PRECISE SELF-LOCALIZATION AND IMPROVED OBSTACLE DETECTION FOR AUTONOMOUS VEHICLES (PRE-SAFE)

by

Ahmet Sinan Kalkan
Hasan Tarık Yumbul
Talha Osman Saraç

CSE4197 / CSE4198 Engineering Project report submitted to Faculty of Engineering
in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE

Supervised by:
Assoc. Prof. Müjdat Soytürk

Marmara University, Faculty of Engineering

Computer Engineering Department

2025

PRECISE SELF-LOCALIZATION AND IMPROVED OBSTACLE DETECTION

FOR AUTONOMOUS VEHICLES (PRE-SAFE)

by

Ahmet Sinan Kalkan
Hasan Tarık Yumbul
Talha Osman Saraç

CSE4197 / CSE4198 Engineering Project report submitted to Faculty of Engineering
in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE

Supervised by:
Assoc. Prof. Müjdat Soytürk
Sign

Marmara University, Faculty of Engineering

Computer Engineering Department

2025

ABSTRACT

The PRE-SAFE (Precise Self-Localization for Autonomous Vehicles) project addresses a key challenge in autonomous vehicle (AV) technology: achieving accurate and real-time localization without relying heavily on GPS. With the growing use of AVs and increasing accident statistics, there is a need for dependable perception and control systems. Our project aims to build a miniature autonomous vehicle prototype that integrates image processing, AprilTags, gyroscope and odometer data, object and unique landmark detection to enhance localization accuracy. The prototype also includes a web-based monitoring system that provides real-time updates on its location and detected objects. Our methodology is built on a layered architecture that uses cameras to recognize AprilTags and unique landmarks for precise positioning, then relies on an onboard gyroscope to estimate the vehicle's yaw, and two tire-RPM sensors supply a rough approximation of how far we've traveled. All of this sensor data is sent to a central control system for real-time analysis and logging. The system was designed to operate in controlled environments with limited space, offering a cost-effective and computationally efficient solution. Results indicate that PRE-SAFE precisely achieves localization within 10 cm error margins. The project lays a foundation for scalable AV systems suitable for education, research, and commercial adaptation.

ACKNOWLEDGEMENTS

We would like to express our gratitude to our advisor, Assoc. Prof. Müjdat Soytürk, for his continuous support, insightful feedback, and encouragement throughout the project. We thank the faculty members of the Computer Engineering Department at Marmara University for their guidance and collaboration. Lastly, we appreciate our fellow teammates for fostering a productive and creative work environment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Autonomous vehicle (AV) technology is rapidly evolving, with the promise of making transportation safer, more efficient, and accessible. However, critical challenges remain in ensuring AVs can reliably detect and interpret their environments—especially in areas where GPS signals are weak or unavailable. Among these challenges, precise localization and robust obstacle avoidance are among the most vital for safe navigation. The PRE-SAFE project addressed these issues through a cost-effective, scalable, and computationally efficient miniature AV prototype. Our system integrated visual markers, object detection, onboard sensor data, and real-time feedback mechanisms to improve localization accuracy and environmental awareness. In addition to solving technical challenges, PRE-SAFE also aimed to contribute to the larger discourse on practical and accessible AV implementations.

1.1 Problem Description and Motivation

Autonomous vehicles have the potential to significantly reduce traffic accidents, which claimed over 1.19 million lives globally in 2021 [1]. However, current AV technologies still face limitations in perception, localization, and real-time decision-making, particularly in GPS-incompatible or dynamic environments. In 2022, the United States alone reported 1,450 self-driving car accidents, the highest ever recorded [2]. These figures illustrate the urgent need for enhanced AV safety systems. Localization often relies on GPS, which is prone to signal loss and inaccuracy, particularly in urban or indoor environments. Therefore, PRE-SAFE was conceived to explore alternative and complementary methods such as visual-based localization using AprilTags and unique landmark detection, sensor fusion through a gyroscope and tire-RPM sensors, and robust obstacle detection with object recognition models. Enhancing AV perception will ultimately improve safety, boost public confidence, and accelerate the adoption of autonomous transportation.

1.2 Main Goal and Objectives of the Project

The primary goal of the PRE-SAFE project was to enhance the perception capabilities of autonomous vehicles by integrating multiple localization and detection techniques into a unified, real-time system. This involved the following specific objectives:

- Design and construct a miniature AV prototype capable of controlled, realistic movement.

- Achieve precise localization using AprilTags, unique landmark detection and onboard sensors.
- Successfully recognize multiple AprilTags in detection range (around 6 meters for an 8.4 cm tag).
- Train an object detection model to recognize a unique landmark and a classification model to detect the relative angle of the detected landmark accurately.
- Implement and train an object detection model to recognize a rare road hazard.
- Develop a real-time low-latency online monitoring platform.
- Incorporate augmented reality (AR) and 3D rendering to simulate datasets for object detection model training purposes.

Together, these objectives aimed to build a cost-effective and practical AV platform for research, education, and future scaling to real-world applications.

2. DEFINITION OF THE PROJECT

2.1 Scope of the Project

The PRE-SAFE project focused on developing a miniature autonomous vehicle equipped with precise self-localization and obstacle detection systems. The system integrated AprilTag and unique landmark detection, gyroscope and tire-RPM sensor data, and object detection algorithms to navigate within GPS-limited environments. Key features included real-time tracking, autonomous navigation in a 10 m² area, and IoT-based remote monitoring. The project also included AR and 3D render-based datasets to enhance detection capabilities.

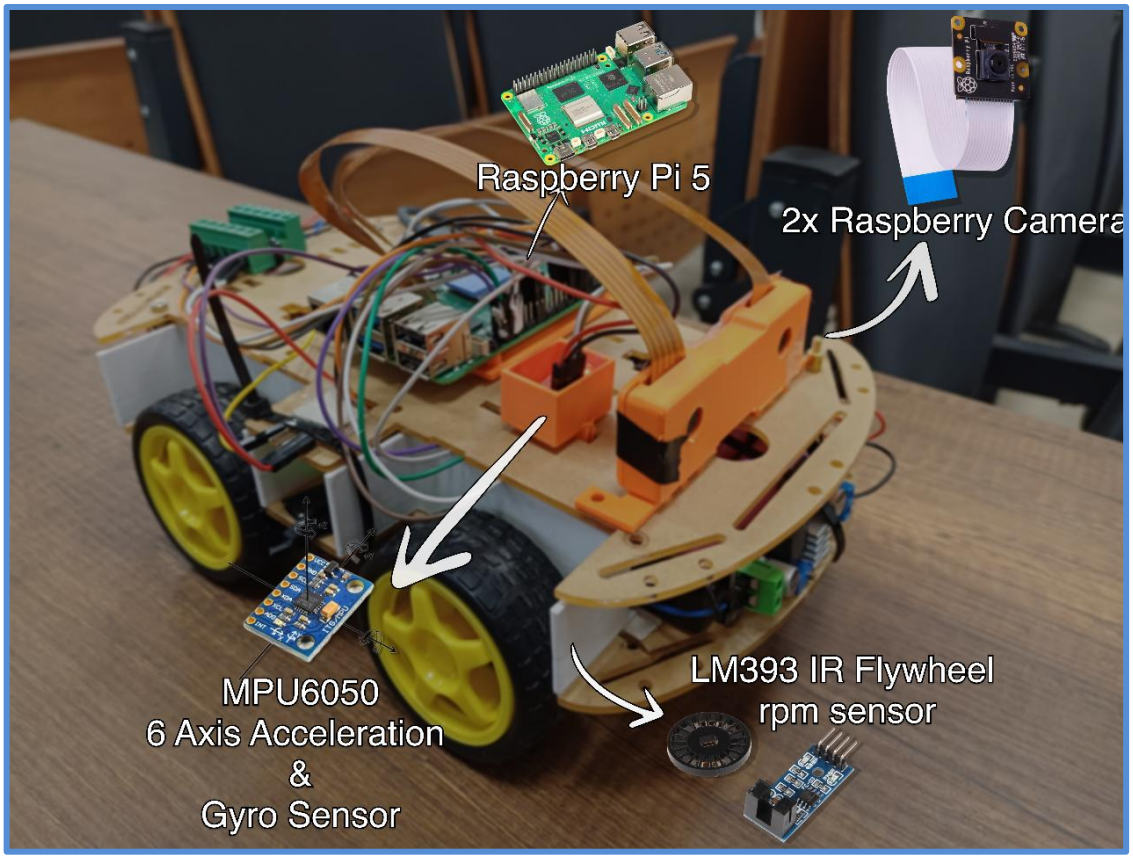

Figure 2.1 The Miniature Autonomous Vehicle Prototype

In-Scope:

- Self-localization using AprilTags and unique landmarks.
- Obstacle detection.
- Vehicle navigation and motion planning in small-scale environments.
- Web-based monitoring system.
- Augmented Reality and 3D render-based training datasets.

Out-of-Scope:

- Full-scale vehicle implementation.
- Cloud-based or long-range communication.
- High-end SLAM systems or LiDAR integration.

2.2 Success Factors

- The vehicle had to localize itself with less than 10 cm error using AprilTags, unique landmark detection and sensor fusion.
- Multiple AprilTags in detection range (around 6 meters for an 8.4 cm tag) had to be detected with an accuracy over 95%.
- The landmark detection model had to maintain over 90% accuracy while the angle classification model had to provide at least 60% accuracy.
- Rare road hazard detection model had to reach over 60% accuracy.
- Web monitoring platform had to deliver updates with less than 1 second delay.
- The AV had to detect AprilTags with 95% success.
- AR and 3D render-based datasets had to support our models to perform significantly better than the base model and other trained models with small real-world datasets.

2.3 Professional Considerations

- Engineering Standards: IEEE software documentation and UML modeling standards were followed.
- Source Control: Git and GitHub were used for version management and team collaboration.
- Sustainability: Energy-efficient hardware components and battery protections were implemented.
- Legal & Ethical: All tests were conducted in controlled environments with compliance to data privacy standards.

2.4 Literature Survey

Many approaches have been proposed in the field of autonomous navigation to enhance obstacle detection and localization for mobile robots and autonomous vehicles. The following studies represent notable methods that are compared with the techniques used in our project.

Image Processing for Localization

Hyunwoo Song et al. [3] introduced a localization technique for mobile robots that relies on image processing, specifically Harris corner detection and Hough transformations, to detect landmarks and convert image coordinates into a spatial coordinate system. This method enables reliable localization in GPS-denied environments such as bridges. Similarly, our project also operates in areas where GPS signals are limited. However, rather than relying solely on image features, our system utilizes AprilTags as visual markers and complements them with sensor data to improve localization accuracy across various environments.

Mapping and Sensor Fusion

Anas Charroud et al. [4] explore a comprehensive range of localization techniques used in self-driving cars, integrating GPS, Inertial Navigation Systems (INS), and sensor fusion methods that combine LiDAR, camera inputs, and GNSS data. This multi-sensor approach addresses challenges such as GNSS inaccuracy and odometry drift in dense urban settings. While our system adopts a more cost-effective and simplified design for small-scale deployment, it mirrors this approach by incorporating accelerometer data to refine positional estimates and mitigate single-sensor limitations.

Obstacle Detection

Both our study and the work of Arvind and Senthilnath [5] address the challenge of obstacle detection in autonomous vehicle environments. Their approach uses reinforcement learning—specifically Q-learning combined with a Multi-Layer Perceptron Neural Network (MLP-NN)—to enable the vehicle to adapt its behavior dynamically based on sensor inputs. Using ultrasonic sensors, their system can detect both static and moving obstacles, adjusting its path through learned actions such as stopping or turning. In contrast, our project focuses solely on the obstacle detection component. We trained a YOLO-based model to identify a potential rare obstacle in the vehicle's environment, such as an overturned truck, but this functionality has not yet been integrated into the car's autonomous navigation system.

IoT Integration and Real-Time Communication

Krasniqi and Hajrati [6] propose a V2X-based IoT infrastructure to support fully autonomous vehicles, utilizing 5G and DSRC technologies to share real-time data between vehicles and roadway infrastructure. Their system enables vehicles to broadcast key safety metrics such as speed, location, and heading to improve coordination and

safety on crowded roads. Our implementation adopts a more constrained but functional IoT model, focused on communication between a centralized monitoring system and the autonomous vehicle. Instead of full V2X capabilities, our system emphasizes essential telemetry—such as position updates and obstacle alerts—suitable for our low-power prototype. This lightweight and efficient setup enables real-time data sharing without requiring advanced communication infrastructure, aligning with the resource limitations and scope of our project.

Table 2.1 summarizes the comparison of these studies and highlights PRE-SAFE's hybrid, lightweight, and scalable nature.

Projects	Localization with AprilTag	Localization with Sensors	Object Detection	Monitoring & Remote Control	IOT
Localization Method Based on Image Processing for Autonomous Driving of Mobile Robot in the Linear Infrastructure	NO	YES	NO	NO	NO
Localization and Mapping for Self-Driving Vehicles	NO	YES	YES	NO	NO
Autonomous Vehicle for Obstacle Detection and Avoidance Using Reinforcement Learning	NO	YES	YES	NO	NO
Use of IoT Technology to Drive the Automotive Industry from Connected to Full Autonomous Vehicles	NO	NO	NO	NO	YES
Precise Self-Localization and Improved Obstacle Detection for Autonomous Vehicles (PRE-SAFE) (Our Project)	YES	YES	YES	YES	YES

Table 2.1 Comparison Between Surveyed Projects and PRE-SAFE

3. SYSTEM DESIGN AND SOFTWARE ARCHITECTURE

3.1 Project Requirements

3.1.1 Functional Requirements

Object Detection

The system was designed to detect and classify objects in the autonomous vehicle's environment with high accuracy. Visual data obtained from the onboard cameras was processed using a custom-trained YOLO model. Detected objects (AprilTags, unique landmark) were labeled and their relative positions and dimensions were calculated.

Localization

The system estimated the vehicle's position with an error margin below 10 centimeters. AprilTag detections and accelerometer data were fused to compute the real-time (X, Z) coordinates of the vehicle. When AprilTag detections failed, the system relied solely on wheel speed sensor data for short-term position estimation. Localization results were visualized on a user interface map for monitoring purposes.

Online Monitoring Platform

A web-based monitoring platform was developed to continuously display the vehicle's live location, movement status, and camera feed. The system allowed users to observe real-time updates through a user-friendly interface and supported features such as remote control and data logging. Any irregularities during operation were reflected on the interface to assist with quick diagnostics.

3.1.2 Nonfunctional Requirements

Performance

The system completed object detection and localization computations within 100 milliseconds. The online monitoring interface maintained an update latency below 1 second during real-time operations.

Reliability

Localization accuracy stayed within a 10 cm threshold for over 95% of test cases, even in GPS-denied environments.

Usability

The monitoring platform featured an intuitive design. First-time users were able to understand core functionality within 5 minutes. Clear error messages and a help section were included.

Security

All communication between the vehicle and monitoring system was encrypted. Access to the monitoring dashboard required user authentication.

Maintainability

The modular software architecture allowed independent updates to components (e.g., object detection, localization) without disrupting other functionalities. Logs were manually generated to assist with fault diagnosis and system debugging.

Portability

The user interface was tested on multiple browsers like Chrome, Firefox, Edge and achieved compatibility these platforms.

3.2 System Design

3.2.1 UML Use case Diagrams

Vehicle

The diagram below outlines how the user initiates core system threads to enable sensing, navigation, and control functionalities.

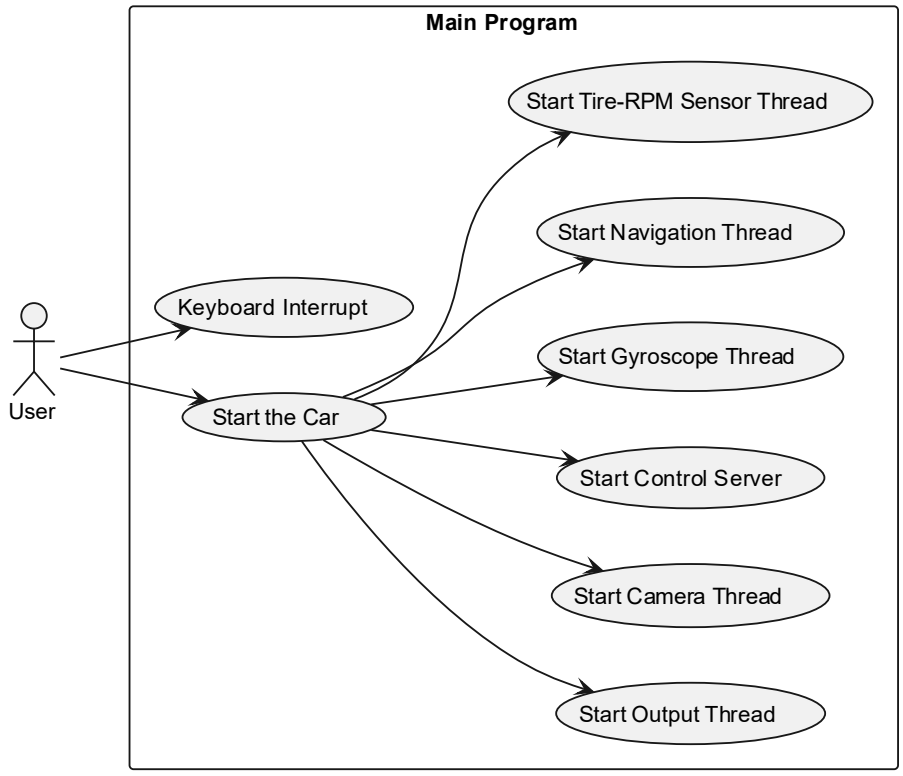

Figure 3.1 Main Program

The following diagram presents how data for terminal output is collected and published by the system.

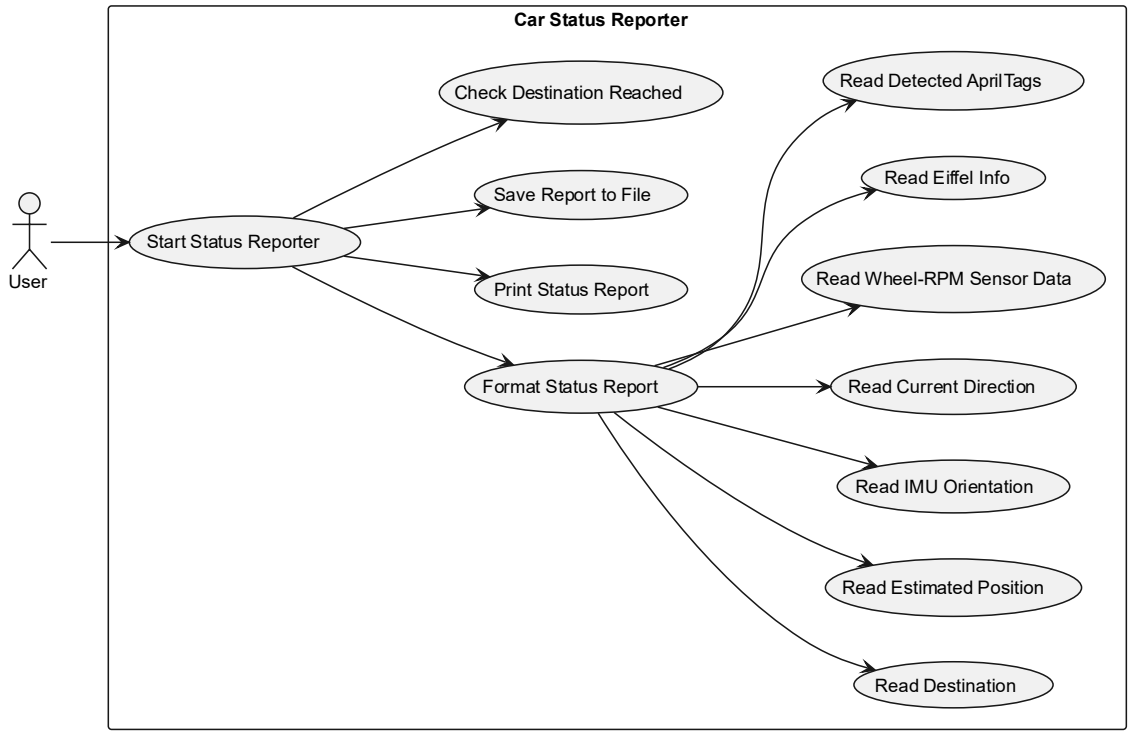

Figure 3.2 Status Reporter Module

This diagram illustrates how video frames are captured from the camera and shared with dependent modules.

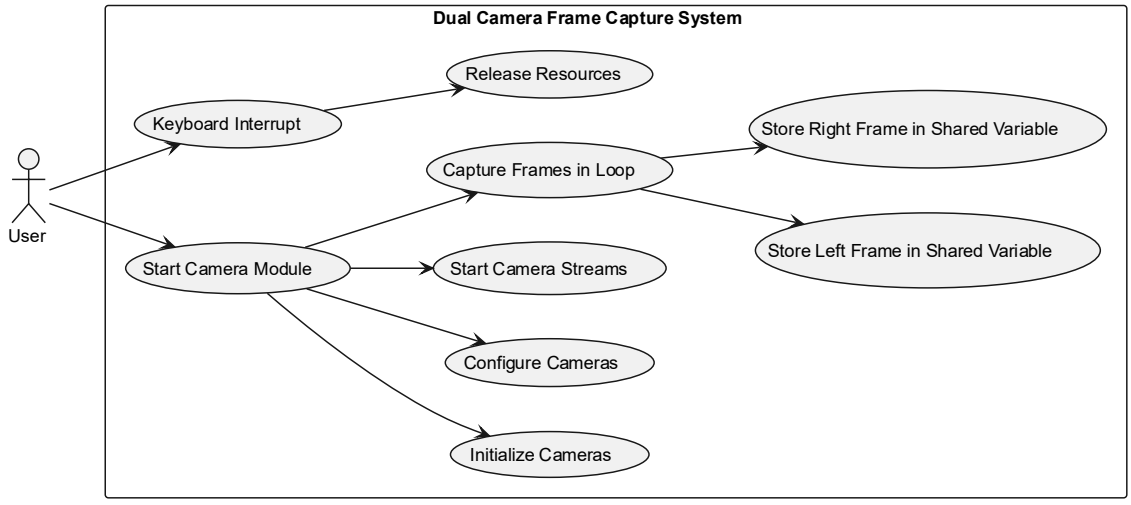

Figure 3.3 Frame Capture Module

The diagram below depicts how the backend server manages command inputs and

live data streaming for user interaction.

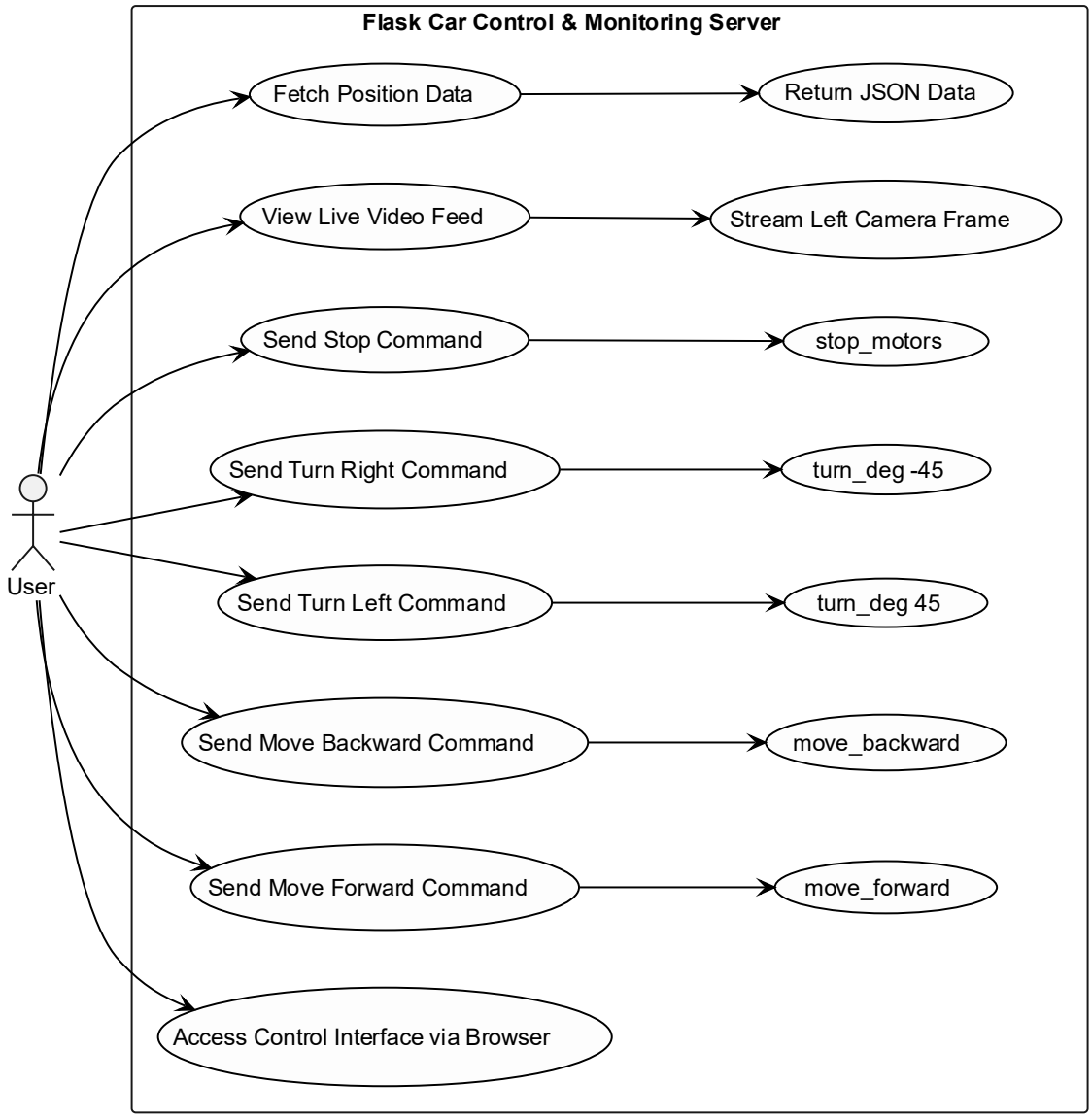

Figure 3.4 Web Interface Backend Module

This diagram explains how the gyroscope is calibrated and initialized to assist in directional awareness.

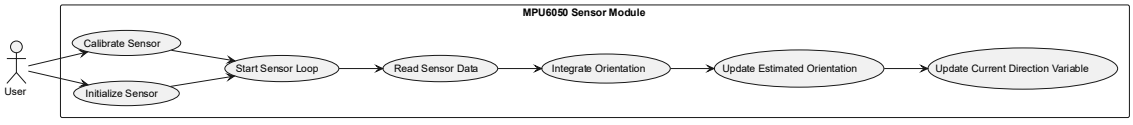

Figure 3.5 Gyroscope Interpreter Module

Depicted here is how angular data from the gyroscope is processed and classified

to estimate the vehicle's heading.

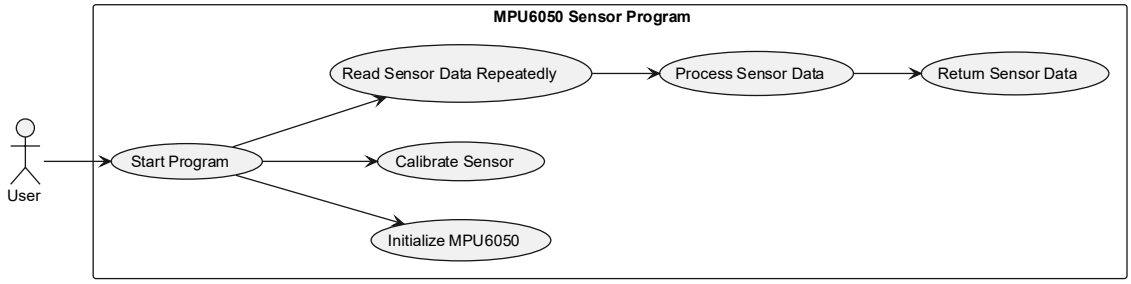

Figure 3.6 Gyroscope Module

Shown here is the interaction between the navigation logic and other modules for path planning and position updates.

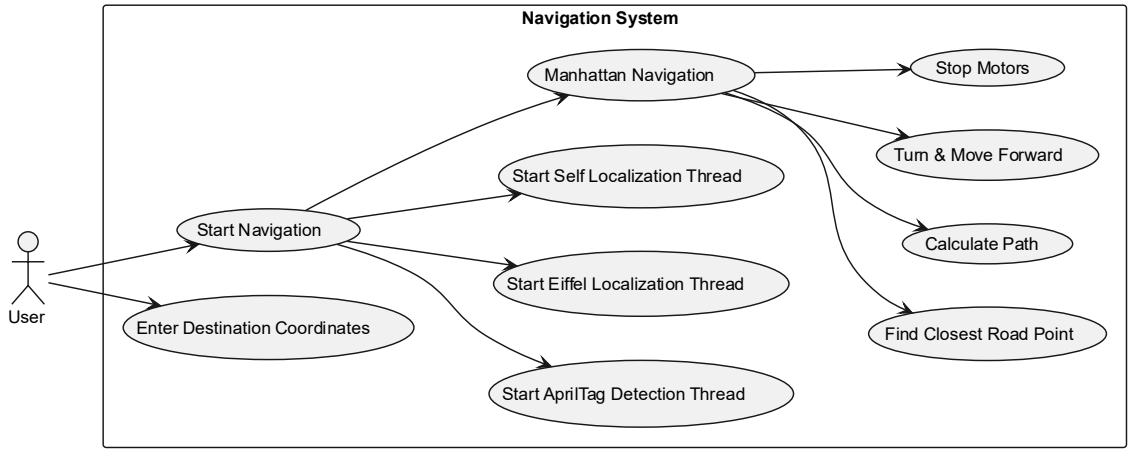

Figure 3.7 Navigation Module

This use case diagram describes how the AprilTag detection system identifies visual markers for localization.

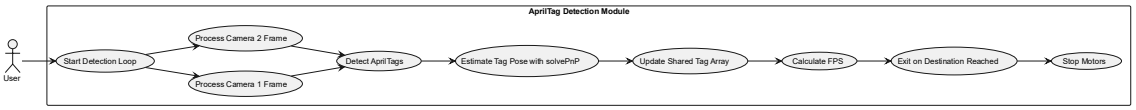

Figure 3.8 AprilTag Detection Module

The diagram below presents how the system uses Eiffel Tower miniature-based methods for determining vehicle position.

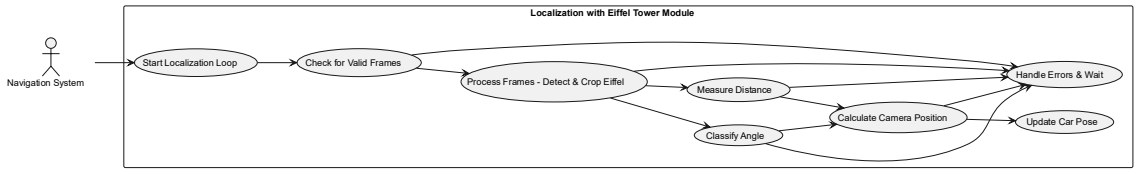

Figure 3.9 Miniature Eiffel Tower-based Localization Module

This diagram outlines how visual data is processed to detect the presence of the Eiffel Tower miniature for localization cues.

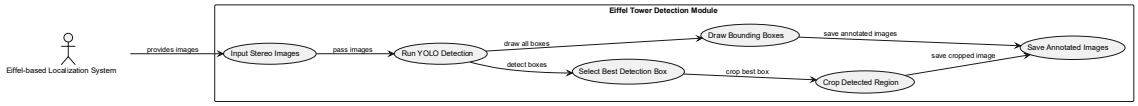

Figure 3.10 Miniature Eiffel Tower Detection Module

The use case diagram here illustrates how the angle between the vehicle and the detected Eiffel Tower miniature is estimated.

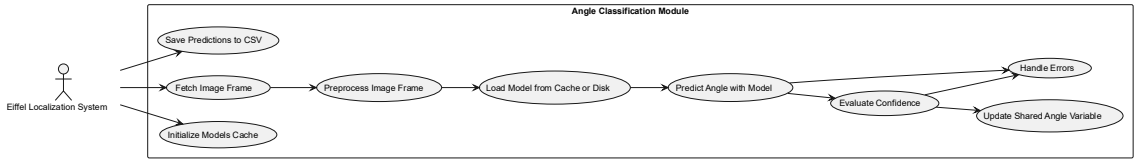

Figure 3.11 Miniature Eiffel Tower Angle Classification Module

Shown here is how single-camera input is used to estimate distance to the Eiffel Tower miniature based on visual cues.

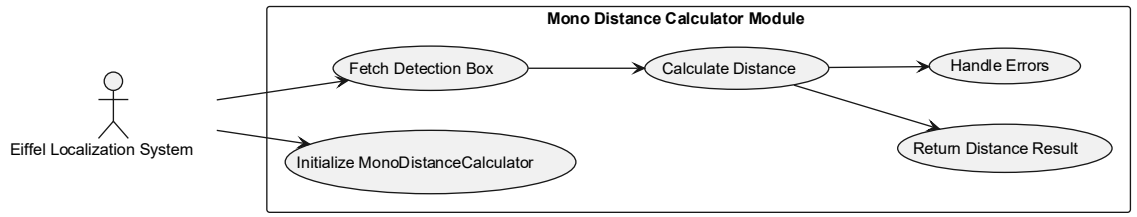

Figure 3.12 Miniature Eiffel Tower Mono Distance Calculation Module

This diagram presents how different positional inputs (e.g., visual, inertial) are combined to produce a refined location estimate.

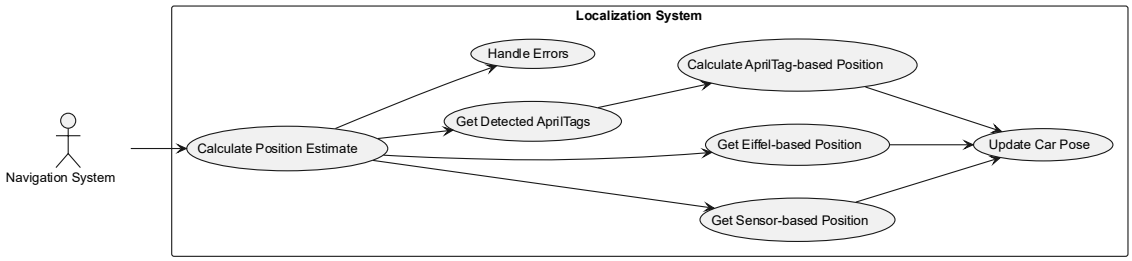

Figure 3.13 Localization Module

The following diagram depicts how the system performs navigation using a Manhattan grid-based road structure.

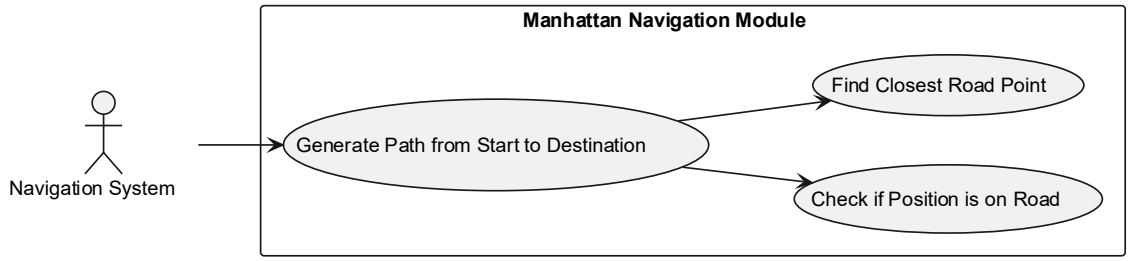

Figure 3.14 Manhattan Navigation Module

This use case shows how a navigable simple road network simulation is generated.

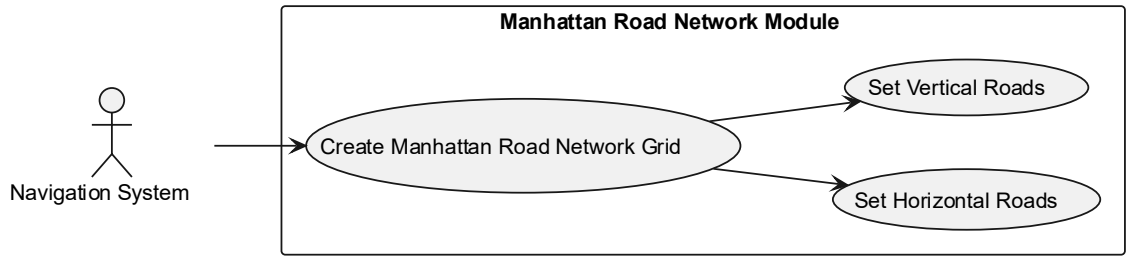

Figure 3.15 Manhattan Road Network Generation Module

The diagram below illustrates how wheel encoder data is gathered and reported to support odometry.

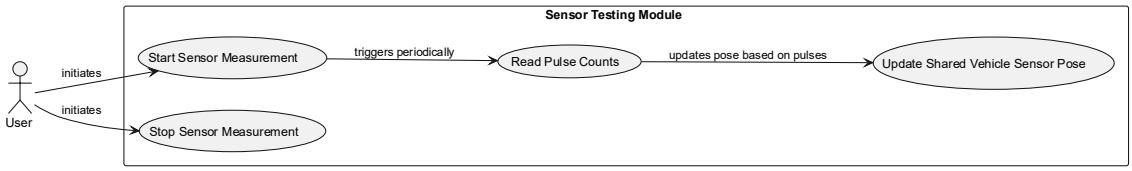

Figure 3.16 Wheel-RPM Sensor Module

This diagram explains how motor control signals are generated based on high-

level motion commands.

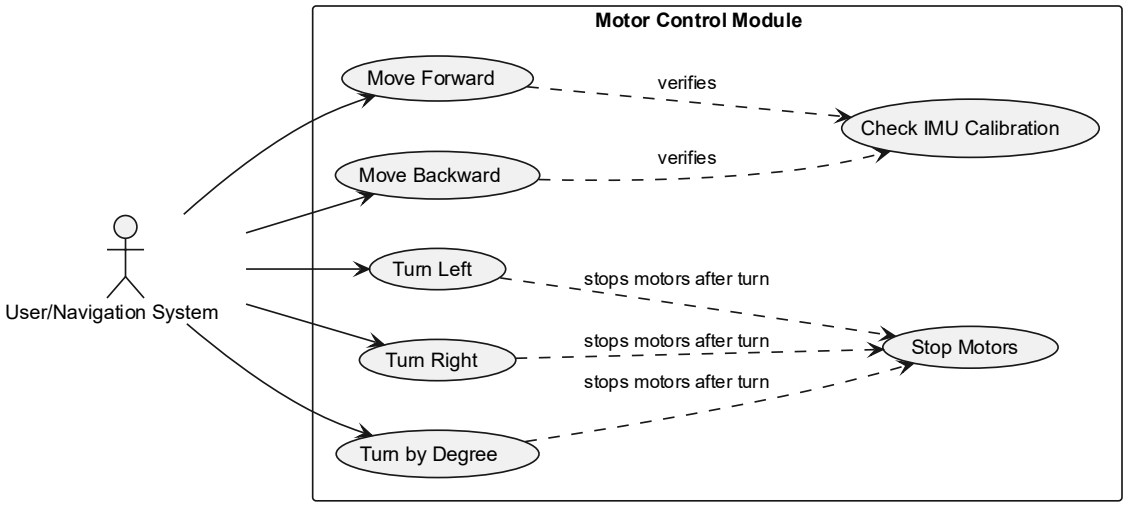

Figure 3.17 Motor Control Module

User Interface

Presented here is how the user interacts with the web-based interface to send commands and receive live updates from the vehicle.

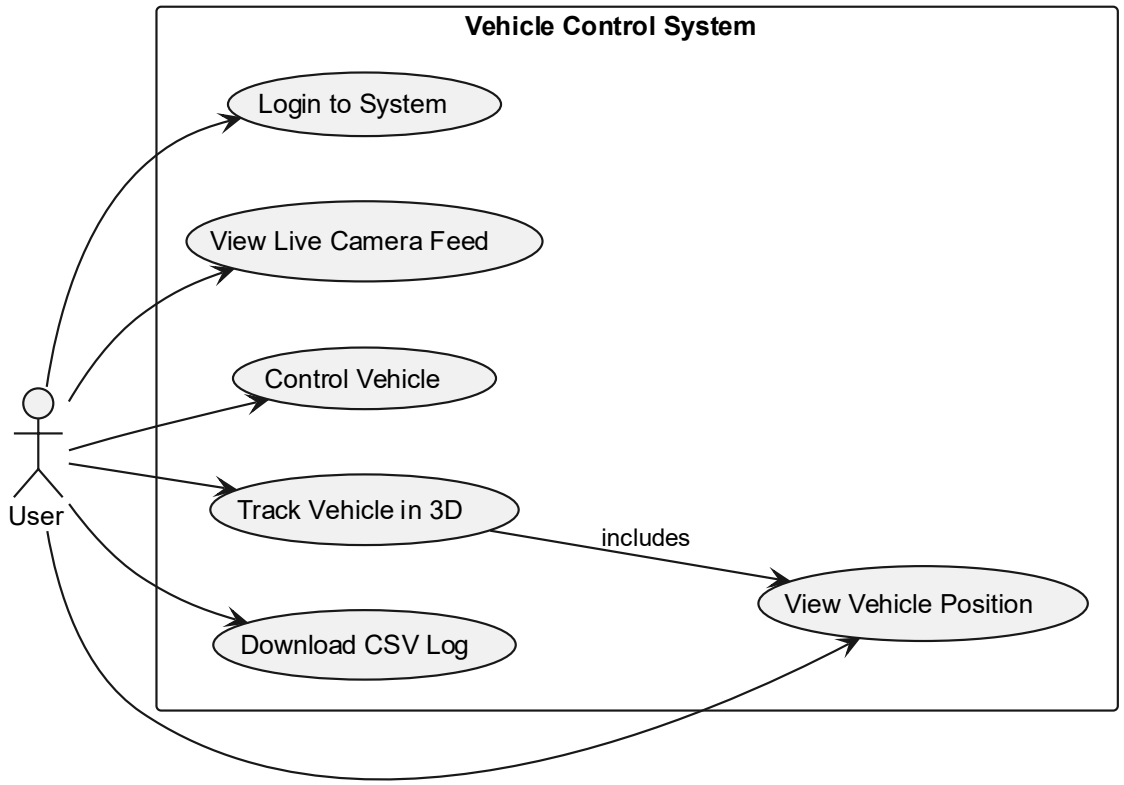

Figure 3.18 Frontend

3.2.2 UML Class Diagrams

Vehicle

The diagram below presents the structure and relationships between core backend modules responsible for sensing, control, navigation, and data communication.

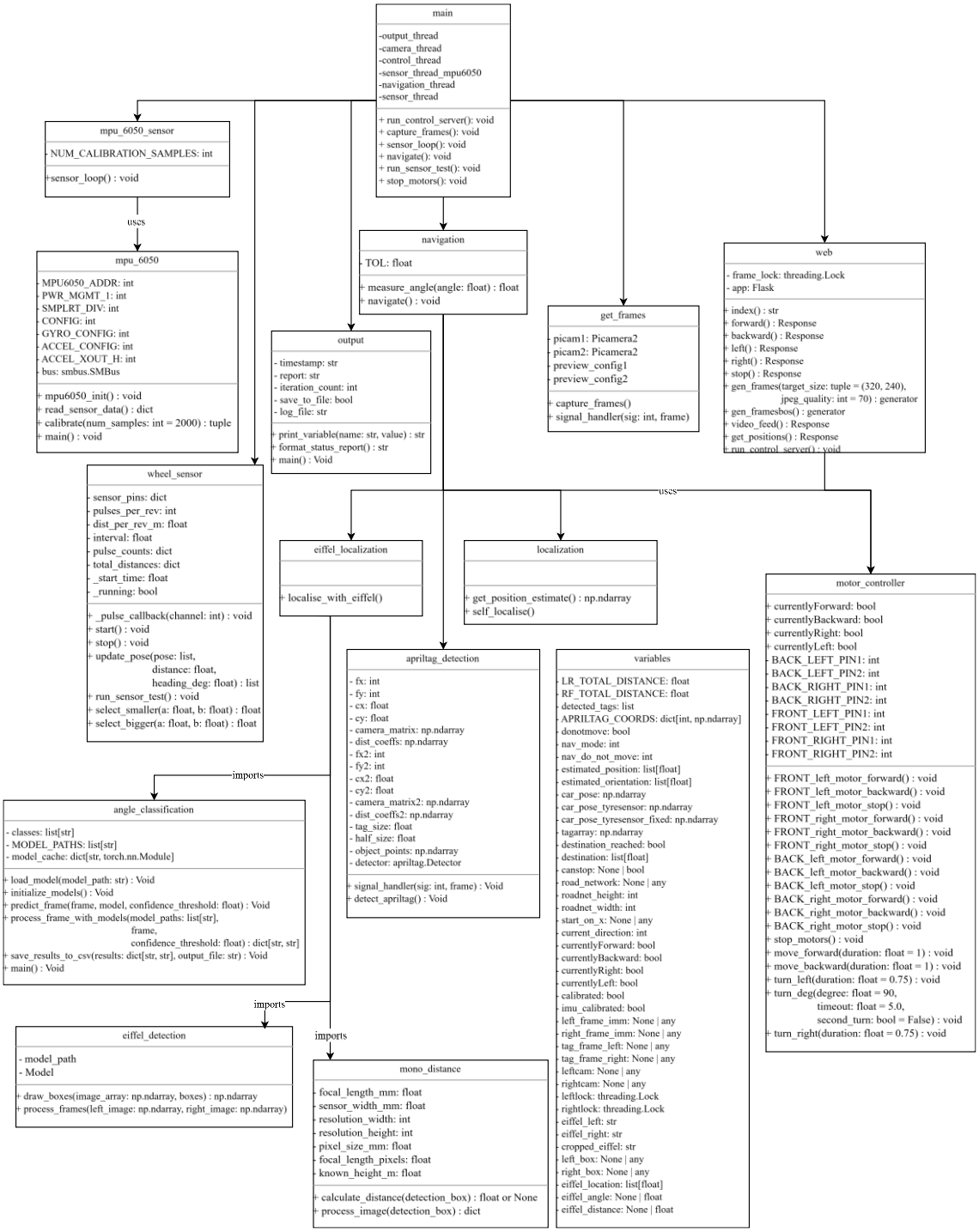

Figure 3.19 Vehicle System Structure

User Interface

This diagram illustrates the frontend architecture, highlighting the components involved in user interaction, data visualization, and vehicle control via the web interface.

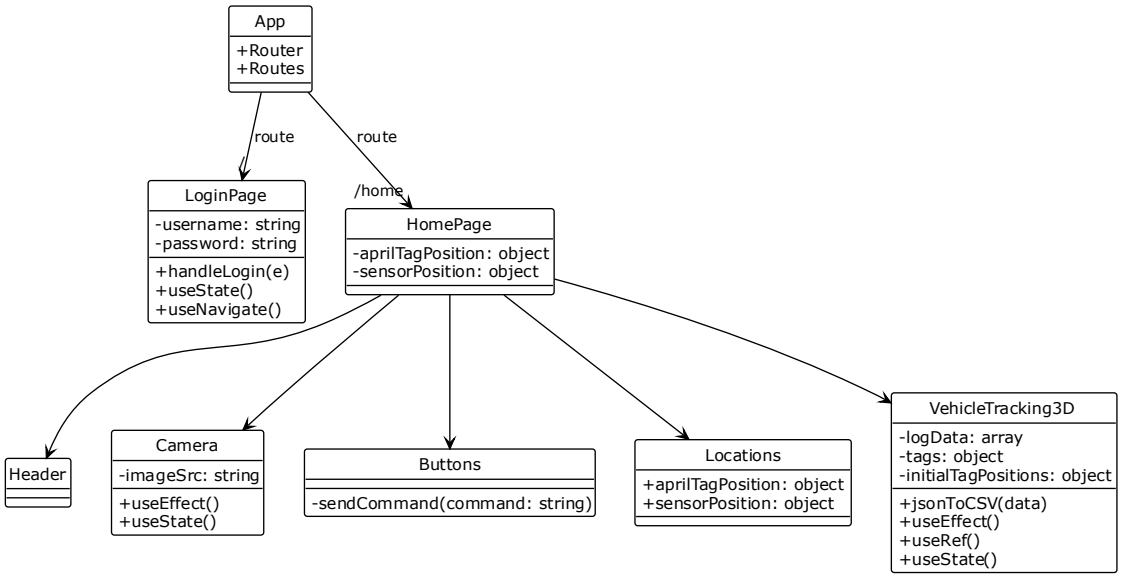

Figure 3.20 User Interface System Structure

3.2.3 User Interface

Login Panel Interface

A secure login screen was implemented to restrict unauthorized access to the Vehicle Control System. Users were required to enter a valid username and password before accessing system features such as live tracking, remote control, and data visualization. This authentication mechanism helped ensure safe and responsible use of the platform during testing and demonstrations.

Figure 3.21 User Interface Login Page

Control Panel Interface

The control panel interface provided real-time visual monitoring and control of the autonomous vehicle. On the left, a live camera feed from the vehicle's onboard camera was displayed, allowing users to observe the physical environment. On the right, a 3D simulation environment was rendered using the Three.js library, where the positions of AprilTags and the vehicle were visualized dynamically.

The 3D interface showed both AprilTag-based (red model) and wheel speed sensor-based (blue model) localization results, enabling comparative analysis. Directional control buttons were provided for manual operation of the vehicle (left, right, forward, backward). Below the visuals, real-time X, Y, Z coordinates were shown for both localization methods, and a "Log Data" button allowed position tracking information to be stored for analysis. This interface served as a central hub for interacting with and evaluating the perception and navigation modules of the system.

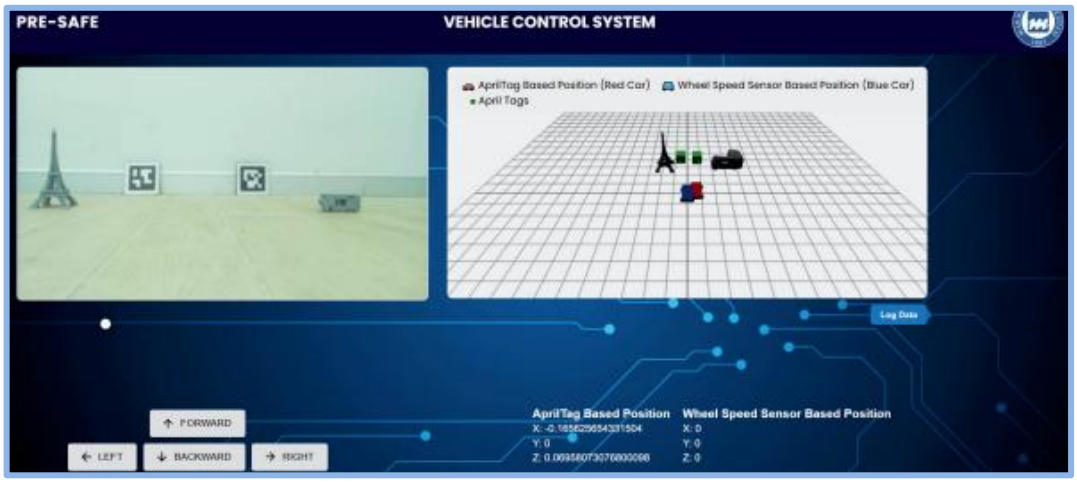

Figure 3.22 User Control and Real-time Status Tracking Interface

3.2.4 Test Plan

The test plan outlined the approach, tools, environment, and timeline for testing activities conducted during the PRE-SAFE project. It ensured a comprehensive evaluation of all major components, including the development of the autonomous vehicle, the implementation of self-localization via AprilTag-based visual positioning fused with speed sensor data, and the integration of object recognition using YOLO and ResNet models. The plan also covered the assessment of system functionalities such as route planning, motor control, obstacle detection, and real-time monitoring through a custom user interface. The test environment incorporated both real-world and augmented datasets to evaluate performance under various scenarios and lighting conditions.

3.3 Software Architecture

- The software architecture of the PRE-SAFE system was designed as a modular and cyclic structure, consisting of four main stages: Perception, Planning, Control, and Monitoring. These components operated in a continuous loop to enable real-time autonomous navigation and decision-making.
- Perception modules processed visual data from onboard cameras using custom-trained object detection models (YOLO) and landmark recognition techniques. AprilTags and wheel-RPM sensor were identified to support self-localization and obstacle detection. The perception output served as the foundation for downstream modules.
- Planning modules received environmental and positional data from the perception layer and executed route optimization, navigation logic, and position

correction. These algorithms determined the most efficient and safe path for the vehicle based on its current state and surroundings.

- Control modules translated the planned path into physical actions. Motor commands were issued to execute movement, turning, and braking operations. The system ensured that control signals were responsive and accurately reflected the decisions made in the planning stage.
- Monitoring modules tracked the vehicle's real-time status and visualized data through a web-based user interface. Users were able to observe vehicle movements, positioning, and sensor feedback via the online platform. This layer also supported remote control and diagnostics.
- The entire architecture was designed to function in a closed-loop system, where perception informed planning, planning triggered control actions, and monitoring provided real-time feedback for verification and adjustment.

4. TECHNICAL APPROACH AND IMPLEMENTATION DETAILS

The PRE-SAFE project incorporates both hardware and software elements. This section outlines the tools, technologies, and implementation applied to each subsystem.

4.1 Hardware Components

- **Raspberry Pi 5:** The main onboard computer used for running all image processing and machine learning tasks. Chosen for its powerful cores.
- **Raspberry Camera Modules 2.1 and 1.3:** Capture a live video feed for object detection and AprilTag recognition, positioned to provide forward facing coverage with cameras spaced five centimeters apart.
- **DC Motors & Motor Drivers:** Motors enable vehicle movement and turning via motor drivers are controlled via GPIO pins.
- **Battery Pack:** The battery pack powers the vehicle and includes over-current fuses.
- **Gyroscope (MPU6050):** Provides yaw information for dead reckoning and localization refinement.
- **RPM Measurement sensor (LM392):** Measures the motor's rotations per minute to calculate vehicle speed and distance traveled.

Each hardware component was integrated on a plexiglass chassis with 3D printed parts to ensure structural stability and modularity.

4.2 Software Components and Tools

- **Python:** Main language used for scripting perception, Planning, control logic, and server communication.
- **OpenCV:** Handles image capture and preprocessing
- **YOLO11:** Object detection model, trained on two custom datasets; one for detecting overturned trucks, another for detecting a unique landmark, which is an Eiffel Tower miniature in our case.
- **Flask:** a web server framework to serve APIs for vehicle status and controls, and to stream camera footage over Http
- **HTML/CSS/JavaScript:** Used for building the frontend dashboard for the monitoring system.
- **Three.js:** Utilized for 3D visualization of the vehicles estimated positions.

4.3 Data Structures and Algorithms

- **Tag Dictionary:** Maps detected AprilTags to known coordinates for rapid localization.
- **Dead Reckoning Algorithm:** Combines gyroscope data with rpm sensor to estimate vehicle displacement.
- **Object Classification Pipeline:** Our object classification pipeline works in two stages. First, a YOLO model locates unique landmark. Then, a ResNet18 model analyzes these landmarks to detect the object's orientation.
- **Path Adjustment Algorithm:** The car navigates using a coordinate-based system, where it determines its axis and executes movements sequentially. The algorithm commands the vehicle to travel a specific distance forward, perform a turn, and then travel another distance to complete the maneuver.

4.4 Networking and Communication Details

- **API (Flask):** We build APIs that provide snapshot data like current coordinates, Current direction and detected tags.
- **Authentication:** Basic password protection is enabled on the interface, and local IP used for access the control system.

4.5 Operating System and Development Environment

- **Operating System:** Debian-based OS running on Raspberry Pi 5, offering lightweight performance with access to necessary GPIO and peripheral libraries.
- **Multithreading and Synchronization:** Multiple Python threads are used to concurrently handle tasks such as image processing, telemetry updates, and control logic. To ensure safe access to shared resources, Python's *threading.Lock* mechanism is used. This prevents race conditions by locking critical variables during updates (e.g., shared sensor data and tag detections).
- **Development Tools:** Visual Studio Code, Cursor, Git for version control, Unity for automatic data generation.

This implementation demonstrates a well-structured, modular approach integrating real-time sensing, visual perception, and online monitoring—all within the constraints of embedded systems development.

5. SOFTWARE TESTING

The software testing phase was crucial to validating the functionality, accuracy, and robustness of each component of the PRE-SAFE system. Tests were conducted in a structured environment using controlled scenarios within a 10 m² indoor testing area.

5.1 Testing Methodology

Each subsystem of the project was tested independently before full system integration. Testing followed a black-box approach for the web interface and a gray-box approach for sensor and control modules. Key tools included logging scripts, latency timers, and simulation environments with repeatable setups.

5.2 Test Scenarios and Results

5.2.1 Localization Accuracy Testing

- Objective: To verify vehicle positioning accuracy with AprilTags and dead reckoning.
- Method: Compare reported coordinates with known ground truth at fixed intervals.
- Result: Localization error remained below 10 cm in all test runs; success rate of 8.4x8.4 cm tag detection within 6 m radius: 97%.

5.2.2 Eiffel Tower Miniature Detection Testing

- Objective: Evaluate the performance of the YOLO-based Eiffel Tower miniature detection model.
- Method: Test the model on a dataset composed of real-world images of the Eiffel Tower miniature taken from various angles, lighting conditions, and distances.
- Result: The model that was trained on 70% 3D-rendered and 30% real Eiffel Tower miniature images achieved 91.7% detection accuracy

5.2.3 Eiffel Tower Miniature Angle Classification Testing

- Objective: Test the effectiveness of a ResNet18-based model in classifying the viewing angle of the Eiffel Tower miniature.
- Method: The model was evaluated using a dataset of real images, each labeled with one of ten discrete viewing angles (e.g., 0°, 15°, …, 165°), omitting the ones that look identical to a previously included angle.

- Result: The ResNet18 model trained on the mixed dataset of 20% real and 80% 3D-rendered images of Eiffel Tower miniature achieved 64.83% classification accuracy.

5.2.4 Obstacle Detection Testing

- Objective: Evaluate the accuracy of the AR-trained obstacle detection model.
- Method: Use a test set of real overturned truck images to test the AR-trained obstacle detection model.
- Result: Model trained fully on AR dataset achieved 68.18% accuracy while YOLO11m model that trained with COCO (Common Objects in Context) dataset achieved 31.82% accuracy.

5.2.5 Monitoring System Performance

- Objective: Measure latency and uptime of the monitoring interface.
- Method: Simulate continuous operation and log delays and disconnections.
- Result: System maintained below 100ms latency with 99.2% uptime over a 30-minute test window.

5.2.6 Manual Control

- Objective: Test manual control function.
- Method: Manually interfere with the vehicle's control system using the user interface.
- Result: Manual control is accessible directly after the autonomous operation completes in all cases.

6. BENEFITS AND IMPACT

The PRE-SAFE project offers several technical, scientific, economic, and societal benefits, making it a valuable contribution to the field of autonomous vehicle systems.

6.1 Benefits and Implications

- **Enhanced Safety:** By improving perception and localization, the system contributes to reducing the risk of accidents, especially in environments where GPS is unreliable.
- **Accessible Research Platform:** PRE-SAFE provides a cost-effective and modular platform suitable for academic institutions, hobbyists, and developers to experiment with autonomous navigation.
- **Educational Value:** It serves as a hands-on tool for teaching topics such as robotics, sensor fusion, embedded systems, and machine learning.
- **Scalable Design:** The architectural choices and modular software components pave the way for scaling to larger systems or real-world autonomous vehicle platforms.

6.2 Scientific Impact

PRE-SAFE introduces a practical hybrid localization method combining AprilTags, landmark detection and sensor feedback. Its successful integration and real-time performance may inspire further academic studies on lightweight AV solutions for resource-constrained environments.

6.3 Economic, Commercial, and Social Impact

- **Prototype Readiness:** PRE-SAFE is a working prototype that can evolve into a commercial-grade testing kit.
- **Mobility Enhancement:** This type of system could eventually be adapted for mobility aids or delivery robots, benefiting socially vulnerable groups.
- **Sustainable Practices:** Emphasis on energy-efficient design and locally sourced components supports environmentally conscious development.

6.4 Potential Impact on New Projects

- **Project Incubation:** PRE-SAFE contributes to ongoing research in smart vehicle systems, AR-assisted data generation, and IoT integration.
- **Curriculum Development:** The modular and interdisciplinary nature of the project encourages its inclusion in university-level embedded systems courses.

6.5 Impact on National Security

- **Infrastructure Monitoring:** With adaptations, similar AVs could be deployed in border or facility surveillance applications, enhancing national security through automation.

In summary, PRE-SAFE is not only a technical achievement but also a platform with tangible educational, commercial, and societal potential.

7. CONCLUSION AND FUTURE WORK

7.1 Conclusion

The PRE-SAFE project successfully delivers a miniature autonomous vehicle system that demonstrates precise self-localization and real-time object detection while also delivering an obstacle detection model capable of recognizing a rare road hazard. Through the integration of visual markers (AprilTags), gyroscope and tire-RPM sensor-based motion estimation, YOLO11 and ResNet18-driven object recognition and classification, the vehicle can navigate GPS-limited environments while maintaining high accuracy and responsiveness. The addition of a web-based monitoring interface enhances transparency and usability, enabling remote observation and control in real-time.

One of the key strengths of the project lies in its modular and scalable architecture. The system is lightweight and cost-effective, making it suitable not only for demonstration and educational purposes but also as a foundational platform for future commercial or research applications. The hybrid localization strategy proved to be reliable under varied conditions, with error margins consistently within acceptable bounds.

Despite its success, the project also revealed areas for improvement. While the YOLO-based detection system worked effectively, its performance could be further optimized using more lightweight or task-specific models for embedded environments.

7.2 Future Work Suggestions

- Integrate lightweight object detection models (e.g., YOLO-Nano, MobileNet) for faster inference on embedded hardware.
- Generate AR-aided simulations to rapidly cover dynamic scenarios, such as moving obstacles or weather conditions.
- Improve dead reckoning algorithms by incorporating gyroscope data and advanced filtering (e.g., Kalman filters).
- Implement cloud-based data logging and monitoring to enable long-term performance analysis.
- Enhance user interface with multi-vehicle monitoring and control features.

Overall, PRE-SAFE represents a meaningful step forward in low-cost autonomous navigation research and sets a strong foundation for follow-up projects in robotics, embedded systems, and smart mobility solutions.

REFERENCES

[1] World Health Organization, Global Status Report on Road Safety 2023, World Health Organization, Geneva, Switzerland, 2023.

[2] National Highway Traffic Safety Administration, Standing General Order on Crash Reporting, 2024, https://www.nhtsa.gov/laws-regulations/standing-general-order-crash-reporting. Date Accessed 13 November 2024.

[3] Song, H., Nakahama, J. & Takada, Y., "Localization Method Based on Image Processing for Autonomous Driving of Mobile Robot in the Linear Infrastructure", Automation Control and Intelligent Systems, Vol. 9, No. 1, pp. 34–45, 2021.

[4] Charroud, A., Moutaouakil, K. E., Palade, V., Yahyaouy, A., Onyekpe, U. & Eyo, E. U., "Localization and Mapping for Self-Driving Vehicles: A Survey", Machines, Vol. 12, No. 2, pp. 118, 2024.

[5] Arvind, C. S. & Senthilnath, J., "Autonomous Vehicle for Obstacle Detection and Avoidance Using Reinforcement Learning", In: Soft Computing for Problem Solving, Springer, pp. 55–66, 2019.

[6] Krasniqi, X. & Hajrizi, E., "Use of IoT Technology to Drive the Automotive Industry from Connected to Full Autonomous Vehicles", IFAC-PapersOnLine, Vol. 49, No. 29, pp. 269–274, 2016.