SmartFlow: Prioritize Emergency Vehicle and Traffic Rules Monitoring System

Submitted in Partial Fulfillment of Requirement for the award of the Degree of

MASTER OF COMPUTER APPLICATIONS

Komal S Kallanagoudar

1MS22MC016

Under the Guidance of

Abhishek K. L.
Assistant Professor
Department of MCA

Department of Master of Computer Applications

RAMAIAH INSTITUTE OF TECHNOLOGY
(Autonomous Institute, Affiliated to VTU)
Accredited by the National Board of Accreditation & NAAC with 'A+' Grade
MSR Nagar, MSRIT Post, Bangalore-560054
www.msrit.edu

2024

RAMAIAH
Institute of Technology

CERTIFICATE

This is to certify that the dissertation work entitled "**SmartFlow: Prioritize Emergency Vehicle and Traffic Rules Monitoring System"** is carried out by **Komal S Kallanagoudar-1MS22MC016**, a bonafide student of Ramaiah Institute of Technology, Bangalore, in partial fulfillment for the award of Master of Computer Applications of the Visvesvaraya Technological University, Belgaum, during the year 2023-2024. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the project report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect to the dissertation work prescribed for the said degree.

Guide
Abhishek K. L.
Assistant Professor
Department of MCA

HOD
Dr. Monica R. Mundada

Name & Signature of Examiners with Date: -

1)

2)

RAMAIAH
Institute of Technology

DECLARATION

I **Komal S Kallanagoudar**, a student of Master of Computer Applications, Ramaiah Institute of Technology, Bangalore hereby declare that the project entitled **"SmartFlow: Prioritize Emergency Vehicle and Traffic Rules Monitoring System"** has been carried out independently under the Guidance of **Abhishek K. L. Assistant Professor, Department of Master of Computer Applications**.

I hereby declare that the work submitted in this project report is my own, except were acknowledged in the text and has not been previously submitted for the award of the degree of Visvesvaraya Technological University, Belgaum or any other institute or University

Place: Bangalore **Komal S Kallanagoudar**

Date: **1MS22MC016**

ACKNOWLEDGEMENT

With deep sense of gratitude, I am thankful to our beloved Principal, **Dr. N.V. R. Naidu**. I would like to express my sincere gratitude to our department for providing us with good infrastructure and the right atmosphere to carry out the project. It was really challenging to carry out the project work within the stipulated time period, which was made possible with the help of many. I do acknowledge the support and encouragement of all of them. I would like to thank our most Respected HoD, **Dr. Monica R. Mundada** for her pillared support and encouragement. I would like to convey my heartfelt thanks to my internal Project Guide **Abhishek K. L**. Assistant Professor, Department of Master of Computer Applications, for giving me an opportunity to embark upon this topic and for her/his constant encouragement.

It gives me great pleasure to acknowledge the contribution of all the people who helped me directly or indirectly realize it. First, I would like to acknowledge the love and tremendous sacrifices that my parents made to ensure that I always get an excellent education. For this and much more, I am forever in their debt.

I am thankful to all my friends for their moral and material support throughout the course of my project. Finally, I would like to thank all the Teaching and Non-Technical Staff of the Department of Master of Computer Applications for their assistance during various stages of my project work.

Komal S Kallanagoudar

ABSTRACT

"SmartFlow: Prioritize Emergency Vehicle and Traffic Rules Monitoring System" aims to address the critical issue of ensuring smooth flow for emergency vehicles while maintaining traffic rule compliance. The increasing urban traffic congestion and the need for efficient emergency response times motivated this work. Ensuring that emergency vehicles can navigate through traffic swiftly and safely is crucial for saving lives and enhancing public safety.

The scope includes the development and implementation of a system that prioritizes emergency vehicles and monitors traffic rule violations. Integration of IoT sensors and real-time data analytics used to implement methods such as algorithmic traffic signal control, data collection from sensors, and real-time processing were utilized to achieve the desired outcomes. The YOLO algorithm is implemented to detect and count the number of vehicles to control congestion and the EAST text detection model is implemented for emergency vehicle detection.

The SmartFlow model demonstrates a significant reduction in emergency vehicle response times and an improvement in traffic law enforcement. The system effectively identified and prioritized emergency vehicles, allowing them to pass through intersections with minimal delay. The SmartFlow system enhances emergency response efficiency and promotes safer urban mobility by ensuring compliance with traffic regulations.

Keywords: Congestion Control, Image processing, Emergency Vehicle Detection, Vehicle Detection, Green Delay optimization, Text Detection

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1. Overview

SmartFlow is an advanced system designed to improve road safety and manage traffic more effectively by integrating state-of-the-art technologies for emergency vehicle detection and real-time monitoring of traffic rules. With urban areas becoming more congested and vehicle numbers increasing, the need for innovative traffic solutions is critical. SmartFlow addresses these challenges using sensors, computer vision, and artificial intelligence (AI) to create a smarter and safer traffic environment. The primary aim is to ensure efficient emergency responses and better compliance with traffic regulations.

A significant feature of SmartFlow is its ability to dynamically control traffic signals based on real-time data. Unlike traditional traffic signals that operate on fixed schedules, SmartFlow adjusts signal timings according to the current traffic conditions. For example, if a particular route is heavily congested, the system can extend the green light duration for that route to alleviate the congestion. This dynamic control reduces wait times and prevents traffic jams.

At the core of SmartFlow is its enhanced traffic monitoring capability. High-resolution cameras are strategically placed at key intersections to continuously monitor traffic flow and detect any rule violations or unusual behavior. These cameras, equipped with advanced image processing algorithms, can identify vehicles running red lights, speeding, or performing other hazardous actions. This real-time data enables traffic authorities to respond swiftly to incidents, improving overall traffic law enforcement.

Objectives:

- Strengthen traffic monitoring efforts with camera technology.
- Controlling the traffic signals at dynamically based on real-time traffic data.
- Adaptive traffic signal systems adjust signal timings to minimize waiting times and reduce idling.
- The emergency vehicle is detected, which gives ambulances priority to pass through traffic lights.

SmartFlow's adaptive traffic signal systems are designed to minimize waiting times and reduce vehicle idling, which helps lower fuel consumption and emissions. By analyzing traffic patterns in real-time, the system can predict peak traffic periods and adjust signal timings to optimize traffic flow. This not only improves the efficiency of traffic movement but also contributes to environmental sustainability by reducing the carbon footprint of vehicular traffic.

A key feature of SmartFlow is its ability to detect emergency vehicles. Using a combination of sensors and advanced algorithms, the system can identify ambulances and fire trucks. When an emergency vehicle is detected, SmartFlow adjusts traffic signals to give it priority, allowing it to pass through intersections quickly. This is crucial in urban environments where timely emergency response can save lives. By ensuring that emergency vehicles reach their destinations without delay.

SmartFlow leverages a range of advanced technologies to achieve its objectives. Computer vision technology helps the system accurately interpret visual data from traffic cameras, identifying different types of vehicles and traffic conditions. AI algorithms process this data in real time, making smart decisions about signal timings and emergency vehicle prioritization. Additionally, the use of Internet of Things (IoT) devices ensures seamless communication between different system components, creating a cohesive and responsive traffic management solution.

1.2.Feasibility Study

Traffic management systems in metropolitan areas would focus on evaluating the technical, economic, operational, legal, environmental, and social aspects of implementing advanced solutions. From a technical perspective, the study would assess the integration of camera-based monitoring, IoT sensors, and AI algorithms into existing infrastructure, ensuring compatibility and scalability across a complex urban network. Legally, regulatory compliance regarding data privacy, traffic management laws, and public safety would be scrutinized. Environmental impact assessments would focus on benefits such as reduced emissions and improved air quality, aligning with sustainability goals. Social feasibility would address public acceptance, community engagement, and equitable distribution of benefits, aiming to mitigate concerns about privacy and social equity.

1.3.Existing System

The existing traffic management systems in metropolitan cities face significant challenges due to the rapid growth in population and vehicle numbers. Traditional methods, such as manual traffic control by police officers and static traffic light systems, struggle to cope with the dynamic and increasing traffic demands. This inefficiency leads to frequent congestion, longer commute times, and heightened environmental pollution from vehicle emissions.

Emergency vehicles, such as ambulances and fire trucks, often find themselves stuck in traffic, unable to navigate swiftly through crowded intersections. This delay in emergency response can have severe consequences, including compromised public safety and increased risk to lives. The lack of real-time monitoring also means that traffic incidents and violations are often detected too late, resulting in delayed interventions and prolonged traffic disruptions.

1.4.Proposed System

The challenge is to design a SmartFlow System integrating camera technology for traffic monitoring and efficient prioritization of emergency vehicles through traffic lights.

The traffic management system of a metropolitan city is a keystone for urban mobility. With the rise of the population, the demand for vehicles has grown and hence the requirement of transportation has also increased. Infrastructural development becomes an indispensable part of complementing population growth to augment urban mobility. The traditional traffic management system is shown not only ineffective for accompanying the increased number of vehicles with the use of police control and traffic light systems but also incompetent enough to handle this growth of traffic on road systems. This traffic congestion consequentially consumes precious working time and eventually leads to environmental pollution for an extended period of vehicle emission. Traffic system utilize the concept of automation with IoT is called as “SmartFlow”. SmartFlow Management System is an advanced and integrated solution designed to optimize traffic flow, reduce congestion, enhance road safety, and improve overall transportation efficiency within urban or metropolitan areas.

2. Literature Survey

Manual Traffic Control Management this refers to the traditional method of traffic management where traffic personnel or officers manually regulate the flow of vehicles and pedestrians at intersections or road junctions. While effective, this approach can be labor-intensive and may not always adapt well to changing traffic conditions. Infrared sensors positioned along roadways offer a solution to managing traffic congestion by detecting vehicle presence at specific distances. This allows the system to monitor the number of vehicles in each lane accurately. When congestion occurs in one lane, the system dynamically adjusts traffic signal timings, allocating more green time to the congested lane to facilitate smoother traffic flow. This adaptive approach optimizes the distribution of green time based on real-time traffic conditions, reducing overall congestion and improving travel times for motorists. However, the initial setup cost for implementing such a system is relatively high, involving investments in infrastructure and technology. Additionally, while the system can efficiently manage regular traffic conditions, it is unable to automate responses to emergencies, such as accidents or roadblocks, which typically require human intervention. Despite these limitations, infrared sensor-based traffic management systems represent a proactive approach to addressing traffic congestion and enhancing overall road efficiency. [1].

Ultrasonic sensors are deployed along roadways to monitor traffic levels effectively. These sensors continuously detect the presence of vehicles in specific lanes and categorize traffic levels into low, medium, or high. This real-time data is then transmitted to a central controller, providing continuous and accurate information about traffic conditions. This enables efficient traffic management, allowing authorities to make informed decisions to optimize traffic flow and reduce congestion. By dynamically adjusting traffic signal timings based on the traffic levels detected by the sensors, the system can respond promptly to changing traffic patterns. However, one limitation of ultrasonic sensors is their limited range, which may necessitate the installation of multiple sensors to cover larger areas adequately. This increases the initial installation and ongoing maintenance costs, potentially impacting the scalability and affordability of the system. [2].

The integration of surveillance cameras and RFIDs along roadways creates a distributed system for traffic management. This system processes sensor data at the node

level and video data at a local server, allowing for real-time analysis of traffic conditions. By calculating cumulative density, the system can effectively regulate traffic flow based on the density of vehicles on the road. Additionally, it addresses the needs of emergency vehicles such as ambulances and fire brigades, ensuring their priority passage through the traffic. Furthermore, the system provides users with insights into congestion status through predictive analysis, allowing for informed decision-making regarding route selection. While this approach reduces latency in detecting and responding to traffic congestion, thereby improving overall system responsiveness, it also presents challenges. Distributed systems can be complex to design, implement, and manage, requiring specialized expertise. Moreover, building and maintaining such a distributed traffic control system can involve significant upfront and ongoing costs, which may impact the feasibility and scalability of the system. [3].

The RFID Tag system is designed with the primary goal of facilitating rapid ambulance response and ensuring uninterrupted passage to the destination. It operates by monitoring traffic lights and enabling the ambulance driver to control them as needed to navigate through intersections without stopping. This approach significantly reduces response times, potentially saving lives by ensuring prompt medical assistance. Moreover, prioritizing ambulance passage at intersections minimizes the risk of accidents and enhances the safety of both patients and emergency responders. However, the effectiveness of the system hinges on the reliability of its technology components, including sensors, communication networks, and traffic control algorithms. Any failures or malfunctions in these components could compromise the system's ability to prioritize ambulance passage effectively, underscoring the importance of robust and dependable infrastructure.[4].

The SIM28 GPS Module and ESP8266 Node MCU Wi-Fi Module are integral components of a project aimed at facilitating the swift movement of ambulances from dispatch to accident zones and hospitals. By leveraging GPS technology, the system ensures accurate navigation, guiding ambulances efficiently through the road network. Additionally, the Wi-Fi module enables real-time communication, allowing for timely updates and coordination between emergency responders and healthcare facilities. This streamlined approach to ambulance dispatch and navigation significantly reduces response times, increasing the likelihood of saving lives and minimizing patient suffering. However,

the effectiveness of the project is contingent upon the reliability of critical infrastructure, such as roads, traffic signals, and communication networks. Any interruptions or failures in these systems could impede emergency response efforts, potentially compromising the effectiveness of the project and the timely delivery of medical assistance to those in need. Therefore, robust contingency plans and maintenance protocols are essential to mitigate the risks associated with infrastructure failures and ensure the seamless operation of the ambulance dispatch system [6].

The Image Processing system, employing Python code with KERAS, OpenCV, TensorFlow, and scikit-learn modules, serves a crucial role in ambulance detection and traffic control. By analysing images captured by cameras, the system identifies ambulances and calculates their current speed, enabling it to predict their arrival times at traffic signals. This prediction capability allows the system to proactively clear the path ahead, facilitating faster response times for emergency vehicles like ambulances. However, the accuracy of these predictions is pivotal to the system's effectiveness. Inaccurate predictions may result in inefficient traffic control, such as prematurely changing signal timings, leading to unnecessary disruptions for other road users. Moreover, if the system's predictions consistently deviate from actual arrival times, it could introduce delays in emergency response efforts, potentially jeopardizing patient outcomes. Therefore, rigorous testing and calibration of the prediction algorithms are essential to minimize inaccuracies and ensure the reliable operation of the Image Processing system in optimizing traffic flow and supporting emergency vehicle response [8].

The integration of GSM and GPS modules in accident detection and prevention systems represents a significant advancement in road safety technology. When an accident occurs, the system swiftly obtains the coordinates of the accident location through GPS technology. These coordinates are then transmitted via the GSM network to registered mobile numbers, notifying relevant parties of the incident in real-time. This proactive approach not only detects accidents but also enables preventive measures to be taken, potentially reducing the frequency and severity of accidents over time. However, there are potential drawbacks to consider. The system may occasionally trigger false alarms or inaccurately detect accidents, leading to unnecessary notifications and potential confusion among emergency contacts. Despite these challenges, the ability of GSM and GPS modules

to detect and notify of accidents promptly underscores their importance in enhancing road safety and facilitating timely response efforts. Continued refinement of these technologies is essential to minimize false alarms and maximize the effectiveness of accident detection and prevention systems [5][14].

The ESP32-based system for accident detection offers a comprehensive approach to enhancing road safety and emergency response. Leveraging the efficient power usage and versatile connectivity options of the ESP32 microcontroller, the system integrates various sensors, including IR, ultrasonic, accelerometer, and MQ-3 gas sensor, to monitor vehicle operations and driver behavior continuously. This multi-sensor approach enables the system to promptly detect accidents by identifying factors such as sudden speed reduction or vehicle tilting. Upon detection, the system utilizes GPS and GSM modules to determine the vehicle's location and promptly notify healthcare centres, facilitating faster response times and ensuring timely assistance to those in need. Additionally, the integration with mobile applications and user-friendly LCD displays enhances communication and coordination, further optimizing emergency response efforts. However, the system faces potential challenges, including the risk of false alarms triggered by sensor readings and dependency on technology, such as Wi-Fi and GPS connectivity, which may introduce the possibility of system failures due to technical issues or signal disruptions. Despite these challenges, the ESP32-based accident detection system represents a significant advancement in road safety and emergency response capabilities, with the potential to save lives and mitigate the impact of road accidents [9].

The method of sound detection harnesses roadside units (RSUs) equipped with specialized technology to detect the distinct siren sounds emitted by emergency vehicles (EVs). By leveraging unique frequencies emitted by these vehicles, the system can swiftly and accurately identify their approach, allowing for timely detection and response. This enables traffic signal controllers to prioritize the passage of emergency vehicles efficiently, ensuring they encounter minimal delays at junctions. However, despite its effectiveness, there are potential drawbacks to this approach. Sound detection technology may occasionally misinterpret non-emergency sounds as siren signals, leading to false alerts and unnecessary prioritization of traffic signals. Such false positives could disrupt traffic flow and compromise the overall effectiveness of the system. Therefore, while sound detection

offers significant benefits in enhancing traffic management and improving emergency vehicle response times, measures must be in place to mitigate the risk of false alarms and ensure the system's reliability in real-world scenarios. [10][11].

The integration of Bluetooth modules into traffic signal control systems provides a wireless communication channel between ambulances and traffic signal controllers, offering flexibility in managing traffic signals without physical connections. This technology allows for smoother and more efficient traffic flow management during emergencies, as ambulance drivers can send commands via their Bluetooth-enabled devices to nearby Bluetooth modules, instructing traffic signals to prioritize the ambulance's passage. However, Bluetooth's limited range, typically up to 100 meters in outdoor environments, poses a significant constraint. This range limitation means that the effectiveness of Bluetooth-based traffic control may be restricted to nearby intersections within range of the ambulance, potentially leaving intersections further away unaffected. In such cases, delays in emergency response and traffic congestion may occur. Thus, while Bluetooth offers a wireless solution for traffic signal control, its range limitations must be considered during implementation to ensure comprehensive coverage and effectiveness in managing traffic during emergencies [7].

The integration of GPS technology with an Android application offers a comprehensive solution for managing emergency situations effectively. When a driver arrives at the accident or fire site, they input the details into the application, which is then stored securely on the Google Cloud service (Firebase). Additionally, the application utilizes GPS services to track the location of emergency vehicle staff in real-time, facilitating better coordination and communication between them and service agencies. Staff members are provided with unique usernames and passwords for authentication, ensuring secure access to route maps and other pertinent information stored in the cloud. The data recorded in the cloud storage is further analyzed by machine learning algorithms to assess the level of emergency and situation accurately. This enables optimized routes for faster response times. However, the reliance on various technologies, including sensors, Wi-Fi modules, and cloud services, introduces the risk of technical failures or malfunctions, potentially disrupting emergency response operations. Despite these challenges, the integration of GPS and an Android application with machine learning capabilities

represents a significant advancement in emergency management, enhancing coordination, and facilitating more efficient response efforts [13].

The utilization of machine learning (ML) and deep learning (DL) algorithms in road safety systems represents a significant advancement in accident prevention and emergency response. By employing technologies such as Face and Eye Detection, FPGA-Based Drowsiness Detection, and Eye Recognition Systems, these algorithms can monitor driver behaviour in real-time and detect potential hazards on the road. Through continuous analysis of facial and eye movements, the system can identify signs of driver fatigue or distraction, issuing warnings to prevent accidents before they occur. Additionally, the system records the location of accidents and sends alert messages to emergency services promptly, enabling them to provide medical assistance in a timely manner. This proactive approach not only improves road safety by preventing accidents but also facilitates prompt medical services, potentially saving lives. However, the implementation of such advanced technologies requires substantial investment in equipment and expertise. Despite the initial challenges, the benefits of these systems in enhancing road safety and saving lives justify the investment and underscore the importance of leveraging technology to address critical safety concerns on our roads [16][17].

The combination of YOLOv3 algorithm and OpenCV provides a powerful tool for detecting traffic violations, such as vehicles jumping red signals, riders without helmets, and drivers without seat belts. YOLOv3, known for its efficiency and accuracy in object detection tasks, reliably identifies these violations in real-time, enabling prompt intervention by law enforcement authorities as violations occur. By leveraging this technology, law enforcement agencies can effectively monitor traffic and enforce regulations, enhancing road safety and compliance with traffic laws [18]. However, the effectiveness of YOLOv3 relies heavily on the availability of extensive annotated data for training. Annotated data is crucial for training the algorithm to accurately recognize and classify traffic violations. Without sufficient data, the performance of the algorithm may be compromised, leading to false detections or inaccuracies in identifying violations. Therefore, while YOLOv3 and OpenCV offer powerful capabilities for traffic violation detection, ensuring access to high-quality annotated data is essential for maximizing their effectiveness in enforcing traffic regulations [12].

3. Hardware and Software Requirements

Hardware Requirements

- Arduino MEGA
- Arduino Uno
- ESP32-cam
- Traffic LEDs
- Jumper wires

Software Requirements

- Arduino IDE
- Python IDLE
- PyCharm
- Xampp database

4. Software Requirements Specification

4.1. Overall Description

4.1.1. Product Perspective

The "SmartFlow: Prioritize Emergency Vehicle and Traffic Rules Monitoring System" project was conceived to tackle the critical challenge of ensuring the swift and safe passage of emergency vehicles amidst growing urban traffic congestion while maintaining strict adherence to traffic regulations. This innovative system is a new, self-contained product specifically designed to address the urgent need for efficient emergency response times and improved public safety.

The increasing density of urban traffic has made it imperative to develop systems that can facilitate the rapid movement of emergency vehicles such as ambulances, fire trucks, and police cars. Delays in emergency response can have dire consequences, making it essential to prioritize these vehicles through intelligent traffic management. SmartFlow is developed to address these issues by integrating modern technologies like IoT, GPS, and real-time data analytics.

SmartFlow utilizes a combination of IoT sensors and real-time data analytics to monitor and manage traffic. An ESP32-camera is employed to capture images of traffic conditions, which are then analyzed to determine congestion levels and identify the presence of emergency vehicles. When an emergency vehicle is detected, a signal is sent to an Arduino Mega, which adjusts the traffic signals to provide a green corridor for the vehicle.

4.1.2. Product Functions

- Image Capturing
- Vehicle Detection and counting
- Emergency Vehicle Detection
- Congestion Control
- Calculating green delay
- Update green signal

4.1.3. User Classes and Characteristics

The SmartFlowt system caters to various user classes with distinct characteristics. These user classes play specific roles and interact with the system differently. Here are some key user classes and their characteristics in a smart traffic management system:

- **Traffic Authorities:**
  Characteristics: Government agencies, traffic control centers, and law enforcement.
  Roles: Monitor and control traffic flow, receive real-time data for decision-making, and enforce traffic regulations.
  Interactions: Access to comprehensive traffic data, control traffic signals, and receive alerts for emergencies.
- **Transportation Planners:**
  Characteristics: Urban planners, city officials, and transportation departments.
  Roles: Analyze traffic patterns, plan infrastructure developments, and optimize transportation systems.
  Interactions: Access historical and real-time traffic data for city planning, implement changes based on insights, and assess the impact of transportation policies.
- **Individual Drivers:**
  Characteristics: Private vehicle owners and drivers.
  Roles: Navigate efficiently, avoid traffic congestion, and receive real-time information about road conditions.
  Interactions: Use navigation apps for real-time traffic updates, receive alerts on alternative routes, and contribute data through connected devices.
- **Pedestrians and Cyclists:**
  Characteristics: Individuals using non-motorized transportation modes.
  Roles: Ensure safe crossings, access pedestrian-friendly routes, and receive alerts on potential hazards.
  Interactions: Receive real-time information on pedestrian-friendly routes, crossing times, and safety alerts through mobile apps or connected devices.

- **IoT Device Manufacturers:**

  Characteristics: Companies producing IoT devices and sensors for traffic management. Roles: Develop and provide sensors for data collection, contribute to technology advancements, and improve system connectivity.

  Interactions: Collaborate with system administrators to ensure compatibility, offer updates for IoT devices, and participate in system improvement initiatives.

- **Emergency Services:**

  Characteristics: Firefighters, paramedics, and law enforcement.

  Roles: Respond to emergencies, navigate through traffic efficiently, and coordinate responses.

  Interactions: Access real-time traffic data for emergency route planning, receive priority signals at intersections and collaborate with traffic authorities for incident management.

4.1.4. Operating Environment

SmartFlow operates within an urban environment, leveraging a diverse array of hardware platforms, operating systems, and software components. The primary hardware includes IoT sensors such as the ESP32-Camera for capturing real-time traffic images and Arduino Mega for processing signals and controlling traffic lights. These devices are essential for monitoring congestion levels and detecting emergency vehicles. The software runs on embedded systems programmed using the Arduino IDE and custom firmware designed for real-time processing. Python is used extensively for developing data analytics algorithms, while OpenCV handles image processing tasks. Databases like MySQL or PostgreSQL store captured images, traffic data, and system logs, ensuring reliable data management.

SmartFlow integrates seamlessly with existing city-wide traffic management systems, ensuring compatibility and smooth operation. It also shares real-time data with emergency response systems and law enforcement databases to monitor traffic conditions, prioritize emergency vehicles, and enforce traffic regulations. This integration enhances emergency response efficiency and improves urban mobility.

4.1.5. Design and Implementation Constraints

The reliance on network connectivity can lead to performance issues in areas with poor coverage. Protecting sensitive traffic data is crucial to prevent breaches that could compromise privacy and system integrity. Ensuring seamless communication between diverse IoT devices and systems with different protocols is essential. The system must be scalable to handle urban growth and changing traffic patterns effectively. IoT devices, especially in remote areas, may face power constraints, impacting reliability. Budget limitations can affect the system's capabilities and the adoption of advanced technologies. Reliable data from sensors is critical for effective decision-making in traffic management. Adherence to local regulations and data privacy standards is essential to avoid legal issues. Harsh weather conditions can impact the performance of IoT devices and sensors. Public understanding and acceptance of the system are crucial for its success. Compatibility issues may arise when integrating IoT technologies with existing traffic management infrastructure.

4.1.6. User Documentation

The SmartFlow System is a technology-driven solution designed to optimize urban traffic control. It employs IoT devices such as cameras to monitor and analyze real-time traffic data, offering features like dynamic traffic signal control, predictive analytics, emergency response systems.

The Key Components are:

- Overview
- System Architecture
- User Interface
- Features

This documentation serves as a guide for users, offering detailed information on setup, features, troubleshooting, security measures, and ongoing maintenance of the SmartFlow system.

4.2.External Interface Requirements

4.2.1. Hardware Interfaces

The hardware infrastructure for a smart traffic management system using IoT comprises various components, including camera, traffic LEDs, a communication network, edge computing devices, Arduino mega, power supply systems.

- ESP32-camera: Camera captures the images of real time traffic and that is uploaded to database and that image is used to detect and count the number of vehicles in order to calculate the green signal delay and also to detect the presences of ambulance.
- Arduino Mega: Arduino is used for both programming the ESP32-camera and also to update the green signal calculated by algorithm on the bases of number of vehicles and also to change green signal when emergency vehicle is detected.
- Communication Network (Jumper wires): Establish a robust communication network to connect all the hardware components. This may include wired connections. Ensure secure and reliable communication for transmitting real-time data between sensors, controllers, and the central system.
- Power Supply: Ensure reliable power supply for all devices, considering backup options such as batteries or generators to prevent system failures during power outages.
- Traffic LED: Integrating traffic LEDs into a smart traffic management system can enhance the efficiency of traffic control and communication. Connect traffic LEDs to intelligent traffic signal controllers that are part of the overall smart traffic management system. Use real-time data from sensors and cameras to dynamically adjust signal timings and optimize traffic flow. Implement adaptive signal timings based on traffic conditions. Adjust the duration of green, yellow, and red lights dynamically to respond to changing traffic patterns.

4.2.2. Software Interfaces

The Arduino Integrated Development Environment (IDE) is an open-source software platform used for programming and developing applications for Arduino microcontroller boards. Arduino is a popular platform for hobbyists, students, and professionals to create a wide range of electronic projects and prototypes. Python IDLE used to write a code for YOLO algorithm to detect the vehicles, count the vehicles and calculate the green delay and also to detect the emergency vehicle.

I. Cross-Platform: Arduino IDE is available for Windows, macOS, and Linux, making it accessible to a broad user base.
II. Code Editor: It provides a text editor where you can write, edit, and save your Arduino sketches (code) written in C/C++.
III. Built-in Libraries: The IDE includes a collection of libraries that simplify interfacing with various hardware components, making it easier to control sensors, displays, motors, and other peripherals.
IV. Compiling and Uploading: Arduino IDE compiles your code into machine-readable binary files and uploads them to the Arduino board over a USB connection.
V. Serial Monitor: It has a built-in serial monitor that allows you to interact with your Arduino projects and view debugging information sent over the serial port.
VI. Board Manager: Arduino IDE supports various Arduino boards, and you can select the appropriate board from the "Boards Manager."
VII. Library Manager: You can easily add and manage third-party libraries for additional functionality.
VIII. Cross-Platform: Python IDLE is available for Windows, macOS, and Linux, making it accessible to a broad user base.
IX. Code Editor: It provides a text editor where you can write, edit, and save your Python scripts with syntax highlighting and auto-completion features to improve the coding experience.

X. Interactive Shell: IDLE includes an interactive Python shell that allows you to test and run Python code snippets in real-time, facilitating rapid prototyping and debugging.

4.2.3. Communications Interfaces

For the communication interface in your smart traffic management system using Arduino and various sensors/controllers, you can utilize a combination of wired and wireless communication protocols.

- **Wired Communication**

  Serial communication using UART (Universal Asynchronous Receiver-Transmitter) is a common and straightforward method to enable data exchange between different Arduino boards. UART allows for two-way communication between devices, using two wires: one for transmitting data (TX) and another for receiving data (RX).

- **Wireless Communication**

  The ESP32 offers robust wireless communication capabilities through both Wi-Fi and Bluetooth. In Wi-Fi Client Mode, the ESP32 connects to a network to send and receive data, ideal for IoT devices communicating with cloud servers. It can also function as a Wi-Fi Access Point, creating a local network for direct device connections without a router, and supports Wi-Fi Direct for direct device-to-device communication. For Bluetooth, the ESP32 includes Bluetooth Classic for continuous data streaming like audio, and Bluetooth Low Energy (BLE) for low-power, periodic data transfer, suitable for short-range communication with sensors and beacons.

4.3.System Features

In the SmartFlow system cameras placed at the traffic junction will capture images of real-time traffic that image is used by the algorithms to detect the number of vehicles to control the congestion and to detect the emergency vehicle to clear the path for an ambulance so that it will not experience the traffic and reach the hospital as soon as possible.

4.3.1. Congestion Control

This feature focuses on managing and alleviating traffic congestion through real-time monitoring and adaptive traffic signal control.

Functional Requirements:

- **REQ-1: Real-Time Traffic Monitoring**:
  The system must continuously monitor traffic conditions using IoT sensors and cameras. Deploy ESP32 cameras at strategic points to capture traffic data and transmit it to the central server to detect and count the number of vehicles.
- **REQ-2: Dynamic Signal Adjustment**:
  The system must dynamically adjust traffic signal timings based on real-time traffic data within 10 seconds of detecting changes. The green delay calculated based on the number of vehicles will be updated via Arduino Mega to optimize traffic flow and reduce congestion.
- **REQ-3: Data Analytics and Reporting**:
  The system must analyze traffic patterns and generate reports to identify congestion trends and inform future traffic management strategies. Utilize data analytics tools and techniques to process collected data and produce actionable insights.

4.3.2. Emergency Vehicle Detection

This feature ensures the swift and secure passage of emergency vehicles by detecting their presence and prioritizing their movement through intersections.

Functional Requirements:

- **REQ-1: Emergency Vehicle Identification**:

  The system must detect and identify emergency vehicles within 5 seconds of their approach. Use image processing and recognition algorithms on data captured by ESP32 cameras to identify emergency vehicles.

- **REQ-2: Signal Priority Activation**:

  When an ambulance is detected, the system must activate signal priority to provide a clear path for emergency vehicles. This involves sending priority signals to the traffic control system via the Arduino Mega, which alters the signal phase to prioritize the emergency vehicle's route.

4.4. Other Nonfunctional Requirements

4.4.1. Performance Requirements

The SmartFlow traffic management system is designed with stringent performance requirements to ensure its effectiveness in real-time traffic monitoring and control.

The system must process real-time traffic data with a latency of less than 1 second. This requirement is essential for making swift decisions and adjustments to traffic signal timings, allowing the system to dynamically respond to changing traffic conditions. By meeting this requirement, SmartFlow optimizes traffic flow and minimizes congestion on roadways. A critical aspect of SmartFlow's functionality is the detection and prioritization of emergency vehicles within 5 seconds of their approach. Swift identification and response are crucial for ensuring the rapid passage of emergency vehicles, thereby enhancing public safety and emergency management.

The system must dynamically adjust traffic signal timings within 10 seconds of detecting changes in traffic conditions. This requirement highlights the system's adaptive capabilities, enabling it to quickly respond to traffic fluctuations and optimize signal timings to reduce congestion.

4.4.2. Safety Requirements

The SmartFlow traffic management system must meet rigorous safety standards to mitigate potential risks of loss, damage, or harm resulting from its operation. A critical requirement is the prevention of traffic signal malfunctions, which could lead to accidents or traffic congestion. To address this, SmartFlow incorporates fail-safe mechanisms such as component redundancy, regular maintenance schedules, and real-time monitoring to promptly detect and rectify any malfunctions. These measures not only ensure continuous operation but also comply with local transportation authority regulations governing traffic signal control systems.

Another essential safety consideration is the reliable operation of emergency vehicle preemption to prevent accidents and delays. SmartFlow employs precise emergency vehicle detection algorithms, robust communication channels, and backup preemption systems to facilitate the swift and secure passage of emergency vehicles through intersections. Compliance with national and local emergency response standards ensures that the system effectively supports emergency management efforts without compromising public safety.

4.4.3. Security Requirements

The SmartFlow system is fortified with a set of security requirements aimed at safeguarding its integrity, data confidentiality, and overall functionality. Firstly, the system is mandated to implement robust access control mechanisms, ensuring that unauthorized access to sensitive components, data, and functionalities is strictly restricted. By enforcing stringent access controls, the system enhances its resilience against potential security threats, safeguarding critical elements from unauthorized manipulation or compromise. This measure not only contributes to the system's security posture but also aligns with industry best practices for access management.

The system places a strong emphasis on physical security. A specific requirement dictates the implementation of physical security measures, including access controls and surveillance, to protect critical hardware components and infrastructure. This multifaceted approach addresses not only virtual vulnerabilities but also the physical aspects of the system.

5. System Design Description

5.1.Introduction

The purpose of this Software Design Document (SDD) is to provide a comprehensive overview of the design and architecture of the SmartFlow system. This document is intended for software developers, system architects, project managers, and stakeholders involved in the development and implementation of the SmartFlow system. It aims to ensure a clear understanding of the design considerations, system architecture, and the functionalities of the SmartFlow system.

The SmartFlow system is designed to manage urban traffic efficiently by prioritizing emergency vehicles and monitoring traffic rule violations. The system integrates IoT sensors, GPS technology, and real-time data analytics to achieve its objectives. It captures images using ESP32 cameras to detect congestion levels and identify emergency vehicles. The system then sends signals to an Arduino Mega to update traffic signals accordingly. Additionally, the system can detect traffic rule violations, capture images of offending vehicles, recognize license plates, and notify drivers of their infractions.

The SmartFlow system faces several constraints that must be addressed for optimal performance and compliance. Firstly, hardware limitations are significant, as the effectiveness of IoT sensors and cameras can be impacted by environmental conditions such as lighting and weather. Integration with existing traffic management infrastructure and emergency response systems is essential for seamless operation, requiring compatibility and possibly retrofitting current setups. Lastly, the system's design must consider scalability to manage increased data volumes and wider deployment areas, ensuring that performance remains consistent as the system expands.

5.2.System Overview

The SmartFlow system is designed to manage urban traffic efficiently by prioritizing emergency vehicles and monitoring traffic rule violations. This system addresses the increasing congestion in urban areas, which poses significant challenges for emergency response times and overall traffic management. The background of the system lies in the

necessity to enhance public safety and optimize traffic flow by utilizing modern technologies such as IoT and real-time data analytics.

5.2.1. System Architecture

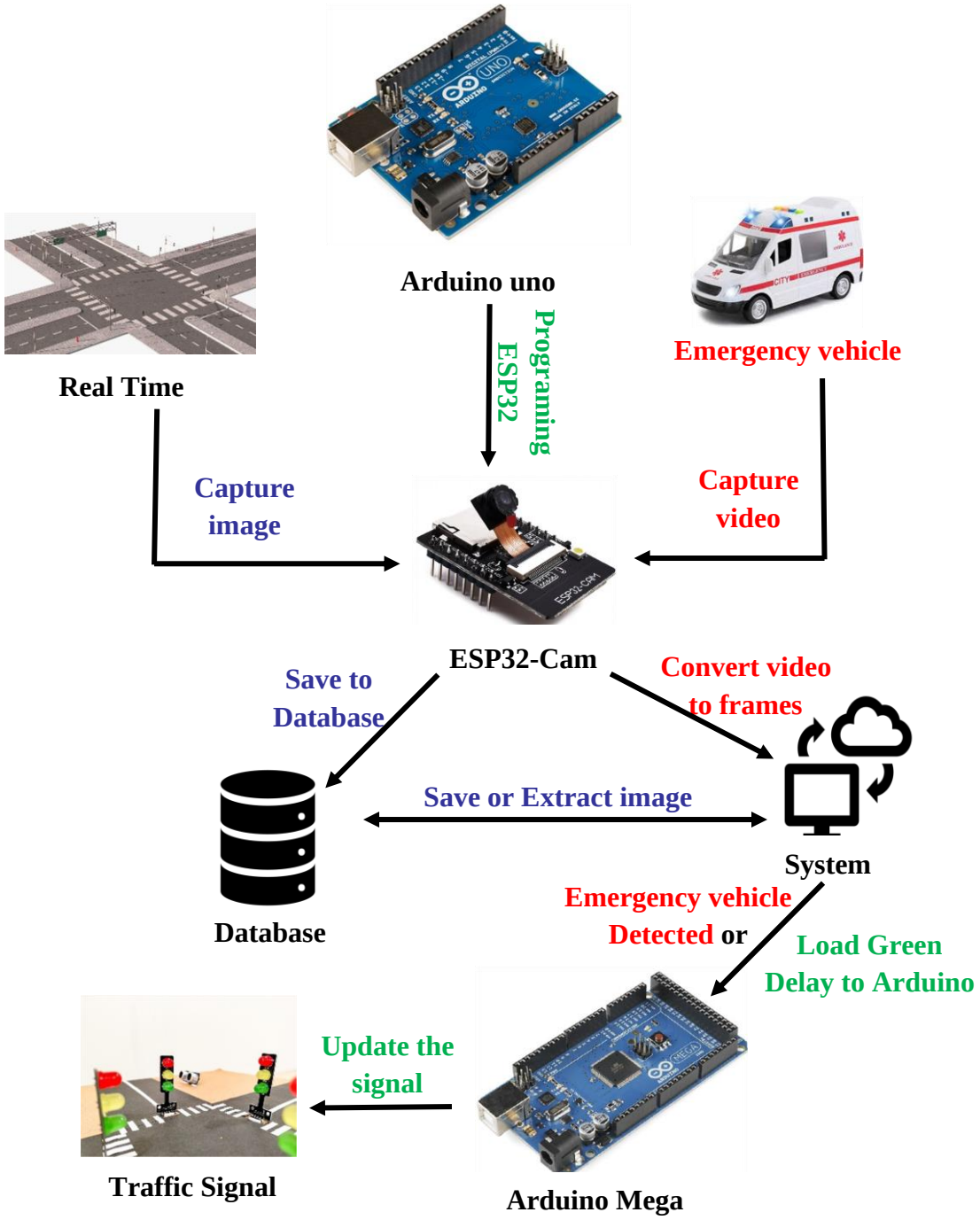

Figure 1: Architecture diagram of SmartFlow

Figure 1 represents the SmartFlow system architecture that integrates various components to optimize traffic management and enhance road safety. At its core, the Arduino Uno microcontroller is used to program the ESP32 camera, which captures real-time images of traffic in a designated lane. These images are stored in a database, serving as a repository for traffic data and allowing for historical analysis. The YOLO algorithm processes these images to count the number of vehicles in the lane, providing essential data for traffic control. To determine the appropriate green light duration, the system uses the formula:

$$\boldsymbol{Green\_delay} = \frac{(\boldsymbol{L}+(\boldsymbol{vehicle_{count}}-\boldsymbol{1})*(\boldsymbol{L}+\boldsymbol{G})}{\boldsymbol{v}} \quad \text{- - - - -> equation(1)}$$

Where: L – the average length of the vehicle, vehicle_count – number of vehicles in one lane, G – inter-vehicle gap and v - Speed of vehicle m/s

The SmartFlow system also incorporates emergency vehicle detection using the EAST detection module. This module analyzes the images to identify the presence of emergency vehicles, such as ambulances. If an ambulance is detected in the image, the system immediately turns the green signal on for that lane, allowing the ambulance to pass through the intersection as quickly as possible. This feature is crucial in ensuring that emergency vehicles can reach their destinations without unnecessary delays, ultimately contributing to faster and more efficient emergency response times.

5.2.2. Architectural Alternatives

Several architectural alternatives were considered for the SmartFlow system:

i. **Centralized Architecture:**
   - All data processing and decision-making occur at a single central server.
   - Pros: Simplifies data management and control.
   - Cons: Potential single point of failure and scalability issues.

ii. **Distributed Architecture:**
   - Data processing is distributed across multiple nodes closer to the data sources.
   - Pros: Enhanced scalability and fault tolerance.
   - Cons: Increased complexity in synchronization and data consistency.

5.3.Functional Design

5.3.1. Describe the functionalities of the system

Data flow Diagram:

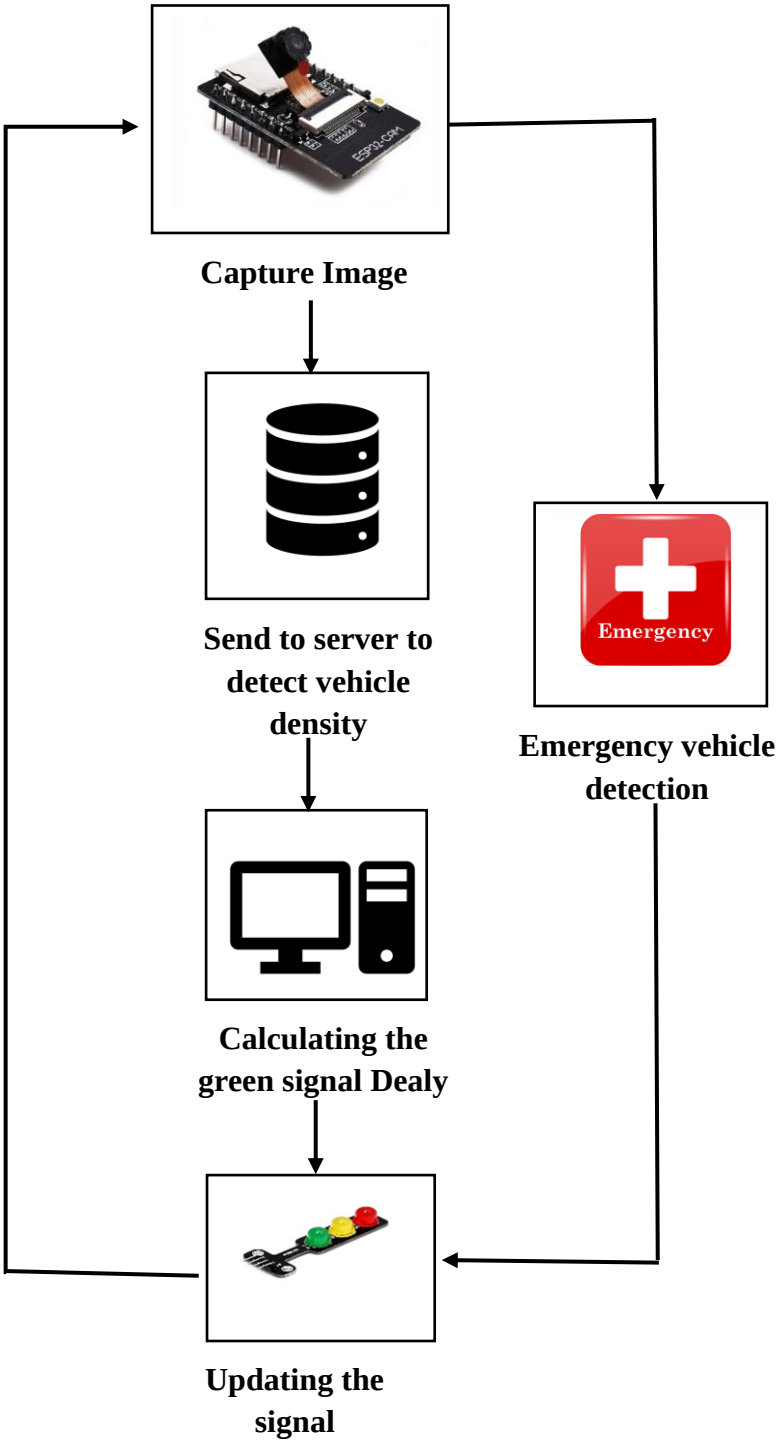

Figure 2: Data flow diagram

As shown in Figure 2, the ESP32 camera captures the image and stores it in the server, the YOLO algorithm processes the image detects the congestion, and calculates the green signal timing with the help of the Green_delay formula, according to that calculation the traffic signal is controlled, the system also includes an emergency vehicle detection feature. It scans the captured images to identify emergency vehicles such as ambulances. When an emergency vehicle is detected, the system immediately activates a green signal for the corresponding lane, ensuring the vehicle can pass through the intersection without delay. This feature is crucial for enabling quick emergency responses and reducing the time it takes for ambulances and other emergency vehicles to reach their destinations.

Use Case Diagram of SmartFlow:

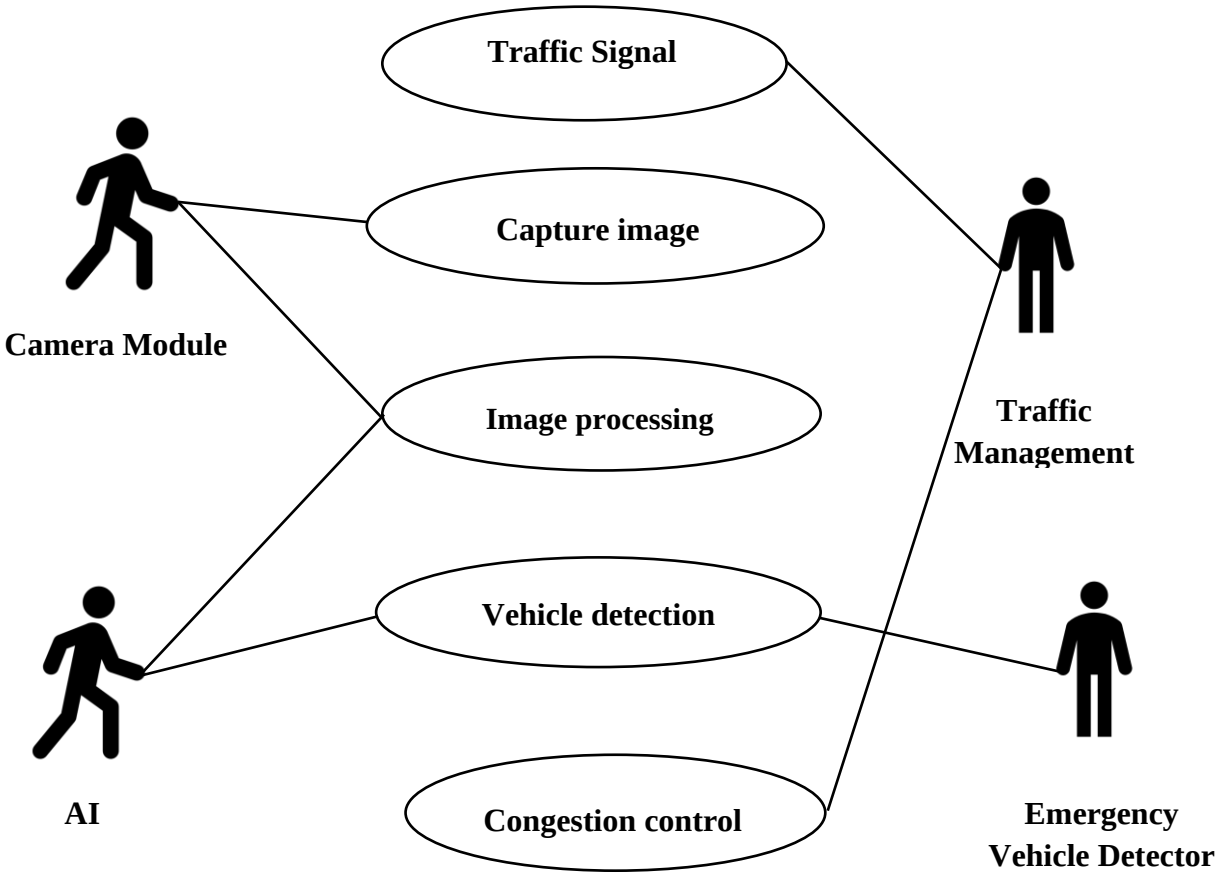

Figure 3: Use case Diagram

Figure 3 represents the use cases and actors in the SmartFlow system which can be described as follows

Use Cases

- **Traffic signal:** The traffic junction where the traffic lights are controlled by the traffic management system.
- **Capture image:** The cameras at junction takes images and send it to the system.
- **Image processing:** The AI algorithm processes the image to detect the number of vehicles and classify them.
- **Vehicle detection:** The AI algorithm uses data set and detect type and number of vehicles and detect the emergency vehicle.
- **Congestion control:** Traffic congestion, were there are more vehicles in one lane and a smaller number of vehicles in another lane leads to traffic.

Actors

- **Camera Module:** Captures the real-time image or video to control the congestion based on the number of vehicles on a lane.
- **AI:** The artificial intelligence algorithm YOLO is used to detect the vehicles and classify them to calculate the green delay.
- **Traffic Management:** The main system that controls the normal cycle of traffic and congestion control of the traffic, it uses the data given by the AI algorithm to control the traffic by adjusting green delay.
- **Emergency Vehicle Detector:** The text detection algorithm is used to detect the emergency vehicle and inform the system to turn on the green signal for an emergency vehicle.

5.3.2. Behavioral design

State Chart Diagram of SmartFlow:

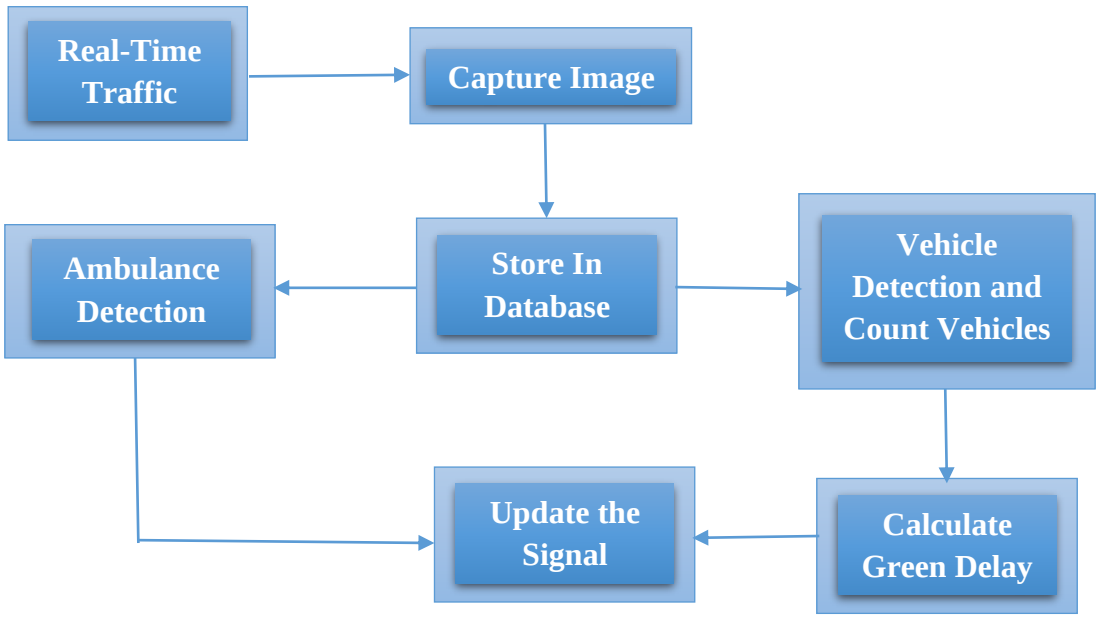

Figure 4: Operational Workflow of SmartFlow

As shown in Figure 4 the process starts with the system in an idle state, waiting to capture an image in the last 5 seconds of green delay. When the camera captures the image of a lane, that image is stored in the database. The AI algorithm then retrieves these stored images to process them and count the number of vehicles in the lane. Based on the vehicle count, the system calculates the duration for the green signal in a lane. As the green signal period for a lane is about to end, the camera captures a new image of the next lane to prepare for the subsequent cycle.

Simultaneously, the AI algorithm continuously analyses the images to detect any emergency vehicles, such as ambulances. If an emergency vehicle is detected, the system immediately overrides the normal operation and activates the green signal for the lane with the emergency vehicle, allowing it to pass through without delay. After the emergency vehicle has passed, the system returns to its idle state, ready to capture the next set of images at a fixed interval, ensuring efficient traffic management and prioritization of emergency vehicles.

Interaction diagram

Sequence Diagram of SmartFlow:

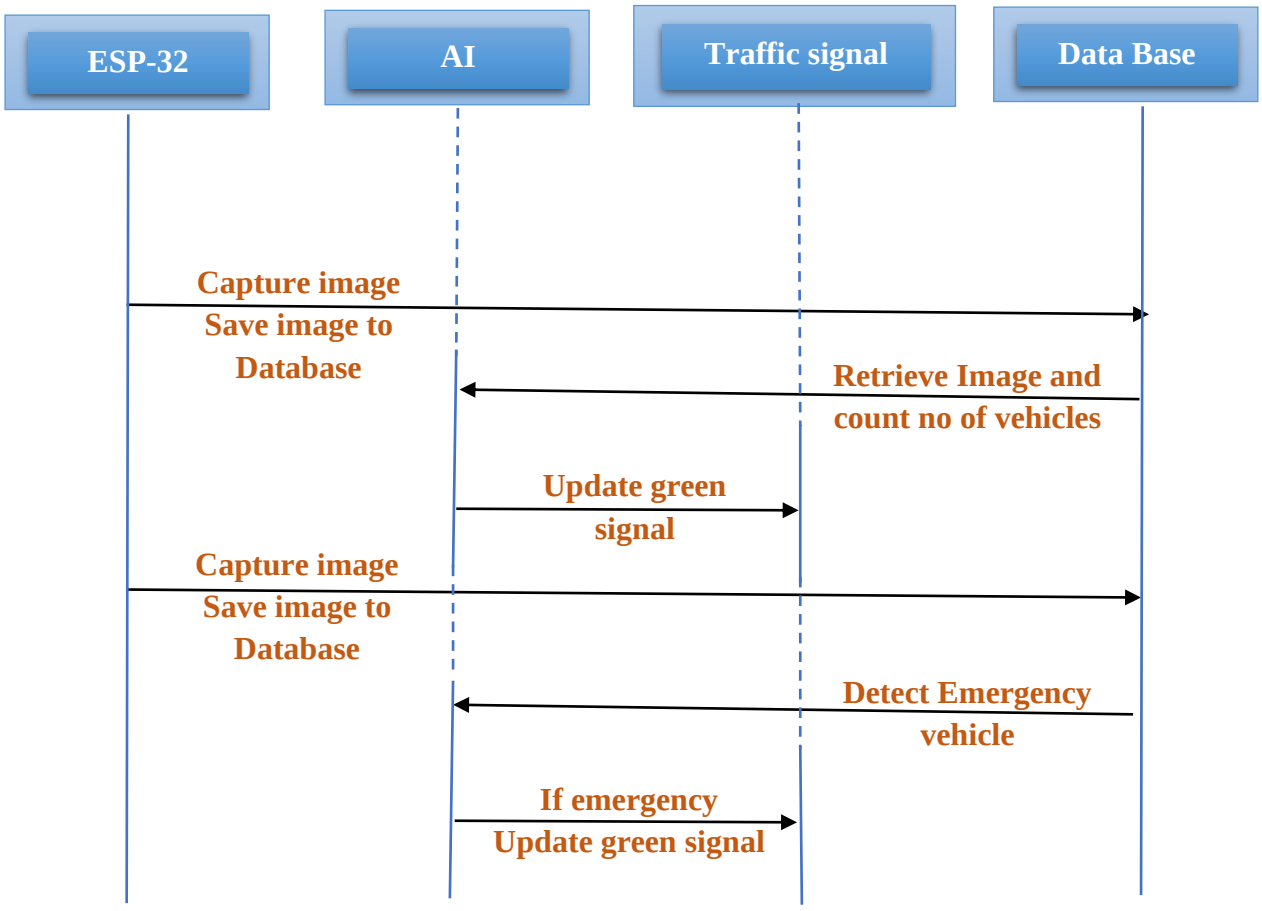

Figure 5: Sequence Diagram

As shown in Figure 5 camera captures the image and stores it in the database. The image is retrieved by an AI algorithm to count the number of vehicles and calculate the green delay, this continues for all other lanes when the last few seconds of the green delay camera takes a picture of another lane. But the camera captures images of all lanes at a fixed time interval and the images are taken by the AI algorithm to detect the emergency vehicle this is a continuous process so that emergency vehicles like ambulances will not suffer from traffic when the ambulance is detected and the green signal is turned on.

6. Implementation

In the SmartFlow implementation, the system begins by establishing an RTSP connection to capture live video feeds for real-time traffic monitoring. It continuously retrieves frames from this stream, which are then stored in a database at regular intervals to facilitate historical traffic analysis. During operation, the system monitors the duration of green signals at intersections. If the delay in switching to green exceeds a set threshold, the system continues to observe traffic conditions until an appropriate time to change the signal timing is determined.An essential feature of SmartFlow is its capability to detect emergency vehicles within the captured frames.

Algorithm for SmartFlow

1. Start
2. Import libraries, cv2,os, pyfirmata, time, threading, pytesseract
3. Set up Arduino board and traffic signal pins
   board = pyfirmata2.Arduino('COM4')
   traffic signals for four signals (signal1, signal2, signal3, and signal4) using pins 2 to 13 for red, yellow, and green lights.
4. Initialize delays (red_delay, yellow_delay, inter_green_delay, initial_green_delay), set next_capture_time, and initialize counters (count) and signal indexing (signal_order, signal_index).
5. Set up RSTP connection
6. Create database connection
7. If failed to read frame from RSTP stream go to step 11 else go to step 5
8. Initialize the database connection
9. Display live transmission from RSTP stream
10. set_light(signal, color, state):
    pin = pins[signal][color]
    board.digital[pin].write(state)
11. traffic_light_cycle(signal, green_delay)
12. Capture the frames from the RSTP stream at regular interval of time and save to database
13. While True:
    14. Detect emergency vehicle

1. Capture image at every 10 seconds and store in ambulance_imges table
2. Retrieve image call Emergency_vehicle()
   ambulance_detected = Emergency_vehicle
3. If ambulance_detected = 1 turn on the green signal else continue
   Traffic_light_cycle(signal,green_delay)

15. Process images and handle traffic

1. If green_delay < 5 seconds go to step 2 else continue
2. Take an image save to the database traffic_images table
3. Retrieve image img
4. Call function congestion control(img)

Module 1: Congestion control

Module one is for congestion control which is detecting and counting the number of vehicles to calculate the green delay in order to reduce the congestion. The process is as shown in the below algorithm.

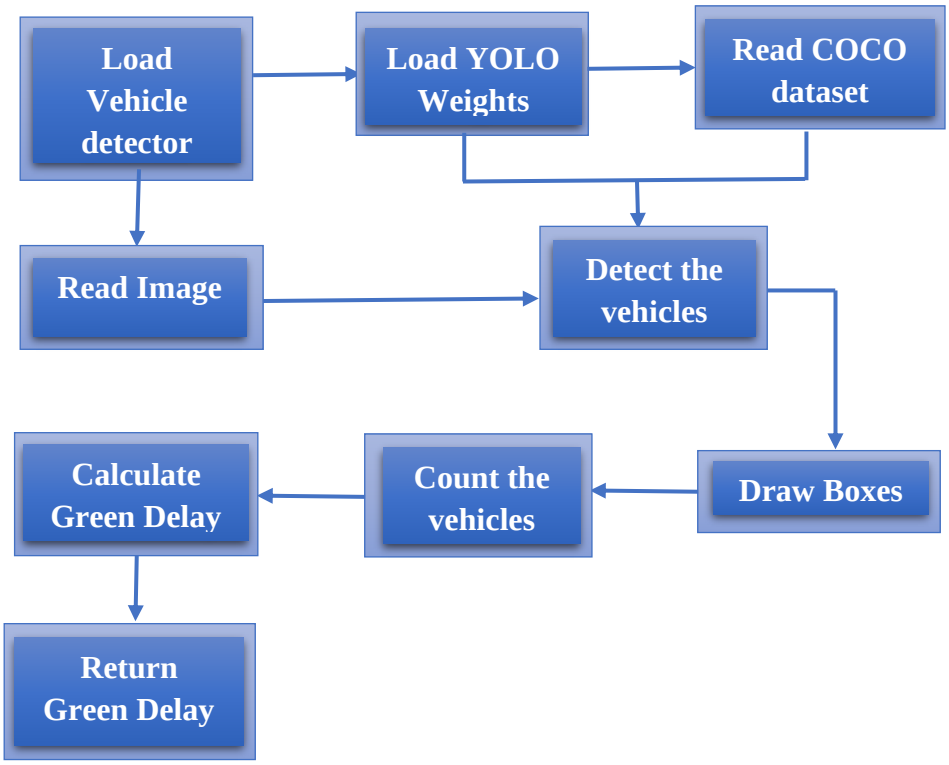

Figure 6: Traffic Congestion Control

As shown in Figure 6 for congestion control, the Vehicle Detector Algorithm plays a pivotal role in managing traffic flow efficiently. This algorithm, integrated with YOLO weights and the

COCO dataset, utilizes deep learning capabilities to detect vehicles within images extracted from the database. Employing OpenCV's cv2.dnn DetectionModel(), the algorithm performs vehicle detection by drawing bounding boxes around identified vehicles and counting these boxes to determine the total number of vehicles in the lane. This count forms the basis for calculating the green delay required for the traffic signal, considering factors such as average vehicle speed. The calculated green delay value is crucial for dynamically adjusting traffic signals via systems like the Arduino Mega, ensuring optimal traffic flow management in real-time scenarios.

Algorithm for Congestion Control

DelayCalculation(img)

1. Load VehicleDetector()
2. Read the img = cv2.imread(img)
3. Call detect_vehicles(img)
4. Draw the boxes to vehicles and count and display the number of vehicles
5. Img_name= os.path.joinbasename(img_path)
6. Return Vehicle_count

VehicleDetector

1. Initialization
    1. Load network net = cv2.dnn.readNet("dnn_model/yolov4.weights", "dnn_model/yolov4.cfg")
    2. Initialize the detection model using cv2.dn DetectionModel()
    3. Specify the classes that correspond to vehicles in the COCO dataset
2. Vehicle Detetion
    1. Initialize vehicles_boxes = [], nmsThreshold, confidenceThreshold
    2. Detect the object in the image using YOLO4 model to get class_ids, scores, boxes
    3. For each iteration

       Compare the ‘score’ is below ‘confidenceThreshold’

       If yes, skip to next detection

       Compare the ‘class_id’ in list of allowed vehicle classes (self.classes_allowed).

       If yes, add the ‘box’ to ‘vehicles_boxes’
    4. Return vehicles_boxes.

In the module one, the process starts with loading the VehicleDetector module, which utilizes the YOLOv4 model for vehicle detection. This model is configured using OpenCV, allowing it to identify vehicles within a given image. Once an image is provided as input, the VehicleDetector module detects vehicles by applying the YOLOv4 model, which outputs bounding boxes around recognized vehicles. After detecting vehicles, the system proceeds to calculate the number of vehicles detected by simply counting the bounding boxes. This also calculate the green delay on the number of detected vehicles and average speed value.

Module 2: Emergency Vehicle Detection

The module 2 is for emergency vehicle detection it uses EAST text detection module for detecting the text written on the ambulance. Once the ambulance is detected it will return true and the green signal for ambulance so that ambulance will reach hospital as soon as possible without any delay.

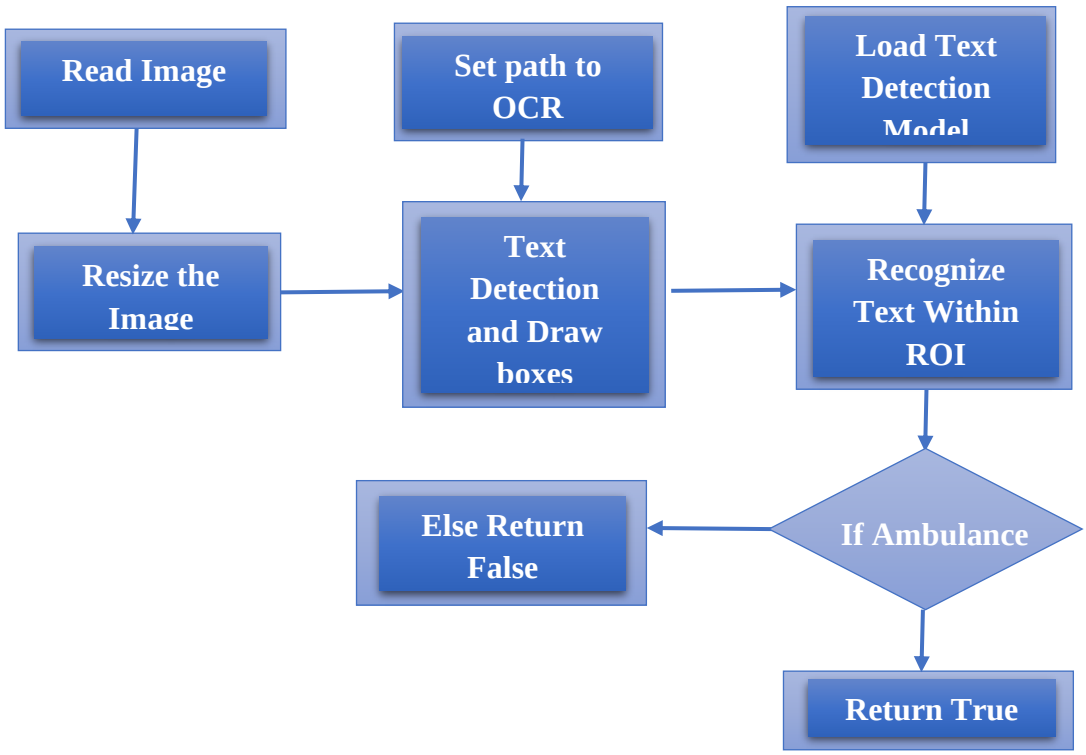

Figure 7: Prioritize Emergency Vehicle Detection

As shown in Figure 7 emergency vehicle detection model utilizes a pre-trained text detection model known as frozen_east_text_detection.pb. When an image is fetched from the database, it is resized and then processed through the EAST model to detect text. Upon detection,

bounding boxes are drawn around the identified text regions. The extracted text undergoes Optical Character Recognition (OCR) using Tesseract. The OCR results are stored in a variable to check that detected word is “AMBULANCE” OR NOT. If it is “AMBULANCE”, the model returns true, indicating the presence of an emergency vehicle; otherwise, it returns false.

Algorithm for Emergency Vehicle Detection

EmergencyVehicle

1. Set the path to the Tesseract OCR engine
2. Load the pre-trained EAST text detection model (frozen_east_text_detection.pb) using OpenCV's DNN module.
3. Read the image using from database
4. Resize the input frame to a fixed size to match the input size expected by the EAST model.
5. Convert the resized frame to a blob suitable for input to the neural network (cv2.dnn.blobFromImage()).
6. Pass the blob through the EAST model (net.forward()) to obtain predictions draw boxes.
7. Text Recognition (OCR) Extract the region of interest (ROI) from the original frame. Apply Tesseract OCR (pytesseract.image_to_string()) to recognize text within the ROI.
8. Return 1 if the text is ‘AMBULANCE’ else 0

Module two works on the text detection and recognition system, the process begins by setting up the Tesseract OCR engine path, which allows the system to leverage Tesseract for Optical Character Recognition (OCR). The pre-trained EAST text detection model, specifically the frozen_east_text_detection.pb file is then loaded using OpenCV's Deep Neural Network module. The system reads the input image from a database, which serves as the source of data for processing. Once resized, the frame is converted into a blob using the cv2.dnn.blobFromImage() function. The blob is then passed through the EAST model using the net.forward() function to obtain text detection and bounding boxes drawn around the detected text areas in the image. And the system extracts the regions of interest (ROIs) from the original frame based on the bounding boxes identified by the EAST model. These ROIs are then processed using Tesseract OCR through the pytesseract.image_to_string() function, which recognizes and converts the text within these regions into a readable string format.

7. Testing

7.1.Description of Testing

Unit Testing

Each functionality of the SmartFlow is tested first the camera is connected to check whether video or image is captured and sending to the system. Next the YOLO algorithm is tested using the test images to detect and count the number of vehicles. The normal working of the traffic signals is tested. The Emergency vehicle detection using text detection module is tested.

Integration

The emergency vehicle detection module and traffic signals both are combined and tested to check the signals turning green immediately for emergency vehicle to pass through junction and camera module and algorithm are combined and tested to solve the traffic congestion based on the number of vehicles present in a lane.

System testing

End to end functionality of the system is tested by integrating the congestion module and emergency vehicle detection module so that the traffic congestion problem is solved based on the number of vehicles present on the lane and also the emergency vehicles like ambulance is given high priority at traffic junction so that it reaches the hospital as soon as possible.

7.2.Test Cases

Table 1: Test Cases for Individual Modules

Test case #	Test case Name	Test case Description	Inputs	Expected Output	Actual Output	Status
1	Testing ESP32 Camera	To check the live streaming capturing the images and storing in the database	Connecting the camera to Arduino uno and uploading code	RSTP live streaming URL	RSTP live streaming URL	Pass
2	Normal Traffic flow	To check whether uploading code to Arduino through PyCharm works or not	Connecting traffic LEDS	The duration of the green signal at each lane is the same	Same Green Delay	Pass
3	Executing YOLO algorithm	To check the YOLO algorithm, detect and count the number of vehicles or not	Some images of vehicles in traffic	Draw boxes for detected vehicles and display count	Output image with boxes drawn for detected vehicles	Pass
4	Text detection	Testing the EAST module for ambulance test detection from the image	Image of ambulance	Draw boxes around detected text ambulance	Error	Pass

Table 2: Test cases for Combined Modules

5	Congestion control	Testing code to count the number of vehicles from image taken from ESP32 camera and change the green signal	Connect camera and database	Green delay on the basis of number of vehicles	Green delay on the basis of number of vehicles	Pass
6	Ambulance Detection	Testing code to detect the ambulance from an image taken by the camera and change green signal	Connect camera and database	Turn on the LED to indicate the presence of an ambulance and change the green signal	Error	Pass
7	Emergency vehicles detection	Combining all the models to test for ambulance detection	Connect camera and database	Turn on the LED to indicate the presence of an ambulance and change the green signal	Error	Fail
8	SmartFlow: Congestion control	Combining all the models to test for congestion control	Connect camera and database	Green delay based on number of vehicles	Green delay based on number of vehicles	Pass

8. Results And Discussion

The SmartFlow system successfully established an RTSP connection for continuous live video feed to monitor real-time traffic. It efficiently captured and stored frames in a database at regular intervals, enabling robust historical traffic analysis. The system demonstrated reliable performance in monitoring and adjusting signal timings at intersections, effectively reducing delays. A key feature of SmartFlow is its ability to detect emergency vehicles within the captured frames. This was achieved with high accuracy, promptly triggering actions such as adjusting signal timings and activating LED lights to indicate the presence of an emergency vehicle. The historical data storage allowed for insightful analysis of traffic patterns, revealing peak hours and common bottlenecks. Despite some challenges like occasional technical issues and environmental factors affecting performance, the system proved to be a valuable tool for improving traffic management and safety. Overall, SmartFlow demonstrated significant potential in optimizing traffic flow and responding effectively to emergencies.

To analyze the performance and impact of the SmartFlow system, we have visualized key data points using graphs. These graphs illustrate the system's efficiency in real-time traffic monitoring, signal timing adjustments, and emergency vehicle detection.

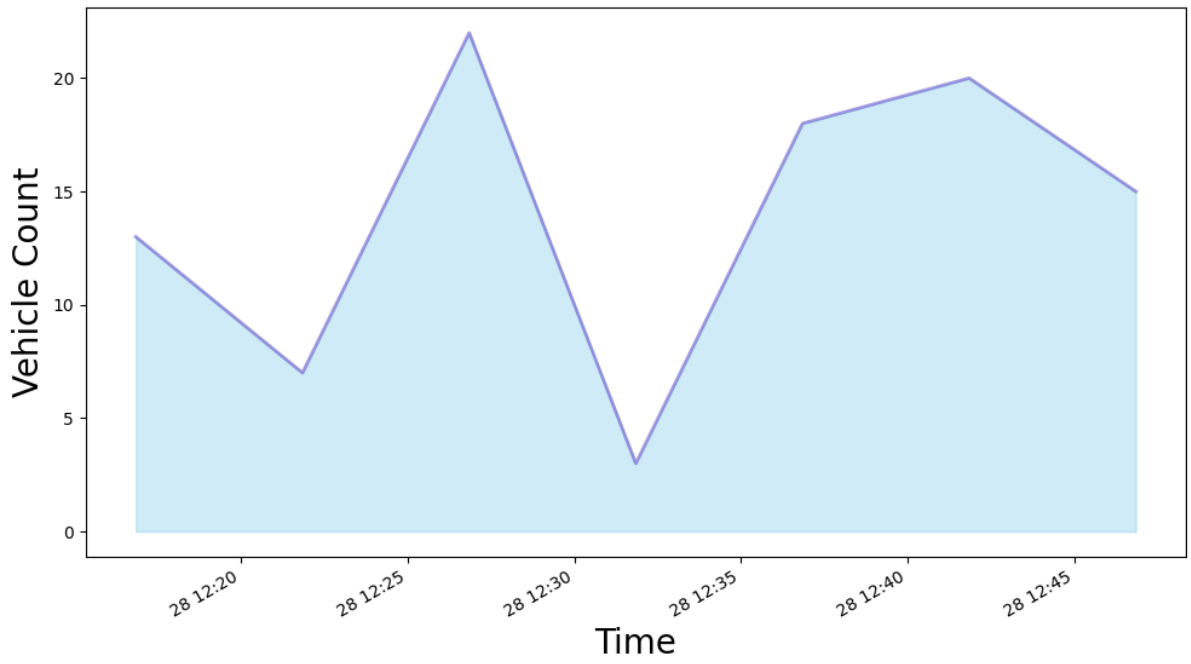

Figure 8: Vehicle Count over Time

Figure 8 illustrates the vehicle count over time across four lanes, each experiencing different delays for red, green, and yellow signals. The graph reveals significant traffic volume trends, highlighting peak periods where vehicle counts are highest. Lane 1 consistently shows the lowest vehicle count, correlating with its minimal signal delays, indicating efficient traffic flow. In contrast, Lane 4 displays the highest vehicle count and substantial delays, particularly for red and green signals, suggesting a need for SmartFlow to optimize signal timings to better manage heavy traffic.

Lanes 2 and 3, despite having green signals, exhibit significant delays, indicating inefficiencies in vehicle movement even when signals are green. This results in higher vehicle counts over extended periods. SmartFlow can address these inefficiencies by adjusting green signal durations in real time to ensure smoother traffic flow. The pronounced red delays in Lanes 3 and 4 contribute to prolonged wait times, leading to increased vehicle counts during these phases. By reducing red delays through adaptive signal control, SmartFlow can alleviate congestion and enhance traffic throughput.

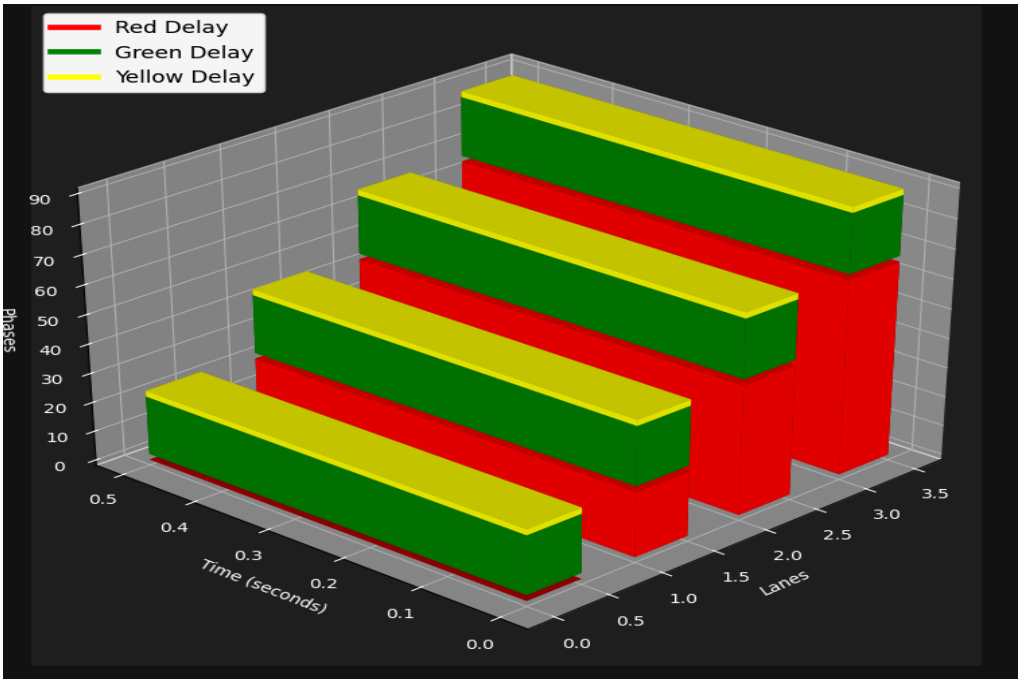

Figure 9: Traffic Analysis with Fixed Green Delay

Figure 9 represents the green, yellow, and red delays in normal traffic junctions when the green delay is fixed for all 4 lanes which is 20 seconds. The waiting time that is red delay will be more.

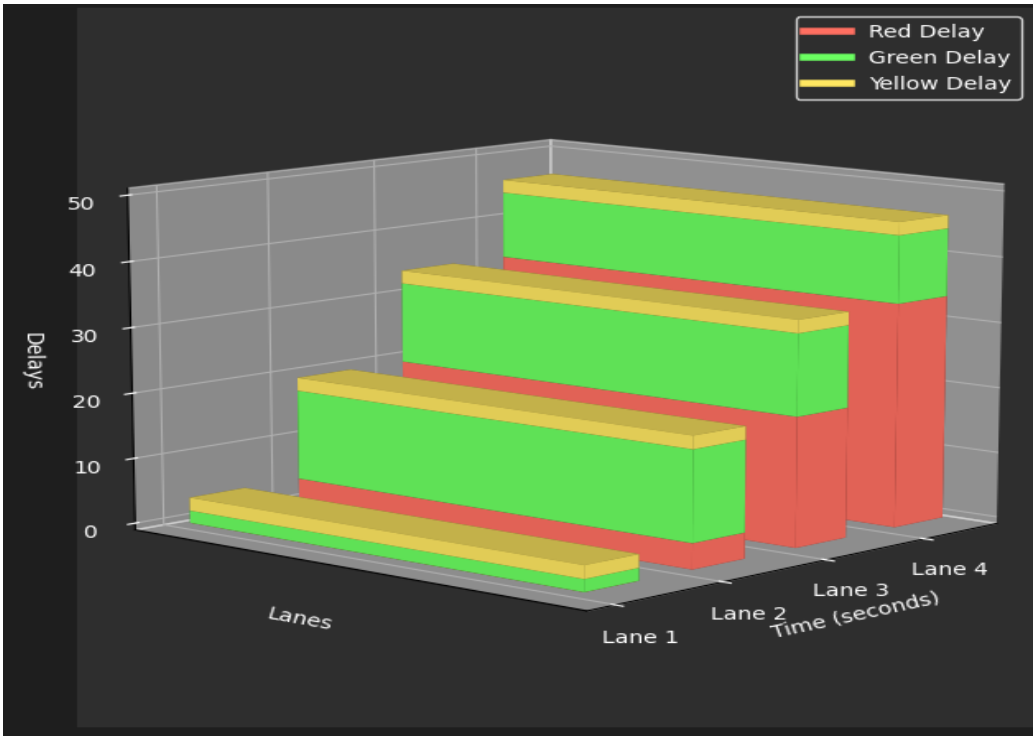

Figure 10 : Dynamic Green Delay Traffic Analyses

Figure 10 represents the green, yellow, and red delay for all 4 lanes when traffic congestion is controlled by dynamic green delay that is based on the number of vehicles present in each lane. The green delay is calculated by using the formula

$$Green\_delay = \frac{(L+(vehicle_{count}-1)*(L+G)}{v}$$

where v is the vehicle speed (10 m/s), is the average length of vehicles (5 meters), and G is the inter-vehicle gap (2 meters). The vehicle counts for lanes 1 to 4 are 3, 15, 20, and 18, respectively. Using this formula, the green delays for the lanes are calculated as 1.9, 14.5, 13.8, and 12.4 seconds, respectively. The total green delay across all lanes sums up to 42.6 seconds. Adding a yellow delay of 2 seconds per lane results in a total time of 50.6 seconds.

In contrast, if a fixed green delay of 20 seconds per lane is used, the total green delay for all lanes is 60 seconds. Including the yellow delay of 8 seconds across all lanes brings the total time to 68 seconds. This comparison shows that the waiting time, represented by the red delay, is significantly higher with fixed green delays. For instance, in the dynamic green delay scenario, the red delay for Lane 2 is 3.9 seconds, while it would be 22 seconds with a fixed green delay.

Similarly, for Lane 3, the red delay is 20.4 seconds with a dynamic green delay compared to 44 seconds with a fixed green delay.

These results indicate that the SmartFlow system, which employs dynamic green delays, effectively reduces waiting times, manages traffic congestion more efficiently, and also conserves energy. By adjusting the green signal duration based on real-time traffic conditions, SmartFlow optimizes traffic flow, reduces unnecessary idling, and enhances overall intersection performance.

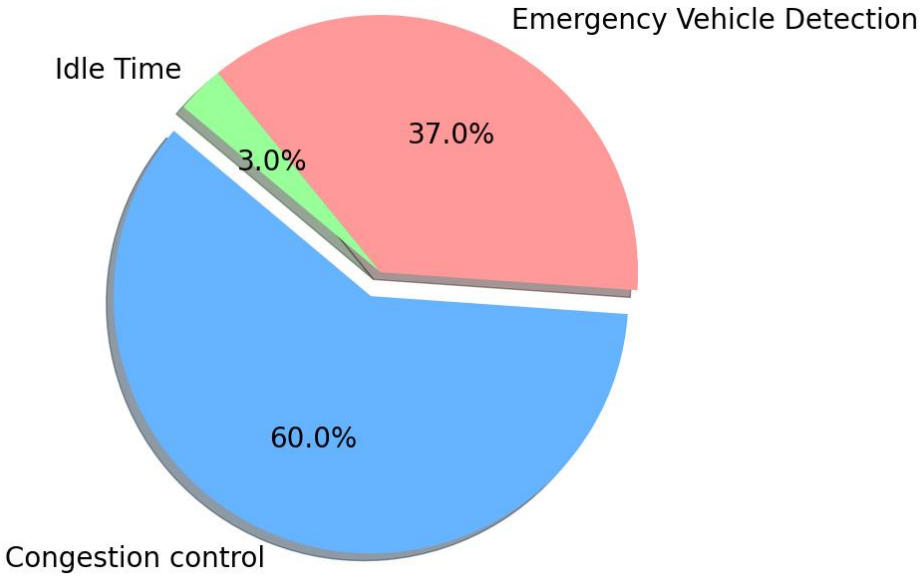

Figure 11: System Performance Metrics

Figure 11 represents the system performance metrics. The Blue color indicates the congestion control that is capturing the image that image is used by the YOLO algorithm for detecting and counting the number of vehicles and calculating the green delay. The red color represents the emergency vehicle detection by the EAST text detection module from the image captured by the camera to provide a smooth flow for emergency vehicles like ambulances. The percentage of emergency vehicle detection depends on the arrival of ambulances in the traffic junctions. Most of the time system does the congestion control. The system will be ideal only for few minutes so it shown only three percentage in the graph.

9. Reports

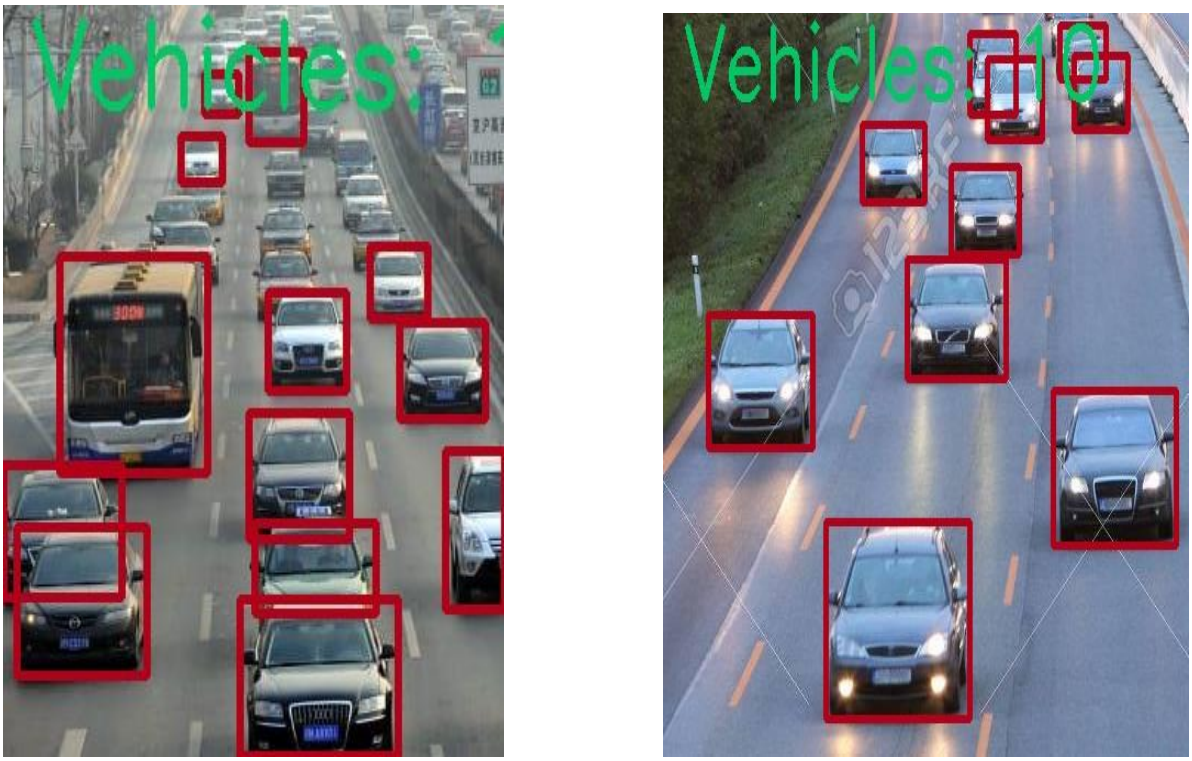

Figure 12: Vehicle Detection and Counting Output

As shown in Figure 12 the YOLO algorithm detects the vehicles draw boxes around it and gives the count of vehicles.

Figure 13: Ambulance Detection Output

Figure 13 shows the output of the EAST text detection module to detect the text Ambulance.

10. Conclusion

The proposed system SmartFlow will control the traffic congestion based on the density that is by calculating the number of vehicles in the lane by taking the image and calculating green delay to adjust green delay according to the number of vehicles and the emergency vehicle is detected from an image taken from the camera and the green signal is given to ambulance so that it does not experience any traffic or delay in reaching the hospital.

The project SmartFlow provides an effective solution for rapid growth of traffic flow particularly in big cities which is increasing day by day and traditional systems have some limitations as they fail to manage current traffic effectively. The SmartFlow is proposed to control road traffic situations more efficiently and effectively. It changes the signal timing intelligently according to traffic density on the particular roadside and regulates traffic flow by capturing the image. The system also makes sure that emergency vehicle like ambulance will not suffer from any interruption due to traffic signal so that ambulances can reach the hospital as soon as possible.

The SmartFlow system is designed for efficient traffic management through real-time monitoring and dynamic signal adjustments based on vehicle counts per lane. It effectively reduces delays by optimizing green signal durations using a formula that considers vehicle speed, average vehicle length, and inter-vehicle gaps. This dynamic approach results in shorter waiting times compared to fixed green delays, significantly enhancing intersection performance and energy conservation. The system also excels in emergency vehicle detection, promptly adjusting signals to prioritize their passage, thereby improving overall traffic flow and safety.

11. Future Scope and Further Enhancement of The Project

The proposed system aims to control traffic congestion using cameras to detect emergency vehicles. Building upon this system, we can implement additional features to enhance its functionality and effectiveness. One significant enhancement is accident detection. By integrating sensors and advanced image processing techniques, the system can detect accidents in real time. When an accident is detected, the system can automatically inform the nearest ambulance service, ensuring a prompt emergency response.

- **Accident Detection:** Implement sensors and advanced image processing to detect accidents in real time.
- **Emergency Response Coordination:** Automatically notify nearest ambulance services upon accident detection for a prompt response.
- **Li-Fi Navigation**: Introduce Li-Fi technology to guide ambulance drivers with real-time traffic updates for optimal route selection. Utilize Li-Fi for dynamic route adjustments based on current traffic conditions to minimize travel time.
- **Traffic Rule Violation Detection**: Enhance the system with high-resolution cameras and license plate recognition to identify and notify drivers of traffic violations.

These points outline how enhancements can be integrated into the proposed system to enhance traffic control, emergency response, and adherence to traffic rules.

12. Bibliography

[1] Pedro Maximino, Rui S. Cruz, Miguel L. Pardal worked on Smart Healthcare Monitoring System For Healthy Driving in Public Transportation published paper in 2023 18th Iberian Conference on Information system and Technologies (2023) https://ieeexplore.ieee.org/document/10211847

[2] S. Mahalakshmi a, T. Ragunthar b, N. Veena a, S. Sumukha a, Pranav R. Deshkulkarni a worked on Adaptive ambulance monitoring system using IOT Published by Elsevier Ltd (2022) https://www.sciencedirect.com/science/article/pii/S2665917422001891

[3] Shruti Gatade, Shreeram V Kulkarni, Samanvitha N worked on Automated Vehicle Accident Detection and Healthcare Unit Alerting Using IoT published in IEEE 2nd Mysore Sub Section International Conference (2022)

[4] Anil Kumar Biswal, Debabrata Singh, Binod Kumar Pattanayak, Debabrata Samanta and Ming-Hourb Yang worked on IoT-Based Smart Alert System for Drowsy Driver Detection (Received 29 December 2020; Revised 18 January 2021; Accepted 10 February 2021; Published 10 March 2021). https://www.hindawi.com/journals/wcmc/2021/6627217/

[5] Sunny Hossain and Farzana Shabnam researched on paper A Comparative Study of IoT Based Smart Traffic Management System published paper in IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (2021) https://ieeexplore.ieee.org/document/9829636

[6] Roopa Ravish, Shanta Rangaswamy, Kausthub Char worked on Intelligent Traffic Violation Detection paper released on 2021 2nd Global Conference for Advancement in Technology (GCAT) Bangalore, India. Oct 1-3, 2021. https://ieeexplore.ieee.org/document/9587520

[7] Dalia Nandi,Krishnendu Choudhury worked on Detection and Prioritization of Emergency Vehicles in Intelligent Traffic Management System released by IEEE 2021

[8] Pankaj Chourasia, Sakshi Choubey, Riya Verma worked on Vehicle Accident Detection, Prevention and Tracking System published by International Research Journal of Engineering and Technology (2020) https://www.irjet.net/archives/V7/i8/IRJET-V7I8445.pdf

[9] Dr. Vikram Bali, Ms. Sonali Mathur, Dr. Vishnu Sharma, Dev Gaur researched on Smart Traffic Management System using IoT Enabled Technology, 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN). https://ieeexplore.ieee.org/document/9362753

[10] Karthik B,Manoj M,Rohit R Kowshik,Akash Aithal,Dr. S. Kuzhalvai Mozhi worked on Ambulance Detection and Traffic Control System published by International Research Journal of Engineering and Technology (2019) https://www.irjet.net/archives/V6/i4/IRJET-V6I4239.pdf

[11] Mohammed Fayaz, Pooja K, Pranitha P Reddy, Swathi T worked onvDensity based Traffic Control System with Ambulance Detection released by International Journal of Engineering Research & Technology (2019) https://www.ijert.org/research/Density-based-Traffic-Control-System-with-Ambulance-Detection-IJERTCONV7IS08100.pdf

[12] Sabeen Javaid, Ali Sufian, Saima Pervaiz and Mehak Tanvee worked on Smart Traffic Management System Using Internet of Things published in International Conference on Advanced Communications Technology (2018) https://www.researchgate.net/publication/324464391_Smart_traffic_management_system_using_Internet_of_Things

[13] Prof. Deepali Ahir, Saurabh Bharade, Pradnya Botre, Sayali Nagane, Mihir Shah worked on Intelligent Traffic Control System for Smart Ambulance paper released in International Research Journal of Engineering and Technology (IRJET) Volume: 05 Issue: 06 June-2018. https://www.irjet.net/archives/V5/i6/IRJET-V5I675.pdf

[14] Varsha Srinivasan, Yazhini Priyadharshini Rajesh, S Yuvaraj and M Manigandan worked on Smart traffic control with ambulance detection released in 2nd International conference on Advances in Mechanical Engineering (ICAME 2018) https://iopscience.iop.org/article/10.1088/1757-899X/402/1/012015/pdf

[15] Sangmesh S B, Sanjay D H, Meghana S, M N Thippeswamy worked on Advanced Traffic Signal Control System for Emergency Vehicles published by International Journal of Recent Technology and Engineering (2018) https://www.ijrte.org/wp-content/uploads/papers/v8i3/C4323098219.pdfC4323098219

[16] Sarfraz Fayaz Khan researched on Health Care Monitoring System in Internet of Things (loT) by Using RFID paper released in 2017 the 6th International Conference on Industrial Technology and Management. https://ieeexplore.ieee.org/document/7917920

[17] Varsha Sahadev Nagmode and Prof.Dr.S.M.Rajbhoj researched on An IoT Platform for Vehicle Traffic Monitoring System and Controlling System Based on Priority published by IEEE in 2017 https://ieeexplore.ieee.org/document/8463825

[18] Patan Rizwan, K Suresh, Dr. M. Rajasekhara Babu worked on Real-Time Smart Traffic Management System for Smart Cities by Using Internet of Things and Big Data released in 2016 International Conference on Emerging Technological Trends https://ieeexplore.ieee.org/document/7873660

[19] Syed Misbahuddin, Junaid Ahmed Zubairi, Abdulrahman Saggaf, Jihad Basuni, Sulaiman A-Wadany and Ahmed Al-Sofi worked on IoT Based Dynamic Road Traffic Management for Smart Cities published by IEEE in 2016 https://www.researchgate.net/publication/304414535_IoT_based_dynamic_road_traffic_management_for_smart_cities

[20] Lien-Wu Chen,Pranay Sharma, and Yu-Chee Tseng, worked on Dynamic Traffic Control with Fairness and Throughput Optimization Using Vehicular Communications released in ieee journal on selected areas in communications/supplement, vol. 31, no. 9, September 2013https://ieeexplore.ieee.org/document/6550865Dynamic_Traffic_Control_with_Fairness_and_Throughput_Optimization_Using_Vehicular_Communications

[21] Mohammad Moazum Wani, Samiya Khan, Mansaf Alam worked on IoT - Based Traffic Management System for Ambulances published by IEEE in 2005 https://arxiv.org/ftp/arxiv/papers/2005/2005.07596.pdf

13. Glossary

Image Capture: The process of taking photographs of the traffic lanes at regular intervals using a camera.

ESP32 Camera: A low-cost microcontroller with an integrated camera module, commonly used for IoT applications. It combines the capabilities of an ESP32 microcontroller with an OV2640 camera sensor, allowing for image and video capture, processing, and transmission over Wi-Fi.

Database: A structured collection of data where captured images are stored for retrieval and processing.

AI Algorithm: A computational procedure used to analyse images, count vehicles, and detect emergency vehicles is YOLO Algorithm (You Only Look Once), A real-time object detection system that divides images into a grid and predicts bounding boxes and probabilities for each region simultaneously. YOLO is known for its speed and accuracy, making it suitable for tasks such as vehicle detection in traffic management systems.

OpenCV's cv2.dnn DetectionModel(): A class in the OpenCV library's Deep Neural Network module used for object detection. This class simplifies the process of loading pre-trained deep learning models and running inference to detect objects in images or videos.

Vehicle Count: The number of vehicles present in a lane, determined by processing the captured images.

Green Delay: The time for which the traffic signal remains green for a particular lane.

Emergency Vehicle: Special vehicles, such as ambulances, that require immediate passage through traffic.

Fixed Intervals: Specific, regular periods at which the system captures images.

Traffic Management: The process of controlling and optimizing the flow of vehicles on the road to reduce congestion and improve safety.

frozen_east_text_detection.pb: This refers to a pre-trained deep-learning model file used for text detection, particularly in scenes with varying orientations and complexities. The ".pb" extension denotes a protobuf file format commonly used to serialize TensorFlow models. The "frozen_east_text_detection" model, often associated with the EAST architecture, is trained to detect text regions in images efficiently.

EAST model (Efficient and Accurate Scene Text Detection): A deep learning model designed for scene text detection, particularly suited for detecting horizontal text in natural scenes.

OCR (Optical Character Recognition): A technology that enables the recognition and conversion of printed or handwritten text into machine-readable text. OCR systems use image processing techniques, often involving pattern recognition and artificial intelligence, to interpret characters from scanned documents, images, or videos.

ROI (Region of Interest): In image processing and computer vision, a specific region or area within an image is selected for further analysis or processing. ROIs are typically identified based on their importance or relevance to the task at hand, such as object detection, feature extraction, or measurement.

Blob: A blob is a standardized format required by neural networks for processing, ensuring that the image data is correctly pre-processed and normalized.

14. User Manual

Introduction

SmartFlow is an advanced traffic management system designed for controlling real-time traffic. It captures live video and images, analyzes traffic conditions, and dynamically adjusts signal timings to optimize traffic flow. The system also includes a critical feature to detect emergency vehicles and prioritize their movement.

Features

- **Real-time Traffic Monitoring:** Establishes an RTSP connection to capture live video feeds for continuous traffic monitoring.
- **Historical Traffic Analysis:** Stores frames at regular intervals in a database to facilitate the analysis of past traffic patterns.
- **Dynamic Signal Adjustment:** Monitors green signal duration and adjusts timing based on current traffic conditions.
- **Emergency Vehicle Detection:** Identifies emergency vehicles in the captured frames and adjusts signals to prioritize their passage.

System Operation

1. **Initialize RTSP Connection:** The system begins by setting up an RTSP connection to capture live video feeds.
2. **Frame Retrieval:** Continuously retrieves frames from the live video stream.
3. **Frame Storage:** Stores frames in a database at regular intervals for historical traffic analysis.
4. **Monitor Signal Duration:** Observes the duration of green signals at intersections.
5. **Signal Timing Adjustment:** Determines the appropriate time to change the signal timing based on observed traffic conditions.
6. **Emergency Vehicle Detection:** Utilizes specialized detection methods to identify emergency vehicles within the captured frames.

7. **Trigger Actions:** Upon detecting an emergency vehicle, the system adjusts signal timings and activates LED lights to indicate the presence of an emergency vehicle.

Emergency Vehicle Detection

1. **Detection Methods:** Uses text detection EAST module for detecting the text ambulance written on the emergency vehicle.
2. **Response Actions:** Adjusts signal timings to prioritize the emergency vehicle and activates LED lights as an indicator.

Congestion Control

1. **Traffic Analysis:** Uses YOLO algorithm to detect and count the number of vehicles and calculate the green delay.
2. **Adaptive Control:** Ensures smooth traffic flow by dynamically adjusting signal timings based on number of vehicles.