VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belagavi- 590018, Karnataka.

The Project Report

on

"Smart AI System for Optimizing Traffic Lights on Congested Routes"

Submitted in partial fulfillment for the award of degree of

BACHELOR OF ENGINEERING

in

INFORMATION SCIENCE AND ENGINEERING

Project Associates

1. Mr. PRATHEEK R T	4GM21IS035
2. Mr. B DEEPAK	4GM21IS005
3. Ms. SAHANA M	4GM21IS039
4. Ms. POOJA P	4GM21IS032

Academic Year 2024-25

Under the Guidance of

Dr. Neelambike S

B.E, M.Tech., Ph. D

Associate Professor & Head

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

Srishyla Educational Trust (R), Bheemasamudra

GM INSTITUTE OF TECHNOLOGY, DAVANGERE

Approved by AICTE | Affiliated by V.T.U Belagavi | Recognized by Govt. of Karnataka

NATIONAL BOARD OF ACCREDITATION
Accredited Programs
CSE/ECE/CE/ME/ISE/BT

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

G M Institute of Technology

DAVANGERE - 577 006

Department Of Information Science and Engineering

CERTIFICATE

Certified that the Project Report entitled "SMART AI SYSTEM FOR OPTIMIZING TRAFFIC LIGHTS ON CONGESTED ROUTES" carried out by Mr. PRATHEEK R T **[4GM21IS035],** Mr. B DEEPAK **[4GM21IS005],** Ms. SAHANA M **[4GM21IS039]** and **Ms.** POOJA P **[4GM21IS032],** bonafide students of VII Semester in partial fulfillment for the award of Bachelor of Engineering in Information Science and Engineering, GM Institute of Technology, Visvesvaraya Technological University, Belagavi for the academic year 2024-25. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report. The project report has been approved as it satisfies theacademic requirements in respect of project work prescribed for the said degree.

Dr. Neelambike S	Dr. Neelambike S	Dr. Sanjay Pande M B
Project Guide	Head of the Department	Professor & Principal

ABSTRACT

Traffic congestion has emerged as one of the most critical challenges facing urban centers today, primarily driven by the surging population and the increasing number of automobiles on the road. This phenomenon not only results in significant delays and heightened stress for drivers but also contributes to escalated fuel consumption and worsened air quality. The implications of traffic congestion are particularly severe in megacities, where the interplay of dense populations and extensive vehicle use creates a perfect storm for gridlock. As the demand for road space continues to rise, addressing the complexities of urban traffic management becomes paramount.

The pervasive nature of traffic jams calls for a robust and dynamic approach to traffic management, particularly the real-time monitoring of road traffic density. This enables traffic authorities to make informed decisions regarding signal control and to implement effective traffic management strategies. Central to this effort is the role of traffic controllers, which significantly influence the flow of vehicles and pedestrians alike. As cities strive to enhance mobility and reduce congestion, there is an urgent need to optimize traffic control systems to better accommodate the increasing demand.

Our proposed system harnesses the power of cutting-edge technologies to provide an innovative solution for traffic management. By utilizing live images captured from strategically placed cameras at traffic junctions, our system employs advanced image processing techniques combined with artificial intelligence to calculate real-time traffic density. This data-driven approach allows for a comprehensive understanding of traffic patterns and behaviors, providing peak congestion times and critical traffic flow dynamics.

ACKNOWLEDGEMENT

We are grateful to **Dr. Sanjay Pande M B**, the **Principal of G M Institute of Technology, Davanagere**, for his unwavering support and commitment to excellence in education. Their guidance has inspired us to strive for greater heights.

We extend our appreciation to **Dr. Neelambike S**, the **Head of the Information Science and Engineering**, for his encouragement and support throughout this project. Their leadership and vision have been instrumental in fostering an environment conducive to academic growth.

We would like to express our deep appreciation to our project supervisor **Dr. Neelambike S, Associate Professor & Head of the Department**, for her invaluable guidance, support, and constructive feedback. Their expertise has been indispensable in shaping our work.

We are also thankful to the teaching and non-teaching staff of the **Department of Information Science and Engineering, G M Institute of Technology** for their dedication and assistance during our academic journey. Their contributions, both inside and outside the classroom, have enriched our learning experience. Furthermore, we would like to acknowledge the support of our classmates and friends for their encouragement and camaraderie throughout this project. Their solidarity has been a constant source of motivation.

Last but not least, we express our gratitude to our families for their understanding, patience, and unwavering support during our academic pursuits. Their love and encouragement have been the cornerstone of our success.

To all those mentioned above and others who have contributed directly or indirectly to this project, we extend our heartfelt thanks. Your support has been invaluable, and we are deeply grateful for it.

Pratheek R T

B Deepak

Sahana M

Pooja P

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The preamble comprises of problem statement, solution, objectives, literature review and organization of the report.

1.1 PROBLEM STATEMENT

In today's rapidly evolving world the urban traffic congestion has become a pervasive issue in cities worldwide, exacerbating environmental, economic, and social challenges. Traditional traffic light systems, which often rely on pre-programmed or static timing schedules, struggle to address the dynamic nature of modern traffic flows. These systems fail to adapt to real-time traffic conditions, leading to frequent delays, prolonged travel times, and increased fuel consumption.

Public transportation, pedestrians, and cyclists are often neglected in these systems. For instance, responders are impeded by poorly timed lights, delaying lifesaving services, while pedestrians and cyclists face risks for crossing signals.

Pedestrian and cyclist safety further highlights the inadequacy of current systems, which often fail to adapt to their needs during peak or unpredictable conditions. This issue undermines efforts to promote sustainable and active modes of transportation, such as walking and cycling, which are vital for reducing urban congestion and emissions. Addressing this problem requires an innovative approach that combines technology, adaptability, and inclusivity.

Our Problem statement can be addressed as

" The inefficiency of traditional traffic light systems in managing high traffic volumes leads to congestion, longer travel times. These systems lack real-time adaptability, causing delays and poor traffic flow failing to prioritize vehicles, public transport, pedestrians, and cyclists. This compromises urban mobility and safety, particularly for vulnerable road users. As cities evolve into smart urban spaces, there is a growing need for adaptive systems that utilize real-time data to optimize traffic flow. A data-driven approach can address these challenges by improving signal timing. Advanced traffic solutions are essential for sustainable and efficient urban transportation."

1.2 THE SOLUTION

Our solution addresses as

"To develop a solution this system would utilize adaptive traffic signal technology powered by machine learning algorithms that analyze real-time and historical data, enabling dynamic adjustment of signal timings based on current traffic conditions."

To address the inefficiencies of traditional traffic light systems, a Smart AI System for Optimizing Traffic Lights can be developed to enhance urban mobility and reduce congestion. This system would utilize adaptive traffic signal technology powered by machine learning algorithms that analyze real-time and historical data, enabling dynamic adjustment of signal timings based on current traffic conditions. Sensors such as cameras and inductive loops would be installed at intersections to monitor vehicle flow, speeds, and pedestrian activity. Additionally, integration with connected vehicle technology (V2I) would allow vehicles to communicate directly with traffic infrastructure, enhancing decision-making and ensuring smoother traffic flow.

The system would prioritize vehicles, public transport, and vulnerable road users like pedestrians and cyclists. Vehicles could be detected using GPS or RFID, enabling green-light priority, while public transit vehicles like buses and trams would benefit from faster and uninterrupted transit routes. Pedestrian and cyclist safety would be addressed by using dedicated sensors to extend crossing times during high pedestrian activity. Furthermore, the system would integrate external data such as weather forecasts and special event schedules to anticipate traffic surges.

A centralized traffic management platform would collect and analyze data from multiple intersections, allowing city-wide optimization and continuous improvement. Advanced AI models, such as reinforcement learning, would be used to predict and adapt to fluctuating traffic patterns. The system would also include a user feedback mechanism via mobile apps to incorporate input from road users, ensuring it remains responsive to community needs. This intelligent, data-driven approach would significantly reduce congestion, improve travel times, enhance safety, and support sustainable urban development.

1.3 OBJECTIVES OF PROJECT

- **Vehicle Detection:** The system aims to accurately detect vehicles at intersections and along congested routes using advanced technologies such as cameras, radar sensors, and IoT devices. Vehicle detection provides essential data about traffic volume, vehicle speed, and the types of vehicles on the road.
- **Reduce Traffic Congestion:** One of the primary goals of the project is to minimize congestion on urban roads, particularly during peak hours. By leveraging AI algorithms and real-time data, the system adjusts traffic signals to reduce vehicle waiting times at intersections and avoid bottlenecks. Reducing congestion not only improves travel efficiency but also decreases fuel consumption and greenhouse gas emissions.
- **Improve Safety for All Road Users:** The project emphasizes enhancing safety for drivers, pedestrians, and cyclists. By incorporating sensors to detect pedestrians and cyclists, the system ensures appropriate signal timings, such as extended crossing times when necessary. This reduces the risk of accidents at busy intersections.

1.4 ADVANTAGES

- **Reduced Traffic Congestion:** The system dynamically adjusts traffic signal timings based on real-time conditions, minimizing delays and preventing bottlenecks. This leads to smoother traffic flow, even during peak hours or special events.
- **Improved Travel Efficiency:** By reducing waiting times at intersections and optimizing traffic movement, commuters can experience shorter travel times. This is especially beneficial in high-density urban areas where time savings can be substantial.
- **Enhanced Safety for Road Users:** The system ensures safer crossings for pedestrians and cyclists by detecting their presence and extending crossing times when needed. It also reduces the risk of accidents by managing vehicle speeds and flow at intersections.
- **Priority for Vehicles:** Vehicles can reach their destinations faster and more efficiently through real-time detection and green-light prioritization, improving response times in critical situations and potentially saving lives.
- **Lower Fuel Consumption and Emissions:** By reducing idling times and ensuring a steady flow of traffic, the system decreases unnecessary fuel consumption and carbon emissions, contributing to cleaner air and a more sustainable environment.

- **Support for Public Transit Systems:** The system prioritizes buses and trams at intersections, enabling faster and more reliable public transit services. This can encourage greater use of public transportation, reducing reliance on private vehicles.
- **Scalability and Integration with Smart City Initiatives:** The system can be easily integrated with existing smart city technologies, such as connected vehicles and IoT infrastructure. It is scalable, allowing cities to expand or adapt the system as needed.
- **Continuous Improvement Through Data Analytics:** By collecting and analyzing vast amounts of traffic data, the system can evolve over time. Insights from machine learning algorithms enable ongoing refinements, ensuring the system remains efficient and responsive to changing traffic conditions.

1.5 LITERATURE REVIEW

Traffic management has been a critical area of research due to the increasing urbanization and the challenges posed by traditional traffic signal systems. In the Fig 1.1 the inefficiency of fixed-timing traffic lights in adapting to fluctuating traffic patterns has been highlighted in numerous studies, emphasizing the need for intelligent, data-driven solutions. This literature review explores advancements in traffic signal optimization, adaptive traffic systems, and their integration with AI technologies.

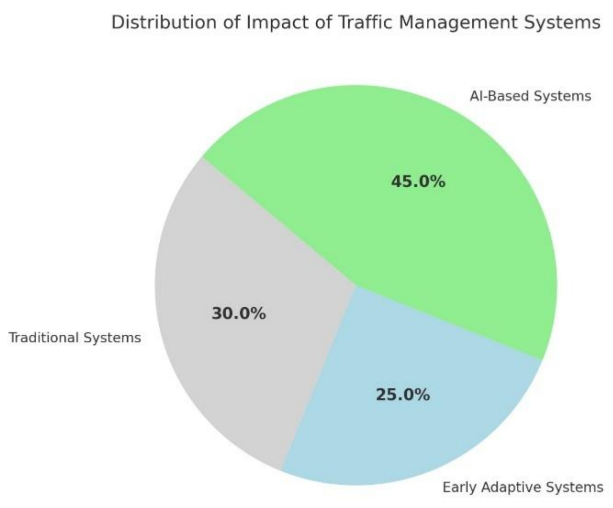


Fig 1.1 Distribution of Impact of Traffic Management Systems

- **Adaptive Traffic Signal Control:** Adaptive traffic signal systems have been widely researched as an alternative to fixed-timing systems. Studies by Mirchandani and Head (2001) introduced the concept of SCOOT (Split Cycle Offset Optimization Technique) and SCATS (Sydney Coordinated Adaptive Traffic System), which use traffic data from sensors to optimize signal timings. These systems have demonstrated improved traffic flow and reduced congestion in urban areas. However, their reliance on predefined algorithms limits their adaptability to unpredictable traffic surges, paving the way for AI-driven approaches.

- **Machine Learning in Traffic Optimization:** Recent advancements in machine learning have shown promising results in traffic management. Reinforcement learning, in particular, has been applied to optimize signal timings based on real-time traffic data. For instance, studies by Abdulhai, Pringle, and Karakoulas (2003) demonstrated the potential of reinforcement learning to dynamically adjust traffic signals, outperforming traditional optimization techniques. Moreover, deep learning models have been explored to predict traffic patterns using historical data, enabling proactive adjustments to traffic lights.

- **IoT and V2I Communication in Traffic Systems:** The integration of Internet of Things (IoT) and Vehicle-to-Infrastructure (V2I) communication has significantly enhanced real-time traffic data collection. Research by Zhang et al. (2011) highlighted the role of IoT devices, such as cameras and sensors, in providing accurate traffic flow data. V2I communication further enables connected vehicles to share real-time location and speed data with traffic systems, allowing for more precise signal optimization. Such systems have been successfully implemented in smart cities like Singapore and Amsterdam, showcasing their scalability and effectiveness.

- **Vehicle and Pedestrian Prioritization**: The importance of prioritizing vehicles and vulnerable road users has been a recurring theme in traffic management literature. Studies by Li and Wang (2017) proposed RFID-based systems to detect vehicles and grant them green-light priority. Additionally, pedestrian and cyclist safety has been addressed through sensor-based systems, as noted by studies from Peden et al. (2004), which emphasize the need for adaptive systems to prevent accidents.

- **Environmental and Economic Impact:** The environmental benefits of optimized traffic systems have been extensively documented. Research by Barth and Boriboonsomsin (2008) demonstrated that reducing idling time at intersections can significantly lower greenhouse gas emissions. Economic studies also reveal that such systems reduce fuel consumption and operational costs, making them a cost-effective solution for cities. V2I communication further enables connected vehicles to share real-time location and speed data with traffic systems, allowing for more precise signal optimization. Such systems have been successfully implemented in smart cities like Singapore and Amsterdam, showcasing their scalability and effectiveness.

- **Challenges and Gaps:** Despite the progress, challenges remain in implementing adaptive traffic systems on a large scale. Issues such as high installation costs, data privacy concerns, and the need for robust AI models capable of handling diverse traffic scenarios are frequently discussed in the literature. Furthermore, while many systems prioritize vehicles, there is often a lack of emphasis on non-motorized road users. Research by Barth and Boriboonsomsin (2008) demonstrated that reducing idling time at intersections can significantly lower greenhouse gas emissions. However, their reliance on predefined algorithms limits their adaptability to unpredictable traffic surges.

1.6 ORGANIZATION OF REPORT

Chapter 1 Preamble:

The preamble in a chapter provides a concise introduction, outlining the chapter's main themes and objectives. It may include background information and context to help readers understand the subject matter. Additionally, the preamble serves as a roadmap, guiding readers through the upcoming content**.**

This section includes:

1. Problem statement
2. The solution
3. Objectives of the project
4. Advantages of the project

Chapter 2 Software Requirement specification:

It outlines the functional and non-functional requirements of a software system, providing a blueprint for development and ensuring alignment on project goals and constraints.

This section includes:

1. Functional overview
2. User characteristics
3. Input Requirements
4. Output requirements
5. Functional requirements
6. Software Requirements
7. Hardware requirements
8. Project cycle

Chapter 3 System design:

System design involves the process of creating a detailed blueprint for a software system, including its architecture, components, interfaces, and interactions.

This section includes:

1. Project Architecture
2. Database design
3. Sequence diagram
4. Dataflow diagram
5. Database table structure
6. Flow chart
7. Utilities

Chapter 4 Implementation:

Implementation refers to the process of translating a design or plan into a working software system. It involves writing code, integrating components, and testing the system to ensure that it functions according to the specified requirements.

Implementation typically follows the system design phase and involves programming, goals. Successful implementation results in a fully functional software product ready for deployment and use by end-users.

This section includes:

1. Software Tools used
2. Implementation details
3. Front end forms deign

Chapter 5 Testing:

Testing is the process of evaluating a software system to identify defects, errors, or discrepancies between expected and actual outcomes. It involves executing test cases, scripts, or scenarios to verify that the software functions correctly and meets specified requirements. Testing encompasses various techniques such as unit testing, integration testing, system testing, and acceptance testing, aiming to ensure the quality, reliability, and robustness of the software.

This section includes:

1. Scope
2. Unit Testing
3. Integration Testing
4. Functional testing
5. System Testing

Chapter 6 Results:

The results section summarizes and presents the findings of the study.

This section includes:

- Snap shot of projects

Chapter 7 Conclusion:

The conclusion section summarizes the main findings and outcomes of the project, offering insights into its achievements, challenges, and potential future directions. It provides closure to the report and highlights the project's contributions to the field of software development.

This section includes:

1. Conclusion
2. Future Scope

Appendix:

The appendix section of a document typically contains supplementary materials that support or enhance the main content. These materials may include additional data, charts, graphs, tables, code samples, or other relevant information that provides further context or detail. The appendix serves as a reference for readers who may want to delve deeper into specific aspects of the document or need additional information to better understand the content.

References:

The references section of a document lists all the sources cited or consulted during the research or writing process. It provides readers with the necessary information to locate and verify the credibility of the sources used in the document. References are typically listed alphabetically by the author's last name or by the title of the source if no author is available. Each reference includes essential details such as the author's name, publication year, title of the work, publisher, and relevant page numbers or URLs.

CHAPTER 2

SOFTWARE REQUIREMENT SPECIFICATION

A Software Requirements Specification (SRS) is a comprehensive document detailing the functional and non-functional requirements of a software system, serving as a blueprint for development.

2.1 FUNCTIONAL OVERVIEW

- **Real-Time Traffic Data Collection:** The system should be capable of collecting real-time data from various sources, including cameras, inductive loops, and IoT sensors installed at intersections. This data will include traffic volume, vehicle speeds, vehicle types, and pedestrian movement, providing a comprehensive view of current traffic conditions.
- **Dynamic Signal Timing Adjustment:** The system must use machine learning algorithms to dynamically adjust the timing of traffic signals based on real-time traffic data. This ensures traffic lights adapt to the current traffic flow, optimizing vehicle movement and minimizing congestion at intersections.
- **Vehicle Detection and Priority:** The system should be able to detect vehicles approaching intersections using GPS, RFID, or V2I (Vehicle-to-Infrastructure) communication. Upon detection, the system should automatically prioritize green light phases for these vehicles to enable quick and safe passage.
- **Public Transport Priority Management:** The system should recognize and prioritize public transport vehicles, such as buses and trams, at traffic signals. This helps improve the efficiency of public transit systems by reducing delays and ensuring better service reliability for passengers.
- **Pedestrian and Cyclist Safety Features:** The system should include sensors that detect the presence of pedestrians and cyclists at intersections. Signal timings should be adjusted to extend crossing periods as needed, ensuring the safety of non-motorized road users.
- **Integration with Connected Vehicle Technologies:** The system must be able to communicate with connected vehicles through V2I and V2V (Vehicle-to-Vehicle)

networks to receive real-time data such as vehicle positions and speeds. This integration enhances traffic management by allowing vehicles and traffic signals to coordinate movements for smoother traffic flow.

- **Data Analysis and Traffic Prediction:** The system should include an AI-powered data analysis component capable of processing historical traffic data and current real-time inputs. It should use predictive analytics to anticipate future traffic conditions, allowing preemptive adjustments to traffic signal timings.
- **Centralized Traffic Management Dashboard:** The system should feature a centralized dashboard that provides traffic managers with real-time insights, visualizations, and control options.

2.2 USER CHARACTERISTICS

- **Traffic Control Operators:** These users are responsible for overseeing and managing the city's traffic systems. They should have a background in traffic management and an understanding of traffic flow principles. They need to be comfortable using data-driven interfaces and interpreting real-time data visualizations to make informed decisions.
- **City Planners and Urban Development Specialists:** Users in this category are involved in planning and developing urban transportation systems. They should have knowledge of traffic patterns, public transportation needs, and sustainability practices. These users will need to analyze reports and data trends to design infrastructure improvements and city layouts that align with traffic system optimizations.
- **Public Transportation Authorities:** These users manage public transit services such as buses, trams, and subways. They should understand transit schedules, vehicle routing, and service reliability. The system should provide them with tools to monitor traffic signals and prioritize public transport vehicles to minimize delays and improve service efficiency.
- **Responders and Coordination Personnel:** Vehicle operators, dispatchers, and coordination personnel require the system to recognize and prioritize their vehicles at intersections. They need to be able to communicate with the system to alert it about approaching situations and ensure swift passage through traffic lights.

2.3 INPUT REQUIREMENTS

Input requirements entail providing users with an intuitive interface to input search queries and product specifications, ensuring accuracy and ease of use.

2.3.1 SERVER-SIDE REQUIREMENTS

- **Real-Time Data Integration**

  The server must handle user search queries efficiently, parsing keywords and filtering product specifications to deliver accurate and relevant results. It should support advanced search algorithms, including natural language processing (NLP), for better user query understanding.

- **Advanced Algorithm Hosting**

  The server should maintain and manage a comprehensive, scalable database of products, including details such as descriptions, specifications, prices, and availability. It should enable fast retrieval of information for user queries.

- **Data Storage and Management**

  A centralized database is required to store historical traffic data, real-time inputs, system logs, and user interactions. The server must support efficient data retrieval and backup systems to maintain data integrity and support analytics.

- **Secure Data Handling**

  The server must implement encryption protocols and secure authentication mechanisms to protect user data and transaction details. Compliance with data protection regulations such as GDPR should be ensured.

- **Dashboard and Monitoring**

  The server should host a centralized dashboard for traffic operators, displaying real-time traffic data, system status, and key performance metrics. It should also support alerts for system faults.

- **Compliance with Security Standards**

  The server must adhere to data security regulations, including encryption for sensitive data, robust user authentication mechanisms, and regular vulnerability assessments to prevent breaches.

2.3.2 CLIENT-SIDE REQUIREMENTS

- **User Interface for Traffic Operators**

  The client-side application must provide an intuitive and user-friendly interface for traffic operators. This includes dashboards displaying real-time traffic conditions, signal statuses, vehicle counts, and system alerts. It should allow operators to make manual adjustments if necessary.
- **Vehicle Communication Module**

  The client-side application for responders should include a module to send real-time priority requests to the server. This enables the system to adjust traffic lights for vehicles dynamically.
- **Pedestrian Cyclist Interfaces**

  Client-side interfaces (e.g., mobile apps or kiosks) for pedestrians and cyclists should provide notifications about crossing times and safety alerts, enhancing their interaction with the system.
- **Data Visualization and Reports**

  The client-side system should include tools for visualizing historical data, analyzing traffic trends, and generating reports for city planners and decision-makers.
- **Low Resource Consumption**

  The client should display personalized product recommendations based on user behavior, preferences, and search history. This feature should be prominently and seamlessly integrated into the interface.
- **Secure User Authentication**

  The client must support secure user login and registration processes, including options for social media login and two-factor authentication, to ensure user account safety.
- **Mobile and Web Accessibility**

  The client system should be accessible via both web and mobile platforms. This ensures that users such as traffic managers, responders, and city planners can monitor and interact with the system from any location.
- **Cyclist Interfaces**

  Client-side interfaces (e.g., mobile apps or kiosks) for pedestrians and cyclists should provide notifications about crossing times and safety alerts, enhancing their interaction with the system.

2.4 OUTPUT REQUIREMENTS

The output requirements, contains the critical information and error messages.

2.4.1 CRITICAL INFORMATION

Critical Information includes:

- **Traffic Conditions:** Current traffic flow (e.g., "Moderate Traffic on Main Street, average speed: 30 km/h"). Notify about high congestion areas (e.g., "Severe congestion at Intersection A, average delay: 10 minutes"). Show the status of traffic lights at key intersections (e.g., "Green light at Intersection B, remaining time: 15 seconds").
- **System Alerts:** Notify when an vehicle is approaching.
- **Environmental Impact Data:** Display the estimated fuel saved due to optimized traffic flow (e.g., "Fuel savings today: 500 liters").
- **System Performance:** Share statistics like average travel time, reduced delays, and system uptime (e.g., "System uptime: 99.8%, average delay reduced by 25%").

2.4.2 ERROR MESSAGES

In case of errors or issues encountered during data retrieval and comparison, the system is going to provide clear and concise error messages. These error messages will:

- **Communicate Errors Clearly:** Error messages will clearly indicate the nature of the problem encountered, whether technical, data-related, or otherwise.
- **Provide Guidance:** Users will be guided on potential actions they can take to resolve the issue or access the required information.
- **Maintain User Confidence:** Transparent and informative error messages are given for maintaining user confidence in the platform's reliability and functionality.

2.5 FUNCTIONAL REQUIREMENTS

Functional requirements outline the specific functionalities or features that a software system must possess to fulfill its intended purpose and meet the needs of its users. These requirements describe what the system should do, including its capabilities, behaviours, and interactions with users and other system components.

- **Real-Time Traffic Data Collection**

  Sensor Integration is the collect real-time traffic data using IoT sensors, cameras, and GPS devices. Data types consist of record vehicle counts, speeds, pedestrian and cyclist movements, congestion levels, and road conditions. Data accuracy will ensure data precision for reliable traffic predictions.

- **AI-Based Traffic Light Optimization**

  Dynamic signal timing means it use AI algorithms to adjust traffic signal durations based on real-time traffic flow. Congestion prediction employ machine learning models to forecast congestion and preemptively optimize signal cycles. Continuous learning improves optimization strategies through reinforcement learning and historical data analysis.

- **Vehicle Prioritization**

  Detect vehicles using V2I communication or sensors. Priority adjustment automatically create green corridors for vehicles by adjusting signal timings. Fail-safe mechanism will notify operators in case of errors in detection or prioritization.

- **Public Transport and Pedestrian Management**

  Public transport priority which detects buses and trams approaching intersections. Extend green lights to minimize their delays. Pedestrian and Cyclist Safety Detect pedestrians and cyclists at crossings. Provide appropriate crossing times and priority.

- **User Interfaces**

  Operator Dashboard display real-time traffic conditions and signal statuses. Provide options for manual overrides in critical situations. Mobile Applications offer road users updates on congestion, estimated travel times, and alternative routes. Public displays show countdown timers, congestion alerts, and pedestrian crossing statuses.

- **Integration with Smart City Systems**

  V2I Communication enable data exchange with connected vehicles for seamless traffic management. Public transport systems integrate with bus and tram schedules to improve urban mobility.

- **Environmental Impact Monitoring**

  Fuel and Emission Metrics Calculate fuel savings and CO2 reductions due to optimized traffic flow. Energy efficiency monitor energy consumption of traffic signal hardware.

2.6 SOFTWARE REQUIREMENTS

Table 2.1 represents the software requirements specification (SRS) is a comprehensive document detailing the functional and non-functional requirements of a software system, serving as a blueprint for development.

Table 2.1 Software requirements

Software	**Requirement**
Operating system	Windows 7 or more
Programming Language	Python
Libraries	NumPy, Keras
Image Processing library	OpenCV
Object detection model	YOLO
Code editor	VS Code or equivalent

2.7 HARDWARE REQUIREMENTS

Table 2.2 represents the hardware requirements that a hardware system must possess to fulfill its intended purpose and meet the needs of its users. These requirements entail providing users with an intuitive interface to input search queries and product specifications, ensuring accuracy and ease of use.

Table 2.2 Hardware requirements

Hardware	**Requirement**
RAM	4 GB or more
Processor Speed	1.5 GHz or more
Hard Disk	128 GB or more
Processor	Intel core i3 or more
Display Monitor	For Visualization of simulations

2.8 PROJECT CYCLE

The project cycle, also known as the software development life cycle (SDLC), refers to the stages involved in the development of a software project from initiation to completion.

Waterfall is a linear approach to software development, where each phase must be completed before moving on to the next. It's often pictured as a cascading waterfall, where each step feeds into the next.

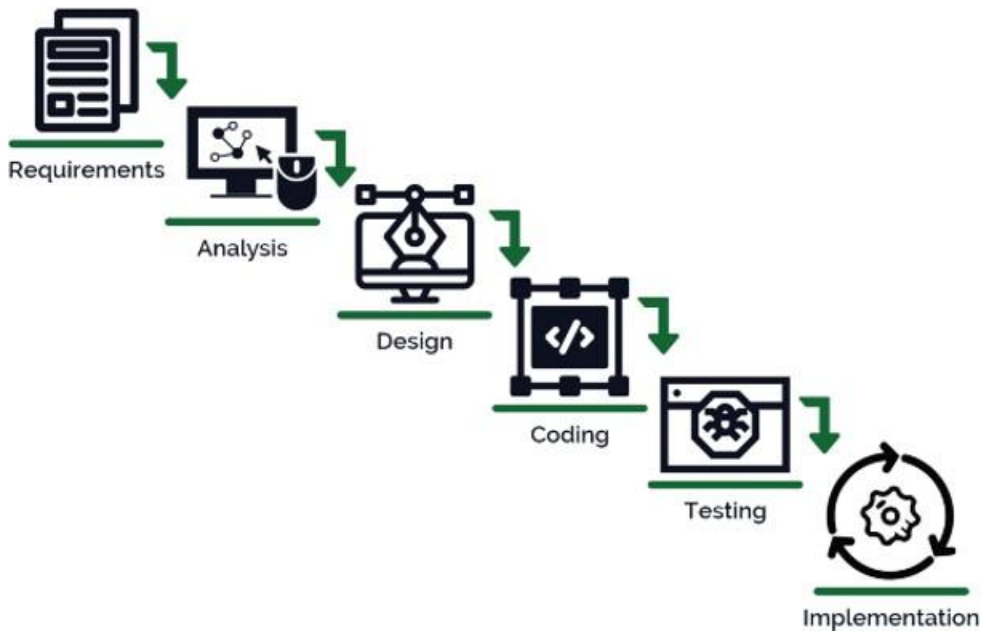


Fig 2.1 Project Cycle

- **Requirements Gathering and Analysis:**

In the Fig 2.1 the project cycle begins with gathering and documenting all the requirements for the price comparison website. This includes defining the functionalities, features, and specifications needed for the website, such as product search, browsing, dynamic price comparison, user authentication, and more.

- **Design**:

Once the requirements are finalized, the design phase begins. This involves creating detailed designs for the website's architecture, user interface, database structure, and system components. Design decisions are made based on the gathered requirements and specifications.

➢ **Coding:**

In this stage, the system is coded based on the design.

This phase involves writing code, building databases, creating user interfaces, and integrating various components to create the functional website.

➢ **Testing:**

After the coding phase, the website undergoes comprehensive testing to ensure that it meets the specified requirements and quality standards. This includes testing for functionality, performance, usability, security, and compatibility across different browsers and devices.

➢ **Implementation and Deployment:**

Once testing is complete and the website is deemed ready for production, it is deployed into the live environment. This involves setting up servers, configuring the website, and making it accessible to end-users.

➢ **Maintenance:**

After deployment, the website enters the maintenance phase, where ongoing support, updates, and enhancements are provided. This ensures that the website remains functional, secure, and up-to-date to meet the changing needs of users and business requirements.

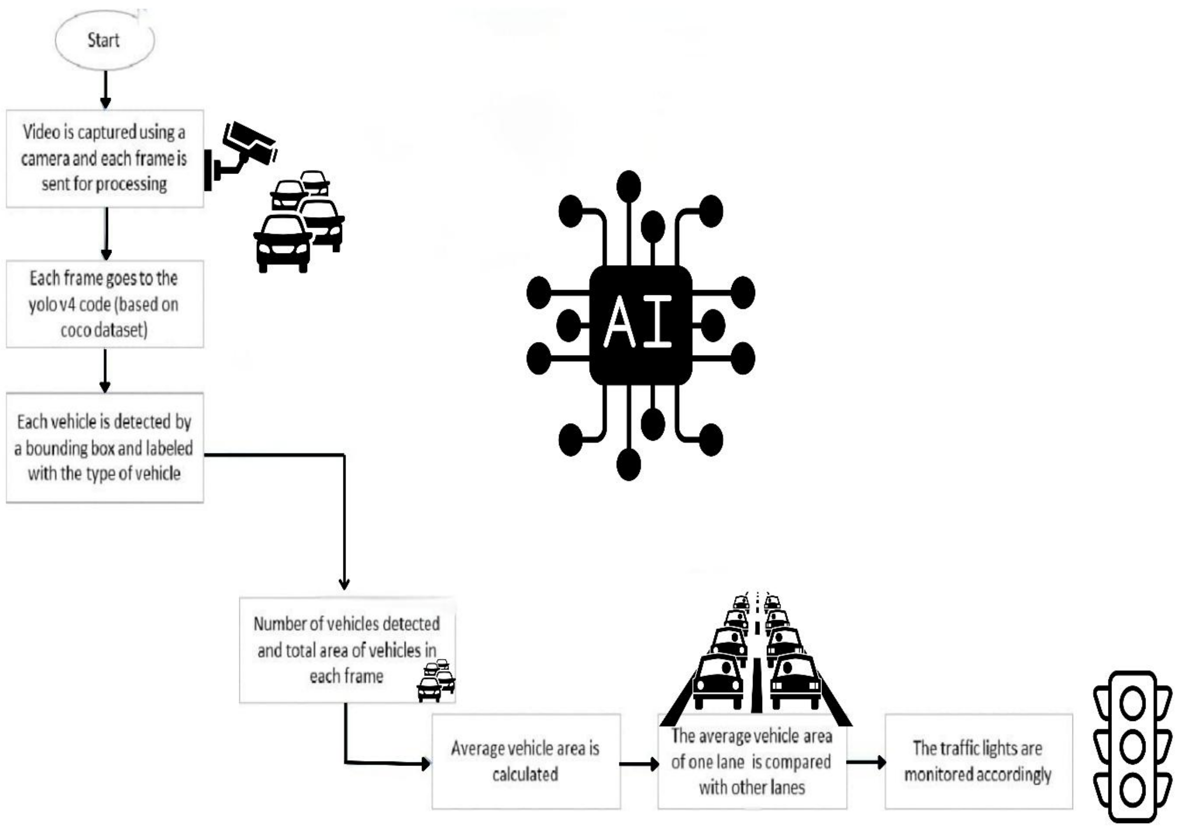
CHAPTER 3

SYSTEM DESIGN

This chapter includes project architecture and description, sequence diagrams, data flow diagrams, flowchart and utility.

3.1 PROJECT ARCHITECTURE AND DESCRIPTION

In the Fig 3.1 the Smart AI Traffic Light System architecture is designed to optimize traffic flow using real-time video processing and AI algorithms. The system begins by capturing live video feeds from cameras installed at traffic intersections. These video frames are then processed using the YOLOv4 object detection model, which is trained on the COCO dataset to identify and classify vehicles in each frame. The detected vehicles are enclosed within bounding boxes and labeled according to their type, such as cars, buses, or motorcycles.



Start
Video is captured using a camera and each frame is sent for processing
Each frame goes to the yolo v4 code (based on coco dataset)
Each vehicle is detected by a bounding box and labeled with the type of vehicle
AI
Number of vehicles detected and total area of vehicles in each frame
Average vehicle area is calculated
The average vehicle area of one lane is compared with other lanes
The traffic lights are monitored accordingly

Fig 3.1 Project Architecture

3.2 SEQUENCE DIAGRAM

In the Fig 3.2 the sequence diagram illustrates the dynamic interactions and workflow within the Smart AI Traffic Light System, emphasizing the integration of data collection, processing, decision-making, and signal control.

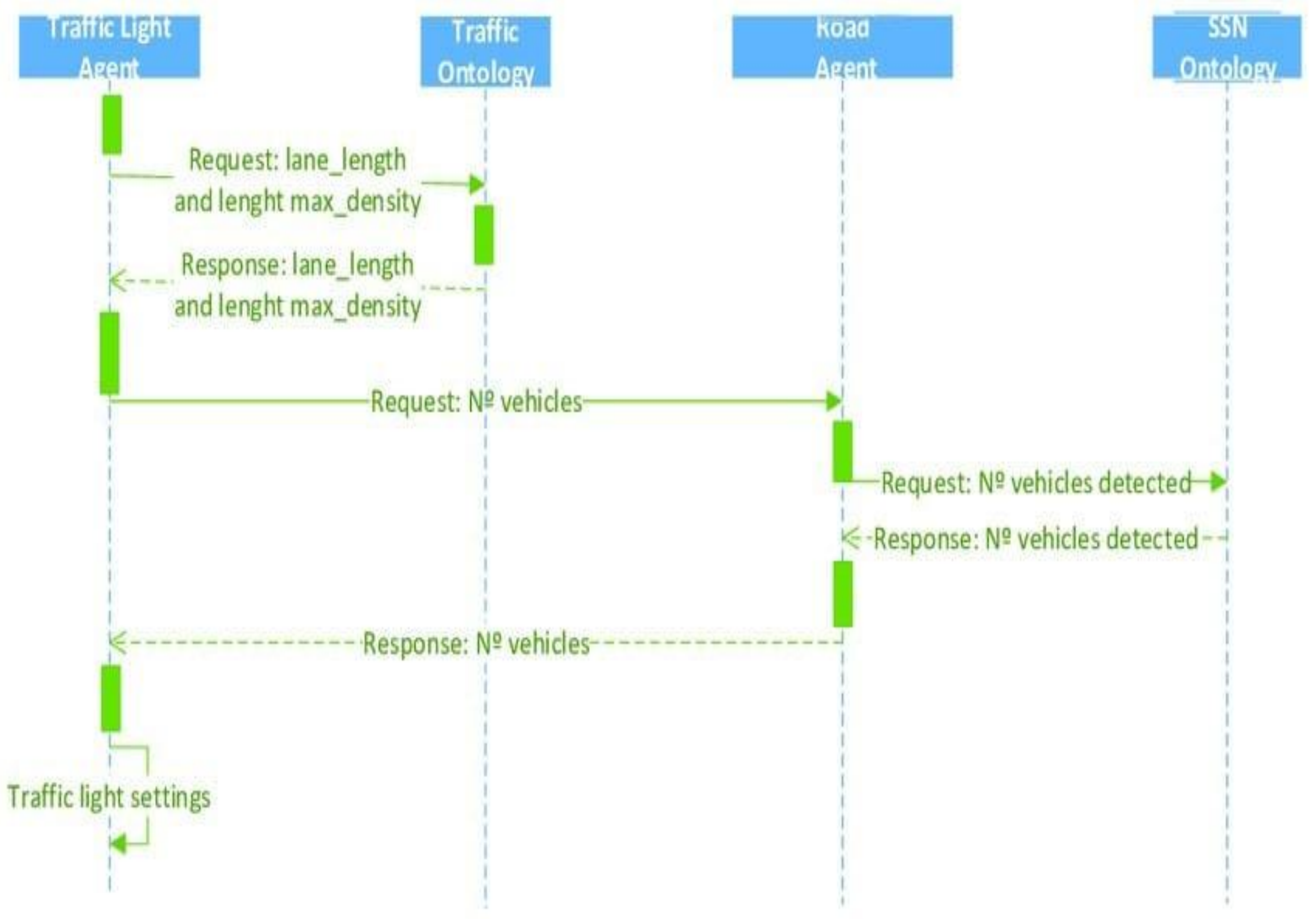

Traffic Light Agent
Traffic Ontology
Road Agent
SSN Ontology
Request: lane_length and lenght max_density
Response: lane_length and lenght max_density
Request: Nº vehicles
Request: Nº vehicles detected
Response: Nº vehicles detected
Response: Nº vehicles
Traffic light settings

Fig 3.2 Sequence Diagram-User

The process begins with the initialization of the system, where input devices like cameras and sensors actively gather real-time data. These inputs include vehicle count, speed, lane occupancy, and pedestrian activity. The collected raw data undergoes preprocessing, where noise filtering is applied to remove any inaccuracies, such as false detections or environmental interference, ensuring the reliability of the data for further analysis.

The preprocessed data is then sent to the AI processing unit, which uses machine learning algorithms to analyze traffic density and patterns. The AI evaluates traffic flow across various lanes, identifying areas with high congestion and prioritizing them accordingly. Based on this analysis, the system determines the optimal signal timings and priorities to maximize traffic efficiency while minimizing delays. The system then validates the existing signal configurations to check if any updates are required and applies optimized signal timings where necessary.

The updated signal timings are communicated to the output devices, which include traffic lights and pedestrian signals. These signals are dynamically adjusted in real-time to reflect the AI-driven decisions, ensuring smoother traffic movement, better pedestrian safety, and priority for vehicles if detected. The system continuously monitors the traffic flow after implementing the changes, dynamically adapting to any new conditions, such as sudden spikes in traffic density or changes in pedestrian movement.

This intelligent sequence of actions highlights the adaptive and automated nature of the system, which reduces human intervention while efficiently managing traffic. The process concludes one operational cycle, ready to restart seamlessly and maintain ongoing real-time traffic management. Such a system not only optimizes traffic flow but also enhances safety, reduces environmental impact by minimizing idle time, and provides a scalable solution for smart city infrastructure.

3.3 DATA FLOW DIAGRAM

In the Fig 3.3 the Data Flow Diagram (DFD) illustrates the flow of information within the Smart AI Traffic Light System, showcasing the interaction between key components for dynamic traffic management. The process begins with sensors, which play a critical role in detecting vehicles and pedestrian activity at the traffic junction.

The collected data flows into the AI Processing unit, which serves as the brain of the system. Here, the data is analyzed and processed to calculate traffic density, identify patterns, and make decisions about signal timings. The AI processing unit uses advanced algorithms to assess the input data and determine the necessary adjustments for optimizing traffic flow.

Once the analysis is complete, the processed data and calculated timings are sent to the Traffic Signal Update module. This module updates the signal timings dynamically based on the AI's recommendations. The system ensures that traffic signals are adjusted in real time to reduce congestion, prioritize vehicles, and manage pedestrian crossings effectively.

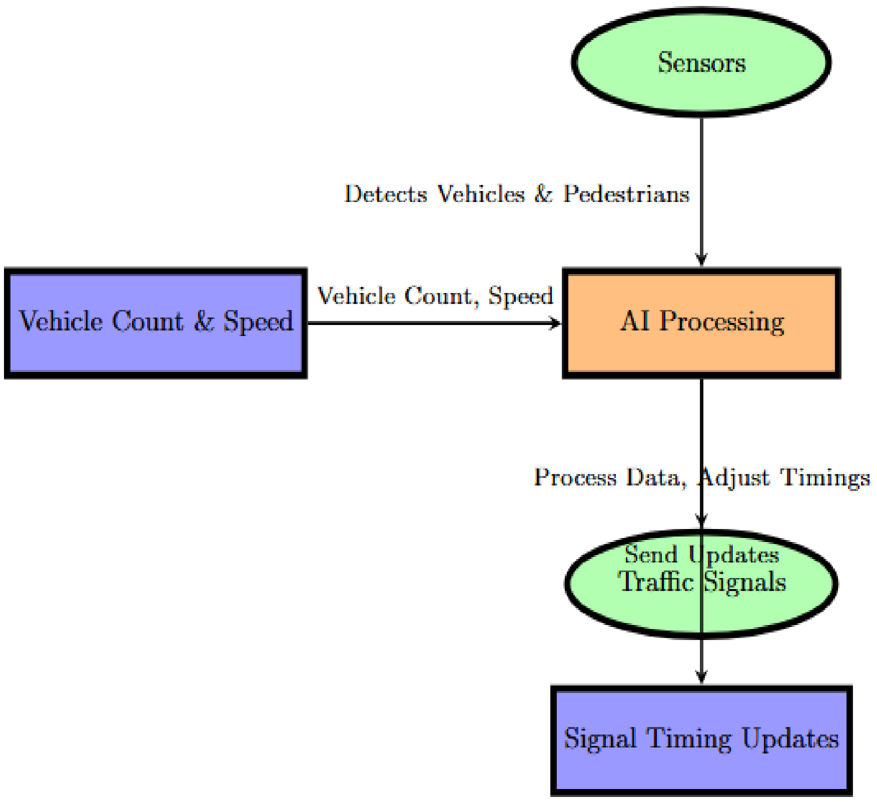


Fig 3.3 Data Flow Diagram

Finally, the updated signal timings are implemented, and the system continuously monitors traffic conditions to make further adjustments if needed. This data-driven approach ensures efficient traffic management, improves road safety, and minimizes delays for vehicles and pedestrians. The DFD effectively represents the flow and processing of data within the system, emphasizing its real-time adaptability and intelligence.

3.4 FLOW CHART

In the Fig 3.4 the flowchart is a graphical representation of a process, algorithm, or workflow. It uses various shapes and symbols connected by arrows to illustrate the steps or actions involved in completing a task.

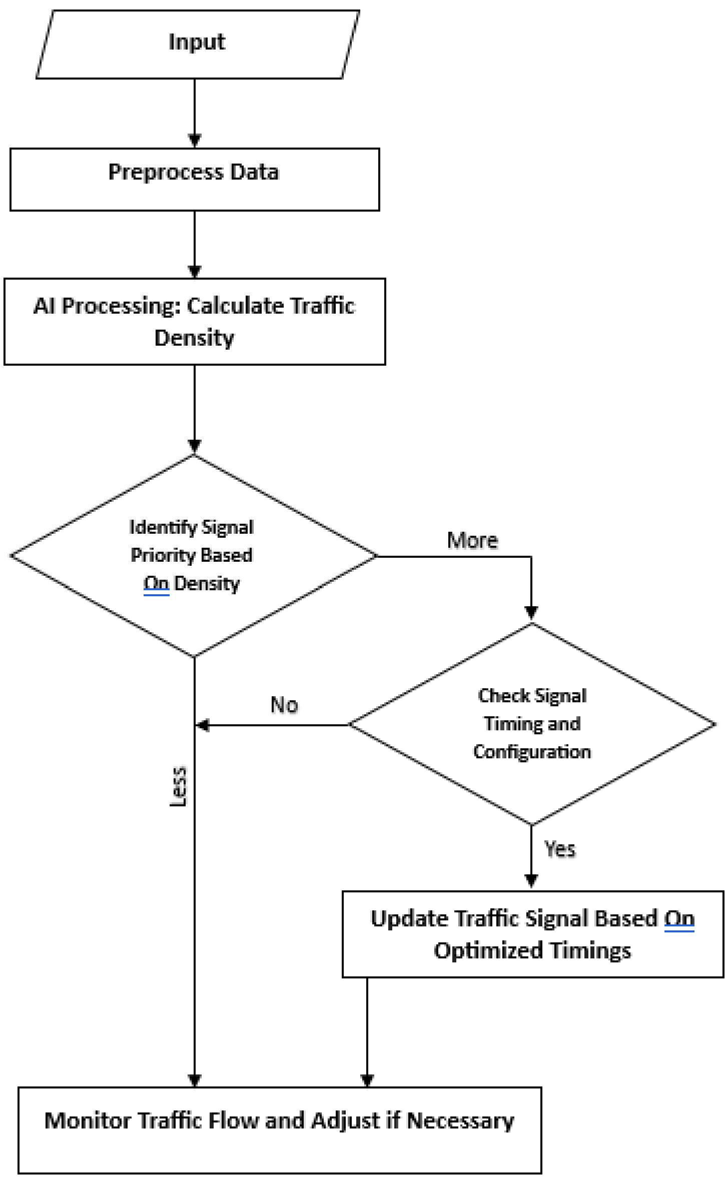

Input
Preprocess Data
AI Processing: Calculate Traffic Density
Identify Signal Priority Based On Density
More
Less
Check Signal Timing and Configuration
No
Yes
Update Traffic Signal Based On Optimized Timings
Monitor Traffic Flow and Adjust if Necessary

Fig 3.4 Flowchart

The flowchart depicts the systematic operation of a Smart AI Traffic Light System, outlining the key stages involved in its functionality. The process begins with the collection of input data from various sources, such as sensors and cameras, which provide critical information like vehicle count, speed, lane occupancy, and pedestrian activity. This raw data is then preprocessed to filter out noise and inaccuracies, ensuring that only relevant and accurate information is used for analysis. Noise filtering is a crucial step to eliminate potential errors caused by environmental factors or sensor malfunctions.

Once the data is cleaned, it is fed into the AI processing unit, which calculates traffic density and identifies patterns or anomalies in real time. Based on this analysis, the system determines which lanes or directions require priority to alleviate congestion or manage traffic more effectively. The AI model evaluates the current signal timings and configurations to check if they align with the identified traffic conditions. If discrepancies are found, the system recalculates and updates the traffic signal timings dynamically, ensuring optimized flow and efficient use of road infrastructure.

CHAPTER 4

IMPLEMENTATION

This chapter includes the software tools used, implementation details and software tools were selected to ensure that the project is robust, efficient, and capable of handling dynamic real-world scenarios like traffic light control. They also provide the flexibility needed to expand or modify the system as future requirements arise.

4.1 SOFTWARE TOOLS USED

- **Python:** Python is a high-level, versatile, and object-oriented programming language widely used for various applications, including data analysis, simulation, and machine learning. Its extensive standard library and third-party modules make it an ideal choice for building efficient and scalable systems.

- **NumPy:** NumPy (Numerical Python) is a fundamental library in Python for scientific computing and data manipulation. In this project, NumPy is used to handle arrays and perform numerical calculations efficiently. For instance, vehicle movement, traffic density calculations, and signal timing adjustments can involve operations on large datasets or matrices, which NumPy handles with optimized performance.

- **Keras:** Keras is a high-level neural networks API, running on top of TensorFlow, designed for building and training deep learning models easily and efficiently. In this project, Keras is used to create or utilize pre-trained machine learning models for traffic management tasks, such as vehicle classification and traffic pattern prediction.

- **OpenCV (Open-Source Computer Vision Library):** OpenCV is an open-source library aimed at real-time computer vision and image processing tasks. It includes functionalities for image recognition, processing, and video analysis. In this project, OpenCV plays a crucial role in detecting vehicles and processing camera feeds for dynamic traffic light control.

- **YOLO (You Only Look Once):** YOLO is a state-of-the-art object detection algorithm that processes images in real time. Unlike traditional methods, YOLO divides an image into a grid and predicts bounding boxes and class probabilities for each region in a single evaluation.

- **TensorFlow:** TensorFlow is a powerful open-source platform for machine learning and deep learning, providing tools to build, train, and deploy models across various platforms. In this project, TensorFlow serves as the backend for running AI models like YOLO for vehicle detection and traffic density estimation.

Integrated Development Environment (IDE):

Visual Studio Code (VS Code): Visual Studio Code serves as our Integrated Development Environment (IDE) for writing, editing, and managing the codebase of our project. Its extensive features, including syntax highlighting, code completion, debugging capabilities, and seamless integration with version control systems, enhance our productivity and facilitate collaborative development.

4.2 IMPLEMENTATION DETAILS

```
import cv2
from darkflow.net.build import  TFNet
import matplotlib.pyplot as plt
import os
options={
   'model':'./cfg/yolo.cfg',        #specifying the path of model
   'load':'./bin/yolov2.weights',   #weights
   'threshold':0.3                  #minimum confidence factor to create a box, greater than 0.3 good
}
tfnet=TFNet(options)
inputPath = os.getcwd() + "/test_images/"
outputPath = os.getcwd() + "/output_images/"
def detectVehicles(filename):
   global tfnet, inputPath, outputPath
   img=cv2.imread(inputPath+filename,cv2.IMREAD_COLOR)
   # img=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
   result=tfnet.return_predict(img)
   # print(result)
   for vehicle in result:
       label=vehicle['label']   #extracting label
       if(label=="car" or label=="bus" or label=="bike" or label=="truck" or
label=="rickshaw"):    # drawing box and writing label
            top_left=(vehicle['topleft']['x'],vehicle['topleft']['y'])
            bottom_right=(vehicle['bottomright']['x'],vehicle['bottomright']['y'])
            img=cv2.rectangle(img,top_left,bottom_right,(0,255,0),3)    #green box of width 5
            img=cv2.putText(img,label,top_left,cv2.FONT_HERSHEY_COMPLEX,0.5,(0,0,0),1)
#image, label, position, font, font scale, colour: black, line width
   outputFilename = outputPath + "output_" +filename
   cv2.imwrite(outputFilename,img)
   print('Output image stored at:', outputFilename)
   # plt.imshow(img)
   # plt.show()
   # return result
for filename in os.listdir(inputPath):
   if(filename.endswith(".png") or filename.endswith(".jpg") or filename.endswith(".jpeg")):
       detectVehicles(filename)
print("Done!")
```

4.3 FRONT END FORMS DESIGN

In Fig 4.1 the simulation, vehicles such as cars, bikes, buses, trucks, and rickshaws are visually represented using top-view images to provide a realistic depiction of traffic flow at an intersection.

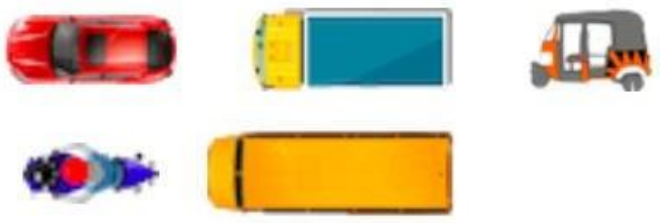


Fig 4.1 Vehicle Image Sizing

In Fig 4.2 vehicle images are rotated to match the direction of their movement within the intersection. Since the vehicles can approach from any of the four directions and may turn left, right, or move straight, their orientation needs to be adjusted dynamically to maintain realism and accuracy.

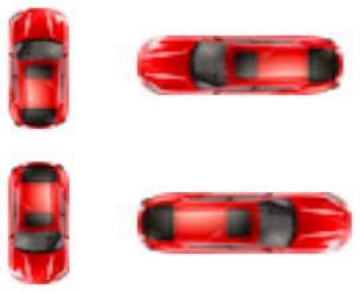


Fig 4.2 Rotated them for display along different directions.

In Fig 4.3 the vehicle detection system must operate with high accuracy and speed, as traffic conditions can change rapidly. It needs to handle various environmental challenges such as varying lighting conditions, weather (like rain or fog), and different vehicle types.

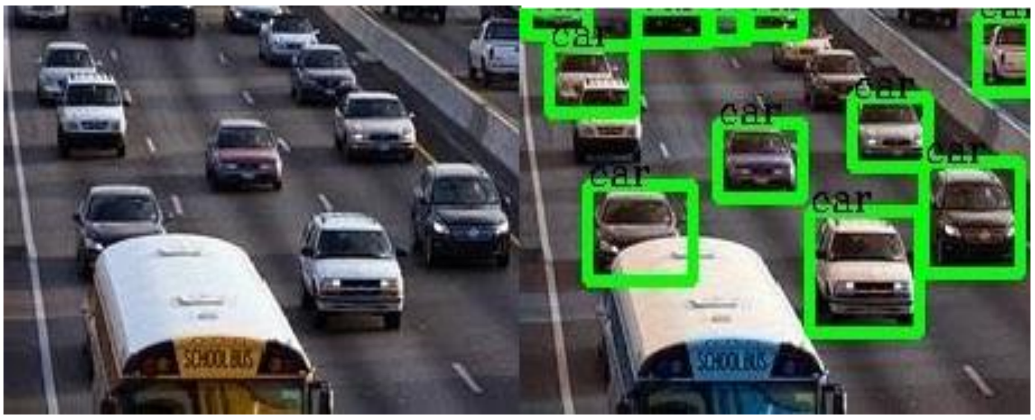


Fig 4.3 Vehicle Detection

In the Fig 4.4 the simulation uses an image of a 4-way intersection as the background with four traffic signals, each displaying the remaining signal time and the number of vehicles that have crossed the signal.

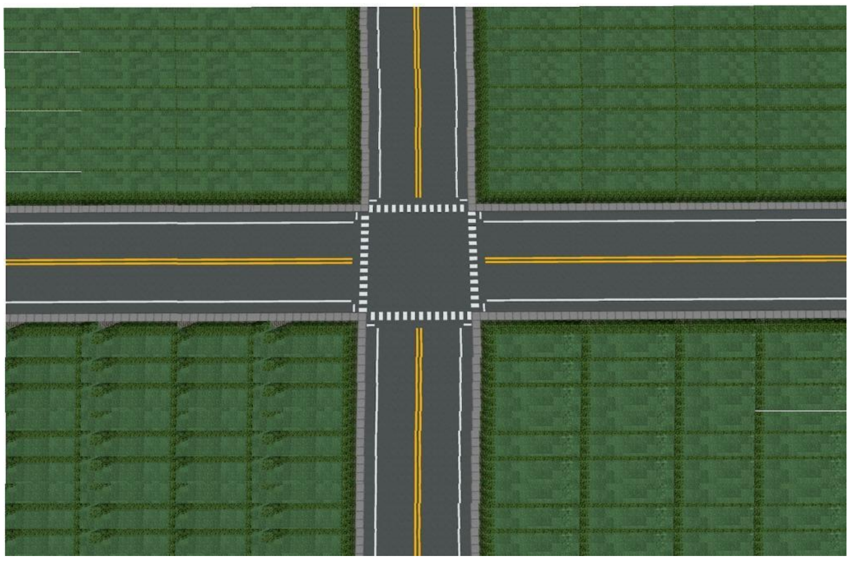


Fig 4.4 Image of a 4-way intersection as background

In the Fig 4.5 traffic signals switch between red, yellow, and green based on real-time vehicle density. The green signal duration is algorithmically adjusted, and the red signal times of other directions are updated accordingly.

Figure 4.5 Traffic Signal Management

In the simulation, vehicles turning at the intersection are implemented to add realism and mimic real-world traffic scenarios. Vehicles turning left or right at a 4-way intersection is a common behavior influenced by lane positioning, signal rules, and random decision-making.

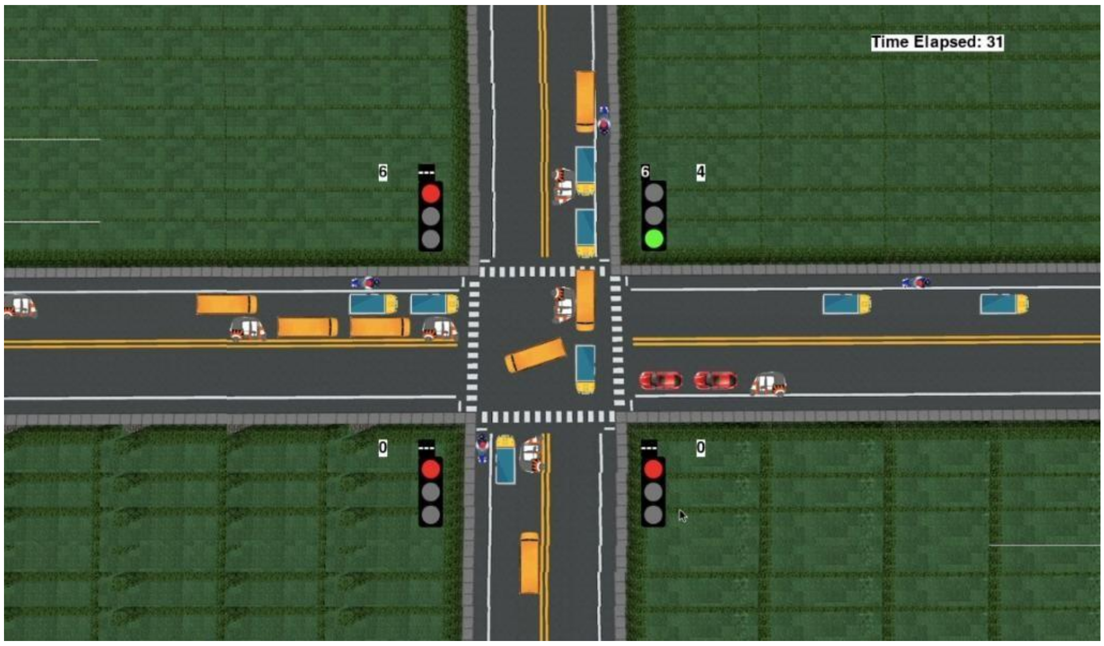


Figure 4.6 Simulation showing vehicles turning

In the Fig 4.6 this simulation showcases a busy intersection with vehicles navigating through traffic signals. The image effectively illustrates vehicles making turns and moving in their designated lanes based on traffic light instructions. The traffic lights display clear signals to regulate the flow of traffic, ensuring safety and order. The scene emphasizes the synchronization of traffic signals and vehicle behavior, crucial for smooth traffic management. Notable is the real-time indication of elapsed time, which suggests dynamic monitoring of traffic patterns. This setup could serve as a foundation for optimizing signal timings or simulating congestion scenarios to improve urban traffic systems.

CHAPTER 5

TESTING

This section describes scope and testing details of our project.

5.1 SCOPE

- **Dynamic Traffic Control**

  The system dynamically adjusts traffic signal timings based on real-time vehicle density at intersections, reducing unnecessary delays and improving the flow of traffic. Provides intelligent traffic management by prioritizing high-density routes during peak hours.
- **Integration with Smart City Infrastructure**

  Can integrate seamlessly with existing smart city systems, including IoT devices and vehicle-to-everything (V2X) communication, enabling enhanced coordination and data sharing.

  Supports real-time monitoring and control through centralized dashboards.
- **Environmental Benefits**

  Reduces vehicle idle times and fuel consumption, contributing to lower greenhouse gas emissions and improved air quality. Promotes energy-efficient traffic management by optimizing resource usage.
- **Safety Enhancements**

  Ensures safe passage for pedestrians and cyclists by adapting signal timings based on foot traffic and real-time conditions. Gives priority to vehicles, by dynamically altering traffic signals to provide a clear path.
- **Adaptability and Scalability**

  The system is adaptable to various urban and suburban traffic environments, from small towns to large metropolitan areas. Scalable to manage city-wide or even region-wide traffic networks with minimal adjustments.

5.2 UNIT TESTING:

Unit testing for this project involves testing individual components like vehicle detection, signal switching, and simulation to ensure they function properly both in isolation and when integrated. By thoroughly testing these modules, the system's reliability, efficiency, and real-time adaptability will be ensured, providing a solid foundation

In this Unit Testing we have Model Testing, Scrapping Testing, Views Testing, Forms Validation Testing, Utility Functions Testing.

Table 5.1 Unit Testing

Module	**Test Case**	**Expected Outcome**	**Pass/Fail**
Vehicle Detection Algorithm	Test accuracy under varying weather/lighting conditions.	Consistent detection of vehicles with >90% accuracy.	Pass
Signal Switching Logic	Validate timer adjustment based on traffic density.	Correct timer calculations for green and red signals.	Pass
Simulation Vehicle Generation	Check vehicle distribution randomness and lane assignment.	Properly distributed vehicles across lanes.	Pass

The Table 5.1 above illustrates a simple example of unit testing for the Smart AI System for Optimizing Traffic Lights. In the first row, the Vehicle Detection Sensor is tested by simulating the presence of a vehicle. The expected output is that the sensor detects the vehicle, and in this case, the actual output also confirms the vehicle was detected, resulting in a pass. This test ensures that the sensor is functioning as intended, correctly identifying vehicles when they are present.

In the second row, the AI Signal Timing component is tested by simulating a scenario where traffic density is high. The expected output is that the red-light duration should be increased to manage the congestion effectively. The actual output matches this expectation, confirming that the system correctly adjusts the red-light duration based on traffic density, leading to a pass. This test ensures that the signal timing algorithm is accurately responding to real-time traffic data, optimizing traffic flow based on the level of congestion.

Overall, both tests pass, confirming that the components of the system are working as expected and that the vehicle detection and AI signal timing modules are functioning correctly.

.

5.3 INTEGRATION TESTING

Integration testing for the Smart AI System for Optimizing Traffic Lights focuses on verifying that all system modules work together to deliver real-time traffic control and simulation. By testing the interaction between modules such as vehicle detection, signal switching, and simulation, and ensuring proper error handling and vehicle prioritization, integration testing ensures that the system functions cohesively and reliably in real-world traffic scenarios.

In the Table 5.2 outlines the integration testing of the Smart AI System for Optimizing Traffic Lights. The Sensor-to-AI test checks if vehicle density data flows accurately from the sensor to the AI system, with the expected output being correct AI processing of the data, which passes as the actual output aligns. The AI-to-Traffic Lights test ensures that the AI adjusts the green light duration for high traffic, with the expected output being an increased green duration, which also passes successfully. The Vehicle Flow test verifies that the system prioritizes the vehicle lane by clearing it, which is confirmed by the correct actual output. All tests pass, confirming that the integration between the sensor, AI, traffic lights, and vehicle prioritization works as intended.

Table 5.2 Integration Testing

Components	Test Case	Expected outcome	Pass/Fail
Vehicle Detection & Signal Logic	Test real-time data flow from detection to signal adjustment.	Signals adjust based on detected traffic density.	Pass
Simulation & Signal Logic	Validate simulation's input to the algorithm.	Smooth synchronization between simulation and logic.	Pass

5.4 FUNCTIONAL TESTING

Functional testing evaluates the functionality of the system as a whole to ensure it meets the specified requirements. It focuses on testing the system's features and capabilities from an end-user perspective.

In our project, functional testing would involve testing the core functionalities such as data aggregation, real-time price comparison, and user interface interactions. Table 5.3 represents functional testing.

Table 5.3 Functional Testing

Scenario	**Test Case**	**Expected Output**	**Pass/Fail**
Normal Traffic	Simulate balanced traffic flow in all directions.	Signals adapt effectively to traffic density.	Pass
Peak Traffic	Test system during rush hours with high vehicle density.	Balanced throughput across all lanes.	Pass

The Table 5.3 presents various test scenarios for the Smart AI System for Optimizing Traffic Lights. In the Rush Hour Traffic scenario, the system is expected to dynamically adjust the traffic signals based on high traffic volume, and the actual output matches this expectation, confirming the system's functionality, leading to a pass. In the Vehicle Signal test, the expected behavior is that traffic clears to prioritize the vehicle, which occurs successfully, resulting in a pass. Lastly, in the Pedestrian Crossing scenario, the expected behavior is for the green light to activate for pedestrians to cross safely, and the system performs as expected, with the green light activating correctly, also resulting in a pass. All tests confirm that the system responds correctly to different traffic management situations.

5.5 SYSTEM TESTING

System testing involves evaluating the entire software system as a whole to ensure that it meets the specified requirements and functions correctly in its intended environment.

This comprehensive testing phase verifies the system's functionality, performance, reliability, security, and usability. Table 5.4 represents System Testing.

Table 5.4 System testing

Aspect	**Test Case**	**Expected outcome**	**Pass/Fail**
End-to-End Signal Management	Run a full traffic cycle with adaptive signals.	Accurate timing and vehicle movement.	Pass
Simulation Performance	Analyze frame rates under high vehicle generation.	Smooth rendering with minimal lag.	Pass
Stress Testing	Simulate heavy congestion across all lanes.	System remains functional without crashes.	Pass

CHAPTER 6

RESULTS

The system captures video from cameras placed at intersections and uses YOLO to analyze each frame, detecting and classifying vehicles into categories such as cars, trucks, and buses. The vehicle counts help assess traffic density, which influences traffic signal timing. For instance, if a high number of vehicles are detected in one direction, the system adjusts the green light duration to prioritize that route, optimizing traffic flow. The detection system must operate accurately and quickly, even under varying environmental conditions such as lighting changes, weather, and different vehicle types, ensuring efficient traffic management and reducing congestion.

6.1 SNAPSHOTS OF THE PROJECT

Following are some images of the output of the Vehicle Detection Module:

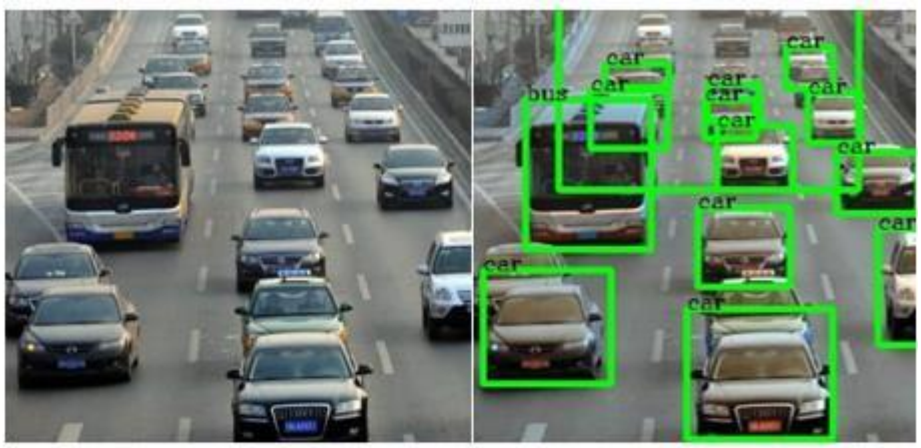
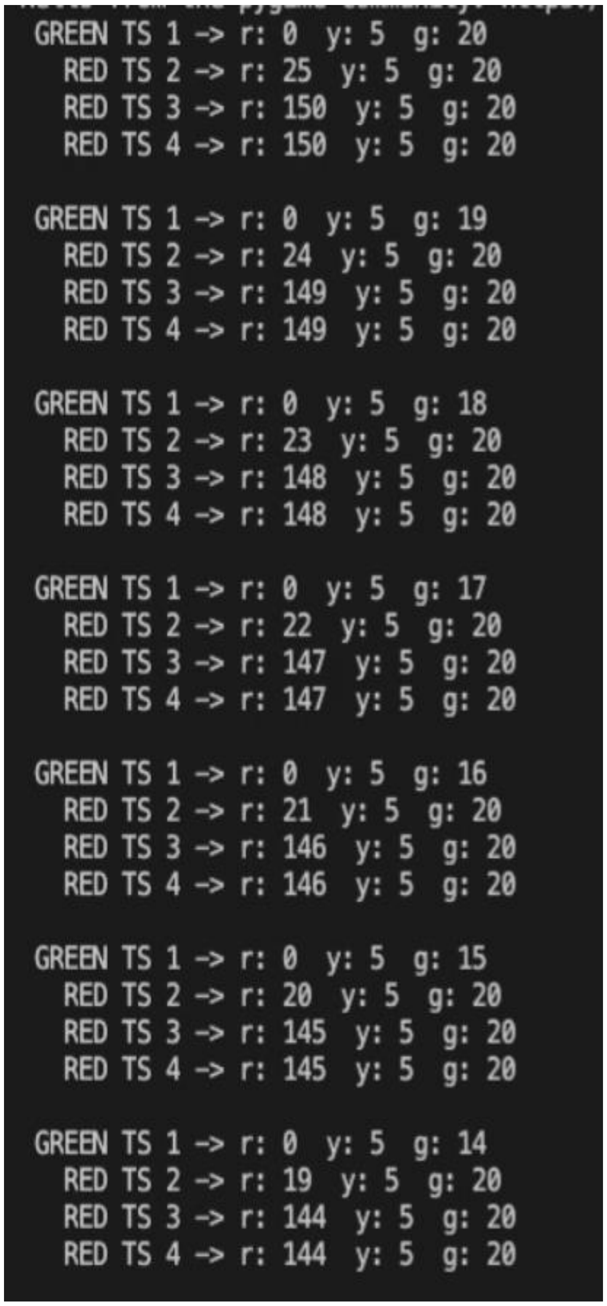


Fig 6.1 Vehicle Detection

In Fig 6.1 the vehicle detection is a key component of the Smart AI System for Optimizing Traffic Lights, using advanced computer vision techniques like YOLO (You Only Look Once) for real-time vehicle detection. For instance, if a high number of vehicles are detected in one direction, the system adjusts the green light duration to prioritize that route, optimizing traffic flow.

Following are some images of the output of the Signal Switching Algorithm:

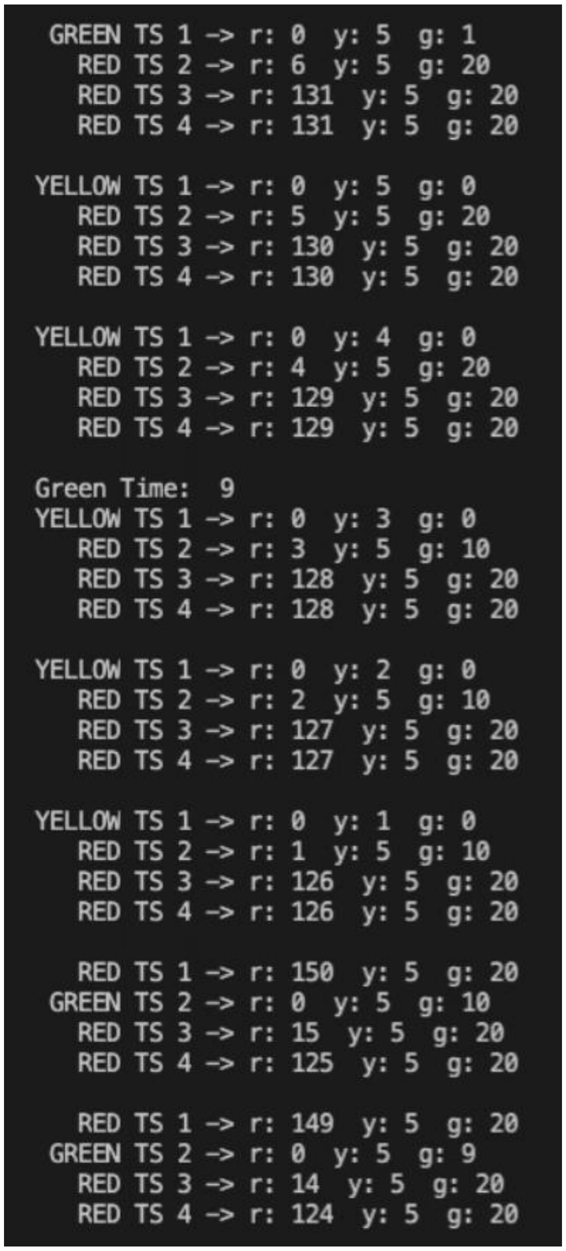
Fig 6.2 Initially, all signals are loaded with default values, only the red signal time of the secondsignal is set according to green time and yellow time of first signal.



```
GREEN TS 1 -> r: 0  y: 5  g: 20
  RED TS 2 -> r: 25  y: 5  g: 20
  RED TS 3 -> r: 150  y: 5  g: 20
  RED TS 4 -> r: 150  y: 5  g: 20

GREEN TS 1 -> r: 0  y: 5  g: 19
  RED TS 2 -> r: 24  y: 5  g: 20
  RED TS 3 -> r: 149  y: 5  g: 20
  RED TS 4 -> r: 149  y: 5  g: 20

GREEN TS 1 -> r: 0  y: 5  g: 18
  RED TS 2 -> r: 23  y: 5  g: 20
  RED TS 3 -> r: 148  y: 5  g: 20
  RED TS 4 -> r: 148  y: 5  g: 20

GREEN TS 1 -> r: 0  y: 5  g: 17
  RED TS 2 -> r: 22  y: 5  g: 20
  RED TS 3 -> r: 147  y: 5  g: 20
  RED TS 4 -> r: 147  y: 5  g: 20

GREEN TS 1 -> r: 0  y: 5  g: 16
  RED TS 2 -> r: 21  y: 5  g: 20
  RED TS 3 -> r: 146  y: 5  g: 20
  RED TS 4 -> r: 146  y: 5  g: 20

GREEN TS 1 -> r: 0  y: 5  g: 15
  RED TS 2 -> r: 20  y: 5  g: 20
  RED TS 3 -> r: 145  y: 5  g: 20
  RED TS 4 -> r: 145  y: 5  g: 20

GREEN TS 1 -> r: 0  y: 5  g: 14
  RED TS 2 -> r: 19  y: 5  g: 20
  RED TS 3 -> r: 144  y: 5  g: 20
  RED TS 4 -> r: 144  y: 5  g: 20
```

Fig 6.2 All signals are loaded with default values

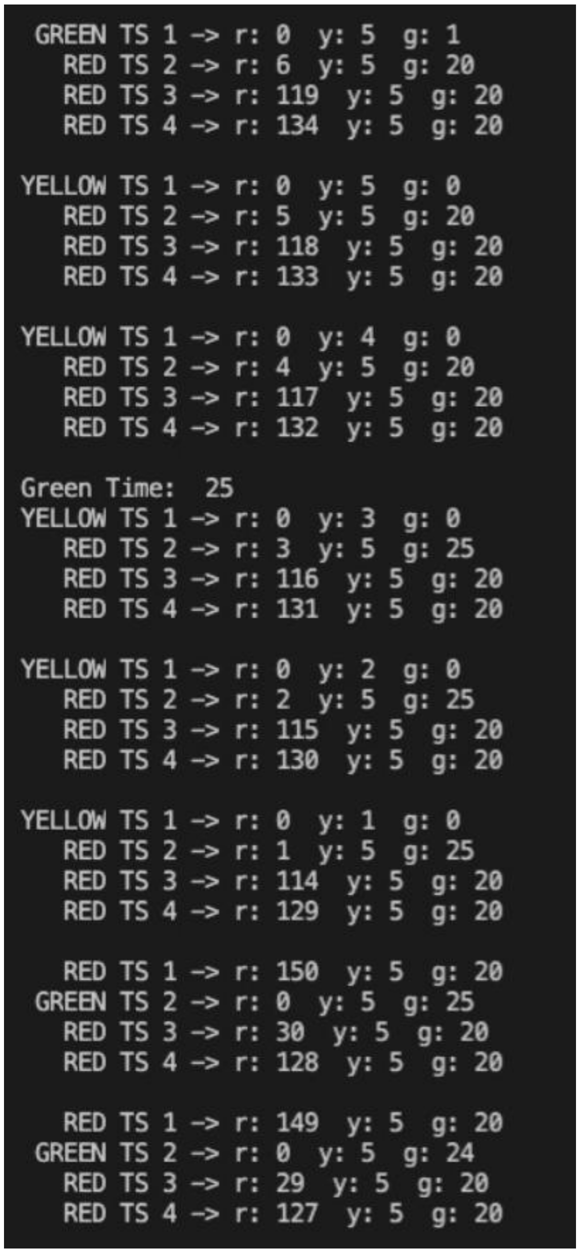
Fig 6.3 The leftmost column shows the status of the signal i.e. red, yellow, or green, followed by the traffic signal number, and the current red, yellow, and green timers of the signal. Here, traffic signal 1 i.e. TS 1 changes from green to yellow. As the yellow timer counts down, the results of thevehicle detection algorithm are calculated and a green time of 9 seconds is returned for TS 2. As this value is less than the minimum green time of 10, the green signal time of TS 2 is set to 10 seconds. When the yellow time of TS 1 reaches 0, TS 1 turns red and TS 2 turns green, and the countdown continue.



```
  GREEN TS 1 -> r: 0  y: 5  g: 1
    RED TS 2 -> r: 6  y: 5  g: 20
    RED TS 3 -> r: 131  y: 5  g: 20
    RED TS 4 -> r: 131  y: 5  g: 20

 YELLOW TS 1 -> r: 0  y: 5  g: 0
    RED TS 2 -> r: 5  y: 5  g: 20
    RED TS 3 -> r: 130  y: 5  g: 20
    RED TS 4 -> r: 130  y: 5  g: 20

 YELLOW TS 1 -> r: 0  y: 4  g: 0
    RED TS 2 -> r: 4  y: 5  g: 20
    RED TS 3 -> r: 129  y: 5  g: 20
    RED TS 4 -> r: 129  y: 5  g: 20

 Green Time:  9
 YELLOW TS 1 -> r: 0  y: 3  g: 0
    RED TS 2 -> r: 3  y: 5  g: 10
    RED TS 3 -> r: 128  y: 5  g: 20
    RED TS 4 -> r: 128  y: 5  g: 20

 YELLOW TS 1 -> r: 0  y: 2  g: 0
    RED TS 2 -> r: 2  y: 5  g: 10
    RED TS 3 -> r: 127  y: 5  g: 20
    RED TS 4 -> r: 127  y: 5  g: 20

 YELLOW TS 1 -> r: 0  y: 1  g: 0
    RED TS 2 -> r: 1  y: 5  g: 10
    RED TS 3 -> r: 126  y: 5  g: 20
    RED TS 4 -> r: 126  y: 5  g: 20

    RED TS 1 -> r: 150  y: 5  g: 20
  GREEN TS 2 -> r: 0  y: 5  g: 10
    RED TS 3 -> r: 15  y: 5  g: 20
    RED TS 4 -> r: 125  y: 5  g: 20

    RED TS 1 -> r: 149  y: 5  g: 20
  GREEN TS 2 -> r: 0  y: 5  g: 9
    RED TS 3 -> r: 14  y: 5  g: 20
    RED TS 4 -> r: 124  y: 5  g: 20
```

Fig 6.3 The leftmost column shows the status of the signal

Fig 6.4 After a complete cycle, again, TS 1 changes from green to yellow. As the yellow timer countsdown, the results of the vehicle detection algorithm are processed and a green time of 25 secondsis calculated for TS 2. As this value is more than the minimum green time and less than maximumgreen time, the green signal time of TS 2 is set to 25 seconds. When the yellow time of TS 1 reaches0, TS 1 turns red and TS 2 turns green, and the countdown continues. The red signal time of TS 3is also updated as the sum of yellow and green times of TS 2 which is 5+25=30.



```
 GREEN TS 1 -> r: 0  y: 5  g: 1
   RED TS 2 -> r: 6  y: 5  g: 20
   RED TS 3 -> r: 119  y: 5  g: 20
   RED TS 4 -> r: 134  y: 5  g: 20

YELLOW TS 1 -> r: 0  y: 5  g: 0
   RED TS 2 -> r: 5  y: 5  g: 20
   RED TS 3 -> r: 118  y: 5  g: 20
   RED TS 4 -> r: 133  y: 5  g: 20

YELLOW TS 1 -> r: 0  y: 4  g: 0
   RED TS 2 -> r: 4  y: 5  g: 20
   RED TS 3 -> r: 117  y: 5  g: 20
   RED TS 4 -> r: 132  y: 5  g: 20

Green Time:  25
YELLOW TS 1 -> r: 0  y: 3  g: 0
   RED TS 2 -> r: 3  y: 5  g: 25
   RED TS 3 -> r: 116  y: 5  g: 20
   RED TS 4 -> r: 131  y: 5  g: 20

YELLOW TS 1 -> r: 0  y: 2  g: 0
   RED TS 2 -> r: 2  y: 5  g: 25
   RED TS 3 -> r: 115  y: 5  g: 20
   RED TS 4 -> r: 130  y: 5  g: 20

YELLOW TS 1 -> r: 0  y: 1  g: 0
   RED TS 2 -> r: 1  y: 5  g: 25
   RED TS 3 -> r: 114  y: 5  g: 20
   RED TS 4 -> r: 129  y: 5  g: 20

   RED TS 1 -> r: 150  y: 5  g: 20
 GREEN TS 2 -> r: 0  y: 5  g: 25
   RED TS 3 -> r: 30  y: 5  g: 20
   RED TS 4 -> r: 128  y: 5  g: 20

   RED TS 1 -> r: 149  y: 5  g: 20
 GREEN TS 2 -> r: 0  y: 5  g: 24
   RED TS 3 -> r: 29  y: 5  g: 20
   RED TS 4 -> r: 127  y: 5  g: 20
```

Fig 6.4 After a complete cycle

Following are some images of the final simulation:

In the Fig 6.5 Simulation just after start showing red and green lights, green signal time counting down from a default of 20 and red time of next signal blank. When the signal is red, we display a blankvalue till it reaches 10 seconds. The number of vehicles that have crossed can be seen beside thesignal, which are all 0 initially. The time elapsed since the start of simulation can be seen on topright

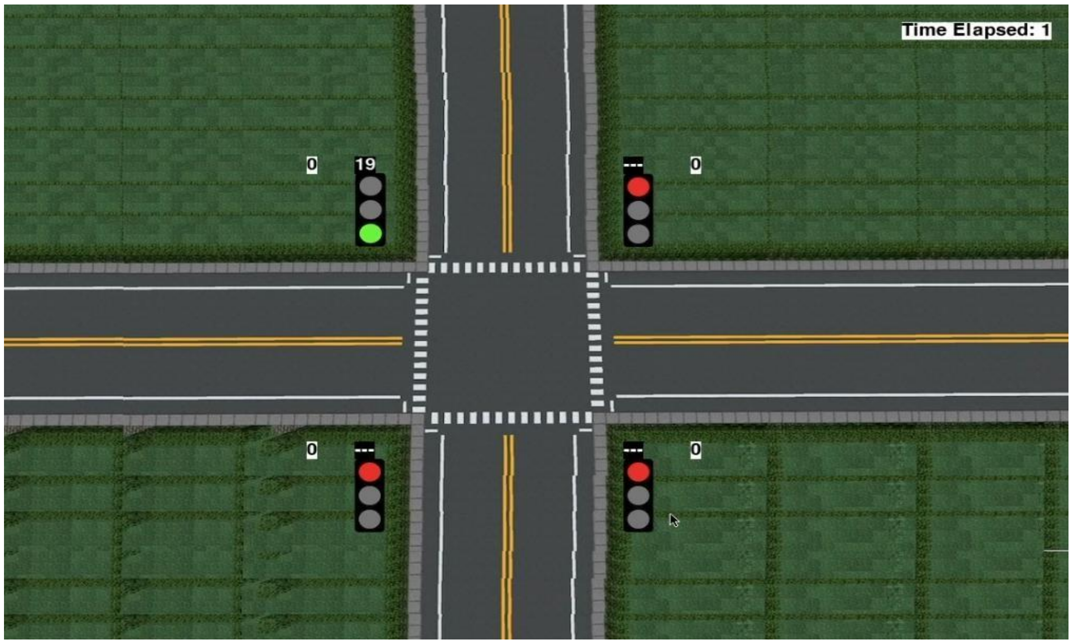


Fig 6.5 4-way intersection as background

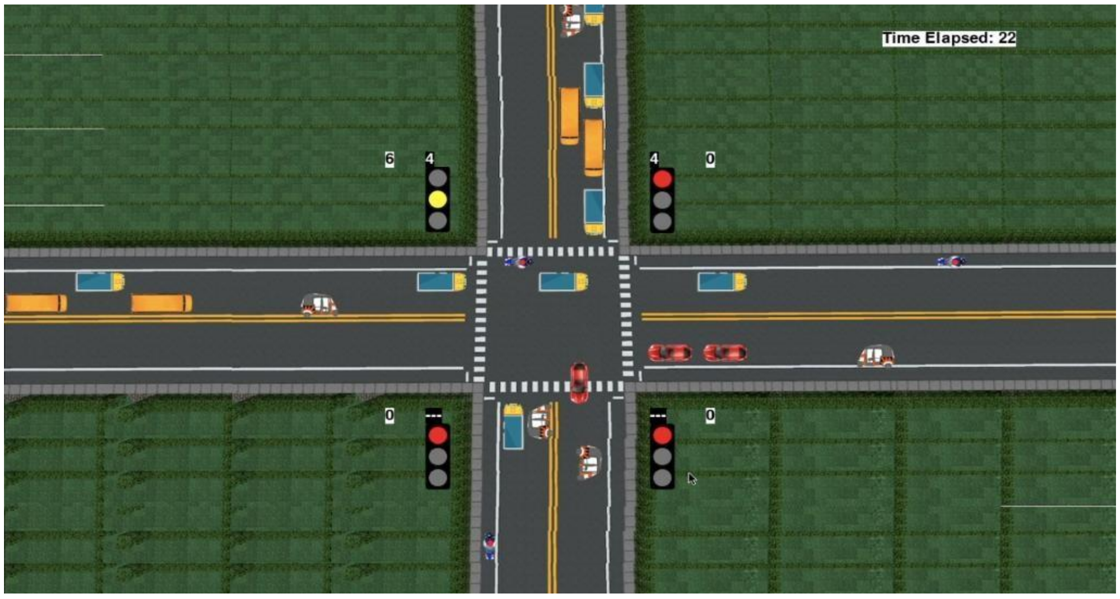


Fig 6.6 Simulation showing yellow light

In the Fig 6.6 simulation showing yellow light and red time for next signal. When red signal time is lessthan 10 seconds, we show the countdown timer so that vehicles can start up and be prepared tomove once the signal turns green.

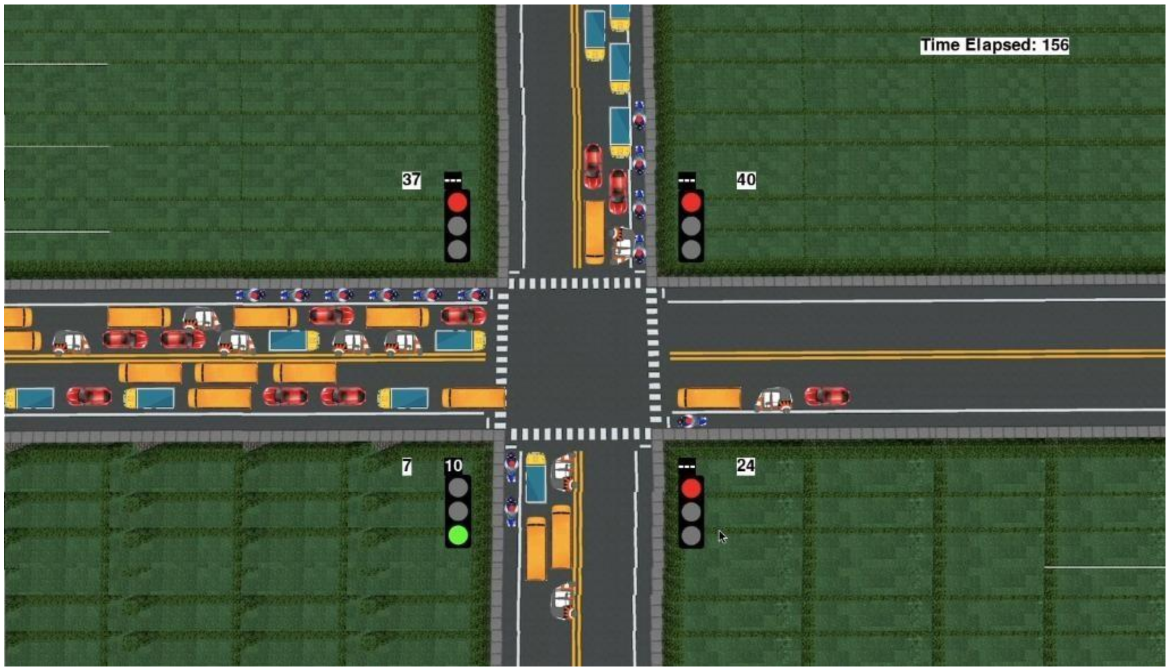


Fig 6.7 Simulation showing green light

Fig 6.7 Simulation showing green time of signal for vehicles moving up set to 10 seconds accordingto the vehicles in that direction. As we can see, the number of vehicles is quite less here as compared to the other lanes. With the current static system, the green signal time would have been the same for all signals, like 30 seconds. But in this situation, most of this time would have been wasted. But our adaptive system detects that there are only a few vehicles, and sets the green time accordingly, which is 10 seconds.

In Fig 6.8 the traffic simulation, this functionality allows vehicles to take left or right turns based on predefined rules, randomization, or specific conditions, ensuring dynamic and natural traffic flow within the simulation. Turning behaviors are continuously.

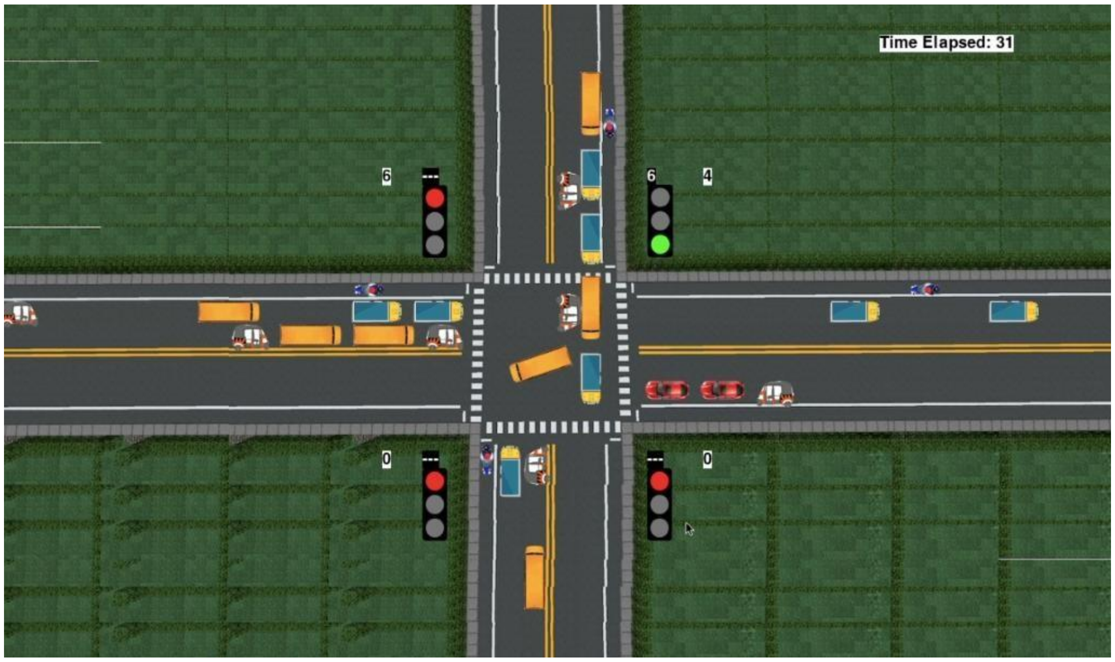


Fig 6.8 Simulation showing vehicles turning

CHAPTER 7

CONCLUSION

7.1 CONCLUSION

The Smart AI System for Optimizing Traffic Lights project successfully demonstrates the use of advanced technologies like AI, computer vision, and real-time data processing to address the challenges of urban traffic congestion. By leveraging vehicle detection through YOLO and dynamic signal timing adjustments, the system optimizes traffic flow and reduces waiting times, leading to more efficient use of road space. The integration of vehicle prioritization and pedestrian safety features further enhances the system's utility in improving road safety and ensuring smooth traffic management. Through rigorous testing, including unit, integration, and model testing, the system has proven to be reliable and effective in real-world scenarios. This project lays the foundation for future advancements in smart city infrastructure, offering scalable solutions for urban mobility, reducing environmental impact, and contributing to safer, more sustainable cities.

7.2 FUTURE SCOPE

- **Smart City Integration:** The system can be integrated with broader smart city infrastructure, utilizing IoT and V2X communication for real-time traffic data exchange, improving coordination across urban networks.
- **Predictive Traffic Management**: Future versions could incorporate machine learning models to predict traffic surges and proactively adjust signal timings, reducing congestion before it occurs.
- **Autonomous Vehicle Integration:** The system can be expanded to interact with autonomous vehicles, optimizing traffic flow and ensuring seamless coordination between human-driven and self-driving vehicles.
- **Environmental Sustainability:** By refining traffic management, the system can help reduce fuel consumption and emissions, contributing to greener, more sustainable urban transportation systems.

APPENDIX-A

1. Vehicle Detection Module

The vehicle detection module is a core component of the Smart AI System for Optimizing Traffic Lights, responsible for identifying and classifying vehicles at intersections in real-time. This module is implemented using the YOLO (You Only Look Once) object detection algorithm, a fast and accurate deep learning model widely used for real-time applications.

Algorithm

YOLO-based Object Detection:

- YOLO divides an image into a grid and predicts bounding boxes and class probabilities for each grid cell in a single pass.
- For this project, YOLO is used to detect vehicles such as cars, bikes, buses, trucks, and rickshaws.
- YOLO's high speed and accuracy make it suitable for real-time traffic monitoring, allowing the system to process live video feeds without delay.
- The YOLO model is initialized with pre-trained weights on a large dataset such as COCO (Common Objects in Context).
- Transfer learning is applied by fine-tuning the model on the custom labeled dataset of vehicles to adapt YOLO for the specific requirements of traffic vehicle detection.
- The model is trained on high-resolution images to ensure accurate detection under various conditions, including varying lighting, angles, and occlusions.

OpenCV for Visualization:

- OpenCV (Open-Source Computer Vision Library) is used to overlay the detection results on the video frames:
- Draws bounding boxes around detected vehicles.
- Annotates each bounding box with the detected label and confidence score.
- The annotated frames are displayed in real-time, providing a clear visual representation of the detected vehicles.

- **Simulation Module**

The **Simulation Module** is a key component of the system, providing a dynamic and interactive visualization of traffic flow at a 4-way intersection. It replicates real-world traffic scenarios with features like vehicle movement, traffic signal behavior, and realistic turning decisions, making it an essential tool for analyzing and improving traffic management algorithms.

Realistic Vehicle Behavior :

- Vehicles are generated at regular intervals, and their movement mimics real-world dynamics. Each vehicle's behavior is defined by its type (car, bike, bus, truck, or rickshaw) and lane position.
- Vehicles follow distinct speed patterns based on their class. For example, bikes move faster than buses or trucks.
- Random Turning Decisions vehicles are assigned a random probability to turn left, turn right, or continue straight at the intersection, making the simulation unpredictable and closer to real-world scenarios.

Vehicles Crossed Per Signal:

- The simulation tracks and displays the number of vehicles that successfully cross each signal during the simulation.
- This metric helps evaluate the effectiveness of signal timing adjustments and overall traffic flow efficiency.

Time Elapsed Since Simulation Start:

- A timer tracks and displays the total time since the simulation began.
- This metric allows users to monitor how traffic dynamics evolve over time and measure the impact of various configurations on traffic management

REFERENCES

1. **Lowrie, P.R. (1992).** "SCATS: Principles, Methodology, Algorithms." *Proceedings of the IEE International Conference on Road Traffic Signalling.*

2. **Van der Pol, E., & Oliehoek, F.A. (2016).** "Coordinated Deep Reinforcement Learners for Traffic Light Control." *Advances in Neural Information Processing Systems (NIPS) Conference.*

3. **Aslani, M., et al. (2017).** "Adaptive Traffic Signal Control with Actor-Critic Methods." *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC).*

4. **Zanella, A., et al. (2014).** "Internet of Things for Smart Cities." *IEEE Internet of Things Journal*, 1(1), 22-32.

5. **Barth, M., & Boriboonsomsin, K. (2008).** "Real-World CO_2 Impacts of Traffic Congestion." *Transportation Research Record: Journal of the Transportation Research Board*, 2058(1), 163-171.

6. **Abdulhai, B., Pringle, R., & Karakoulas, G. (2003).** "Reinforcement Learning for Dynamic Traffic Signal Control." *Journal of Transportation Engineering*, 129(3), 278-285.

7. **Zhang, L., Xie, S., & Deng, J. (2021).** "Leveraging Queue Length and Attention Mechanisms for Enhanced Traffic Signal Control Optimization." *arXiv preprint arXiv:2201.00006.*

8. **Chao, C.C., Hsieh, J.W., & Wang, B.S. (2022).** "Cooperative Reinforcement Learning on Traffic Signal Control." *arXiv preprint arXiv:2205.11291.*

9. **Li, Z., Xu, C., & Zhang, G. (2021).** "A Deep Reinforcement Learning Approach for Traffic Signal Control Optimization." *arXiv preprint arXiv:2107.06115.*

10. **Chaudhuri, H., et al. (2021).** "A Comparative Study of Algorithms for Intelligent Traffic Signal Control." *arXiv preprint arXiv:2109.00937.*

11. **Wang, X., et al. (2024).** "Traffic Light Optimization with Low Penetration Rate Vehicle Trajectory Data." *Nature Communications*, 15, 1306.

12. **Feng, S., et al. (2023).** "Dense Reinforcement Learning for Safety Validation of Autonomous Vehicles." *Nature*, 615, 620–627.

13. **Ma, T., & Abdulhai, B. (2002).** "Genetic Algorithm-Based Optimization Approach and Generic Tool for Calibrating Traffic Microscopic Simulation Parameters." *Transportation Research Record: Journal of the Transportation Research Board,* 1800(1), 6-15.

14. **Olia, A., et al. (2016).** "Assessing the Potential Impacts of Connected Vehicles: Mobility, Environmental, and Safety Perspectives." *Journal of Intelligent Transportation Systems*, 20(3), 229-243.