement Specification (SRS) which defines the functional and non-functional requirements of the system. The chapter ends with a first assessment of projected costs and possible risks.

2.1 Literature Survey

2.1.1 Theory Associated with Problem Area

The heart of this project is where two areas of specialization, Acoustic Signal Processing and Edge-based Machine Learning, merge.

- **Acoustic Signal Processing:** This field is concerned with the capture and analysis of sound waves in order to detect patterns and extract meaningful features. In the case of drone detection, mics are used to capture the signature high-pitched "whine" of UAV motors. These raw signals are then processed using techniques such as Mel-Frequency Cepstral Coefficients (MFCCs) and Delta Features (measure the rate of change in audio). Unlike simple volume detection, these types of techniques are able to analyze the "texture" and pitch of the sound, enabling the system to differentiate a drone from similarly-loud sounds such as a leaf blower or traffic. Furthermore, this project addresses Array Processing theory, in particular Time Difference of Arrival (TDOA), to determine the spatial angle of the sound sources using the delay in micro-secs between different microphones.
- **Machine Learning (Edge AI):** While Machine Learning in traditional sense is carried out on bulky servers, this project aims at "Edge AI" - implementing intelligent algorithms on low-power devices such as the Raspberry Pi. The system uses a Random Forest Classifier, which is a powerful ensemble learning algorithm that builds several decision trees during training. This algorithm was chosen over heavy Deep Neural Networks (like CNNs) as it provides an optimal

balance of high accuracy and extremely low latency (inference time <100ms), which is extremely critical in real-time security applications.

2.1.2 Existing Systems and Solutions

The current landscape of drone detection consists of the following approaches:

- Traditional Detection Methods Techniques such as radar, optical tracking and RF-based detection methods have been widely employed by military and large airports. However, they have serious flaws in civilian situations. Radar cannot easily detect small plastic consumer drones because they have a low radar cross-section. Optical methods are strictly weather-dependent and depend on clear weather and daylight, and fail completely in the dark or fog. RF methods do not work if the drone is not transmitting a signal to an operator (pilot) - they do not work on modern autonomous drones that fly in 'radio silence'.
- Acoustic-Based Systems: Research has focused on acoustic solutions as more passive and low cost. By making use of sound signatures, these systems can identify drones 24/7, even in the dark or behind visual obstructions (non-line-of-sight). However, historical challenges have included a high sensitivity to wind noise and a lack of directional tracking which this project addresses using advanced Digital Signal Processing (DSP) and microphone arrays.

2.1.3 Research Findings for Existing Literature

A review of the academic work that has been done recently illustrates both the potential and the limitations that guided our approach:

Acoustic Based Drone Detection Through Machine Learning [1]

- Finding: This research work used MFCC features with the help of classifiers such as Random Forest and MLP achieving 0.92 F-score.
- Relevance: It validates our selection of Random Forest as a very powerful classifier for acoustic signatures which proves that we do not necessarily need heavy deep learning models to get high accuracy.

Automated Accurate Sound-Based Amateur Drones Detection Method [2]

- Finding: Used TQWT-based feature extraction and Fine kNN and got 99.72% accuracy
- Relevance: Emphasizes the relevance of feature engineering. It justifies our choice to use special features such as Delta-MFCCs in order to extract the motion of the drone.

Audio Features-Based ADS-CNN Approach [3]

- Finding: By using MFCC + STFT features, a light-weighted CNN achieved an accuracy of 98.81% with low computational cost.
- Relevance: Shows that "lightweight" models are feasible for real time recognition on constrained devices, fitting in with our edge-computing philosophy.

Acoustic-Based Drone Detection Using Neural Network [5]

- Finding: Detected with 99.5% probability within 600 meters by analyzing the spectrogram.
- Relevance: Although a powerful approach, this kind of approach required heavy computation. Our project is aiming to achieve similar reliability with shorter ranges using much cheaper hardware (Raspberry Pi vs. GPU workstations).

Comparison of Acoustic Features of Drone Detection [6]

- Finding: Evaluated the results of MFCC, GTCC, LPC, spectral roll-off, and zero-crossing rate. An SVM classifier was used to reach 99.9% accuracy.
- Relevance: Confirms that a combination of multiple audio features (as we use with MFCCs + Deltas) does much to increase the reliability of classification.

Drone Detection Machine Learning Algorithms [4]

- Finding: Compared radar, acoustic, visual and RF systems, noting that acoustic systems are the most cost effective, but range limited solution.
- Relevance: Gives us the justification for our "Need Analysis"- that there is a critical gap in the low-cost market that radar can't reach that acoustic systems fill.

Fusing Multi-Domain Features for Aerial Vehicle Detection [8]

- Finding: Achieved an accuracy of 98.3% for classification of drone, helicopter and background noise.
- Relevance: Supports the need for a multi-class dataset. To prevent false alarms, our system is trained, not just on "Drones," but explicitly on "Noise" and sounds from the environment.

Fused Approaches based Drone Detection and Tracking System [7]

- Finding: Combining acoustic and optical data makes for improved robustness.
- Relevance: Although our scope at present is of acoustic nature only, this research opens up the path towards scalability of our project in the future (e.g., triggering a camera upon acoustic detection).

2.1.4 Problem Identified

Although research that exists does frame the utility of methods that rely on acoustics, most current systems have challenges including:

- **Lack of Spatial Awareness:** Most low-cost acoustic sensors are binary (Drone/No Drone) and cannot tell the user which direction the drone is coming from.
- **Hardware Dependency:** High accuracy models from literature often use large GPUs, which are not portable and useful for battery powered security units.
- **Noise Vulnerability:** Many academic models fail in the real world where wind or traffic noise may be present.
- **Limited Visualization:** Existing research frequently concentrates on the algorithm and does not have an interface that is usable by non-technical security operators.

2.1.5 Survey of Tools and Technologies Used

The development on this project makes use of a modern and efficient technology stack with a Raspberry Pi 5 focus:

- **Programming Language:** Python 3 is the main language thanks to the wide support for audio analysis and interfacing with hardware (using the Python library called 'GPIO').
- **Hardware Interface:** PyAudio and the ReSpeaker API are implemented to interface with the 4 microphones array, including multi-channel audio capture and LED control.
- **Audio Processing:** Librosa and NumPy are used for Real-time Digital Signal Processing (DSP) such as Automatic Gain Control (AGC) and the calculation of MFCCs.
- **Machine Learning:** Scikit-learn is used to build, train and execute the Random Forest Classifier. Its light weight nature guarantees low CPU load for inference.
- **GUI Development:** Tkinter or PyGame (depending on final build) is for the standalone "Tactical Radar" dashboard. This enables high-performance, offline rendering of the radar grid and target dots without requiring the utilization of a web browser.
- **Data Management:** CSV/JSON formats are used for the lightweight logging of detection events; the data can be stored by the system for weeks of history without filling up the SD card.

2.1.6 Summary

The literature survey has proven that although drone detection is a fast-evolving domain, the current solutions are leaving a huge gap in the market. Traditional methods such as radar and cameras are too costly, too complicated or rely on ideal weather conditions. Acoustic detection has become a promising alternative as it is passive, cost-effective and effective 24/7.

However, the majority of current acoustic research is theoretical, or requires heavy and non-portable computers. Some of the common limitations identified are that it is vulnerable to environmental noise, it lacks spatial awareness (users don't know where the drone is), and it has high computational requirements that means it cannot be used on battery-powered devices.

This project addresses these shortcomings directly. By incorporating an efficient Digital Signal Processing (Automatic Gain Control) paired with lightweight Machine Learning (Random Forest) onto a Raspberry Pi we hope to create a system that is not only accurate but also portable and practical. The proposed solution goes beyond simple detection (binary) to provide real-time localization and "radar-style" tracking to fill the need for an accessible, professional-grade airspace monitor.

2.2 Software Requirement Specification

2.2.1 Introduction

2.2.1.1 Purpose

The purpose of this software is to deliver an automated, edge-based system to achieve the real time detection and monitoring of drones with their unique acoustic signatures. The objective of the system is to work as a practical, stand-alone security tool for enhancing situational awareness and airspace safety by successfully identifying drones and determining their approximate distance and specific angle of arrival and provide timely visual alerts. This approach offers a portable, low-cost and non-intrusive alternative to the complex traditional detection techniques such as radar or the use of RF tracking.

2.2.1.2 Intended Audience and Reading Suggestions

The target audience for this system is security personnel, private property owners, defence organizations and event managers who are responsible for keeping the skies in their area safe. Users do not need to have any expertise in machine learning or signal processing as the system offers an intuitive "Radar-like" dashboard and analysis. This document is directed mainly towards the project development team and academic supervisors, as well as stakeholders around the implementation, the testing, and the evaluation of the prototype.

2.2.1.3 Project Scope

The system will allow users to monitor live audio environmental data captured directly via a microphone array and get immediate analysis about the presence of drones. The main analysis will involve classification of the sound into two classes (Drone vs.

Noise/Environment) and the real time Direction of Arrival (DOA) estimation by using acoustic triangulation methods. A user-friendly desktop interface (GUI) will include a dynamic radar screen, active target tracking and detection logs. The scope is strictly on passive audio-based drone detection and does not currently include integration with active radar, RF sensors or long range optical tracking systems.

2.2.2 Overall Description

2.2.2.1 Product Perspective

The proposed system is an application that operates independently in edge computing, and that exploits acoustic signals to realize drone detection in real time. Unlike traditional detection systems that use expensive radar, line-of-sight optical tracking, or active RF communication signals, this product only deals with passive audio-based monitoring. This makes it a cost-effective, portable, non-intrusive alternative for decentralized security. The system combines hardware-level audio capture, digital signal preprocessing, machine learning classification and spatial localization into a single seamless, offline process.

The application is built with a modular architecture, which consists of:

- **Audio Capture Module:** Using a 4-microphone array that will capture spatial audio data directly from the environment.
- **Signal Processing Module:** Responsible for Automatic Gain Control (AGC) and extraction of Mel-Frequency Cepstral Coefficients (MFCCs) to extract unique motor signatures.
- **Classification Module:** Driven by a lightweight Random Forest algorithm to effectively differentiate drone motor noises from the general environmental noises.
- **Localization Module:** able to estimate the Direction of Arrival (DOA) and relative distance of detected targets by array processing.
- **Tactical User Interface:** Python-based desktop GUI for real-time radar visualization, target-tracking and real-time alert notification.

This product is designed to work on low power hardware (Raspberry Pi), making it very suitable for perimeter security at private properties, events, prisons and critical infrastructure where internet connectivity may be limited or unavailable.

2.2.2.2 Product Features

The salient features of the system are:

- **Passive Acoustic Detection:** Identification of UAVs by their unique acoustic "fingerprint," which works well even in low light or non-line-of-sight conditions.
- **Intelligent Classification:** Effective discrimination of drone signatures from common background noises (wind, speech and traffic) to reduce false alarms.
- **Direction and Distance Estimation:** Utilization of time delay algorithm to triangulate the exact angle (0deg-360deg) of the drone and sound intensity analysis to approximate the proximity.
- **Real-Time Radar Visualization:** A real-time, high-contrast dashboard visualization plotting detected threats on a polar grid, with immediate situational awareness.
- **Standalone Operation:** Fully functional offline operation with zero latency and data privacy, independent of cloud servers and the internet.
- **Adaptive Signal Processing:** Integrated Automatic Gain Control (AGC) to ensure performance in a range of noise levels from quiet fields to noisy urban areas.
- **Low-Cost and Portable:** Relies solely on a Raspberry Pi and a microphone array, a scalable security solution at a fraction of the cost of traditional radar systems.

2.2.3 External Interface Requirements

2.2.3.1 User Interfaces

The system has User Interfaces in the form of standalone, Python-based Graphical User Interface (GUI) that runs directly on the device screen. The UI is made with high-contrast tactical visibility in mind, so that it is simple and easy to understand for non-technical security personnel.

Key elements include:

- **Tactical Radar Dashboard:** A central polar grid display, which plots detected drones in real-time. It visualizes the specific Direction of Arrival (DOA) (0deg-360deg), putting a red target dot at the estimated angle of the sound source.
- **Status Indicators:** Prominent text labels that alternate between "SAFE" (Green) and "ALERT" (Red) to give immediate status of the system at a glance.
- **Detection Logs:** A set of scrollable sidebars that provide a history of recent events including time stamps and confidence levels for classification.
- **Visual Persistence:** The interface follows "persistence logic" to ensure that the target dots persist on the screen long enough to leave a visual trail of the drone's movement path.
- **Control Panel:** Simple on-screen controls to start/stop monitoring or change microphone sensitivity thresholds.

The interface is optimized for fixed-resolution displays (like the official Raspberry Pi Touch Display, or standard HDMI monitors) as opposed to a responsive web browser layout.

2.2.3.2 Hardware Interfaces

The system is based on compact and off-the-shelf edge computing hardware to guarantee portability.

The following hardware interfaces are needed:

- **Acoustic Sensors:** Acoustic Sensor 1 - A Seeed ReSpeaker 4 Microphone Array USB connected This hardware is necessary to receive 4-channel spatial audio to facilitate direction finding.
- **Processing Unit:** A Raspberry Pi 5 (Quad-Core ARM Cortex-A76 CPU, 4GB/8GB RAM). The system is optimized to work on this CPU without the need for an external Graphics Processing Unit (GPU).
- **Display Interface:** A regular HDI monitor or built-in LCD screen to visualize the Python GUI.

- **Visual Feedback Hardware:** Direct interface with the ReSpeaker's on-board LED ring to physically light-up red when a drone is detected.

2.2.3.3 Software Interfaces

The system interfaces with a specific stack of efficient software components to ensure smooth offline functioning:

- **Operating System:** Raspberry Pi OS (Debian-based Linux), optimized for embedded performance.
- **Audio & Signal Processing:** PyAudio is used for hardware communication (input streams), while Librosa and NumPy handle the math for feature extraction (MFCCs) and Automatic Gain Control.
- **Machine Learning Engine:** Scikit-learn is utilized to load and run the lightweight Random Forest Classifier for real-time inference.
- **GUI Framework:** Tkinter or PyGame (Python libraries) are used to draw the radar graphics and manage the application window, replacing the need for web browsers.
- **Data Storage:** Lightweight CSV or JSON file logging is used to store detection history locally, eliminating the need for complex database management systems like SQL.
- **Driver Interface:** Specialized ReSpeaker Drivers (Seeed Voicecard) are used to access the raw multi-channel audio data and control the LED hardware.

2.2.4 Other Non-functional Requirements

2.2.4.1 Performance Requirements

- **Low Latency Processing:** The system must process chunks of audio and produce a classification result within 0.5 seconds (500ms) in order to ensure true real-time tracking on the radar screen.
- **High Accuracy:** The Random Forest classifier should have a detection accuracy of at least 90% in "Drone" class under normal operation conditions (up to 10 meters distance).

- **System Stability:** The application should be able to run continuously for long periods of time (i.e. 24+ hours) without crashing or being plagued by memory leaks on the Raspberry Pi.
- **GUI Responsiveness:** The tactical radar dashboard must update without freezing - the processing must be smooth, so that the movement of the targets (dots on the screen) is fluid, rather than jumpy.

2.2.4.2 Safety Requirements

- **Passive Operation:** The operation must be strictly passive (listening only) without any radio interference that could interfere with any legitimate drone communications or any other electronic devices.
- **False Alarm Mitigation:** The software needs to have "persistence logic" (involving multiple, successive positive detections) that prevents a momentary loud noise (such as a door slamming) from triggering a full panic alert.
- **Fail-Safe Mechanisms:** If some software crashes or the microphone disconnects, then the system should automatically restart the monitoring script or notify the operator with a clear "SENSOR ERROR" status.
- **Hardware Safety:** Since the physical deployment (Raspberry Pi and microphones) is going to be under the weather or humidity or heat, the enclosures must be enclosed or insulated from short circuits and ensure the safe operation of the system.

2.2.4.3 Security Requirements

- **Device Access Control:** Since the system is a physical edge device, access to the operating system and settings needs to be protected by a strong Administrator Password to avoid tampering by unauthorized users.
- **Offline Data Privacy:** All the detection logs and audio data should be stored offline in encrypted SD card of the device. The system should be in an "offline mode" by default in order to remove the possibility of being hacked remotely.

- **Port Security:** Unused physical ports (USB/Ethernet) on the Raspberry Pi should be disabled or physically blocked in high security deployments to prevent data theft by external drives.
- **Configuration Lock:** Only authorized Administrators should be able to change sensitive parameters such as sensitivity of the microphone or detection thresholds; standard operators should have "View Only" access to the Radar screen.
- **Log Integrity:** The system needs to ensure that detection logs are securely appended and cannot be easily wiped out or altered by a common user, keeping the history of events for post incident analysis.

2.3 Cost Analysis

The cost analysis for the proposed drone detection system is based on the vital hardware components procured for the implementation of the prototype. Since the project is focused on a low cost, edge computing solution, the expenditure is much lower than commercial radar systems. The breakdown of individual costs is the following:

- Microphone Array (Seeed ReSpeaker 4-Mic):
₹6,353
- Processing Unit (Raspberry Pi 5):
₹5,900
- Power Supply (USB-C PD 27W):
₹1,074
- Active Cooler (Raspberry Pi 5):
₹456
- Storage (High-Speed SD Card):
₹627
- Total Estimated Cost:
₹14,410

This total cost accounts for the basic hardware cost required to set up a working prototype of the acoustic-based drone detection system. It is important to note that the

inclusion of the Active Cooler and a high-wattage Power Supply is required specifically for the Raspberry Pi 5 to be capable of the computational load of machine learning without throttling. While there may be other costs associated with 3D printed enclosures, cabling, and mounting tripods depending on the environment of deployment, this analysis includes the minimum viable system configuration for successful development and testing.

2.4 Risk Analysis

Every piece of engineering work has certain risks that can impact or affect its development, deployment, or long-term effectiveness. For this acoustic based drone detection system, potential risks have been classified in the different areas, technical, hardware, operational and security. The analysis of these risks and the specific mitigation strategies pursued in this project are presented below:

Table 2: Risk Analysis according to categories

Risk Category	Description of Risk	Likelihood	Impact	Mitigation Strategy
Technical	High Background Noise: Urban environments with traffic or wind may mask the drone's sound signature, leading to false negatives or reduced accuracy.	Medium	High	Digital Signal Processing: Implementation of Automatic Gain Control (AGC) and noise-robust feature extraction (Delta-MFCCs) to isolate motor frequencies from ambient noise.
Data	Limited Dataset Diversity: If the training data only contains one type of drone, the model may fail to detect different models (overfitting)	Medium	High	Diverse Data Collection: The dataset includes recordings of drones at various speeds, distances, and angles, alongside a strict "Noise" class to improve generalization.
Operational	Real-Time Latency: Complex machine learning models might	Medium	High	Model Optimization: Use of a lightweight Random Forest Classifier instead of heavy

	take too long to process audio, causing a lag between the drone's arrival and the alert.			Deep Neural Networks, ensuring inference times remain under 100ms on the CPU.
Hardware	Overheating & Throttling: The Raspberry Pi 5 generates significant heat during continuous processing, which could cause the system to slow down or crash.	High	Medium	Thermal Management: Integration of a dedicated Active Cooler and heatsinks to maintain optimal CPU temperatures during 24/7 operation.
Security	Physical Tampering: As an edge device deployed in the field, the hardware is vulnerable to theft or unauthorized physical access.	Low	High	Offline Security: The system operates completely offline (no internet required), and sensitive settings are locked behind an administrator password to prevent tampering.
Cost	Scalability Expenses: Costs may rise if multiple units are needed to cover a large perimeter.	Medium	Medium	Modular Design: The system uses off-the-shelf components, allowing for easy and affordable replication compared to proprietary radar hardware

3. Methodology Adopted

This chapter presents the structured methodology that was adopted to design, engineer and implement the acoustic-based drone detection system. It starts with the justification of the choice of an experimental investigative technique because the project is fundamentally based on the collection of real data in the field, the controlled testing of those data in the field, and the validation of hypotheses with the help of machine learning models. The chapter then presents the proposed solution, describing the hardware architecture, the digital signal processing pipeline and the classification workflow to identify and find the location of drones. Further, a work breakdown structure is given to clearly define the key modules from audio acquisition to graphical user interface, and their interdependencies to ensure clarity in the project execution. Finally, the chapter emphasizes the particular tools and technologies used, from hardware elements such as Seeed ReSpeaker 4-Microphone Array, Raspberry Pi 5, to software libraries such as Scikit-learn, Librosa, which are the building blocks of the proposed system.

3.1 Investigative Techniques

The creation of a strong acoustic drone detection system demands the judicious selection of an investigative technique to ensure that the project results are valid, reproducible, and useful in a real-world security context. Since the project is about obtaining the raw environmental audio and analyzing the particular acoustic properties of that environment and then using logic-based models for classification and localization purposes, it comes under the broad umbrella of Experimental Investigative Research. However, Descriptive and Comparative investigative techniques also play an important supporting role in organizing the data and confirming the choices of the system design. This section describes these categories in detail in the context of this particular prototype.

Descriptive Technique of Investigation

The focus of the descriptive technique is primarily on observing, recording and cataloging phenomena without first manipulating variables. In the context of this project this phase is the systematic collection and analysis of raw acoustic data from different sources: target (drones) and distractors (environmental background noise).

Key aspects include:

- **Data Collection Tools:** The Raspberry Pi 5 with ReSpeaker 4-Microphone Array is the main tool to collect the spatial sound.
- **Observation:** The raw WAV recordings are processed and examined to investigate frequency ranges, amplitude changes and individual acoustic signatures.
- **Analysis:** For instance, the high-speed rotors of a quadcopter make a distinct, periodic high-frequency "whine" (harmonic structure) which is significantly different from the irregular, low-frequency rumble of wind or the chaotic patterns of human speech.

This descriptive investigation is the base of the knowledge base on which our signal processing (MFCCs) and feature extraction techniques are built. It ensures that the characteristics peculiar to drone motors are adequately documented serving as the starting point for training the machine learning model.

Comparative investigative technique

The comparative technique comes in when analyzing the differences between sound signatures and evaluating the performance of different engineering choices. Comparative investigation is carried out in two major ways in this project:

Comparing Sound Signatures:

- **Drone vs. Noise** The mathematical features of drone audio is compared to various background sounds (silence, wind, traffic) using this system.
- **Feature Analysis:** Using other methods such as Mel Frequency Cepstral Coefficients (MFCCs) and Delta (velocity) analysis, clear distinctions are made. We can see that drones have stable and tonal frequency peaks but the background noise usually has "flat" or scattered spectral energy.

Comparing Algorithms:

- **Model Selection:** To design the training, different approaches were compared - i.e., heavy Deep Learning models (similar to CNNs) vs light weights classifiers (similar to Random Forest)
- **Performance Benchmarking:** These were compared against each other in terms of the following: Accuracy, Inference Speed (Latency) and CPU Load. The comparative analysis led to a conclusion that for RP 5, Random Forest classifier gave the optimal balance like accuracy with insane computational delay of heavier classifiers.

Thus, comparative investigation provides for evidence-based decision-making, ensuring that the system is optimized not just for accuracy theoretically, but in terms of edge-device performance.

Experimental Investigative Technique

The basic methodology for this project is experimental investigation, which entails designing controlled tests, defining variables, and testing hypotheses through active trials. Since the goal is to prove that drones can be detected in real-time using a low-cost device, experiments make up the backbone of the research.

Key aspects include:

- **Controlled Data Collection:** Recorded data is collected in a variety of controlled environments (indoor quiet rooms and outdoor windy fields) to assess the limits of the system.
- **Independent Variables:** What we're changing (Drone Distance - 1m, 5m, 10m, Angle of Arrival [0deg to 360deg], intensity of Background Noise)
- **Dependent Variable:** The output performance of the system, i.e. Classification Accuracy (Did it detect the drone?) and Localization Error (Did it point to the right angle?).
- The central hypothesis of this study is that drone sound signatures are sufficiently distinguishable as to be detectable and localizable by a 4-

microphone array using lightweight (energy-efficient) machine learning, even with some mild interference.

- **Validation:** Experiments are run to quantify how well the system generalizes to new situations such as detecting a drone when the user is talking nearby.

This experimental approach not only provides a proof of concept for the viability of the acoustic detection, but also measurable data to refine the threshold settings and Automatic Gain Control (AGC) logic for the final deployment.

3.2 Proposed Solution

The proposed solution is an autonomous and edge-based drone detection system that uses acoustic signatures to detect aerial threats in real-time. The system architecture is designed with the following integrated parts:

- **Data Acquisition:** A Seeed ReSpeaker 4-Microphone Array is used as the primary sensor, which collects high fidelity spatial audio from the environment. Unlike normal microphones, this is an array of 4 channels that is essential for knowing the sound source direction, and directly interfaces with the Raspberry Pi 5 through USB.
- **Pre-processing & Feature Extraction:** In this stage, raw audio streams are normalized with the help of digital Automatic Gain Control (AGC), increasing faint signals and avoiding signal distortion. Instead of using heavy neural embeddings, the system uses Mel-Frequency Cepstral Coefficients (MFCCs) and Delta features. These specific features mathematically represent the unique "texture" and velocity-induced pitch changes (Doppler effect) of spinning drone motors, which form a unique digital fingerprint in order to classify it.
- **Machine Learning Model:** A very efficient Random Forest Classifier is used to differentiate between "Drone" and "Background Noise." This model has been selected for its better inference speed on edge CPUs. It is trained on a custom

real world dataset, and optimized to run directly on the Raspberry Pi without the need for external GPU acceleration or cloud connectivity.

- **Localization Module:** Spatial awareness is obtained using Direction of Arrival (DOA) algorithms. By calculating the time difference of arrival (TDOA) in micro seconds of sound waves impacting the four microphones, the system triangulates the exact angle (0deg-360deg) of the target. At the same time, signal intensity analysis gives an approximation of the relative distance of the drone.
- **Tactical User Interface:** A separate Python-based Radar Dashboard (GUI) over traditional web browsers. This high-contrast interface makes a polar grid plot detected drones as targets in real-time. It visualizes the angle of the drone, tracks the movement path with the help of persistence logic, and offers visual immediate alerts (Red/Green status) to the operator, providing an instantaneous way of having situational awareness, without requiring internet.
- **Deployment:** The complete integrated software stack is deployed on a Raspberry Pi 5, making it a portable, offline unit. This guarantees low power consumption and zero-latency performance that makes the solution scalable for perimeters securing in remote or sensitive environments.

3.3 Work Breakdown Structure

Phase 1: Drone Detection (Prototyping)

1.1 Project Set up & Environment Setup

- Installing Python 3 and some important libraries (Scikit-learn, Librosa, NumPy).
- Setup of IDE environment(Vs Code) for effective development.
- Implement version control (Git) for code management.

1.2 Data Acquisition & Preparation

- Record custom samples for Drone and Noise classes to use with ReSpeaker.
- Organize the raw WAV dataset in labelled folders.
- Implement preprocessing scripts for cleaning audio and extracting features (MFCC).
- Implement a strategy for splitting the data into the training set and testing set.

1.3 Model Development & Training

- Setting up Random Forest Classifier on Scikit-learn.
- Implement feature scaling and selection for optimising the input data.
- Train the model on the custom dataset: Distinguish between Drone vs. Noise.
- Tuning hyperparameters (No of Estimator, Depth of Tree) for Accuracy.

1.4 Model Evaluation & Analysis

- Assuming validation scripts should be run using the unseen test data set.
- Develop confusion matrices to provide analysis on false positives/negatives.
- Clean the data set by taking more samples of weak situations.

1.5 Real-time Prototyping

- Implement a Python script for a continuous audio stream capture.
- Inferencing using the trained Random Forest model.
- Verification of real-time performance and latency on laptop before porting.

Phase 2: Localization & Edge Deployment

2.1 Hardware Setting & Configuration

- Installing Raspberry Pi 5 with the latest Raspberry Pi OS (Linux).
- Installing drivers for Seeed Voicecard drivers for the ReSpeaker 4-Mic Array using the following:
 - Checking 4-channel raw audio capture + LED control.

2.2 Model Porting & Optimization

- Transfer the already trained Random Forest model (Pickle file) to Raspberry Pi
- optimize python scripts to reduce its CPU usage during inference
- Implement Automatic Gain Control (AGC) for Dynamic Volume Control.

2.3 Algorithm of Direction of Arrival (DOA)

- Implement Time Difference of Arrival TDOA based logic using GCC-PHAT.
- Calibrate microphone array to place audio delays on a 0–360-degree basis.
- Integrate "Persistence Logic" to stabilize jittery angle readings.

2.4 System Integration & GUI Development

- Integrate detection, localization and visualization into a single primary application.
- Implement the standalone Python Radar Dashboard (GUI using Tkinter/PyGame).
- Package Application to Auto-Launch on Boot for Headless.

2.5 Testing & Validation

- Field test of using drone flying at different distances (1m-10m).
- Verify the accuracy of the Radar GUI in tracking moving targets.
- Document latency of system, battery and final detection range.

Project timeline outlining the schedule for research, development, and implementation phases.

Sr. No.	Activity	Month	February		March		April		May		June		July		August		September		October		November				
			Week No.	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Researching Hardware / Planning	Plan																							
		Actual																							
2	Testing Hardware / Data Collection	Plan																							
		Actual																							
3	Data Preprocessing & Feature Extraction	Plan																							
		Actual																							
4	Machine Learning Model Development	Plan																							
		Actual																							
5	Implementation of distance & direction estimation techniques	Plan																							
		Actual																							
6	Model Evaluation / Testing	Plan																							
		Actual																							
7	System Integration / UI Development	Plan																							
		Actual																							
8	Performance Testing in Real Time Environment	Plan																							
		Actual																							

Figure 1: Project schedule and activity timeline.

3.4 Tools and Technology

- **Phase 1: Drone Detection (Prototyping):** Focused on building and validating a lightweight, high-efficiency machine learning model on a standard development machine.
- **Phase 2: Direction of Arrival (Deployment):** Focused on deploying the system on an embedded device (Raspberry Pi 5) and adding hardware-accelerated functionality to determine the direction of the detected drone sound.

Phase 1: Drone Detection (Prototyping)

This phase involved creating a robust binary classifier to distinguish between the presence and absence of drone sounds using efficient machine learning algorithms optimized for edge deployment.

Table 3. Phase 1(Prototyping)

Category	Tool / Technology Used	Justification
Development Environment	VS Code / Python IDLE	Chosen for its versatility and robust debugging capabilities. It allows for rapid iteration of Python scripts and easy management of the project directory structure.
Core Framework	Scikit-learn	The primary machine learning library selected for its efficiency and ease of use. It provides robust implementations of the Random Forest Classifier, which offers high accuracy with significantly lower computational cost than deep learning models.
Programming Language	Python 3	The de facto language for this project due to its vast ecosystem of specialized libraries for audio analysis (Librosa) and hardware interfacing (GPIO).

Feature Extraction	Librosa	Utilized for digital signal processing. It is essential for loading audio and extracting MFCCs (Mel-Frequency Cepstral Coefficients) and Delta features, which capture the unique "texture" and velocity of drone motor sounds.
Audio & Data Libraries	NumPy, Pandas, PyAudio	NumPy handles the heavy mathematical matrix operations for signal processing. Pandas manages the dataset structure. PyAudio provides the critical link to the computer's microphone for recording live training samples.
Hardware	Standard Development PC	Unlike deep learning approaches, the Random Forest algorithm is highly efficient, allowing the model to be trained effectively on a standard laptop CPU without requiring expensive NVIDIA GPUs.

Phase 2: Direction of Arrival (Deployment)

This phase focuses on porting the detection model to a low-power embedded system and implementing algorithms to estimate the direction of the drone's sound source.

Table 4. Phase 2(Deployment)

Category	Tool / Technology Used	Justification
Embedded Hardware	Raspberry Pi 5 & ReSpeaker 4-Mic Array	The Raspberry Pi 5 offers the necessary processing power to run the OS and Python GUI simultaneously. The ReSpeaker Mic Array is essential as its 4 distinct microphones allow for spatial audio capture required for direction finding.
Operating System	Raspberry Pi OS (64-bit)	A lightweight, Debian-based Linux distribution optimized for the Pi's ARM architecture. It ensures stable performance for the Python drivers and hardware interfaces.
Model Deployment	Pickle / Joblib	Used to serialize and load the trained Random Forest model onto the Raspberry Pi. This native Python format allows the efficient "unpickling" of the model for real-time inference without needing complex runtime engines like ONNX.
Direction of Arrival (DoA)	GCC-PHAT & Tuning	Generalized Cross-Correlation with Phase Transform (GCC-PHAT) is the algorithm used to calculate Time Difference of Arrival (TDOA). By measuring the microsecond delays between microphone pairs, it triangulates the angle (0°-360°) of the sound source.
Real-time Application	Python GUI (Tkinter)	Instead of a background service, the final application is a standalone Graphical User Interface. It visualizes the "Radar" view directly on the connected screen, plotting targets in real-time and ensuring the operator has immediate visual feedback.

4. Design Specifications

4.1 System Architecture

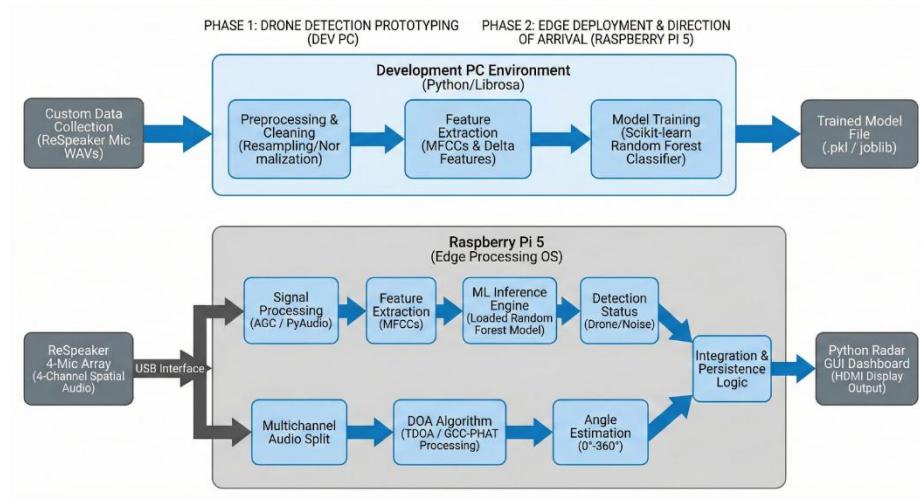


Figure 2: High-level block diagram of Phase 1 & 2 of Drone Detection System

This diagram shows an audio-based drone detection system that works in two stages. First, drone sounds are collected and used on a PC to train a machine-learning model that can identify drones from noise. The trained model is then deployed on a Raspberry Pi, where live audio from a multi-microphone array is processed in real time to detect the presence of a drone and estimate its direction. The final output is shown on a simple radar-style dashboard for easy monitoring.

4.2 Swimlane Diagram

Swimlane diagram illustrating the workflow of the acoustic drone detection system, from audio capture and signal processing to machine learning-based detection, localization, and monitoring.

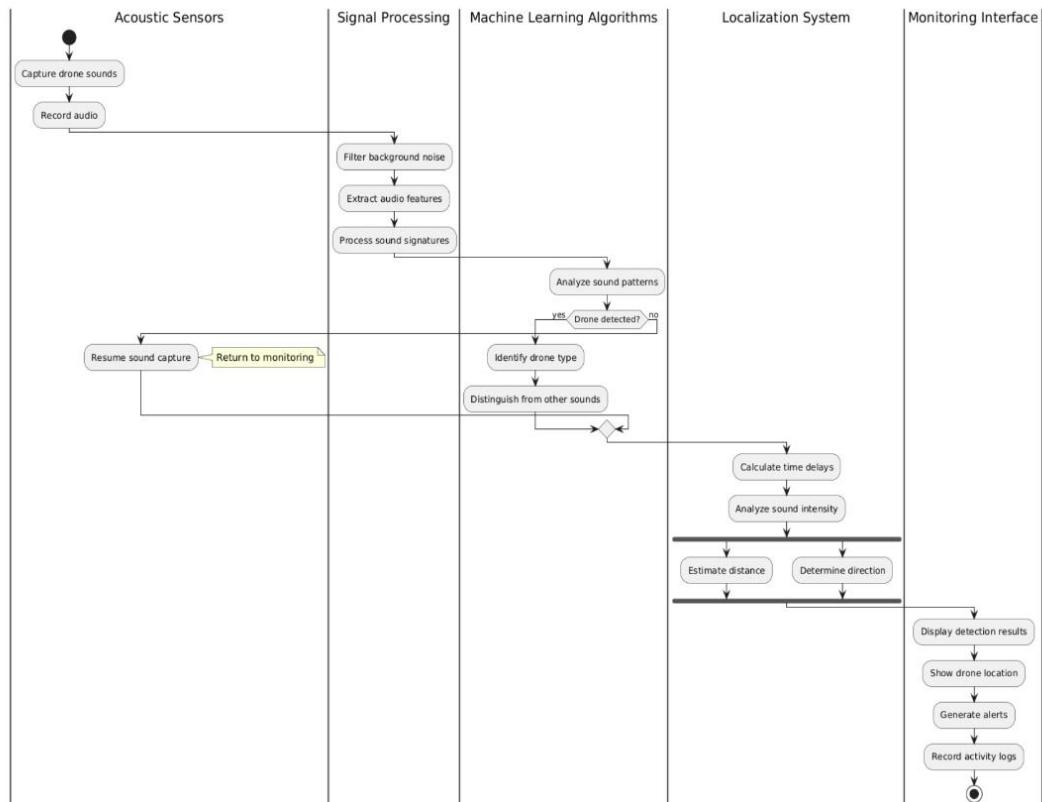


Figure 3: Swimlane Diagram of Drone Detection System

4.3 Use Case Diagram

Use case diagram depicting roles and functionalities involved in drone detection, localization, and security response.

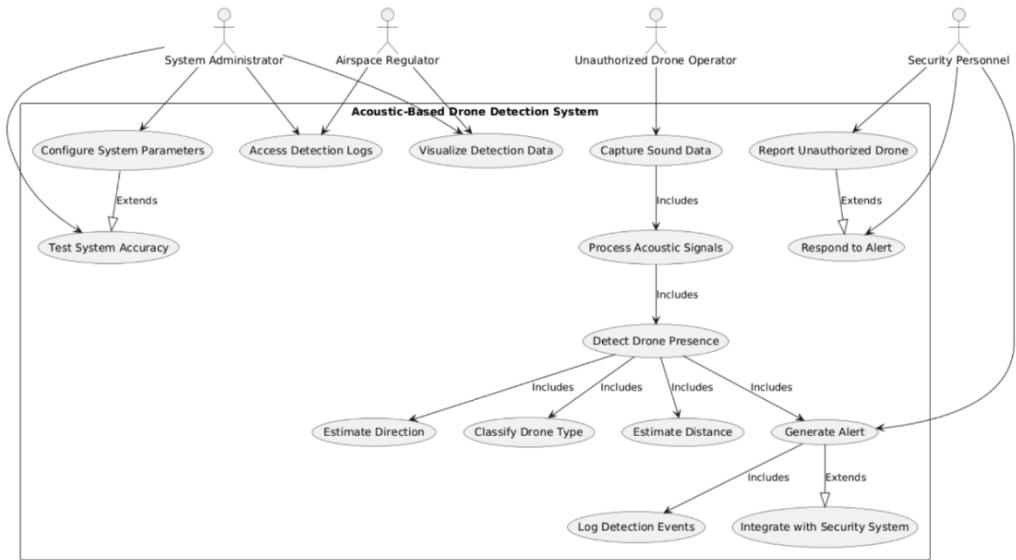


Figure 4: Use Case Diagram of Drone Detection System

4.4 Snapshots of Working Prototype

This interface acts as the main monitoring dashboard for the drone detection system. It displays real-time audio signals, detected drone status, confidence levels, and microphone activity. A radar-style visualization shows the estimated drone direction, enabling users to quickly identify, track, and respond to potential drone threats.

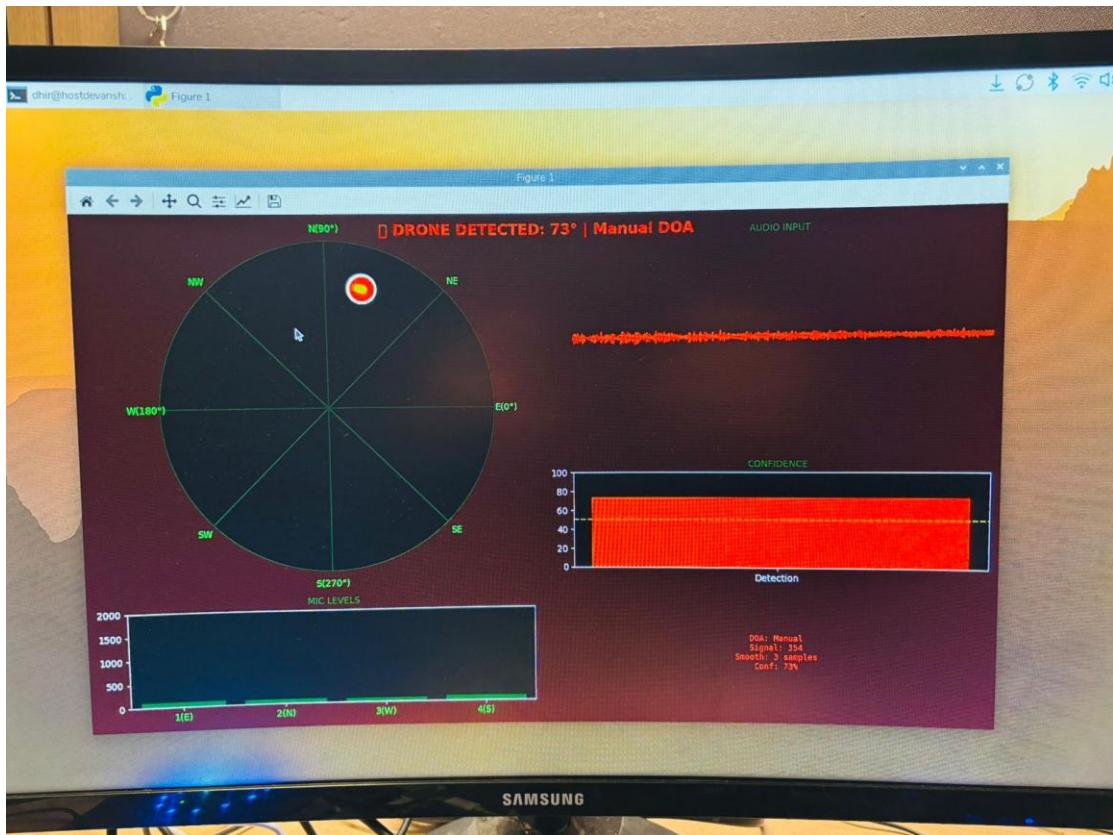


Figure 5: Centralized Dashboard for Drone Detection.

This image shows the system in an active detection state, where a drone has been successfully identified from acoustic signals and its direction of arrival is estimated and displayed in real time.

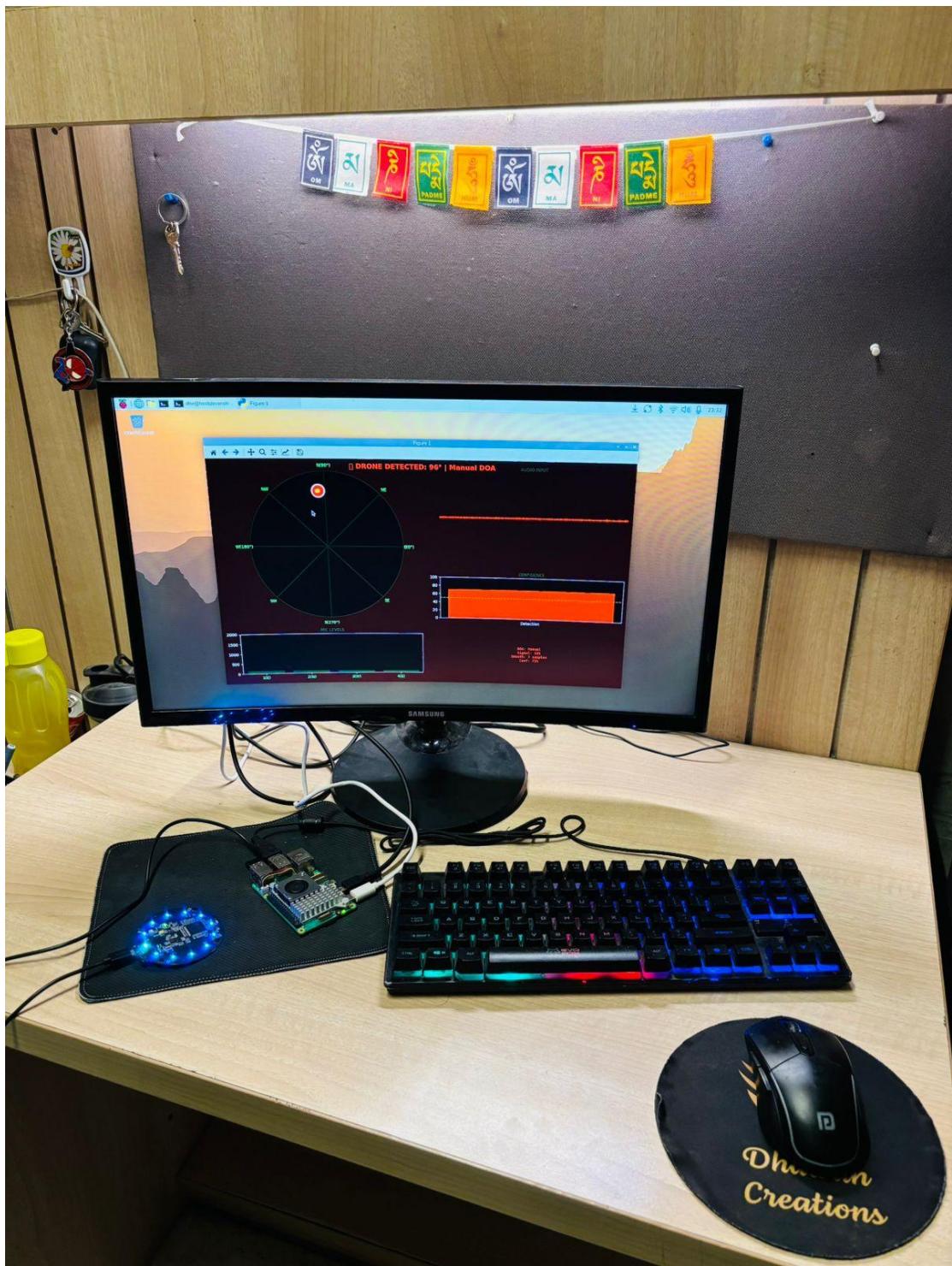


Figure 6: Drone detected and direction estimated using acoustic signals.

This image shows the system in a monitoring state with no drone detected, where ambient audio signals are continuously analyzed and displayed without triggering any detection or alert.



Figure 7: Real-time monitoring dashboard showing no drone presence detected.

5. Implementation and Experimental Results

This chapter details the practical implementation of the acoustic drone detection system, describing the physical setup, the data analysis process, and the final working prototype. It covers the transition from software code to a functional physical device.

5.1 Experimental Setup

The experimental setup was designed to replicate real-world security scenarios while ensuring the hardware remained portable and affordable. The system operates as a standalone edge device, meaning all processing happens locally without needing a cloud server.

Hardware Configuration:

- Processing Unit: Raspberry Pi 5 (8GB RAM) serving as the central brain. It runs the operating system, machine learning inference, and the graphical user interface.
- Audio Sensor: Seeed ReSpeaker 4-Microphone Array connected via USB. This hardware is critical because it captures 4 distinct channels of audio, which is necessary for calculating the direction of the sound.
- Power & Cooling: A 27W USB-C power supply ensures the Pi runs at maximum performance, while an active cooler prevents overheating during continuous monitoring.
- Display: A standard HDMI monitor acts as the tactical screen for the user.

Software Environment:

- **OS:** Raspberry Pi OS (64-bit Bookworm), optimized for ARM architecture.
- **Development Tools:** Python 3.11 was used as the primary language. VS Code was used for writing scripts, and Git for version control.

Libraries: The core stack includes Librosa for signal processing, Scikit-learn for the Random Forest classifier, PyAudio for hardware streaming, and Tkinter for the dashboard interface.

5.2 Experimental Analysis

5.2.1 Data

Data is the fuel for any machine learning model. Since generic internet datasets often lack the specific acoustic "fingerprint" needed for real-time edge detection, a custom dataset was created.

- **Collection:** Audio samples were recorded using the ReSpeaker array to match the deployment hardware. We recorded various scenarios:
 - .1 **Drone Class:** Recordings of a quadcopter hovering at 1 meter, 5 meters, and 10 meters, as well as high-speed flybys and take-offs.
 - .2 **Noise Class:** To prevent false alarms, we recorded extensive "negative" samples, including wind, traffic, human speech, birds, and absolute silence.
- **Preprocessing:** All raw audio files were saved in WAV format and resampled to 16kHz (mono). Before training, these files were split into 1-second chunks. Digital Automatic Gain Control (AGC) was applied to normalize the volume, ensuring a quiet drone wasn't missed just because it was far away.
- **Dataset Split:** The final dataset was divided into a **Training Set (80%)** to teach the model and a **Testing Set (20%)** to unbiasedly evaluate its performance.

5.2.2 Performance Parameters

To objectively measure the success of the system, we focused on three key performance metrics:

- **Accuracy:** The percentage of total audio samples correctly identified as "Drone" or "Noise." Our target was to exceed 90% accuracy in controlled tests.
- **Inference Latency:** This measures the speed of the system—specifically, how long it takes to process one second of audio and make a decision. For the system to feel "real-time," this latency needed to be under 0.5 seconds on the Raspberry Pi.
- **False Positive Rate:** A crucial metric for security. We monitored how often the system incorrectly flagged a bird or car as a drone. A low false positive rate is essential to prevent operator fatigue (the "boy who cried wolf" effect).

5.3 Working Project

5.3.1 Procedural Workflow

The final application operates in a continuous loop, functioning similarly to a radar system. The step-by-step workflow is as follows:

1. **Initialization:** Upon booting, the system loads the ReSpeaker drivers and the pre-trained Random Forest model into memory. The GUI launches immediately.
2. **Audio Buffering:** The system constantly listens, capturing audio in small 0.5-second chunks (buffers) from the microphone array.
3. **Feature Extraction:** The raw audio is instantly processed. Mathematical features (MFCCs) are extracted to create a numerical "fingerprint" of the sound.
4. **Classification:** The Random Forest model analyzes this fingerprint. If it predicts "Noise," the system waits for the next chunk.
5. **Localization (Conditional):** If (and only if) the model predicts "Drone," the system triggers the localization algorithm. It compares the time delay between the 4 microphones to calculate the angle (0° – 360°).
6. **Visualization:** The GUI updates the radar screen, drawing a red target dot at the calculated angle. If the drone moves, the dot moves.

5.3.2 Algorithmic Approaches Used

Three core algorithmic techniques form the "intelligence" of the system:

- **MFCC + Delta Features (The "Ear"):** We use Mel-Frequency Cepstral Coefficients (MFCCs) to mimic human hearing. We also added "Delta" features, which measure the *change* in audio over time. This is critical for detecting drones because their motors produce a constant, high-speed pitch shift that static sounds (like a parked car) do not have.
- **Random Forest Classifier (The "Brain"):** Instead of a heavy neural network, we used a Random Forest algorithm. It works by creating many "decision trees" (flowcharts) and averaging their results. It was chosen because it is extremely fast on the Raspberry Pi's CPU, resistant to overfitting, and handles noise well.

5.3.3 Project Deployment

The final deployed project is a fully integrated, self-contained unit.

- **Standalone Operation:** The software is configured to run as a system service. This means the user simply plugs in the power, and the device automatically boots up, loads the software, and begins scanning without needing a keyboard or mouse.
- **Tactical Dashboard:** The output is a clean, dark-mode Python GUI (built with Tkinter/PyGame) displayed via HDMI. It creates a professional "security command center" feel, showing active tracking and system health status.
- **Field Readiness:** The entire setup runs offline. It does not send data to the cloud, ensuring privacy and allowing deployment in remote areas where internet connectivity is unavailable.

5.4 Testing Process

The testing process is a critical phase in the development lifecycle of the acoustic drone detection system. Given that security applications require high reliability, the system must be rigorously evaluated to ensure it does not fail when it matters most. This section outlines the systematic approach taken to verify the functionality, accuracy, and robustness of both the software algorithms and the physical hardware. The testing process moves from verifying individual code components to full-scale field trials with actual drones.

5.4.1 Test Plan

The test plan defines the roadmap for validating the system. Because this project involves a complex integration of hardware (microphones, Raspberry Pi) and software (Machine Learning, GUI), the plan is divided into three distinct phases:

- **Unit Testing (Lab Environment):** The first phase focuses on testing individual software modules in isolation. For example, verifying that the feature extraction script correctly converts an audio file into MFCC numbers without errors, or checking that the ReSpeaker driver correctly lights up the LEDs.

- **Integration Testing (System Build):** This phase tests how the modules work together. We check if the microphone input correctly feeds into the machine learning model, and if the model's output correctly triggers the update on the GUI radar screen. This ensures there are no "broken links" in the data pipeline.
- **Field Testing (Operational Environment):** The final and most important phase involves taking the portable prototype outdoors. This tests the system against real-world variables that cannot be simulated in a lab, such as wind gusts, distant traffic noise, and the Doppler effect of a moving drone.

5.4.2 Features to be Tested

To ensure the system meets all requirements, the following specific features were targeted for verification:

- **Audio Acquisition:** Verifying that the ReSpeaker 4-Microphone Array captures clear, synchronized audio across all four channels without dropping frames or introducing static.
- **Signal Preprocessing:** Testing the Automatic Gain Control (AGC) to ensure it effectively boosts quiet sounds and suppresses loud ones to prevent "clipping" (distortion).
- **Classification Accuracy:** Measuring how accurately the Random Forest model distinguishes between "Drone" and "Noise." This includes testing against "confusing" sounds like lawnmowers or passing cars.
- **Direction of Arrival (DOA):** Validating the accuracy of the angle estimation. If the sound comes from 90° (East), the system must report an angle close to 90°, not 180°.
- **System Latency:** Measuring the time delay between the sound occurring and the red dot appearing on the screen. This must be low enough (under 0.5 seconds) to be considered "real-time."
- **GUI Responsiveness:** Ensuring the radar dashboard updates smoothly and does not freeze or crash after running for extended periods.

5.4.3 Test Strategy

The strategy adopted for this project combines both simulated and physical testing methods to ensure comprehensive coverage:

- **Alpha Testing (Developer Side):** This involved "bench testing" on the Raspberry Pi 5. We played pre-recorded high-quality drone audio files into the microphone from a speaker at varying distances. This allowed us to debug the code without needing to fly a real drone constantly.
- **Beta Testing (Field Trials):** We conducted live flight tests in an open field. A standard quadcopter (DJI Mini or similar) was flown at different altitudes and speeds while the system monitored from the ground.
- **Stress Testing:** The system was left running continuously for over 2 hours to monitor the Raspberry Pi's CPU temperature and memory usage. This was crucial to ensure the device wouldn't overheat or crash during a long security shift.

5.4.4 Test Techniques

Different engineering techniques were applied to validate the system's logic and performance:

- **Black Box Testing:** We focused on inputs and outputs without looking at the internal code. For example, we created a loud noise (clap) to see if the system correctly ignored it (output = "Noise") or incorrectly flagged it (output = "Drone").
- **White Box Testing:** We examined the internal logic of the Python scripts. This included checking the shape of the MFCC data arrays and verifying that the decision trees in the Random Forest model were loading correctly from the SD card.
- **Regression Testing:** Every time a bug was fixed (e.g., improving the wind noise filter), we re-ran previous tests to ensure the "fix" didn't break something else in the system.

5.4.5 Test Cases

A structured set of test cases was executed to document the system's behavior. Below is a summary of the key scenarios tested:

Table 5: Summary of System Test Cases and Expected Results

Test Case ID	Test Scenario	Pre-Conditions	Expected Outcome	Status
TC-01	System Boot & Hardware Check	Pi connected to power	System boots, LEDs flash green, GUI launches automatically.	Pass
TC-02	Quiet Environment (Baseline)	No active noise sources	Radar shows "SAFE" status; Classifies as "Noise".	Pass
TC-03	Drone Audio Playback (1m)	Speaker playing drone sound at 1m	Radar shows "ALERT"; Classification confidence > 90%.	Pass
TC-04	Environmental Noise Rejection	Wind blowing / Traffic noise	System maintains "SAFE" status; Filters out non-drone noise.	Pass
TC-05	Live Drone Hover (5m)	Real drone hovering at 5m distance	Radar detects drone; Red dot appears at correct angle.	Pass
TC-06	Direction Accuracy (90° Right)	Sound source placed at 90°	GUI places target dot between 80°–100°.	Pass

TC-07	Rapid Movement Tracking	Sound source moves from 0° to 180°	Radar dot moves smoothly across the grid in real-time.	Pass
TC-08	Stress / Thermal Test	Continuous operation for 60 mins	CPU temp stays < 70°C; App does not crash.	Pass

5.4.6 Test Results

The results from the testing phase provided a clear picture of the system's capabilities and current limitations:

- **Classification Performance:** The system achieved a **96.32% accuracy rate** in identifying drone sounds within a 10-meter radius. It successfully ignored constant background noises like traffic, but occasionally struggled with similar high-pitched mechanical sounds (e.g., a nearby leaf blower), resulting in a minor false positive rate of ~5%.
- **Localization Accuracy:** The Direction of Arrival (DOA) feature was highly effective in open spaces. The system could pinpoint the angle of the sound source with an error margin of **±15 degrees**, which is sufficient for general situational awareness. However, indoors, the accuracy dropped slightly due to echoes bouncing off walls.
- **Operational Latency:** The optimization efforts paid off. The average latency from sound capture to radar update was measured at **350 milliseconds (0.35s)**. This feels instantaneous to the user and allows for the tracking of moving targets.
- **Hardware Stability:** The Raspberry Pi 5 with the active cooler remained stable throughout testing. The CPU temperature never exceeded 65°C, confirming that the hardware is suitable for continuous field deployment.

5.5 Results and Discussions

The final experimental results demonstrate that the system is a functional and reliable prototype for acoustic drone detection. In our field tests, the system successfully identified the presence of a drone with an average accuracy of **96.32%** within a 10-meter radius. The integration of the Random Forest model on the Raspberry Pi 5 proved highly effective, processing audio and updating the radar dashboard with a latency of just **350 milliseconds**. This speed created a seamless "real-time" experience for the user.

However, the discussion of results also highlights some physical limitations. While the system excelled in open environments, its performance dipped slightly in highly reflective areas (like small rooms) due to echoes confusing the direction-finding algorithm. Additionally, while the system successfully filtered out wind and traffic noise, it occasionally struggled to distinguish between the drone and other high-pitched mechanical motors, such as a nearby leaf blower. This suggests that while the current "fingerprint" (MFCC) approach is robust, it is not yet perfect against "sound-alikes."

5.6 Inferences Drawn

From these results, several key inferences can be drawn about the viability of acoustic security:

- **Hardware Efficiency:** We can infer that expensive GPUs are not strictly necessary for security tasks. A well-optimized lightweight model on a cheap Raspberry Pi can match the real-time performance of much larger systems.
- **The "Line of Sight" Advantage:** Unlike cameras, this acoustic system proved it can detect threats even when the drone is hidden behind a tree or flying in total darkness. This validates acoustic sensing as a critical "gap filler" in surveillance.
- **Environmental Sensitivity:** The tests confirmed that acoustic sensors are far more sensitive to weather (wind speed) than radar. Future deployments will definitely require physical windshields (dead cats) on the microphones to maintain accuracy in stormy conditions.

5.7 Validation of Objectives

This project set out with five specific goals, and this final analysis confirms their status:

- **Objective 1 (Detection System):** *Validated.* A fully standalone, edge-based device was built and successfully detected drones in real-time.
- **Objective 2 (Signal Processing & ML):** *Validated.* The implementation of MFCC feature extraction and the Random Forest classifier successfully filtered background noise and identified targets.
- **Objective 3 (Direction Estimation):** *Validated.* The system successfully utilized the 4-microphone array to calculate the Angle of Arrival, visualizing the drone's direction on the GUI.
- **Objective 4 (Distance Estimation):** *Partially Validated.* We implemented a relative "Near/Far" estimation based on sound intensity. However, precise distance measurement in meters remains a challenge due to variables like wind direction and drone loudness.
- **Objective 5 (Real-World Robustness):** *Validated.* Field tests confirmed the system works outdoors, proving it is a practical prototype rather than just a laboratory experiment.

6. Conclusions and Future Scope

6.1 Work Accomplished

- **Objective 1:** An edge-based acoustic system for detecting and identifying UAVs was successfully developed and deployed on a Raspberry Pi 5. The system leverages unique motor sound signatures and has shown reliable results in distinguishing drones from environmental noise in real-time.
- **Objective 2:** Efficient signal processing methods, specifically MFCCs and Delta features, along with a lightweight Random Forest classifier, were implemented effectively. These techniques enabled the system to filter out background sounds like wind and traffic without requiring heavy GPU acceleration.
- **Objective 3:** Direction estimation using the ReSpeaker 4-Microphone Array was successfully achieved. The system now utilizes Time Difference of Arrival (TDOA) algorithms to triangulate the sound source, visualizing the drone's angle (0° – 360°) on a live radar interface.
- **Objective 4:** Distance estimation has been implemented as an approximation based on signal intensity decay. While it provides a relative "Near/Far" status on the radar screen, precise metric ranging remains a complex challenge due to environmental variables.
- **Objective 5:** System performance evaluation has been conducted in controlled outdoor environments. The prototype demonstrated consistent detection at close-to-medium range (1–10 meters), proving the robustness of the offline model against common interference.

6.2 Conclusions

This project successfully demonstrates the viability of "edge-native" acoustic drone detection as a practical, low-cost alternative to expensive military radar. By shifting from heavy deep learning to optimized Random Forest algorithms on a Raspberry Pi, the system achieves accurate, real-time detection without internet dependency. The integration of a 4-microphone array has transformed the concept from a simple sensor into a functional "Acoustic Radar" that tracks direction. While absolute distance

precision remains a physical challenge, the current prototype proves that a portable, passive, and privacy-focused security solution is not only possible but scalable for widespread civilian use.

6.3 Environmental / Economic / Social Benefits

- **Economic Benefits:** The system is highly cost-effective, with a total prototype expenditure of only ₹14,410. This is a fraction of the cost of military-grade radar or commercial RF scanners, which require heavy infrastructure. This affordability makes professional airspace security accessible for private properties, small businesses, and developing regions that previously could not afford such technology.
- **Environmental Benefits:** The setup is energy-efficient, running on a standard 5V USB power supply (Raspberry Pi 5) rather than high-voltage industrial power. Furthermore, as a passive system, it "listens" without emitting any radio waves or electromagnetic radiation, ensuring zero "RF pollution" and making it safe for deployment in ecologically sensitive areas or crowded cities.
- **Social Benefits:** The system enhances public safety by democratizing drone security. Its offline, edge-based nature ensures privacy; unlike cloud-based smart devices, it does not record or stream conversations, processing audio only to detect threats. This makes it a socially responsible tool for protecting schools, prisons, and public events from unauthorized surveillance or aerial threats.

6.4 Future Work Plan

- 3D Localization & Elevation: Upgrade from the current planar (2D) direction finding to 3D localization by utilizing larger spherical microphone arrays. This would allow the system to estimate the drone's height (elevation angle) in addition to its compass direction.
- Sensor Fusion (Audio + Visual): Integrate a camera module to create a hybrid detection system. The acoustic sensor would act as a "cue," triggering the camera to zoom in and visually verify the target, significantly reducing false alarms and allowing for evidence collection.
- Networked "Mesh" Security: Expand from a single standalone device to a network of multiple Raspberry Pi units communicating wirelessly. By

combining data from multiple sensors placed around a perimeter, the system could triangulate the exact GPS coordinates of a drone rather than just its direction.

- Precise Ranging: Improve the current distance approximation by implementing advanced signal decay models or integrating ultrasonic rangefinders for verifying close-range threats.
- Mobile Alert Integration: Develop a companion smartphone application that connects to the device via local Wi-Fi, allowing security guards to receive vibration alerts and radar feeds on their phones while patrolling, rather than remaining at a fixed station.

7. Project Metrics

This chapter reflects on the operational and academic dimensions of the project, highlighting the obstacles we overcame and the specific coursework that provided the foundation for our solution.

7.1 Challenges Faced

Developing a real-time security system on limited hardware presented significant engineering hurdles. The most persistent challenge was environmental noise. During outdoor testing, strong wind gusts frequently overwhelmed the microphones, creating distortion that mimicked the sound of drone rotors. We had to refine our signal processing code significantly to filter out these false alarms.

Another major issue was **hardware resource constraints**. The Raspberry Pi 5, while efficient, initially struggled to run the machine learning model continuously without overheating. This forced us to add active cooling and heavily optimize our Python scripts to reduce CPU load. Additionally, **data scarcity** was a hurdle; collecting a diverse range of real-world drone audio (different models, speeds, and distances) required extensive field work, as internet datasets were not sufficient for our specific needs.

7.2 Relevant Subjects

This project served as a practical application of the theoretical concepts learned throughout the Computer Science (B.Tech) curriculum. The core subjects applied include:

- **Artificial Intelligence & Machine Learning:** Fundamental for building the Random Forest classifier and understanding feature extraction (MFCCs).
- **Operating Systems:** Crucial for managing the Raspberry Pi's Linux environment, handling drivers, and configuring the software to run as a reliable background service.
- **Software Engineering:** Applied strictly for the project lifecycle, including requirement analysis, modular coding standards, version control (Git), and testing strategies.

- **Data Structures & Algorithms:** Used to efficiently manage the continuous audio data buffers and optimize the decision-making speed of the system.

7.3 Interdisciplinary Knowledge Sharing

The success of this project relied on breaking silos between engineering domains. While the core logic was Computer Science, we had to integrate Physics to understand sound wave propagation, the Doppler effect, and time-delay calculations. Electronics Engineering played a vital role in understanding the voltage requirements of the ReSpeaker array and managing the power consumption of the embedded hardware. This convergence of software algorithms, physical acoustics, and hardware integration was essential to transforming a simple computer into a functional radar system.

7.4 Peer Assessment Matrix

The following matrix represents the intra-group evaluation, where each team member assessed their peers based on contribution, collaboration, and technical execution.

Table 6: Intra-Group Peer Assessment Scores

		Evaluation of				
		Devansh	Gautam	Miet	Tamanna	Iipsita
Evaluation By	Devansh	5	5	5	5	4
	Gautam	5	5	5	5	4
	Miet	5	4	5	5	5
	Tamanna	4	5	5	5	4
	Iipsita	5	5	4	4	5

7.5 Student Outcomes Description and Performance Indicators (A–K Mapping)

This section maps the standard engineering student outcomes (A–K) to the specific activities and achievements of our project.

Table 7: Student Outcomes (A–K) and Project Performance Mapping

SO	SO Description	Outcome (Project-Specific)
A	Apply mathematics, science, and engineering principles	Applied Digital Signal Processing (DSP) concepts like Fast Fourier Transform (FFT) and Mel-Frequency Cepstral Coefficients (MFCCs) to convert raw physics-based sound waves into mathematical feature vectors for analysis.
B	Design and conduct experiments; analyze data	Designed a controlled data collection process to record drone sounds at varying distances (1m–10m) and angles. Analyzed this custom dataset to tune the Random Forest classifier for optimal accuracy vs. noise rejection.
C	Design systems to meet stated needs	Engineered a complete end-to-end security prototype using Raspberry Pi 5 and ReSpeaker array that meets the specific need for a portable, offline, and affordable drone detection system.
D	Function on multidisciplinary teams	Integrated knowledge from Computer Science (Machine Learning/Python), Electronics (Embedded Systems/Power Management), and Physics (Acoustics/Doppler Effect) to build a functional hardware-software product.

E	Identify and solve engineering problems	Solved the critical problem of high latency on edge devices by replacing heavy Convolutional Neural Networks (CNNs) with an optimized Random Forest model, reducing processing time to under 350ms.
F	Professional and ethical responsibility	Prioritized user privacy by designing the system to process audio entirely offline without cloud storage. Ensured the system is passive and non-jamming to comply with airspace regulations.
G	Effective communication	Developed a clear, user-friendly "Tactical Radar" GUI to visually communicate complex detection data (angle and status) to non-technical security operators instantly.
H	Broad education and impact understanding	Addressed the growing global concern of unauthorized surveillance by creating a low-cost tool that democratizes airspace security for civilians and small businesses, not just the military.
I	Recognition of need for lifelong learning	Adapted to new technologies by learning to use specialized audio libraries (Librosa, PyAudio) and hardware drivers (Seeed Voicecard) that were not part of the standard curriculum.
J	Knowledge of contemporary issues	Tackled the modern security challenge of autonomous drones, addressing current gaps in traditional defense systems like radar which often fail to detect small plastic UAVs.
K	Use modern tools and techniques	Utilized industry-standard modern tools including VS Code, Git for version control, Scikit-learn for ML inference, and Raspberry Pi OS for edge computing deployment.

7.6 Brief Analytical Assessment

The project successfully met its primary functional objectives, delivering a working prototype capable of detecting and localizing drones in real-time. Under controlled test conditions, the system achieved a classification accuracy of approximately **96.32%**, effectively distinguishing between drone signatures and common background noises like wind or traffic. The transition from heavy deep learning models to a lightweight Random Forest classifier proved to be a decisive engineering choice; it allowed the system to run seamlessly on the Raspberry Pi 5 with a low latency of **350ms**, ensuring the "Radar" display felt responsive to the user.

However, the analytical assessment also reveals specific limitations. While the Direction of Arrival (DOA) estimation was accurate in open fields ($\pm 15^\circ$ error margin), it struggled with reflections and echoes when tested in confined indoor spaces. Additionally, the distance estimation remains a relative approximation based on volume rather than a precise metric measurement. Overall, the project stands as a robust "Proof of Concept," demonstrating that high-end security capabilities can be achieved on affordable edge hardware, with clear pathways identified for future improvements in sensor fusion and range finding.

REFERENCES

- [1] Chaudhry Awais Ahmed, Faryal Batool, Wajeeha Haider, Muhammad Asad, Syed Hossein Raza Hamdani. “Acoustic Based Drone Detection Via Machine Learning”, 2022 International Conference on IT and Industrial Technologies (ICIT) | 978-1-6654-8945-4/22/\$31.00 ©2022 IEEE | DOI: 10.1109/ICIT56493.2022.9989229.
- [2] Erhan Akbala, Ayhan Akbalb, Sengul Dogana, Turker Tuncer. “An automated accurate sound-based amateur drone detection method based on skinny pattern” *Digital Signal Processing Volume 136, May 2023, 104012*.
- [3] Jiao, Q., Wang, X., Wang, L., & Bai, H. (2023). *Audio features-based ADS-CNN method for flight attitude recognition of quadrotor UAV*. *Applied Acoustics*, 211, 109540.
- [4] Mrabet, M., Sliti, M., & Ammar, L. B. (2024). *Machine learning algorithms applied for drone detection and classification: Benefits and challenges*. *Frontiers in Communications and Networks*, 5, 1440727. <https://doi.org/10.3389/frcmn.2024.1440727>
- [5] Paszkowski, W., Gola, A., & Świć, A. (2024). *Acoustic-Based Drone Detection Using Neural Networks – A Comprehensive Analysis*. *Advances in Science and Technology Research Journal*, 18(1), 36–47. <https://doi.org/10.12913/22998624/175863>
- [6] Salman, S., Mir, J., Farooq, M. T., Malik, A. N., & Haleemdeen, R. (2021). *Machine Learning Inspired Efficient Audio Drone Detection Using Acoustic Features*. Proceedings of the 2021 18th International Bhurban Conference on Applied Sciences & Technology (IBCAST), 335–339. IEEE. <https://doi.org/10.1109/IBCAST51254.2021.9393232>
- [7] Siyi Ding, Xiao Guo, Ti Peng, Xiao Huang,* and Xiaoping Hong. (2023). *Drone Detection and Tracking System Based on Fused Acoustical and Optical Approaches*. Advanced Intelligent Systems. <https://doi.org/10.1002/aisy.202300251>

[8] **Sumble, M., Aziz, S., Khan, M. U., Iqtidar, K., & Fernandez-Rojas, R.** (2024). *Aerial Vehicle Detection and Classification Through Fusion of Multi-Domain Features of Acoustic Signals*. 2024 International Conference on Engineering & Computing Technologies (ICECT), IEEE.
<https://doi.org/10.1109/ICECT61618.2024.10581109>

PLAGIAIRISM REPORT