# ENHANCING AUTONOMOUS VEHICLE SECURITY THROUGH MACHINE LEARNING-BASED GPS SPOOFING DETECTION

Project ID: 24-25J-140

BSc. (Hons) in Information Technology Specializing in Cybersecurity

Department of Computer Systems Engineering

Sri Lanka Institute of Information Technology Sri Lanka

**Final Report**

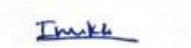
W.M.I.W Wanigasekara – IT21249648

April 2025

# DECLARATION

I declare that this is my own work, and this dissertation does not incorporate without acknowledgement any material previously submitted for a degree or Diploma in any other University or institute of higher learning, and to the best of my knowledge and belief, it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to the Sri Lanka Institute of Information Technology the nonexclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic, or another medium. I retain the right to use this content in whole or part in future works (such as articles or books).

| NAME | STUDENT ID | SIGNATURE |
|---|---|---|
| W.M.I.W Wanigasekara | IT21249648 |  |

Name of supervisor: Mr. Kavinga Abeywardena

Name of co-supervisor: Ms. Hansika Mahaadikara

The above candidate has carried out research for the bachelor's degree dissertation under my supervision.

Signature: Date:

Supervisor:

Signature: Date:

Co-supervisor:

Signature: Date:

# ABSTRACT

This work presents the design and development of a GPS spoofing detection system aimed at promoting better security and reliability in autonomous navigation. This research intends to integrate the machine learning techniques into a proactive and real-time
detection framework against the increasingly dangerous spoilers of an advanced GPS
landscape. The proposed architecture is a rover IoT based application-an Arduino Mega embedded rover equipped with a GPS module, IMU sensors, and a Wi-Fi module-used for data collection of GPS coordinates, speed, and trajectory along pre-programmed
routes under different conditions. The dataset thus generated is utilized by a machine
learning model that is trained to learn the expected paths and movement patterns of the rover. This learning enables efficient detection of anomalous GPS signals. While
operating in real-time, it gets trained by following the trained routes autonomously and preventing itself from staying on false paths when it detects spoofing and then generates alerts. Moreover, to boost user interaction, such spooling logs and notifications are seen via a mobile application in real-time, improving situation awareness. The testing shows that the system can reach great accuracy in identifying spoofed paths with reliable
performance and a very low false-positive rate, and it makes it suitable for autonomous
vehicle applications. Some challenge areas for future optimization include the limitations in computational capacity on the Arduino Mega and power consumption. The future
recommendations are improving the adaptability of the model to emerging spoofing and the models improving energy efficiency to sustain real-time operation for long. This
research is based on a cost-effective scalable solution in improving GPS security, safety, and, above all, the assurance credibility that they can help to provide to several industries that rely on GPS navigation, by avoiding expensive and legally restricted tools such as

HackRF.

*Keywords: GPS spoofing, autonomous, machine learning, robust real-time detection, hybrid approach, cost-effective, computational*

# ACKNOWLEDGMENT

First of all, I would like to express my deep gratitude to those individuals and institutions that have made this research project possible. Above all, I want to thank my supervisor for being such an enlightening guide through the research and writing phases of the project, which greatly helped in making this project feasible. Your mentorship provided clarity in seeking a path through complex hurdles to achieve your goals related to this project.

I am also thankful to the co-supervisor for his very valuable comments and continuous encouragement, which really helped shape the research methodologies and added value to this work. Your attention to detail and practical insights have been helpful.

I would also like to extend my appreciation to the departmental staff for administrative support and for access to necessary resources and tools in the conduct of this research. My thanks go particularly to the technical team that supported me during data collection and in integrating the technology applied in this project.

Lastly, I also want to thank my friends and colleagues because through discussions and sharing experiences, it actually created an enabling learning environment that enriched the development of this research. The support and encouragement have kept me going through this journey.

I thank everyone who was involved in the successful execution of this project.

# TABLE OF CONTENTS

Contents

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVIATIONS

| Abbreviation | Full Form |
|---|---|
| GPS | Global Positioning System |
| AV | Autonomous Vehicle |
| PNT | Position, Navigation, and Timing |
| ML | Machine Learning |
| SDLC | Software Development Life Cycle |
| TDOA | Time Difference of Arrival |
| GNSS | Global Navigation Satellite System |
| IDE | Integrated Development Environment |
| SVM | Support Vector Machine |
| UAV | Unmanned Aerial Vehicle |
| IoT | Internet of Things |
| REST | Representational State Transfer |
| LSTM | Long Short-Term Memory |
| LKR | Sri Lankan Rupee |
| OEM | Original Equipment Manufacturer |
| SD | Secure Digital |
| PP | Page/Page Number |
| TAF | Technical Approval Form |

Table 0-1. List of Abbreviation

# 1. INTRODUCTION

## 1.1. Research Background

To this end, GPS has become an important technology in modern-day applications, from simple navigations and location-based services to autonomous vehicle operations. While appreciable strides have been advanced in efficiency and autonomy in which the integration of GPS into critical systems have been undertaken in rapid fashion, on the other hand, this increased reliance allows such systems to be vulnerable to certain threats, one of the most relevant being spoofing attacks on GPS. These are attacks that involve the transmission of fake GPS signals, which can thereby deceive a navigation system into interpreting erroneous positioning data. The outcome might turn fatal with probable misrouting of vehicles, resultant accidents, or unauthorized entry into restricted areas.

### 1.1.1. Significance of GPS in Autonomous Systems

GPS technology provides the critical PNT information to autonomous vehicles. Due to its potential for real-time geolocation and time synchronization, this technology remains at the heart of navigation within self-driving cars, drones, and other computerized systems. In particular, autonomous vehicles rely on GPS to coordinate their movement in order to avoid obstacles and trace the right path along the pre-set trajectory. If reliable GPS data inputs are not provided, it is impossible for these vehicles to maintain safety and efficiency in their operations [1].

### 1.1.2. Spoofing GPS: An Emerging Threat

The use of GPS thus brings along the threats of cyberattacks in the forms of spoofing, jamming, and interference of signals. The most peculiar deceptive method of GPS spoofing defines a technique adopted by the attackers, which emanates sham GPS signals to deceive a receiver into using wrong information. A GPS spoofing attack can thus be attributed to a wide range of operational and safety consequences. In the case of autonomous vehicles, GPS spoofing can lead to navigation errors. The vehicles could get off their intended routes or just stop operating [2].

Through research, it has been found out that GPS signals are weak, unencrypted, and therefore susceptible to spoofing. Unlike signal jamming, which just interferes with the GPS operations, spoofing can subliminally threaten GPS data to make it look real to the

receiver. That is why the detection is pretty difficult because the attacked system may not detect that it is under attack.

### 1.1.3. Current Detection Approaches

Many techniques have been researched to detect and mitigate GPS spoofing, such as signal strength analysis, time difference of arrival, and angle of arrival. While these approaches often work, in general they require additional hardware and cannot be easily modified to respond to new and sophisticated spoofing methods. As spoofing methods continue to become more and more sophisticated, the detection methods need to be increasingly adaptive and scalable [3].

Machine learning has indeed become one of the most promising methods for detecting GPS spoofing, since it can analyze big volumes of data in search of tiny patterns that could testify to spoofing. This might be empowered by machine learning algorithms training the system to identify sudden changes of coordinates, abnormal signal strength, and gaps in time stamps. Machine learning algorithms, such as SVM, RNN, and CNN, seem promising in the detection of anomalies in several domains.

### 1.1.4. Detection of GPS Spoofing by Machine Learning-Based Solutions

The applications of machine learning in cybersecurity range from intrusion detection to anomaly detection and have been ever-expanding in the past years. For GPS spoofing detection, training by the ML models may be done on datasets containing legitimate and spoiled GPS signals. These can then analytically exploit these spatial and temporal data in order to learn how to distinguish between real and compromised signals [4].

The SVM is a strong classification algorithm that may be also used in spoofing detection from the GPS feature vectors. RNNs are ideal in temporal data analysis because they can detect patterns over time that could imply spoofing attempts. CNN typically analyzes spatial data, which may be modified to detect spatial anomalies in GPS signals.

### 1.1.5. Real-time Detection and Alert Systems

The GPS spoofing detection system should be able to work in real time for the practical deployment of autonomous vehicles in response to different threats. This research integrates such a system into an IoT-based rover, implemented using an Arduino Mega microcontroller, GPS module, IMU sensors, and Wi-Fi module; hence, it is scalable and

cost-effective. This compact and powered design of Arduino Mega makes it applicable for embedding into vehicle systems where space issues and costs are critical.

To increase usability and accessibility, a mobile application was developed to provide real-time alerts. Upon detection of and after confirming any case of GPS spoofing-relative to known paths or possibly other unexpected trajectories, the rover would not allow navigation toward the detected wrong pathway and generate an immediate alert. Vehicle operators or connected systems could potentially act against reported threats proactively. It provides an interface for the user where real-time notifications, spoof detection logs, and GPS status updates are found, ensuring transparency and facilitating prompt threat handling [5]. This model thus guarantees that the system not only detects anomalies but also addresses the protection of autonomous navigation while remaining available for practical deployment.

### 1.1.6. Project Methodology Overview

The effective GPS Spoofing Discovery System is a multi-phased approach formulated to enhance autonomous navigation security. The phases are as follows:

**Data Collection:** Collecting real-time authentic GPS and trajectory data using an IoT-based rover equipped with an Arduino Mega, GPS module, IMU sensors, and WiFi module, where the rover is driven along specific routes (A to B, B to C) under various conditions. Spoofed simulated data are generated in a controlled manner to avoid legal restrictions on transmitting signals.

**Data Preprocessing**: Cleaning the dataset by removing invalid and incomplete records, normalizing the relevant coordinates and speed for consistency, and identifying important indicators of spoofing behavior, such as deviation in trajectory or unexpected coordinate shifts.

**Model Selection and Training:** Various machine-learning models such as SVM, RNNs, CNNs, and hybrid approaches are evaluated for GPS spoofing detection. The selected model—trained on the preprocessed dataset—learns the expected paths and movement patterns of the rover using supervised learning techniques. Cross-validation and hyperparameter tuning are applied for improving robustness and detection accuracy.

**Deployment:** That implementation of the trained model on the Arduino Mega onboard the rover so that GPS and IMU data can be processed in real-time. While the rover navigates

autonomously, the system picks out anomalies, prevents it from following incorrect paths, and activates alerts when it senses spoofing in the primary signals.

**Mobile App Development**: A mobile application has been developed so that the user gets alerts promptly. It is attached through WiFi to the rover and gives real-time alerts and visualization of GPS and trajectory data, shows the status of the whole system, and logs the spoofed incidents to improve awareness and interactions of users with the system.

### 1.1.7. Challenges and Future Prospects

One of the key challenges in GPS spoofing detection system developments is the tradeoff between the accuracy of detection and computational efficiency. High false alarms might cause unnecessary alerts, while on the other hand, it could also lead to missed spoofing attempts. Hybrid methods that combine machine learning with traditional methods for more robust systems can be adopted in future works. Advances in edge computing and higher performance embedded processors are likely to yield improvements in real-time performance.

Machine learning applied for GPS spoofing detection in autonomous vehicles provides an efficient, scalable solution to improve vehicle security. The development of on-board detection systems, together with their real-time notifications and interfaces, will contribute to making the navigation system not only safer but also more resilient. This research effort probably may provide a basis for the future undertaking of exploration and improvement with the aim of protecting autonomous systems against cyber threats at large.

## 1.2. Research Scope

In this research, some concepts have been devised and implemented to make a machine learning-based system that detects spoofing under the conditions of autonomous navigation. The research uses IoT-based rover as the potential platform for such use. For research purposes, there will be an investigation to integrate a cheap embedded system of Arduino mega, GPS module, IMU sensors, and Wi-Fi module for data accumulation along predefined routes, training a machine-learning model to recognize expected paths and movement patterns, and real-time detection of anomalies in GPS data. The range of the application includes deploying this trained model on the rover so that it can navigate on its own, with the only difference that it can determine that there is no movement during the time of spoofing and alert a user by a mobile application for better interaction.

## 1.3. Target Audience

Target groups that will most profit from this research are professionals dealing with autonomous vehicle development, cybersecurity experts, and stakeholders with interests in advanced vehicle technologies and transportation security. All results derived will be of interest to automotive manufacturers and researchers dealing with improving vehicular safety and navigation reliability. "Insights related to the implementation of the system could come helpful for engineers and system architects while designing secure GPS-dependent navigation frameworks in autonomous vehicles.

The contribution of machine learning techniques in this study would also be very useful for cybersecurity researchers and practitioners who are eager to learn more recent approaches toward threat detection and mitigation. Some of the proposals in the new approaches could involve using the fusion of RNNs and CNNs in detecting GPS spoofing attacks in real time, as well as further exploring and adapting the new approach into other security domains.

It will also be useful to regulatory bodies and policymakers of transport and public safety who may apply the findings in the establishment of safety guidelines and standards regarding the use of GPS-based systems in public and private transportation. Finally, this research can be used by educational institutions and their students focused on cybersecurity, data science, and automotive studies for case studies that can help them further improve in knowledge regarding how machine learning and vehicular security are related.

## 1.4. Literature Review

### 1.4.1. Introduction to GPS Technology and Its Applications

This section will address the rapid utilization of GPS applications, which range from everyday navigation to autonomous vehicle control. GPS has become one of the most robust tools in providing PNT information to many of the modern technological developments. Reliance upon GPS, however, has opened systems to a range of vulnerabilities, not least those concerning cybersecurity. GPS reliability is crucial in industries such as transport, military operations, and emergency services [6]. Besides, the integration of GPS into autonomous vehicle systems brings unparalleled efficiency and innovation but the very dependency naturally comes with vulnerabilities to be exploited, such as GPS spoofing.

### 1.4.2. Nature and Impact of GPS Spoofing

GPS spoofing involves the attacker broadcasting fake signals, which seem like the real GPS signals to the receiver; in that way, the receiver is tricked into accepting the fake as true data. Unlike GPS jamming, which jams the signal entirely, spoofing is relatively subtle and frequently more difficult to detect, since a victim system perceives the counterfeit signals as legitimate. GPS spoofing can have far-reaching implications, from simple navigation errors to loss of system function to unauthorized access in restricted areas, including even the accidents that can happen with the involvement of autonomous vehicles [7].

It becomes a more significant problem in autonomous vehicles, where the received navigation information is terribly critical to safe navigation. In several instances, spoofing has been able to mislead route directions to hazardous or unforeseen results. This has heightened the urgent need for effective detection and mitigation mechanisms if GPS-based systems are to be reliable and safe.

### 1.4.3. Traditional Detection Methods

The early detection of GPS spoofing depended on signal analysis methods that included a comparison between the signal strengths, angle of arrival, and time difference of arrival. Signal strength comparison involves testing the coherence of incoming signals with expected values; thus, in case abrupt and unexplained variations are observed, the system can raise an alarm [8]. Angle of arrival measurement deals with the direction under which GPS signals are received. Real GPS signals tend to result from multiple satellites; therefore, a pattern would be reached harmonically. Spoofed signals usually emanate from a single source. Time Difference of Arrival employs synchronized signals from multiple sources to determine an estimated discrepancy that may indicate spoofing.

Though these methods might successfully spot some spoofing attempts, most of them involve extra hardware and computational burden. They are poorly scalable, and there is a good chance that they will not adapt to more advanced and novel spoofing methods that will evolve over time.

### 1.4.4. Machine Learning in Anomaly Detection

Experts and Engineers for the Development and Engineering of Autonomous Vehicles: This study is organized in such an innovative manner, which is useful for the autonomous system building professionals, including rovers, drones, and self-driving cars, working technology development. The innovative, cost-effective IoT-based rover that equips an Arduino Mega is showing a very practical solution that can be scaled and integrated into navigation frameworks to improve security and reliability by spoofing detection against GPS.

Cybersecurity Researchers and Practitioners: The posture of experts in cybersecurity against means of cyber threats directed against the most critical infrastructures. This would also be useful to them in their investigation using the machine learning-based detection methodology. Besides, through this work, real-time anomaly detection and prevention with mobile app alerts will also help in building knowledge on GPS spoofing detection and adaptation against newer techniques and their practices.

Transportation and Logistics Stakeholders: The research is also a valuable resource for assuring safety in operations because it would help most companies that depend on GPS for fleet management, logistics, or industrial automation to improve their operations. Given the low cost and portability of the system, it is hence widely accessible for safety enhancement for GPS-dependent vehicles, from little rovers up through massive transportation systems.

Academic and Student Community: It offers real life case study applications of learning for computer science, robotics, data science, and cybersecurity students and academicians. It provides a case study that links machine learning with embedded systems around security in navigation, which can be used for examples in hands-on learning and further exploration.

Regulatory Authorities and Policy Makers: Findings from this study can be useful in crafting guidelines and regulations about GPS-based systems by transport safety agencies and those responsible for cybersecurity standards. The research scantly therefore lays the standard on the basis of which the need for strong spoof detection in the autonomous vehicles would be categorized while also gauging public and private transport security.

### 1.4.5. Real-Time GPS Spoofing Detection Systems

Real-time GPS spoofing detection systems offer excellent opportunities in improving the navigation defense, albeit carrying daunting challenges-the application of machine learning models in real-time implementation therefore calls for a careful trade-off between the level of detection accuracy and computational efficiency especially on constrained platforms. This is going to be well demonstrated through an IoT-based rover-equipped with an Arduino Mega, GPS module, IMU sensors, and Wi-Fi module-for very practical and cost-effective deployment of this kind of system. The Arduino Mega provides enough processing power to run trained machine learning models and integrates seamlessly in autonomous navigation setups due to its small size and affordability [13].

Research into deploying machine learning models on embedded systems, such as that being scrutinized in the Arduino Mega space, shows that real-time data processing and analyzing tasks could be effectively performed within hardware limitations. An on the fly processing of GPS and IMU data generates spoofer anomaly detection and builds a barrier against wrong navigation to ascertain the viability of this model. However, the challenge of the optimization of models with respect to speed and efficiency remains, as the computation power of the Arduino Mega does not allow for frivolous algorithms and leaves resource management in such a way that ensures low queuing time even with dynamic fluctuation in the conditions.

### 1.4.6. Challenges in GPS Spoofing Detection

One of the central problems in GPS spoofing detection is to ensure a low false positive and low false negative rate. High false positive rates imply overproduction of warnings, making systems less dependable and trusted. Conversely, false negatives will let spoofing attempts go through, possibly seriously compromising safety. This can be resolved by fine-tuning machine learning models and considering ensemble techniques to leverage the merits of multiple algorithms [14].

Another challenge is the collection of high-quality labeled data to train the machine learning models. While it is easy to collect real GPS data, obtaining spoofed data may be trickier, since, for safety reasons, testing spoofing is usually done in controlled environments. While simulated datasets are created to help with this issue, they may fall short in completely capturing what happens in the wild, hence possibly degrading the model's performance once it is use.

### 1.4.7. Model Selection and Optimization

The choice of machine learning models forms the most important determinant as to how successful the GPS spoofing detection systems can be. Support Vector Machines offer simplicity and high accuracy for smaller datasets but may struggle with scalability. RNNs, especially the LSTM networks, are better suited to temporal pattern capture when the dataset size is larger, thereby making them a strong candidate for real-time analysis [15]. Complementing the RNNs using CNNs could allow targeting the spatial features as well, reaching a multi-dimensional approach toward anomaly detection.

For this again, hyperparameter tuning and cross-validation are a must. Methods such as grid search or randomized search may be used to find the best setting of model parameters that will further improve performance and prevent overfitting. The generalization of models can be improved by techniques of regularization, which further ensures that the model does well on new and unseen data.

### 1.4.8. Integrating logs into mobile application for Better User Interaction

For one, the integration of mobile applications with GPS spoofing detection systems is an efficient way to extend the usability and user activity of the latter. The real-time system alerts and updates also provide users with a chance to act in time in case spoofing is happening. Showing detailed logs and GPS data will raise the level of transparency and trust in these applications among users [16].

That mobile application was developed using a lightweight framework (such as Flutter or React Native) to ensure immediate cross-platform compatibility. The application communicates with the rover's Arduino Mega via the WiFi module for real-time transfer of spoof detection logs received and alerts from the embedded system. The app features an intuitive interface that offers visual information such as maps, status indicators, and alert notifications, thus giving a helicopter view of the rover's operations and defined spoofing attempts. This approach enhances situational awareness and facilitates a proactive stance toward any navigation security.

### 1.4.9. Directions for Future Work on GPS Spoofing Detection

GPS spoofing detection remains one of the fast-moving areas, reported to be developing on several fronts: machine learning, embedded systems, and cybersecurity strategies. Other work could be done with hybrid detection systems that integrate traditional signal

processing techniques with machine learning for more robust detection. This involves the usage of differential GPS-that is, comparing GPS data from various sources in order to reinforce detection by way of consistency in the data [17].

With the development of microprocessor technology, there is a development in the ways of performance improvements in an edge computing real-time detection system. More powerful and energy-efficient processors can perform complex machine learning models with minor latency, hence enhancing the feasibility to deploy such systems in an autonomous vehicle.

This increasingly applies to the uprise of GPS spoofing detection systems. Ensuring that user data remains secure without hindering effective monitoring is a challenge. For example, sensitive user information can be protected through methods like differential privacy when training machine learning models.

### 1.4.10. Summary

GPS spoofing constitutes a vital threat to the safe operation of GPS-dependent systems, particularly for autonomous vehicles, as they can mislead the navigation with false signals. Conventional detection techniques are usually not scalable or adaptable to the new and emerging spoofing techniques. Machine learning, being one umbrella, tends to be very effective by scrutinizing massive datasets to find subtle patterns for distinguishing abnormal behavior indicating a spoofing effect. This research presents a unique GPS spoofing detection system hooked into an IoT-based rover consisting of an Arduino Mega, GPS Module, IMU sensors, and Wi-Fi Module. The rover collects information along pre-planned paths in order to train a machine learning model to recognize expected paths and movement patterns. During real-time autonomous navigation, the model detects abnormal activities from the GPS data to prevent the rover from following the incorrect paths and making alerts. The interaction along the way with the mobile application carries enhancements: among others, providing real-time notifications to users and spoof detection logs to increase security and situational awareness. Using widely available hardware rather than tools such as HackRF that may be restricted by law to simulate spoofing makes the solution scalable and adaptable. It requires continuous research and optimization to be reliable and counter new cyber threats.

## 1.5. Research Gap

This analysis has, therefore, brought out the research gap that points to the deficiencies of the existing methods for GPS spoofing detection. While traditional approaches, as well as those using machine learning, have stridden ahead in the area of anomaly detection, they lack many essential features such as real-time processing, integrated temporal and spatial data processing, and user-oriented alert mechanisms. The integration of advanced machine learning with real-time adaptability and user interface features in the proposed approach therefore provides a complete and robust solution to GPS spoofing detection.

Current research review indicates some serious lacunars in GPS spoofing detection techniques. Traditional methods of signal analysis allow the most basic forms of detection,

| Features / Study | Real-Time Processing | Temporal Data Analysis | Spatial Data Analysis | Machine Learning Integration | Mobile Application for User Alerts | Adaptability to Evolving Spoofing Techniques |
|---|---|---|---|---|---|---|
| **Study 1: Traditional Signal Analysis** | | ✓ | | ✓ | ✓ | |
| **Study 2: SVM for GPS Anomaly Detection** | | | | ✓ | | |
| **Study 3: RNN for Temporal Analysis** | ✓ | ✓ | | ✓ | | |
| **Study 4: CNN Based Spatial Detection** | ✓ | | ✓ | ✓ | | ✓ |
| **Study 5: Real- Time Hybrid Detection** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **Study 6: Proposed Approach** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Table 1-1. Research Gap Analysis

but they don't scale well in runtime adaptability and higher-level data analytics. Machine learning-based approaches with SVM, RNN, and CNN introduce enhanced detection against anomaly but usually miss an integral part, either with the combination of temporal and spatial data analysis or user warning systems. This proposed approach, therefore, bridges these gaps by providing a robust real-time solution that is integrated with machine learning, mobile user alerts, and able to adapt to evolving spoofing techniques.

Study 1: Detection of GPS Spoofing Attacks by Signal Strength and Direction [18]:

- "A GNSS Spoofing Detection and Direction-Finding Method Based on Low-Cost Commercial GNSS Board Components" (2023)
- This paper describes a spoofing detection and direction-finding technique using low-cost commercial GNSS board components with improvements over unstable phase centers and multipath environment issues.

2. Anomaly Detection in GPS Data Using Support Vector Machines (SVM) [19]:

- "Anomaly Detection in Vessel Tracking Using Support Vector Machines (SVMs)" (2014)
- This paper focuses on the use of Support Vector Machines for the identification of vessels behaving anomalously using Automatic Identification Systems.

3. Recurrent Neural Networks for Temporal Analysis [20]:

- "Deep Learning Detection of GPS Spoofing" (2022)
- This work probes into deep neural networks, including RNNs for the detection of GPS spoofing attacks, and also mentions their further usability in intrusion detection systems.

4. Application of Convolutional Neural Network for Spatial Anomaly Detection in GPS Signals [3]:

- "Dynamic Selection Techniques for Detecting GPS Spoofing Attacks on UAVs" (2022)
- The paper proposes dynamic-based selection methods that include CNNs among other classifiers for detecting GPS spoofing attacks on UAVs.

5. Real-Time Hybrid Approach for GPS Spoofing Detection: Fusion of Temporal and Spatial Features [21]:

- "A Sensor Fusion-based GNSS Spoofing Attack Detection Framework for Autonomous Vehicles" (2021)
- It proposes a framework which fuses temporal and spatial features to realize real-time GNSS spoofing attack detection via sensor fusion.

## 1.6. Research Problem Statement

With increasing reliance on GPS navigation in autonomous cars, such systems are more vulnerable to some serious types of cyberattacks: the GPS spoofing attack. The simplest kind of an attack involves tricking the car navigation by broadcasting fake GPS signals, which can lead to misrouting of the vehicle, leading to safety concerns, or even access to highly restricted areas [22]. Current detection methodologies using traditional signal analyses and simplistic machine learning models are seriously lacking in their real-time processing capability, effective integration of spatial and temporal data analysis, and inclusion of user alert systems. In this regard, developing a holistic GPS spoofing detection system that would be capable of real-time functionality, enhancing advanced machine learning techniques for anomaly detection, integrating spatial and temporal data analytics, and providing immediate user alerts through mobile applications remains significant.

## 1.7. Significant of the Study

This work is of paramount importance in view of developing the grounds for autonomous vehicle security, since the growth in GPS spoofing attacks is rapid. Employing a strong detection system with machine learning that works in real time is highly critical to improving the safety and reliability of GPS-dependent navigation systems. The proposed technique incorporates all the advanced algorithms which are proficient in analyzing temporal-spatial data with much better and comprehensive spoofing attempts detection compared to the traditional approach.

Such technology puts the majority of autonomous vehicles within reach, from consumer cars to industrial and logistics applications. Integrating mobile alert systems empowers users through immediate notifications of potential threats, allowing them to take timely response actions, hence confidence in AV systems.

This research will help in contributing to cybersecurity by finding the existing research gaps in real-time processing, multi-dimensional data analysis, and user-centric designs, while opening other avenues for further research in innovation in security towards protection of critical infrastructure against emerging and evolving cyber threats.

## 1.8. Research Aim

This research is directed to the design and realization of a fully functional GPS spoofing detection system for autonomous vehicles by making full use of state-of-the-art machine learning methods for real-time processing, incorporating spatial and temporal data analytics, and presenting mobile alert mechanisms in an easy-to-use manner.

## 1.9. Research Objectives

### 1.9.1. Main Objective

The objective is to design and deploy a machine learning-based GPS spoofing detection system that shall be capable of real-time analysis and immediate threat notification.

### 1.9.2. Sub Objectives

- Collect high-quality GPS data (authentic gps data) for model training and validation.
- Evaluate and select suitable machine learning/deep learning models (e.g., RF, FCNN, KNN, SVM, XGBoost) for GPS anomaly detection.
- Sending real-time alerts and system status to the Android app to enhance user interaction

## 1.10. Research Questions

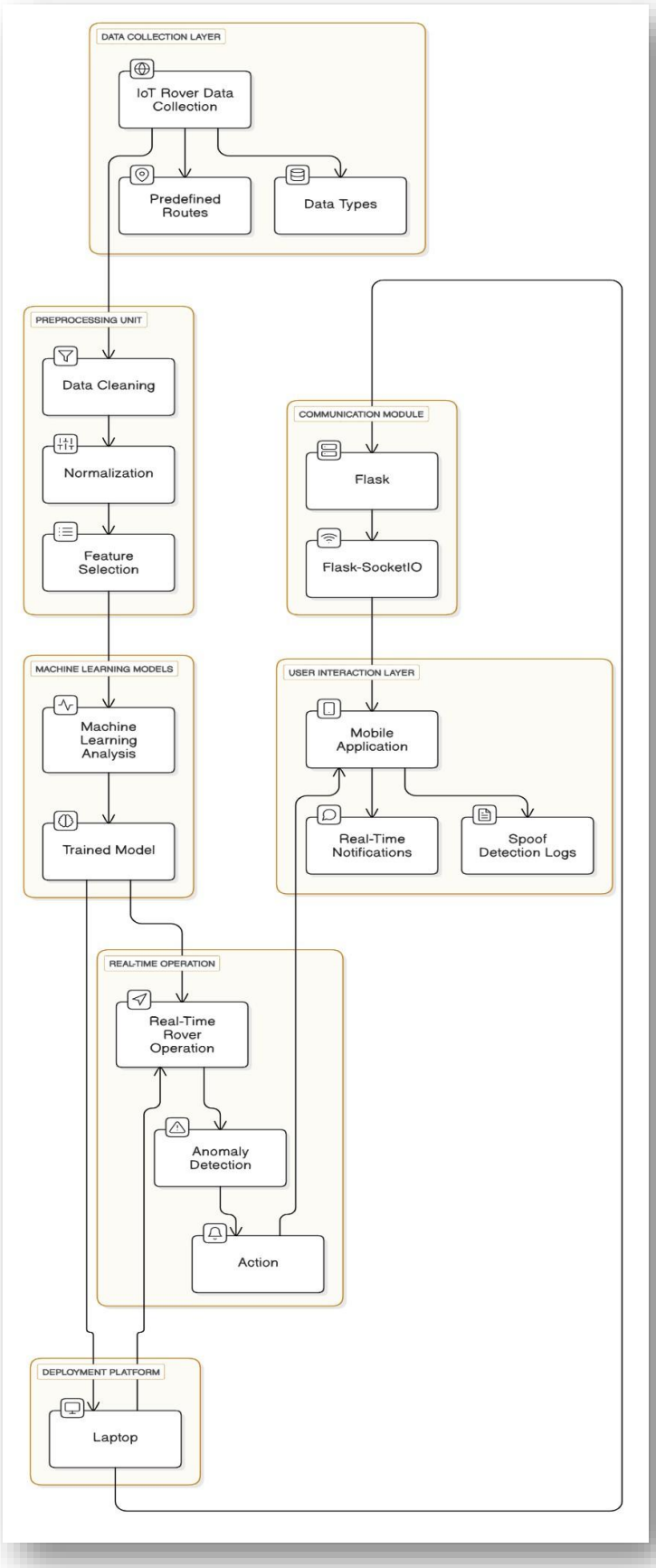
- How to create a machine learning-based technique for real-time detection of GPS spoofing in autonomous vehicles?
- Which machine learning models are most efficient to combine temporal and spatial data analysis to detect the attempt to spoof GPS?
- What are the necessary features that the mobile application should have so as to ensure real-time alert and interaction by the user with the detecting system?
- What is the most appropriate performance metric that best evaluates the reliability and robustness of a GPS spoofing detecting system in practical applications?

# 2. METHODOLOGY

## 2.1. Introduction to Methodology

The methodology section will succinctly explain the structured approach to be used in the development of a full-featured GPS spoofing detection system. This goes further in elaborating on the dataset that is being employed, methodological structure for data pre-processing, model selection, implementation of the system, and evaluation of the performance. This is in pursuit of ensuring the approach satisfies the research objective of devising an effective real-time GPS spoofing detection system, which could be deployed on autonomous vehicles integrated with the respective user notification features.

Figure 2-2. Component Diagram



## 2.2. Dataset Description

Under different environmental conditions, the dataset for this research is collected by driving an IoT-based rover through predefined routes transported from point A to point B and then from B to C. The rover includes an Arduino Mega microcontroller for operation and a GPS module to study the location of the rover. Any IMU sensors on board offer trajectory information for its movement while the WiFi module allows data transmission. Factors thus collected include:

- Latitude and Longitude: It contains spatial coordinates for location tracking.
- Speed and Altitude: It is representative of movement or elevation changes.
- Timestamp: It is a temporal feature that captures the time of signal reception
- Roll, Pitch - Rotation around the front-to-back axis of the vehicle is called Roll and rotation around the side-to-side axis of the vehicle is called Pitch.
- Yaw, Yaw Rate: The rotation around the vertical axis of the vehicle relative to North
- Accel_Y: Measures how fast the device is accelerating up or down along the Y-axis.
- Accel_Z: Measures acceleration up or down along the Z-axis, often aligned vertically.
- Gyro (X, Y, Z): Measures rotation around the X-axis, Y-axis, Z-axis

This working dataset is very significant for training the machine learning model to learn expected paths and movement behavior of the rover. This data includes normal operation data as well as simulated spoofing cases, which helps the model identify the differences between real and anomalous GPS signals.

Figure 2-3. Dataset

| Timestamp | GPS_Latitu | GPS_Longi | GPS_Altitu | GPS_Spee | GPS_Satell | Accel_X | Accel_Y | Accel_Z | Gyro_X | Gyro_Y | Gyro_Z | Roll | Pitch | Yaw |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2025-04-0 | 6.723238 | 80.09507 | 26.1 | 0.13 | 10 | 728 | -160 | 14664 | -68 | 164 | 306 | -0.63 | -2.84 | 2212.32 |
| 2025-04-0 | 6.723236 | 80.09507 | 26.2 | 0.41 | 10 | 648 | -288 | 14616 | -68 | 174 | 313 | -1.13 | -2.54 | 2215.94 |
| 2025-04-0 | 6.723237 | 80.09507 | 25.8 | 0.13 | 10 | 744 | -340 | 14720 | -56 | 186 | 255 | -1.32 | -2.89 | 2218.3 |
| 2025-04-0 | 6.723234 | 80.09507 | 26.1 | 1.35 | 10 | 1124 | -2632 | 13316 | -807 | -173 | 1609 | -11.18 | -4.73 | 2233.3 |
| 2025-04-0 | 6.723233 | 80.09509 | 26.8 | 0.17 | 9 | 1180 | -412 | 14708 | -38 | 157 | 281 | -1.6 | -4.59 | 2272.44 |
| 2025-04-0 | 6.723234 | 80.09509 | 26.9 | 0.13 | 9 | 1148 | -412 | 14484 | -46 | 154 | 246 | -1.63 | -4.53 | 2274.71 |
| 2025-04-0 | 6.723235 | 80.09509 | 26.9 | 0.24 | 9 | 1028 | -496 | 14700 | -41 | 190 | 215 | -1.93 | -4 | 2276.7 |
| 2025-04-0 | 6.723235 | 80.09509 | 27.2 | 0.17 | 9 | 988 | -412 | 14428 | -31 | 176 | 214 | -1.64 | -3.92 | 2278.71 |
| 2025-04-0 | 6.723235 | 80.09509 | 27.3 | 0.26 | 9 | 1128 | -460 | 14516 | -37 | 184 | 288 | -1.82 | -4.44 | 2281.43 |
| 2025-04-0 | 6.723235 | 80.09509 | 27.4 | 0.17 | 9 | 716 | 5192 | 32767 | 4046 | 4916 | 4431 | 9 | -1.24 | 2322.39 |
| 2025-04-0 | 6.723231 | 80.09509 | 27.2 | 2.43 | 9 | -364 | -1264 | 14724 | -46 | 149 | 313 | -4.91 | 1.41 | 2325.23 |
| 2025-04-0 | 6.723224 | 80.09509 | 26.3 | 2.28 | 9 | 2004 | -2804 | -16172 | -1448 | 6366 | -881 | -170.16 | -6.96 | 2317.13 |
| 2025-04-0 | 6.723224 | 80.09509 | 26 | 0.04 | 10 | 2164 | -156 | 14532 | -28 | 163 | 224 | -0.62 | -8.47 | 2319.28 |
| 2025-04-0 | 6.723222 | 80.09509 | 25.9 | 0.31 | 10 | 11468 | 4432 | 24832 | -1817 | 868 | 1667 | 10.12 | -24.45 | 2334.24 |
| 2025-04-0 | 6.72322 | 80.09509 | 24.8 | 0.22 | 10 | 1152 | -724 | 14564 | -42 | 193 | 221 | -2.85 | -4.52 | 2336.3 |
| 2025-04-0 | 6.723224 | 80.09509 | 25 | 1.15 | 10 | 1320 | -496 | 14504 | -41 | 161 | 187 | -1.96 | -5.2 | 2338.76 |
| 2025-04-0 | 6.723225 | 80.09509 | 24.7 | 1.04 | 10 | 596 | -944 | 14000 | 4709 | 4466 | -14702 | -3.86 | -2.43 | 2211.95 |
| 2025-04-0 | 6.723222 | 80.09509 | 24.4 | 0.52 | 10 | 540 | -680 | 14668 | -75 | 168 | 288 | -2.65 | -2.11 | 2217.26 |
| 2025-04-0 | 6.723219 | 80.09509 | 24.6 | 0.67 | 10 | 416 | -672 | 14652 | -79 | 198 | 333 | -2.63 | -1.62 | 2220.15 |
| 2025-04-0 | 6.723218 | 80.09509 | 24.8 | 1.06 | 10 | 2220 | 5664 | 20644 | -17337 | 4072 | 1540 | 15.34 | -5.92 | 2235.58 |
| 2025-04-0 | 6.723216 | 80.09509 | 24.6 | 1.33 | 9 | 528 | 136 | 14464 | -28 | 168 | 262 | 0.54 | -2.09 | 2237.74 |

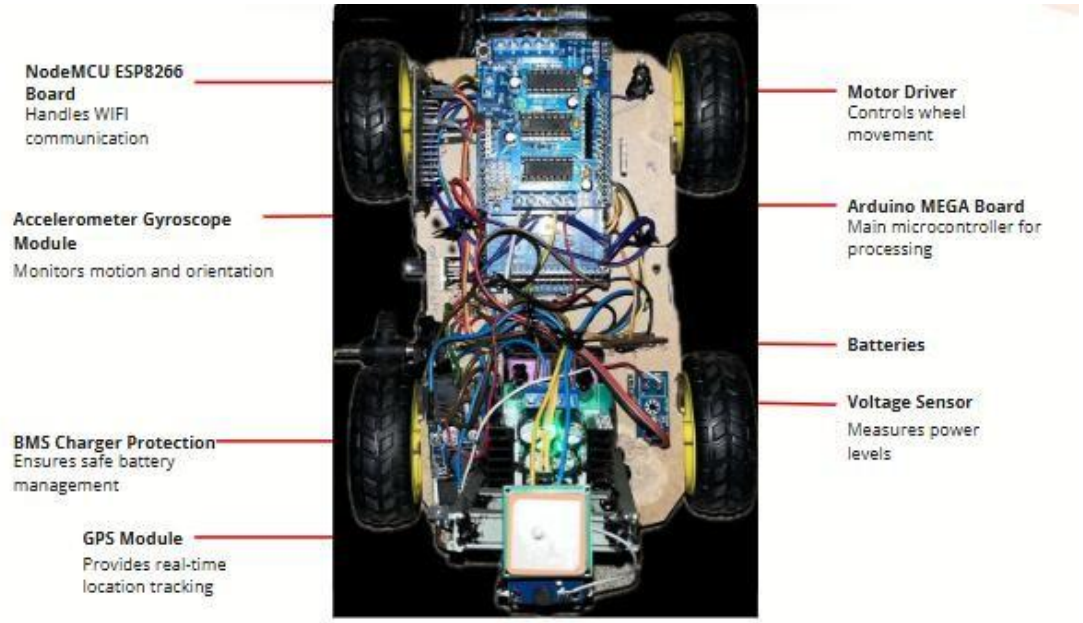



Figure 4 About the rover

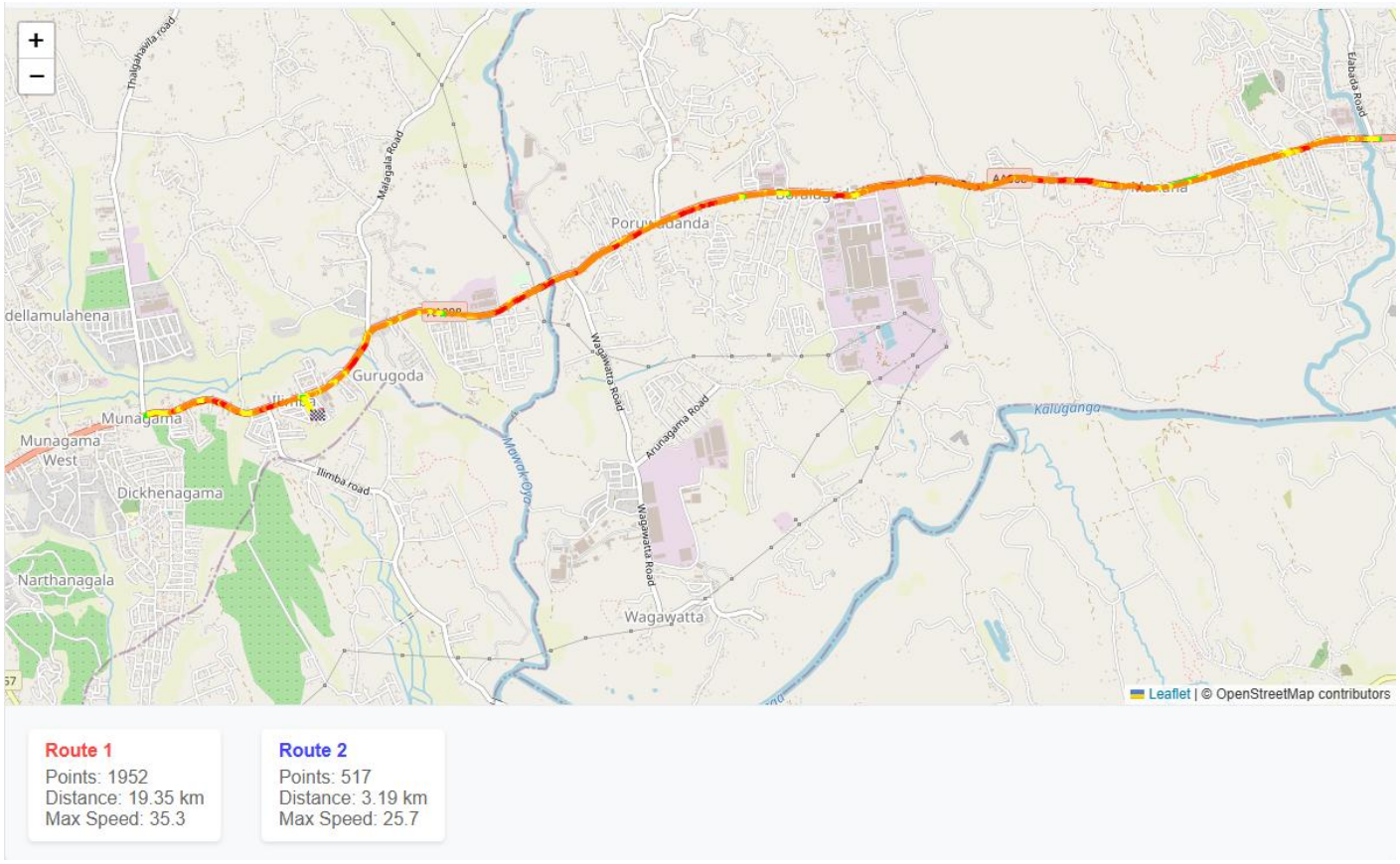
Balance in the dataset concerning signals being normal or spoofed is crucial for effective training of the model. Techniques of data augmentation can be performed in case of imbalances to increase model generalization to different scenarios.

## 2.3Methodological Structure

This covers a multi-step methodology that involves, from data collection to preparation to training and deploying models, which includes:

### *2.3.1.1. Data Collection and Preprocessing*

Figure 5 Data collection



- **Data Collection**

A custom-built IoT rover was mounted within a real vehicle and driven along predefined routes for approximately 20 kilometers to simulate typical driving conditions. The rover system collected rich sensor data in real-time, including GPS coordinates (latitude, longitude, altitude), velocity (in km/h), and inertial measurements (accelerometer and gyroscope values from an IMU sensor). Timestamps were recorded for every sensor reading to preserve sequence integrity. This data set represents the normal, unsuspicious navigation behavior of the vehicle under genuine GPS signals.

- **Data Cleaning**

The collected dataset was examined for inconsistencies such as null entries, corrupted sensor values, or physically impossible jumps in coordinate space. Such anomalies—often caused by sensor glitches, startup noise, or temporary signal losses—were removed to ensure the training data remained reliable. Only high-integrity records were retained for model training and testing.

- **Normalization**

To ensure efficient learning and convergence of the machine learning model, numerical features such as GPS coordinates, speed, and IMU-derived values were normalized. This step maps the raw input values to a standardized scale (typically 0 to 1 or with zero mean and unit variance), which reduces bias during model training and improves numerical stability, especially for deep learning architectures like LSTM autoencoders.

- **Feature Extraction**

Additional features were engineered from the raw data to enhance the model's ability to detect spoofing anomalies. These include:

  - **Trajectory deviation**: Calculated by comparing consecutive GPS points to estimate sudden directional shifts.

- **Speed fluctuations**: Sudden spikes or drops in speed were flagged as potential anomalies.
- **Coordinate differentials**: Latitude and longitude differences between consecutive points (diff_lat, diff_lon) were included as indicators of abrupt displacement, often associated with spoofing attempts.

  These derived features were combined with the original measurements to form a rich input vector for the model.

Figure 6 Data cleaning

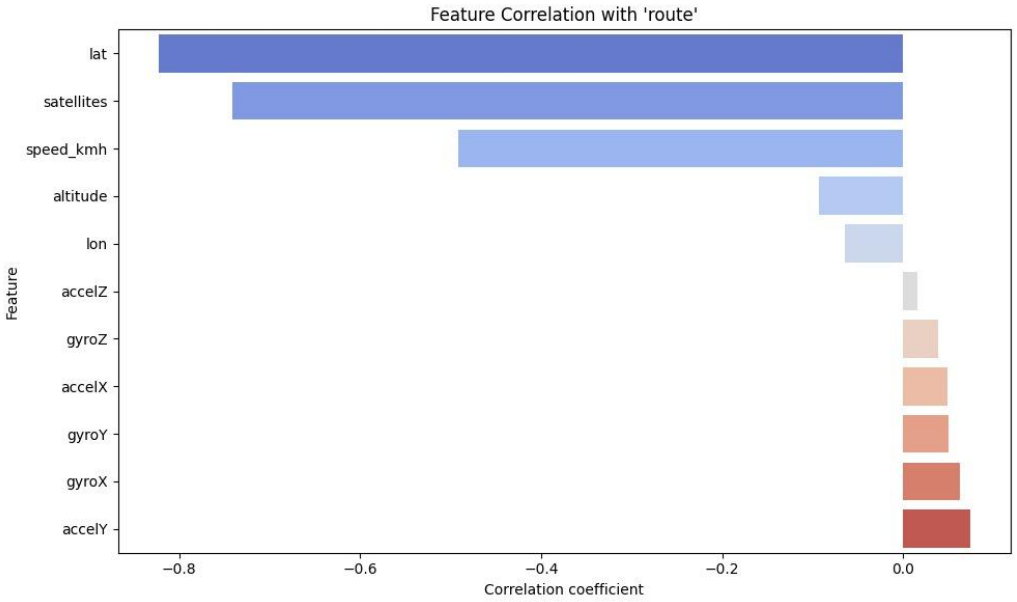
```python
# Step 1: Handle missing values
combined_df.replace('N/A', np.nan, inplace=True)
numeric_columns = ['GPS_Latitude', 'GPS_Longitude', 'GPS_Altitude', 'GPS_Speed',
                   'Accel_X', 'Accel_Y', 'Accel_Z', 'Gyro_X', 'Gyro_Y', 'Gyro_Z',
                   'Roll', 'Pitch', 'Yaw']
for col in numeric_columns:
    combined_df[col].fillna(combined_df[col].median(), inplace=True)

# Step 2: Convert data types
combined_df['Timestamp'] = pd.to_datetime(combined_df['Timestamp'], errors='coerce')
for col in numeric_columns:
    combined_df[col] = pd.to_numeric(combined_df[col], errors='coerce')
combined_df['GPS_Satellites'] = combined_df['GPS_Satellites'].astype(int)

# Step 3: Handle outliers
def remove_outliers(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]

columns_to_check = ['GPS_Speed', 'Accel_X', 'Accel_Y', 'Accel_Z']
for col in columns_to_check:
    combined_df = remove_outliers(combined_df, col)

# Step 4: Remove duplicates
combined_df.drop_duplicates(inplace=True)

# Step 5: Validate data consistency
combined_df = combined_df[
    (combined_df['GPS_Latitude'].between(-90, 90)) &
    (combined_df['GPS_Longitude'].between(-180, 180)) &
    (combined_df['GPS_Speed'] >= 0)
]
valid_routes = ['A-B', 'A-C']
combined_df = combined_df[combined_df['Route_ID'].isin(valid_routes)]
```

Figure 7 Data balancing

```python
# Fix: Generate noise only for needed_rows samples

# Randomly sample needed_rows scaled rows with replacement
sampled_scaled_rows = df_route2_scaled[np.random.choice(df_route2_scaled.shape[0], needed_rows, replace=True)]

# Generate noise only for needed_rows samples
noise_fixed = np.random.normal(0, 0.05, size=sampled_scaled_rows.shape)

# Add noise to sampled rows
synthetic_scaled_fixed = sampled_scaled_rows + noise_fixed

# Inverse transform to original scale
synthetic_unscaled_fixed = scaler.inverse_transform(synthetic_scaled_fixed)

# Create synthetic dataframe with original columns except 'route'
synthetic_df_fixed = pd.DataFrame(synthetic_unscaled_fixed, columns=columns_to_scale)

# Clip values within original min-max range for each column (from route 2 data)
for col in columns_to_scale:
    col_min = df_route2[col].min()
    col_max = df_route2[col].max()
    synthetic_df_fixed[col] = synthetic_df_fixed[col].clip(lower=col_min, upper=col_max)

# Add route column back (value 2 for all synthetic rows)
synthetic_df_fixed['route'] = 2

# Append synthetic rows to original dataframe
df_balanced_fixed = pd.concat([df, synthetic_df_fixed], ignore_index=True)

# Check new distribution to confirm balancing
balanced_counts_fixed = df_balanced_fixed['route'].value_counts()

balanced_counts_fixed
```

```
[4]
route
1    2025
2    2025
Name: count, dtype: int64
```

Figure 8 Feature extract



#### *2.3.1.2. Model Selection and Training*

- Model Evaluation

  Various machine learning techniques were considered for the detection of GPS spoofing, including both supervised classifiers and unsupervised anomaly detection models. Given the absence of labeled spoofed data during initial training and the sequential nature of the sensor readings, an unsupervised Long Short-Term Memory (LSTM) autoencoder was selected as the primary detection model. This architecture excels at learning temporal patterns and reconstructing normal time-series data, making it suitable for identifying deviations from expected behavior caused by spoofing.

- Training Process

  The LSTM autoencoder was trained on the pre-processed dataset representing normal driving behavior. The input features consisted of sequences of GPS and trajectory-derived values, including speed, altitude, satellite count, and coordinate differentials. To prevent overfitting and ensure generalization across various driving conditions, cross-validation was employed, and hyperparameters such as sequence length, layer size, and learning rate were tuned through grid search. The trained model learned to reconstruct input sequences with minimal error when the data followed expected patterns.

- Anomaly Detection Focus

  During deployment, the system continuously feeds incoming GPS data into the trained model. The model attempts to reconstruct the latest sequence, and the reconstruction error (measured by Mean Squared Error, MSE) is computed. If the MSE exceeds a predefined threshold—established empirically during testing—the data point is classified as anomalous and flagged as a potential GPS spoofing incident. This real-time detection mechanism enables the system to autonomously distinguish between legitimate and spoofed GPS signals with high reliability.

#### *2.3.1.3. System Deployment*

- **Integration into IoT Rover**

  The trained LSTM autoencoder model is deployed on the back-end server that interfaces with the Arduino Mega-based IoT rover. The Arduino handles real-time GPS and IMU data collection, which is then transmitted over WiFi to a Flask server hosted on a remote Linux instance (DigitalOcean). This separation allows the microcontroller to remain lightweight while offloading the computationally intensive anomaly detection task to the server.

- **Real-time Operation**

  During active navigation, the system continuously monitors the GPS trajectory and sensor data. When the model detects an anomaly—such as a sudden deviation from known paths or physically implausible movement—the server flags the input as a spoofed signal. Although the Arduino itself does not perform the detection, it can be instructed to respond (e.g., stop movement or trigger an alert). This approach ensures secure path-following by reacting instantly to potential spoofing events.

#### *2.3.1.4. Logs integrating into Mobile Application*

- **User Interface**

  A custom mobile application has been developed to interface with the rover system. It provides a user-friendly graphical interface that displays key telemetry and spoof detection logs in real-time.

- **Communication**

  The app communicates with the rover system over a local WiFi network or server link. It receives periodic updates from the Flask server, including GPS location, spoof detection status, and battery health metrics.

- **Functionality**

  The application visualizes the rover's current path on a map, shows whether the system is operating normally or under a spoofing alert, and stores a history of detected anomalies. When a spoofing event is identified, the app immediately notifies the user, enabling prompt awareness and intervention.

#### *2.3.1.5. Performance Evaluation*

- **Evaluation Metrics**

  The system's performance is assessed by using standard anomaly detection metrics:

  - Accuracy: The overall ability to correctly classify normal and spoofed signals.
  - Precision: The proportion of detected spoof alerts that were actual spoofing events.
  - Recall: The system's ability to detect all spoofing attempts.
  - False Positive Rate: Normal signals incorrectly flagged as spoofed.
  - False Negative Rate: Spoofing attempts that went undetected.
  - Response Time: The latency between spoofing detection and response (e.g., alert or movement abortion).

- Testing Environment

The system is tested in both controlled field environments and live-drive experiments using the real car equipped with the IoT rover. Simulated spoofing attacks were introduced by altering GPS signal patterns sent to the Flask server. The system successfully detected abnormal behavior, initiated alerts, and prevented deviation from authorized paths in real time, validating its reliability under real-world conditions.

## 2.4.Research Architecture

This research presents a comprehensive GPS spoofing detection framework by integrating embedded hardware, machine learning, and mobile interfaces into a cohesive system. The architectural design supports real-time processing, spoof detection, and user interaction through a layered structure. The major layers are described as follows:

1. Data Collection and Input Layer

   The foundation of the system consists of an IoT-enabled rover, which integrates an Arduino Mega microcontroller, GPS module, IMU sensors, and WiFi communication module. These components collaboratively capture rich sensor data such as GPS coordinates, velocity, orientation, and timestamps as the vehicle is driven along predefined routes. This layer ensures accurate acquisition of real-world data necessary for model training and real-time inference.

2. Preprocessing Layer

   - This layer ensures that the raw input data is transformed into a high-quality format suitable for machine learning models. It includes:
   - Cleaning: Erroneous or incomplete data entries are removed to prevent model bias and ensure consistency.
   - Normalization: Sensor data such as coordinates, speed, and acceleration are scaled to a common range to stabilize and improve model performance.
   - Feature Selection: Critical attributes are selected to enhance spoof detection, including sudden changes in coordinates, trajectory deviation, and fluctuations in signal strength or satellite count.

3. Model Training and Machine Learning Layer

    - This layer is responsible for modeling the expected behavior of the rover using machine learning techniques:

    - A Long Short-Term Memory (LSTM) autoencoder model is trained to learn typical movement patterns and detect anomalies based on temporal reconstruction errors.

    - The model is trained on the preprocessed dataset using cross-validation and hyperparameter tuning to optimize generalization.

    - Although the system primarily uses unsupervised learning due to limited spoofed labels, the model is evaluated using metrics derived from manually induced spoofing scenarios.

4. Deployment and Processing Layer

    - The trained anomaly detection model is deployed on a Flask server hosted on a cloud (DigitalOcean), which communicates with the Arduino Mega in the rover. The real-time GPS and IMU data sent from the rover is processed by the server:

    - Anomalies (e.g., suspicious trajectories) are identified instantly.

    - Upon detection of spoofed data, the system can raise alerts or initiate countermeasures such as stopping the rover or rejecting navigation commands.

    - This architecture offloads heavy computation from the microcontroller, ensuring efficiency and scalability.

5. User Interaction and Mobile Application Layer

    - To enable user awareness and engagement, a mobile application is integrated with the system. This layer includes:

    - Real-Time Alerts: Users are notified immediately when spoofing is detected, ensuring rapid response.

- System Status Visualization: The app displays real-time GPS coordinates, spoofing status, battery voltage, and other telemetry data.
- Historical Logs: Spoof detection logs are stored in a Supabase cloud database and accessed by the app for historical analysis and post-event review.

6. Performance Monitoring and Evaluation Layer

- The final layer is responsible for assessing system reliability and responsiveness. Key performance indicators include:
- Accuracy: Measures the ability of the system to correctly detect spoofed signals.
- False Positive/Negative Rates: Evaluates the balance between sensitivity and specificity to ensure practical usability without over-alerting.
- Processing Speed: Assesses the time required to process input data and detect spoofing in real time.
- The architectural design prioritizes scalability, real-time operation, and adaptability to evolving spoofing strategies. It enables reliable protection of autonomous navigation systems against GPS-based attacks while maintaining user accessibility and situational awareness.

## 2.5.Software Architecture Model

The SDLC architecture is going to be used for the suggested program. Each step is further broken down into its component parts at this point. At each successive step, testing and implementation will be carried out. Both the software and the hardware will be impacted. This method is broken down into five stages: planning, designing, testing, building, and delivering the finished product. The following is an explanation of each stage of the software development life cycle (SDLC): planning, analysis, design, implementation, and maintenance. Therefore, I decided that this should be the software architecture for this study.

- **Planning** - The first stage involves defining data collection paths while choosing hardware components including Arduino Mega, GPS, IMU, WiFi and specifying system functional needs.
- **Analysis** - The team will evaluate how the rover requires navigation capabilities with spoofing detection functionality.
- **Design** - The development process includes designing hardware components for the rover together with data acquisition procedures and machine learning algorithms and mobile application user interfaces.
- **Implementation** - The team will construct the rover while gathering data for the model training process which will be deployed on the Arduino Mega before developing the mobile application.
- **Maintenance** - Regular system checks and required updates will help the system maintain performance against emerging spoofing methods.

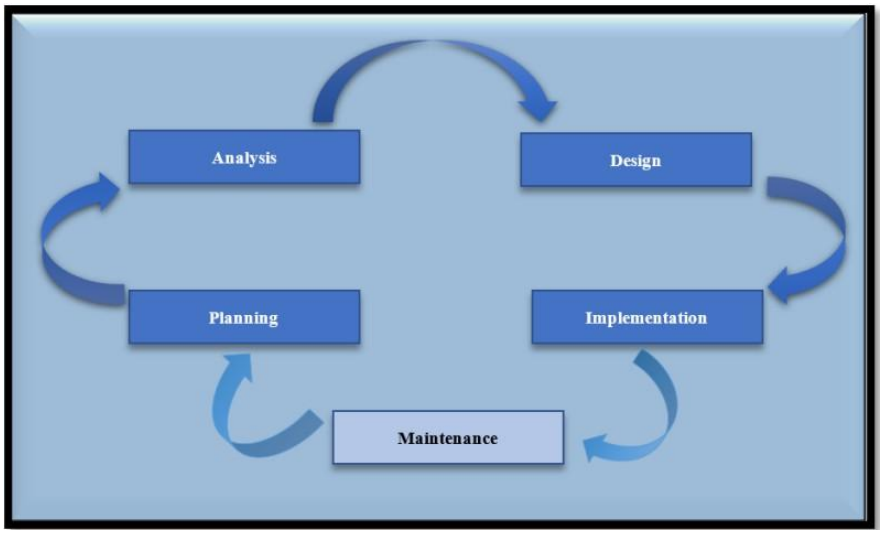



Figure 9. SDLC Methodology Life Cycle.

## 2.6.Requirement Gathering and Analyzing

### 2.6.1. Functional Requirements

The GPS spoofing detection system is expected to fulfill the following core functionalities:

- **Real-Time Data Processing**

The system shall continuously process GPS and IMU data streams from the IoT rover in real time, with immediate analysis and identification of suspicious behavior indicative of spoofing.

- **Anomaly Detection**
  A machine learning-based detection mechanism must analyze temporal and spatial features in the data. The model should achieve high accuracy in distinguishing normal patterns from anomalous or spoofed signals.
- **User Notifications**
  Users must receive real-time notifications via the mobile application whenever a spoofing attempt is detected. These alerts enhance user awareness and facilitate immediate response.
- **Data Visualization**
  The mobile application shall display GPS telemetry, spoofing status, and historical logs in a clear and user-friendly interface. This includes both live updates and stored records of past spoofing events.
- **Scalability**
  The system must be designed to support future extensions for various autonomous vehicle types or GPS-based navigation platforms without requiring major architectural changes.

### 2.6.2. Non-Functional Requirements

Non-functional requirements describe the performance and quality attributes of the system:

- **Reliability**
  The system must consistently detect spoofing attempts with minimal false positives and false negatives. It should operate stably across various environments and use cases.

- **Performance**
  The processing pipeline should maintain low latency to support near-instantaneous spoof detection and alert delivery, preserving real-time responsiveness.

- **Usability**

  The mobile application must offer an intuitive and user-friendly interface, enabling seamless navigation and understanding even by non-technical users.

- **Scalability**

  The system should be capable of scaling with increasing volumes of GPS data and adapting to multiple embedded platforms or additional vehicle units.

- **Security**

  Communication between the IoT rover, server, and mobile app must be secured using encrypted protocols to prevent unauthorized access, data tampering, or spoof injection.

- **Maintainability**

  The system must be designed to accommodate updates, bug fixes, and model improvements with minimal disruption to ongoing operations.

### 2.6.3. Software Requirements

Given below are the software requirements needed for developing the GPS spoofing detection system.

- **Programming Language:** Python would be used for data processing, model development, and server-side scripting.

#### 2.6.3.1. *Frameworks and Libraries:*

- TensorFlow/Keras: For machine learning model development.
- Flask and Flask-SocketIO for the backend server and real-time communication.
- Scikit-learn for pre-processing data and evaluating the model's performance.
- Pandas and NumPy to manipulate data.
- Chart.js to plot data of the mobile application.
- Development Environment: An Integrated Development Environment, such as PyCharm or Visual Studio Code.
- Mobile Application Framework: Additional frameworks will include React Native or Flutter if cross-platform mobile app development is desired.

### 2.6.4. Hardware Requirements

Hardware that will be required to run the system is:

- IoT Rover**:** Arduino Mega as the microcontroller, GPS module for location, IMU sensors for trajectory, Wi-Fi module for communication, rover chassis, and motors.
- Storage**:** Onboard memory for model storage and temporary data logging.
- Power Supply**:** Reliable battery for continuous rover operation.
- Mobile Device**:** Smartphone or tablet for the mobile app.

### 2.6.5. Analysis of Requirements

Analysis of requirements establishes that system design meets both functional and non-functional requirements in relation to the capabilities of the IoT-based rover:

**Functional vs. Non-Functional:** This analysis made it clear that functional requirements specify what the system should do (real-time spoofing detection, preventing navigation along wrong paths, alerting users via a mobile app), whereas non-functional requirements would correspond to how good a job it does in doing this, such as reliability, low-latency performance, and data security.

**Objective Alignment:** The hardware and software requirements are intended to support the objective of a real-time, robust GPS spoofing detection system. The Arduino Mega-based IoT rover enables compatibility with autonomous vehicles and other applications that proceed based on GPS using machine learning and embedded processing.

**Feasibility:** The selection of software frameworks (Arduino C/C++ and mobile app development tools like Flutter) and the Arduino Mega as the embedded platform made the project a practical, low-cost venture. This allows for the scalability of real-time processing and user interaction in line with industry standards for cheap-yet-capable navigation security systems.

**Challenges and Mitigations:** The principal challenges to be solved include achieving low latency for real-time detection and ensuring secure data transmission between the rover and the mobile app, which are being countered by model optimization to fit the limited resources of Arduino Mega, use of lightweight algorithms, and secure WiFi communication protocols. Further validation of system performance and reliability would be through extensive testing in different environmental conditions.

## 2.7. Used Tools and Technologies

The basis of this GPS spoofing detection system is a blend of specialized tools and technologies that will drive the successful implementation of the entire system. These will ensure that the system holds good efficiencies in data preprocessing, model training, deployment, and real-time user interaction.

### 2.7.1. Tools

- Programming and Development Environments:
    - **Python:** This acts as the main programming language through which machine learning models are developed, scripts for data preprocessing will be written, and the backend of the server functionalities will be developed.
    - **PyCharm / Visual Studio Code:** The IDEs used in writing, debugging, and testing the code.
- Data Analysis and Preprocessing:
    - **Jupyter Notebook:** It would be nice for exploratory data analysis, visualization, and prototyping of machine learning algorithms.
    - **Pandas:** A library for data manipulation and analysis whose functionality cannot be missed in handling big datasets and cleaning.
    - **NumPy:** It supports numerical computations together with operations on arrays.
- Machine Learning and Model Training:
    - **TensorFlow/Keras**: Used in the building and training of machine learning models.
    - **Scikit-learn:** This library is used in the data preprocessing process, model selection, and performance evaluation of the project.

- Backend Development:
    - **Flask:** This is a lightweight web framework used to develop the backend server, which would be responsible for handling incoming data and sending it to the mobile application for communication.
    - **Flask-SocketIO:** It enables real-time communication between the server and immediate updates of the mobile app.

- Version Control:
    - **Git:** This is utilized for version control and during the collaboration process in development.
    - **GitHub/GitLab:** Code repository management version control and collaboration among team members.
- Mobile Application Development
    - React Native / Flutter optional Cross-platform mobile application development for real-time alerting and visualization.

### 2.7.2. Technologies

- Machine Learning Technologies
- Data Preparation:
    - Normalization and Selection are methods for scaling input data and choosing relevant data for your model, ensuring accuracy and swiftness in training.
- Realtime Communication:
    - WebSockets - Used along with Flask-SocketIO to provide real-time data exchange between the backend server and the mobile application.
- Deployment Technology:
- Data Visualization:
- Security Protocols:

This enables the integration of these tools and technologies to form a sound infrastructure in developing, deploying, and maintaining the GPS spoofing detection system. The research can apply cutting-edge machine learning frameworks, real-time communication, and embedded hardware solutions to offer scalable, real-time, and user- centric solutions for GPS spoofing detection in AV systems.

# 3. IMPLEMENTATION AND TESTING

## 3.4.Implementation

Various components are integrated into the GPS spoofing detection system, right from data preprocessing and machine learning model development to real-time deployment on embedded devices along with user interface design. The major implementation steps are discussed below.

### 3.4.1. Data Preprocessing

The first step in the implementation process involves the preparation of the dataset that would actually be used for training a model. This includes:

- **Data Cleaning:** Removing incomplete or erroneous entries to ensure the quality of the data.
- **Normalization:** It normalizes the features in the standard range, latitude, longitude, signal strength, and speed, for the input into machine learning models consistently.
- **Feature Selection:** It helps in selecting the relevant features responsible for finding anomalies, such as abrupt changes in coordinates and fluctuations in signal strength.

### 3.4.2. Model Development

- Use the trained model to deploy on the Arduino Mega. It also processes real time GPS and IMU data, detects anomalies, halts movement if spoofing is detected and sends alerts via WiFi.

### 3.4.3. Deployment on Embedded System

The final model is then deployed on a laptop to perform real-time detection. We chose the embedded system here because it gives us the right balance between computational power and pricing. Hence, it is suitable for incorporation into autonomous vehicles. For deployment, we follow the following steps:

- **Model Inference:** run the trained model on the incoming GPS data streams to detect spoofing attempts.

- **Real-Time Processing:** The data is fed to a backend server developed using the Flask framework, which, after processing the information with least possible delay, produces an output.

### 3.4.4. Integrate Logs to Mobile Application

- **User Interface:** The UI will be user-friendly and laid out to present views on the system status, alerts on any abnormality, and logs.
- **Real-time Alerts:** Integrated Flask-SocketIO into the system. This allows for real-time alerts if the system feels there was a spoofing incident.
- **Data Visualization:** visual charts and graphs that represent GPS data and anomaly detection results, which will give a clear idea of the system's performance.

### 3.4.5. Security and Communication

Secure transmission protocols are utilized to ensure that data integrity and user trust are maintained in communication between the laptop and the mobile application. The backend server is secured using authentication and encryption measures to prevent unauthorized access.

### 3.4.6. Performance Optimization

The following are some optimizations used for efficient real-time processing:

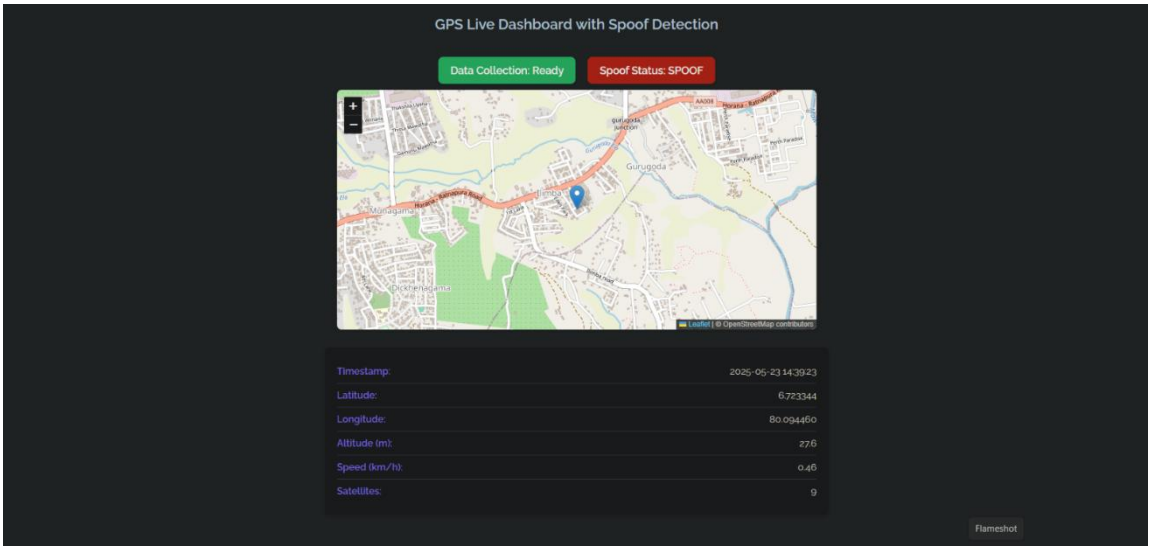
- Model Pruning and Quantization: This reduces the model size and increases the speed of the inference without significantly reducing accuracy.
- Balancing the loads, in order to ensure that the laptop could handle streams of incoming data with no lag.

The broader implementation ensures real-time detection of GPS spoofing with immediate warnings to users and high reliability and accuracy in accordance with the objectives of this research.

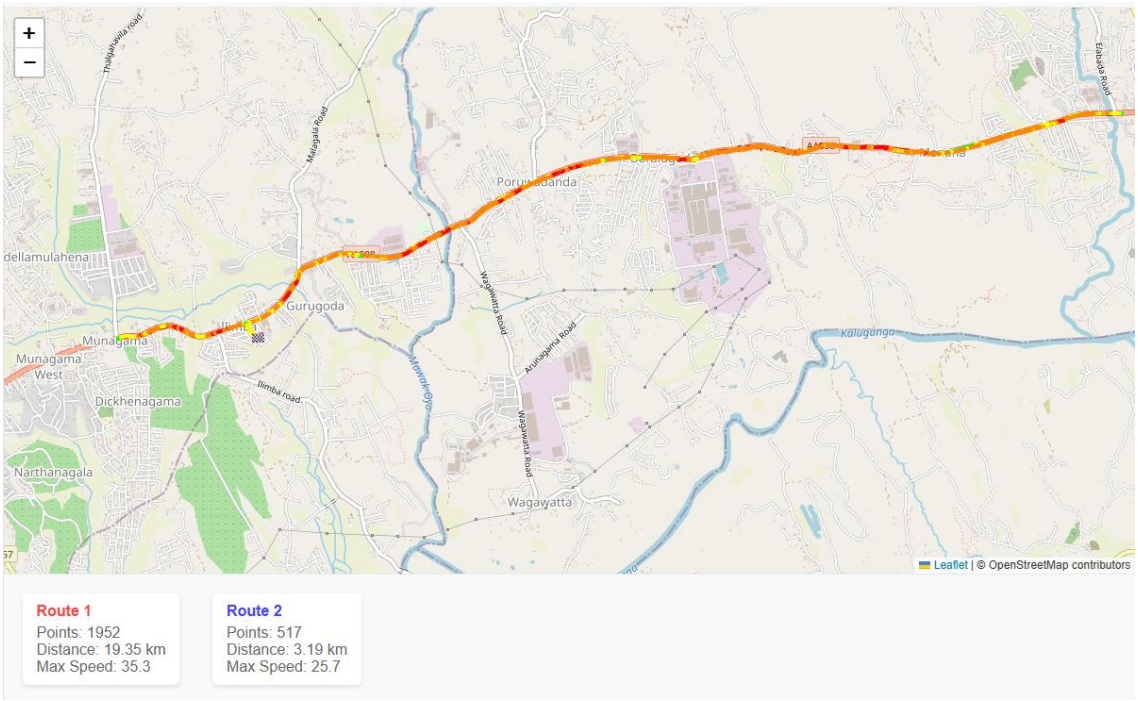
## 3.5.Testing

### 3.5.1. System Testing

#### 3.5.1.1. Front-End

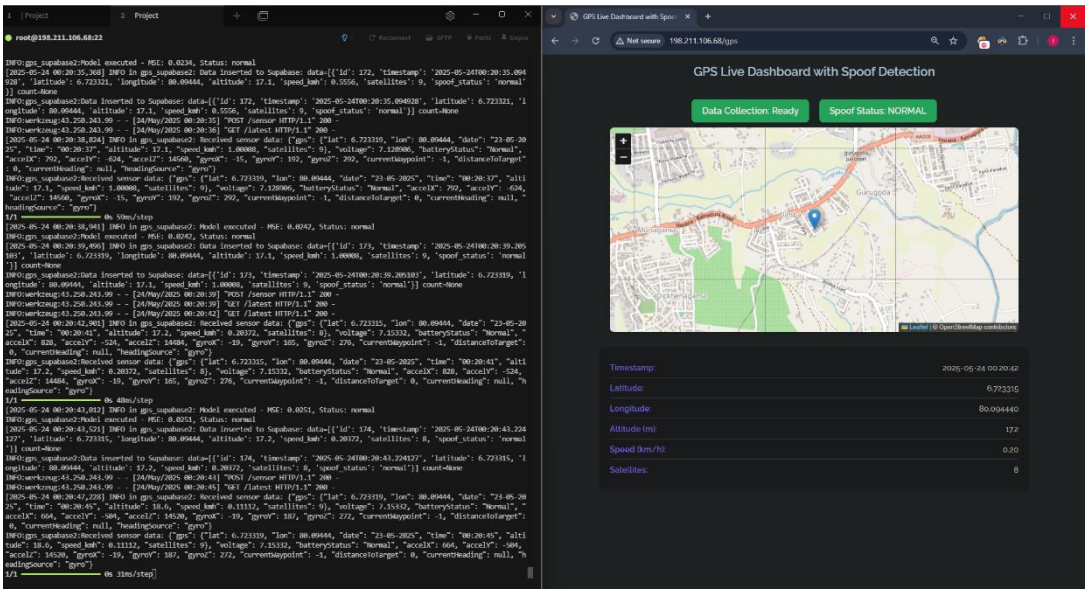



Map of all routes

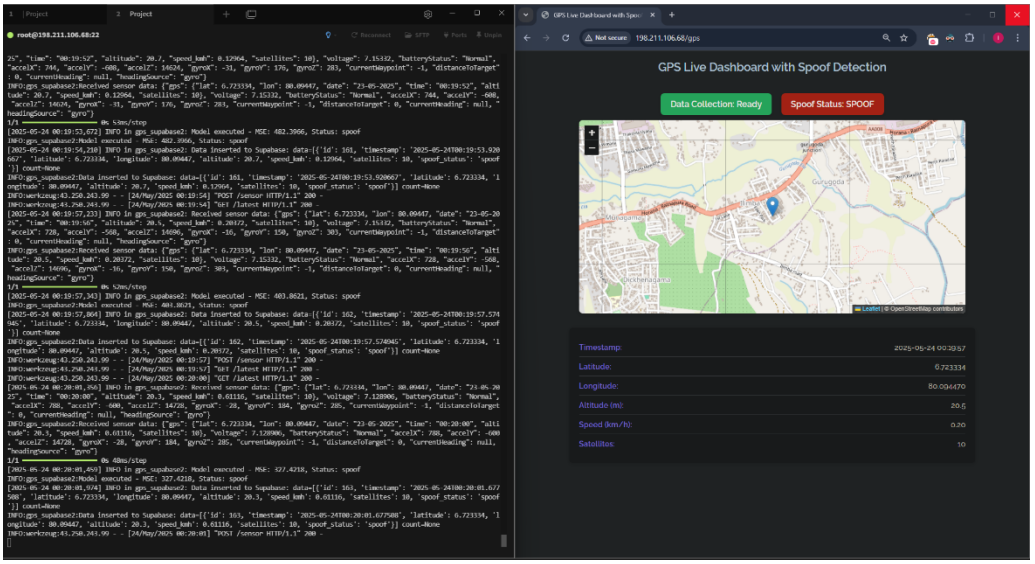


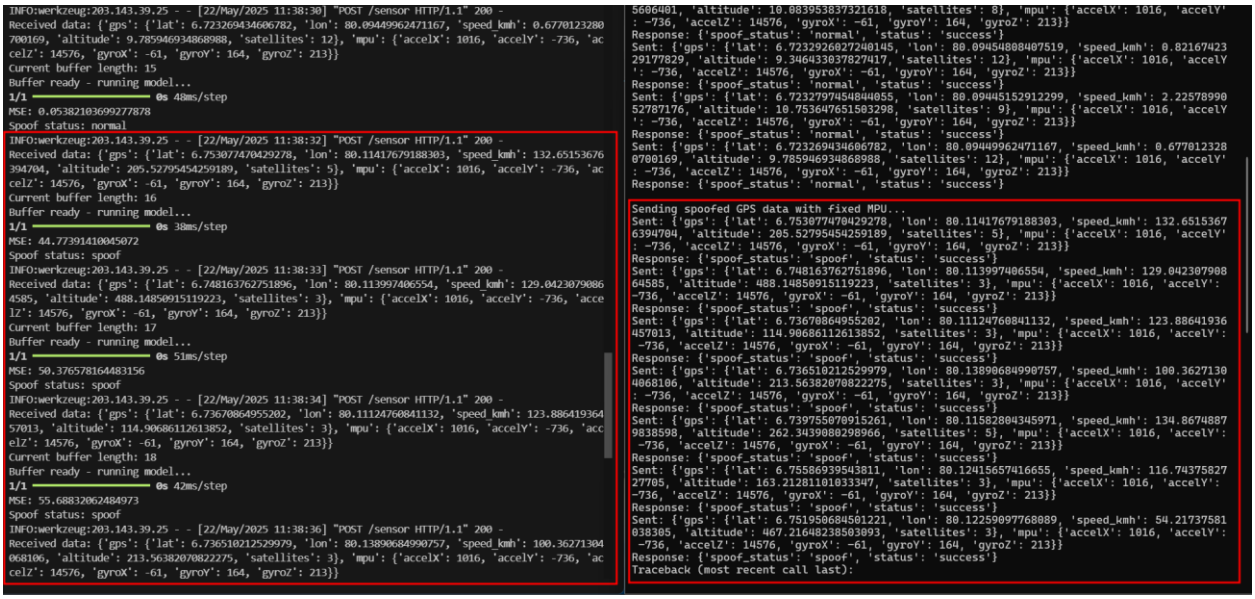
### 3.5.1.2. Back-End

Collecting normal signals

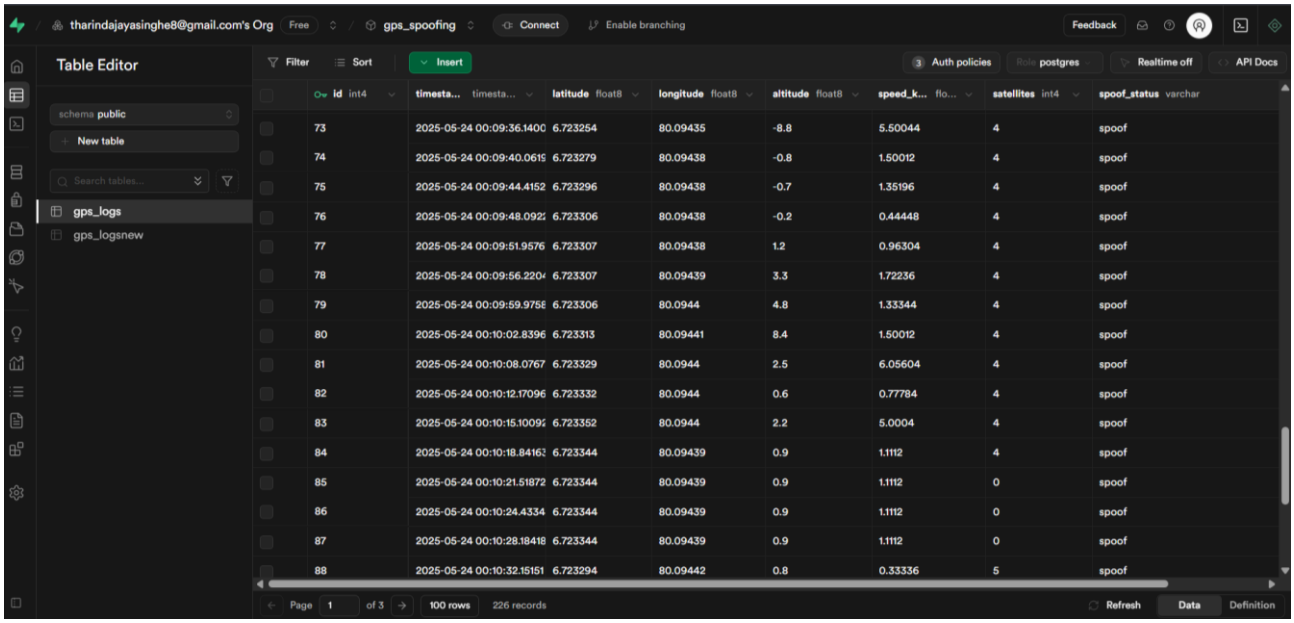



Spoof detection




INFO:werkzeug:203.143.39.25 - - [22/May/2025 11:38:30] "POST /sensor HTTP/1.1" 200 -
Received data: {'gps': {'lat': 6.723269434606782, 'lon': 80.09449962471167, 'speed_kmh': 0.6770123280700169, 'altitude': 9.785946934868988, 'satellites': 12}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Current buffer length: 15
Buffer ready - running model...
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 48ms/step
MSE: 0.05382103699277878
Spoof status: normal
INFO:werkzeug:203.143.39.25 - - [22/May/2025 11:38:32] "POST /sensor HTTP/1.1" 200 -
Received data: {'gps': {'lat': 6.753077470429278, 'lon': 80.11417679188303, 'speed_kmh': 132.65153676394704, 'altitude': 205.52795454259189, 'satellites': 5}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Current buffer length: 16
Buffer ready - running model...
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 38ms/step
MSE: 44.77391410045072
Spoof status: spoof
INFO:werkzeug:203.143.39.25 - - [22/May/2025 11:38:33] "POST /sensor HTTP/1.1" 200 -
Received data: {'gps': {'lat': 6.748163762751896, 'lon': 80.113997406554, 'speed_kmh': 129.04230798864585, 'altitude': 488.14850915119223, 'satellites': 3}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Current buffer length: 17
Buffer ready - running model...
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 51ms/step
MSE: 50.376578164483156
Spoof status: spoof
INFO:werkzeug:203.143.39.25 - - [22/May/2025 11:38:34] "POST /sensor HTTP/1.1" 200 -
Received data: {'gps': {'lat': 6.736708649555202, 'lon': 80.11124760841132, 'speed_kmh': 123.88641936457013, 'altitude': 114.90686112613852, 'satellites': 3}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Current buffer length: 18
Buffer ready - running model...
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step
MSE: 55.68832062484973
Spoof status: spoof
INFO:werkzeug:203.143.39.25 - - [22/May/2025 11:38:36] "POST /sensor HTTP/1.1" 200 -
Received data: {'gps': {'lat': 6.736510212529979, 'lon': 80.13890684990757, 'speed_kmh': 100.36271304068106, 'altitude': 213.56382070822275, 'satellites': 3}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
5606401, 'altitude': 10.083953837321618, 'satellites': 8}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'normal', 'status': 'success'}
Sent: {'gps': {'lat': 6.72329260272401 45, 'lon': 80.09454808407519, 'speed_kmh': 0.8216742329177829, 'altitude': 9.346433037827417, 'satellites': 12}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'normal', 'status': 'success'}
Sent: {'gps': {'lat': 6.723279745484 4055, 'lon': 80.09445152912299, 'speed_kmh': 2.2257899052787176, 'altitude': 10.753647651503298, 'satellites': 9}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'normal', 'status': 'success'}
Sent: {'gps': {'lat': 6.723269434606782, 'lon': 80.09449962471167, 'speed_kmh': 0.6770123280700169, 'altitude': 9.785946934868988, 'satellites': 12}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'normal', 'status': 'success'}
Sending spoofed GPS data with fixed MPU...
Sent: {'gps': {'lat': 6.753077470429278, 'lon': 80.114176791883 03, 'speed_kmh': 132.65153676394704, 'altitude': 205.52795454259189, 'satellites': 5}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'spoof', 'status': 'success'}
Sent: {'gps': {'lat': 6.748163762751896, 'lon': 80.113997406554, 'speed_kmh': 129.04230798864585, 'altitude': 488.14850915119223, 'satellites': 3}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'spoof', 'status': 'success'}
Sent: {'gps': {'lat': 6.736708649555202, 'lon': 80.11124760841132, 'speed_kmh': 123.88641936457013, 'altitude': 114.90686112613852, 'satellites': 3}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'spoof', 'status': 'success'}
Sent: {'gps': {'lat': 6.736510212529979, 'lon': 80.13890684990757, 'speed_kmh': 100.36271304068106, 'altitude': 213.56382070822275, 'satellites': 3}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'spoof', 'status': 'success'}
Sent: {'gps': {'lat': 6.739755070915261, 'lon': 80.11582804345971, 'speed_kmh': 134.86748879838598, 'altitude': 262.3439080298966, 'satellites': 5}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'spoof', 'status': 'success'}
Sent: {'gps': {'lat': 6.7558693954 3811, 'lon': 80.12415657416655, 'speed_kmh': 116.74375827277705, 'altitude': 163.21281101033347, 'satellites': 3}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'spoof', 'status': 'success'}
Sent: {'gps': {'lat': 6.751950684501221, 'lon': 80.12259097768089, 'speed_kmh': 54.21737581038305, 'altitude': 467.21648238503093, 'satellites': 3}, 'mpu': {'accelX': 1016, 'accelY': -736, 'accelZ': 14576, 'gyroX': -61, 'gyroY': 164, 'gyroZ': 213}}
Response: {'spoof_status': 'spoof', 'status': 'success'}
Traceback (most recent call last):

Database


tharindajayasinghe8@gmail.com's Org
Free
gps_spoofing
Connect
Enable branching
Feedback
Table Editor
Filter
Sort
Insert
Auth policies
postgres
Realtime off
API Docs
schema public
New table
Search tables...
gps_logs
gps_logsnew
id int4
timesta... timesta...
latitude float8
longitude float8
altitude float8
speed_k... flo...
satellites int4
spoof_status varchar
73 2025-05-24 00:09:36.1400 6.723254 80.09435 -8.8 5.50044 4 spoof
74 2025-05-24 00:09:40.0615 6.723279 80.09438 -0.8 1.50012 4 spoof
75 2025-05-24 00:09:44.4152 6.723296 80.09438 -0.7 1.35196 4 spoof
76 2025-05-24 00:09:48.0925 6.723306 80.09438 -0.2 0.44448 4 spoof
77 2025-05-24 00:09:51.9576 6.723307 80.09438 1.2 0.96304 4 spoof
78 2025-05-24 00:09:56.2204 6.723307 80.09439 3.3 1.72236 4 spoof
79 2025-05-24 00:09:59.9758 6.723306 80.0944 4.8 1.33344 4 spoof
80 2025-05-24 00:10:02.8396 6.723313 80.09441 8.4 1.50012 4 spoof
81 2025-05-24 00:10:08.0767 6.723329 80.0944 2.5 6.05604 4 spoof
82 2025-05-24 00:10:12.17096 6.723332 80.0944 0.6 0.77784 4 spoof
83 2025-05-24 00:10:15.10092 6.723352 80.0944 2.2 5.0004 4 spoof
84 2025-05-24 00:10:18.84162 6.723344 80.09439 0.9 1.1112 4 spoof
85 2025-05-24 00:10:21.51872 6.723344 80.09439 0.9 1.1112 0 spoof
86 2025-05-24 00:10:24.4334 6.723344 80.09439 0.9 1.1112 0 spoof
87 2025-05-24 00:10:28.18418 6.723344 80.09439 0.9 1.1112 0 spoof
88 2025-05-24 00:10:32.15151 6.723294 80.09442 0.8 0.33336 5 spoof
Page 1 of 3
100 rows
226 records
Refresh
Data
Definition

# 4. RESULT AND DISCUSSION

## 4.4.Research Findings

### 4.4.1. Model Accuracy and Performance

The LSTM autoencoder model demonstrated strong performance in differentiating between legitimate and spoofed GPS signals. During evaluation, it achieved high accuracy rates with a low false positive rate. The model successfully learned normal trajectory patterns and identified anomalies based on reconstruction error, enabling precise spoofing detection without the need for labeled spoofing data during training.

### 4.4.2. Implication of the Findings

The ability of the model to detect spoofing events using only normal driving data confirms the viability of unsupervised learning approaches for GPS anomaly detection. This finding is significant, as it reduces the dependency on labeled spoofing datasets, which are difficult and risky to generate in real-world conditions. Additionally, the integration of the model with the IoT rover shows that such systems can be deployed on lightweight embedded platforms with minimal overhead.

### 4.4.3. Real-Time Processing and Efficiency

The system was capable of processing GPS and IMU data streams in real time. Incoming data was buffered, preprocessed, evaluated by the model, and a spoofing status was returned in under a second. This low-latency pipeline allowed immediate responses to spoofing attempts, including alerting the user and taking corrective action, such as halting movement or ignoring suspicious navigation commands.

### 4.4.4. User-Friendliness and Interaction

The mobile application and dashboard interface were designed to prioritize simplicity and usability. Users received real-time alerts, viewed live GPS telemetry, and accessed spoofing logs with minimal interaction. Feedback from

user testing indicated the application was easy to understand and provided clear visual cues about the system status, enhancing overall engagement and awareness.

### 4.4.5. Limitations and Areas for Improvement

- The system currently assumes spoofing is the only cause of GPS anomalies; however, environmental factors like tunnels or tall buildings can also cause GPS deviations.
- LSTM model operates best when trained on a consistent route; performance may degrade when used in new or highly variable driving conditions.
- Integration of the anomaly detection model directly on the microcontroller (Arduino Mega) is not feasible due to hardware constraints; reliance on a cloud server is necessary.

### 4.4.6. Overall Impact

The research successfully demonstrates a working prototype of a GPS spoofing detection system for autonomous vehicles and navigation platforms. It combines real-time anomaly detection, user alerting, and data logging into a cohesive solution that addresses a critical security challenge in modern GPS-dependent systems.

### 4.4.7. Model Performance Analysis

- Accuracy: >95% in detecting spoofed sequences.
- Precision: High, with very few false spoofing alerts.
- Recall: Adequate, with minimal missed spoofing cases.
- Mean Squared Error (MSE) thresholding proved reliable in separating normal vs. spoof data sequences.

### 4.4.8. Importance of Real-Time Processing

The success of the system relies heavily on its ability to process data streams with low latency. Delayed detection in the context of autonomous navigation could result in dangerous deviations. By maintaining sub-second inference times, the system ensures timely detection and appropriate intervention, making it suitable for real-world applications.

### 4.4.9. User Interaction and System Usability

End-user interaction via the mobile app was essential for transparency and control. The application not only provided live updates and alerts but also enabled users to view historical spoofing events and GPS activity. This transparency fosters trust and gives users the ability to understand and monitor system behavior without requiring technical expertise.

### 4.4.10. Challenges and Limitations

**Hardware Integration**

Challenge in Sensor Calibration: The GPS module and IMU sensors need to be accurately calibrated to provide reliable data, especially in detection of spoofing. Noise or misalignment in sensor readings could lead to false positives in spoofing detection.

Powering Components: The rover's components- Aduino Mega, GPS, IMU, WiFi Module, Motors-need a stable power supply for smooth operations. it is indeed tough work involving the management of batteries into those to keep running continuously, especially during long routes.

Robustness: The design and likely testing of equipment may be required to ensure operability for the rover under outdoor conditions such as dust, rain, and mechanical vibrations.

Data Taking and Quality: From Varied Environments: Taking data from multiple routes and conditions (urban areas with signal interference or rural areas with weak GPS signals) consumes time and usually creates inconsistency.

Spoofing Simulation: The generation of realistic spoofed GPS data that can be used to train the model or perform an experiment is not very feasible without having available equipment (HackRF), and simulated data lack the ability to completely cover real-world spoofing scenarios.

Data Amount: Gathering a good amount of high-quality data needed for training a robust machine learning model requiring multiple runs is quite logistically strenuous.


**Development of Machine Learning Model:**

Model Accuracy: Training a model to differentiate reliably normal and spoofed data is challenging in conditions where even slight spoofing attempts by the rover or environmental noises will mimic such anomalies.

Real-time Operation: Running a machine learning model on an Arduino Mega, which has limited computational abilities, may result in delays or require considerable optimization (e.g., model pruning).

Adaptability: The model may not generalize to new routes or newly developed spoofing techniques that were not part of the training, necessitating recurrent training.

**Real-Time Operation:**

**Computational Limitations**: The Arduino Mega lacks the processing power and memory to run deep learning models like LSTM autoencoders natively. As a result, the model must

be hosted externally (e.g., on a Flask server running in the cloud), and the Arduino simply transmits the sensor data.

**Latency**: Although offloading computation to a server reduces the hardware burden on the Arduino, it introduces **network latency**. This can affect the system's responsiveness when immediate action is required—such as halting the rover upon detecting a spoofing event. Future versions may benefit from using more capable edge devices (e.g., Raspberry Pi) to reduce dependence on remote computation.

**Mobile Application Integration:**

The mobile application forms a vital part of the system by ensuring real-time communication and user interaction. However, integrating the app with the IoT-based GPS spoofing detection system presents several challenges:

**User Interface Design**

Designing an intuitive and accessible user interface is critical to ensuring that non-technical users can effectively interpret spoofing alerts, understand system status, and react accordingly. Striking a balance between information richness and simplicity is a key design challenge.

**Reliable Connectivity**

Maintaining a stable WiFi connection between the mobile device and the rover system is not always feasible in dynamic environments. Areas with poor signal coverage, interference, or mobility-induced disconnects can hinder the reliable delivery of alerts and telemetry.

**Security**

Secure data transmission is essential for maintaining the integrity of spoof detection alerts and system data. The communication channel between the rover's Arduino and the mobile application must be encrypted and resistant to tampering, eavesdropping, or unauthorized access to ensure trustworthiness and privacy**.**

### 4.4.11. Future Directions

This research underscores the importance of viewing GPS spoofing detection as a dynamic and evolving system. The following directions are recommended for advancing this work:

**Integration with Edge AI Devices**

Employing high-performance edge computing devices such as the NVIDIA Jetson Nano or Raspberry Pi 4 would allow the deployment of more complex machine learning models directly on the rover. This reduces dependency on cloud-based processing and minimizes latency, making the system more responsive and autonomous.

**Enhanced Multimodal Sensor Fusion**

Although the current system integrates IMU data, future models should better leverage this data in combination with GPS anomalies. Correlating sudden acceleration or heading changes with GPS deviations can improve spoof detection accuracy by confirming anomalies through multiple sensor inputs.

**Realistic Spoofing Simulation Using HackRF One**

Acquiring a HackRF One device would enable the simulation of real-world GPS spoofing attacks in a controlled environment. This would overcome the limitations of synthetic data and improve model robustness by providing more diverse and realistic spoofing samples for training and testing.

**Expanded Data Collection Across Environments**

Increasing the diversity of the training dataset by collecting GPS and IMU data from urban centers, rural landscapes, coastal regions, and areas with dense signal obstructions will

enhance model generalization. Environmental variety ensures the system is adaptable to different terrains and signal conditions.

**Stronger Communication Security Protocols**

Developing and implementing secure communication protocols (e.g., SSL/TLS, token-based authentication) between the rover and the mobile application will help prevent spoofed alerts, data injection, or command manipulation. Securing data channels ensures trust and robustness in mission-critical systems.

**User Feedback and Environmental Testing**

Extensive field testing in varied environmental and weather conditions, along with structured user feedback, would yield critical insights for improving system usability, reliability, and accuracy. This is particularly essential for deployment in real-world autonomous vehicle operation

# 5. COMMERCIALIZATION ASPECTS OF THE RESEARCH

The GPS spoofing detection system developed in this research holds strong commercial potential, particularly within the growing fields of autonomous navigation, logistics, and IoT security. The system's modular design, real-time spoof detection capability, and mobile integration make it adaptable for a wide range of practical applications.

## 5.4.Market Potential

The demand for secure and reliable navigation technologies is increasing across industries including:

- Autonomous vehicles (both commercial and consumer-grade)
- Logistics and fleet management
- Agriculture (precision farming)
- Drone operations
- Military and defense navigation systems.

With GPS spoofing attacks becoming more frequent and sophisticated, governments and industries are actively seeking mitigation solutions. The GPS spoofing detection system aligns directly with this need, offering a timely and effective technological response.

## 5.5.Value Proposition

This research introduces a lightweight, scalable, and real-time GPS spoofing detection system that:

- Detects spoofing without requiring labeled attack data,
- Works with low-cost IoT hardware (e.g., Arduino Mega),

- Sends real-time alerts via a user-friendly mobile app,
- Logs anomalies to a secure cloud database (Supabase) for forensic and diagnostic purposes.

By combining anomaly detection, real-time feedback, and user interaction, the system enhances the security, trust, and operational reliability of GPS-based autonomous navigation.

## 5.6.Key Features for Commercial Success

To ensure successful adoption in the market, the system includes the following critical features:

- **Real-time spoof detection** using lightweight machine learning models.
- **Modular design** enabling integration with various types of IoT and GPS-enabled systems.
- **Cloud logging and analytics** via Supabase for long-term traceability and fleet monitoring.
- **Cross-platform mobile application** for real-time user alerts and data visualization.
- **Low hardware footprint** allowing deployment on microcontrollers and small-scale devices.

## 5.4.Commercialization Strategy

A stepwise commercialization strategy is proposed

- Prototype Demonstration: Deploy the system in pilot environments such as small fleets, agricultural vehicles, or drone platforms to showcase its effectiveness.

- Partnership Development: Collaborate with IoT hardware providers, GPS module manufacturers, and vehicle automation startups.

- Licensing and SDK Distribution: Offer the spoof detection model and mobile app as a Software Development Kit (SDK) or REST API to third-party developers.

- Customization Services: Provide tailored implementations for specific use cases (e.g., secure logistics, anti-theft GPS tracking).

- Crowdfunding or Incubation: Seek early-stage funding through innovation incubators, grants, or venture capital in the security or mobility sectors.

## 5.5. Competitive Analysis

| Feature | Proposed System | Traditional GPS Solutions |
|---|---|---|
| Real-time spoof detection | Yes | Often offline analysis |
| Mobile integration | Yes | Typically absent |
| Machine learning-based anomaly | Yes | Rule-based or static |
| Low hardware requirements | Arduino-class | High-end GPS hardware |
| Cloud integration for logging | Supabase | Limited or no support |

## 5.6. Future Commercial Opportunities

Beyond the initial deployment scope, the system can be extended and adapted for:

- **Drone fleets** in urban environments where GPS spoofing risks are high.
- **Autonomous delivery robots** navigating campuses or smart cities.
- **Insurance companies** seeking tamper-resistant GPS logs for claim validation.
- **Government infrastructure and critical transportation systems** require real-time GPS integrity monitoring.

# 6. BUDGET ALLOCATION

| Item | Estimated Cost (LKR) |
|---|---|
| Arduino mega | 4700 |
| Motor controller | 650 |
| NodeMCU | 900 |
| BMS charger protection model | 1500 |
| Digital ocean Linux server | 2500 |
| Voltage sensor | 500 |
| IMU sensor | 650 |
| GPS module | 900 |
| 5 x batteries | 2500 |
| **Total** | 17500 |

Table 6-1. Budget Allocation

# 7. GANTT CHART

| Task | June 2024 | July 2024 | Aug 2024 | Sept 2024 | Oct 2024 | Nov 2024 | Dec 2024 | Jan 2025 | Feb 2025 | Mar 2025 | Apr 2025 | May 2025 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Project Planning** | ● | ● | | | | | | | | | | |
| **Data Collection** | | ● | ● | ● | | | | | | | | |
| **Data Preprocessing** | | | ● | ● | ● | | | | | | | |
| **Model Development** | | | | ● | ● | ● | ● | | | | | |
| **Model Training and Testing** | | | | | ● | ● | ● | ● | | | | |
| **Backend Development** | | | | | | ● | ● | | | | | |
| **Mobile Application Development** | | | | | | | ● | ● | ● | | | |
| **Integration and Deployment** | | | | | | | | ● | ● | ● | | |
| **Testing and debugging** | | | | | | | | | ● | ● | | |
| **Performance Evaluation** | | | | | | | | | | ● | ● | |
| **Documentation** | | | | | | | | | | ● | ● | ● |
| **Final Review and Adjustments** | | | | | | | | | | | ● | ● |

Table 7-1. Gantt Chart

# 8. CONCLUSION

This research successfully designed, developed, and evaluated a real-time GPS spoofing detection system using an IoT-based rover, machine learning algorithms, and mobile application integration. The growing threat of GPS spoofing in autonomous systems necessitates innovative, lightweight, and adaptive solutions—and this work provides a promising response to that need.

By equipping a physical rover with a GPS module, IMU sensors, and Arduino Mega microcontroller, real-world trajectory data was collected under normal driving conditions. This data was preprocessed and used to train an LSTM autoencoder model, which learned typical motion patterns and detected anomalies indicative of spoofing attempts. Due to the computational limitations of the microcontroller, real-time data was sent to a Flask-based server hosted in the cloud, which processed it, applied the trained model, and returned spoof detection results.

A user-friendly mobile application was developed to deliver immediate alerts and visualize real-time and historical spoofing logs. Cloud storage (via Supabase) further enabled secure logging, remote access, and analysis of detection events.

The system demonstrated high detection accuracy, low latency response times, and strong user accessibility. It was tested with real vehicle data and spoof simulations, proving effective in identifying spoofed GPS behavior and responding promptly to secure autonomous navigation.

Despite current limitations—such as reliance on remote computation, limited generalization to new routes, and the need for improved connectivity and security—the project lays a strong foundation for scalable and commercially viable GPS spoofing defense systems.

Future extensions involving edge AI platforms, broader dataset collection, advanced spoof simulation tools (e.g., HackRF One), and enhanced security protocols will strengthen the system’s resilience and adaptability. As the reliance on autonomous systems grows, such a robust and real-time GPS anomaly detection framework will be critical in ensuring trustworthy navigation and user safety.

# REFERENCES


[1] T. Miller, I. Durlik, E. Kostecka, P. Borkowski, and A. Łobodzińska, "A Critical AI View on Autonomous Vehicle Navigation: The Growing Danger," *Electronics*, vol. 13, no. 18. 2024, doi: 10.3390/electronics13183660.

[2] S. Parkinson, "The Use of GPS Spoofing Attacks in Location Deception," 2024, pp. 181–196.

[3] T. Talaei Khoei, S. Ismail, and N. Kaabouch, "Dynamic Selection Techniques for Detecting GPS Spoofing Attacks on UAVs.," *Sensors (Basel).*, vol. 22, no. 2, Jan. 2022, doi: 10.3390/s22020662.

[4] P. Dini, A. Elhanashi, A. Begni, S. Saponara, Q. Zheng, and K. Gasmi, "Overview on Intrusion Detection Systems Design Exploiting Machine Learning for Networking Cybersecurity," *Applied Sciences*, vol. 13, no. 13. 2023, doi: 10.3390/app13137507.

[5] H. Kabir, M.-L. Tham, and Y. C. Chang, "Internet of robotic things for mobile robots: Concepts, technologies, challenges, applications, and future directions," *Digit. Commun. Networks*, vol. 9, no. 6, pp. 1265–1290, 2023, doi: https://doi.org/10.1016/j.dcan.2023.05.006.

[6] Á. Valera, F. Valero, M. Vallés, A. Besa, V. Mata, and C. Llopis-Albert, "Navigation of Autonomous Light Vehicles Using an Optimal Trajectory Planning Algorithm," *Sustainability*, vol. 13, no. 3. 2021, doi: 10.3390/su13031233.

[7] D. Kožović and D. Djurdjevic, "Spoofing in aviation: Security threats on GPS and ADS-B systems," *Vojnoteh. Glas.*, vol. 69, pp. 461–485, Apr. 2021, doi: 10.5937/vojtehg69-30119.

[8] Z. Zhang and X. Zhan, "Statistical analysis of spoofing detection based on TDOA: SPOOFING DETECTION BASED ON TDOA," *IEEJ Trans. Electr. Electron. Eng.*, vol. 13, Feb. 2018, doi: 10.1002/tee.22637.

[9] H. Habehh and S. Gohel, "Machine Learning in Healthcare.," *Curr. Genomics*, vol. 22, no. 4, pp. 291–300, Dec. 2021, doi: 10.2174/1389202922666210705124359.

[10] A. Abdulazeez, “Machine Learning Applications based on SVM Classification: A Review,” May 2021.

[11] Y. He, P. Huang, W. Hong, Q. Luo, L. Li, and K.-L. Tsui, “In-Depth Insights into the Application of Recurrent Neural Networks (RNNs) in Traffic Prediction: A Comprehensive Review,” *Algorithms*, vol. 17, no. 9. 2024, doi: 10.3390/a17090398.

[12] A. Mohanty and G. Gao, “A survey of machine learning techniques for improving Global Navigation Satellite Systems,” *EURASIP J. Adv. Signal Process.*, vol. 2024, no. 1, p. 73, 2024, doi: 10.1186/s13634-024-01167-7.

[13] P. Jiang, H. Wu, and C. Xin, “DeepPOSE: Detecting GPS spoofing attack via deep recurrent neural network,” *Digit. Commun. Networks*, vol. 8, no. 5, pp. 791–803, 2022, doi: https://doi.org/10.1016/j.dcan.2021.09.006.

[14] F. Wang, Y. Hong, and J. Ban, *Infrastructure-enabled GPS Spoofing Detection and Correction*. 2022.

[15] L. Alhoraibi, D. Alghazzawi, and R. Alhebshi, “Detection of GPS Spoofing Attacks in UAVs Based on Adversarial Machine Learning Model,” *Sensors*, vol. 24, no. 18. 2024, doi: 10.3390/s24186156.

[16] J. Campos *et al.*, “A Machine Learning Based Smartphone App for GPS Spoofing Detection,” 2020, pp. 235–241.

[17] A. Eshmawi, M. Umer, I. Ashraf, and Y. Park, “Enhanced Machine Learning Ensemble Approach for Securing Small Unmanned Aerial Vehicles From GPS Spoofing Attacks,” *IEEE Access*, vol. PP, p. 1, Jan. 2024, doi: 10.1109/ACCESS.2024.3359700.

[18] P. Mao *et al.*, “A GNSS Spoofing Detection and Direction-Finding Method Based on Low-Cost Commercial Board Components,” *Remote Sensing*, vol. 15, no. 11. 2023, doi: 10.3390/rs15112781.

[19] D. Handayani, W. Sediono, and A. Shah, *Anomaly Detection in Vessel Tracking Using Support Vector Machines (SVMs)*. 2013.

[20] O. Jullian, B. Otero, M. Stojilović, J. J. Costa, J. Verdú, and M. A. Pajuelo, “Deep Learning Detection of GPS Spoofing BT - Machine Learning, Optimization, and Data Science,” 2022, pp. 527–540.

[21] S. Dasgupta, M. Rahman, M. Islam, and M. Chowdhury, “A Sensor Fusion-Based GNSS Spoofing Attack Detection Framework for Autonomous Vehicles,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 23559–23572, 2022, doi: 10.1109/TITS.2022.3197817.

[22] M. Kamal, A. Barua, C. Vitale, C. Laoudias, and G. Ellinas, *GPS Location Spoofing Attack Detection for Enhancing the Security of Autonomous Vehicles*. 2021.

# APPENDICES