AI Based Traffic Manegement System

Nishant Tyagi
Dept of ECE,
RajKumar Goel Institute of Technology,
Ghaziabad,India
tyagi.nis2021@gmail.com

Tithi Dey
Dept of ECE,
RajKumar Goel Institue of Technology ,
Ghaziabad,India
tithi.dey125@gmail.com

Vaibhav Sharma
Dept of ECE,
RajKumar Goel Institue of technology,
Ghaziabad, India
vaibhavfec@rkgit.edu.in

Abstract— Urban traffic congestion significantly hampers the timely response of emergency vehicles, such as ambulances, often resulting in critical delays that can impact patient outcomes and save-or-loss-of-life situations. Traditional traffic signal systems operate on fixed timing schedules and lack the ability to detect or prioritize emergency vehicles in real time, causing ambulances to be stuck at red lights during urgent journeys. To address this pressing issue, this paper proposes a smart, automated ambulance detection system that leverages the YOLOv8 object detection algorithm combined with an Arduino Uno microcontroller to dynamically control traffic signals at intersections.

The system employs a webcam to continuously monitor traffic, analyzing video frames in real time to identify ambulances using the YOLOv8 model implemented in the Spyder IDE environment. Once an ambulance is detected, the system sends a serial communication signal to the Arduino Uno, which overrides the standard traffic light sequence and turns the signal green for the ambulance's lane for a fixed duration of 30 seconds. This allows the ambulance to pass through the intersection swiftly and safely, minimizing delays caused by traffic congestion. After the set interval, the traffic lights resume their normal red-green cycle, ensuring minimal disruption to overall traffic flow.

By combining real-time computer vision with embedded hardware control, the proposed system offers an efficient and cost-effective solution to improve emergency response times and enhance urban traffic management as part of the broader smart city initiatives.

Keywords—YOLOv8

INTRODUCTION

With rapid urbanization and rising vehicle numbers, traffic congestion poses serious challenges, especially for emergency vehicles like ambulances. Delays caused by traffic jams can critically impact emergency response times and endanger lives. Traditional traffic systems with fixed signal timings cannot adapt in real time to emergency scenarios, highlighting the urgent need for intelligent and adaptive traffic management solutions.

This project proposes an automated ambulance detection and traffic signal control system using the YOLOv8 object detection algorithm integrated with an Arduino Uno

microcontroller. A webcam continuously monitors traffic.and YOLOv8 detects ambulances in real time through video analysis performed in the Spyder IDE. Upon detection, the system sends a signal via serial communication to the Arduino, which temporarily overrides the traffic lights to turn green for the ambulance lane for 30 seconds, allowing swift and uninterrupted passage. After the allotted time, normal signal cycles resume to maintain overall traffic flow.

By combining computer vision techniques with embedded hardware control, this system aims to improve emergency vehicle response efficiency while promoting the development of smarter, more responsive urban traffic infrastructure.

A. Problem Statement

Conventional traffic light systems rely primarily on fixed timers or pre-programmed cycles that control the duration of red, green, and yellow signals at intersections. While this approach can manage regular traffic flow under normal conditions, it lacks the ability to respond dynamically to real-time traffic scenarios, especially in critical situations involving emergency vehicles such as ambulances. During emergencies, every second counts, and delays caused by waiting at red lights can lead to life-threatening situations. Unfortunately, current static traffic control systems are not equipped to detect or prioritize ambulances on the road, resulting in significant delays in emergency response times.

This limitation represents a major challenge in urban traffic management, where increasing vehicle density exacerbates congestion and makes it more difficult for ambulances to navigate through traffic efficiently. The inability to adapt traffic signals in real time can cause ambulances to get stuck in jams, increasing response times and jeopardizing patient outcomes. Therefore, there is a pressing need for intelligent traffic control systems capable of recognizing emergency vehicles and dynamically adjusting signal states to prioritize their passage.

Our proposed system addresses this critical gap by integrating advanced computer vision techniques with microcontroller-based traffic signal control. Using an object detection algorithm, the system can continuously monitor traffic and identify ambulances in real time. Once an ambulance is detected, the system sends a command to the microcontroller to override the normal traffic light cycle, switching the signal to green for the ambulance's lane. This dynamic control ensures that ambulances receive an uninterrupted path through intersections, significantly reducing response delays. By combining real-time detection with automated signal management, our approach enhances emergency response

capabilities and offers a smart, responsive solution to modern urban traffic challenges.

II. LITERATURE SURVEY

1) *Smart Traffic Management Using AI and Deep Learning*

Recent research highlights the integration of artificial intelligence (AI) and deep learning to improve traffic management systems. Traditional rule-based systems are being replaced by models that learn patterns from traffic data. Deep learning architectures like Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks are used to forecast traffic density and vehicle movement patterns. These models analyze large datasets in real time, enabling adaptive signal control that adjusts according to congestion levels. Such intelligent systems enhance the efficiency of urban transport networks by reducing delays and improving emergency response times. Studies have also shown that AI-driven systems can outperform human-managed or fixed-timing traffic signals, especially in dynamic traffic scenarios. The use of AI in traffic control not only improves flow but also supports predictive maintenance of infrastructure and incident detection, making the systems more proactive rather than reactive.

2) *Reinforcement Learning for Adaptive Signal Control*

Reinforcement learning (RL) has emerged as a promising approach for adaptive traffic signal control. Unlike traditional methods, RL algorithms learn optimal actions through interactions with the environment, receiving rewards for favorable outcomes. In the context of traffic lights, RL agents are trained to minimize queue lengths and waiting times. Algorithms like Q-learning and Deep Q-Networks (DQNs) have been employed in simulations to manage signal phases dynamically based on vehicle count and flow patterns. These studies demonstrate that RL-controlled intersections adapt quickly to fluctuating traffic and can prioritize emergency vehicles when integrated with detection modules. The effectiveness of RL in managing multi-intersection networks has also been explored, showing potential for city-wide scalability. The key advantage lies in its ability to improve over time through continuous learning, making RL a core component in the evolution of smart traffic management systems.

3) *Integration of Edge Computing in Traffic Systems*

Edge computing is transforming intelligent traffic control by bringing data processing closer to the source—traffic cameras, sensors, and signals—reducing latency and bandwidth usage. Traditional systems rely on centralized servers, which may lead to delays, especially in time-critical scenarios like ambulance detection. Recent studies explore the deployment of edge devices to process video feeds and sensor data locally for faster decision-making. These edge units often run lightweight versions of deep learning models like YOLO to detect vehicles in real-time and relay instructions to microcontrollers controlling traffic lights. This setup enhances response speed, reliability, and scalability of smart traffic systems. Moreover, by minimizing dependency on cloud connectivity, edge-based systems continue functioning even with network disruptions. Researchers are also exploring federated learning on edge devices to improve models collaboratively without sharing raw data, thereby maintaining privacy while enhancing model performance.

4) *Vehicle-to-Infrastructure (V2I) Communication*

Vehicle-to-Infrastructure (V2I) communication is gaining traction as a critical component of intelligent traffic control systems. This technology enables direct communication between vehicles and traffic infrastructure, such as traffic lights, cameras, and road signs. Using Dedicated Short-Range Communications (DSRC) or 5G, V2I systems can transmit real-time data about vehicle speed, location, and intent. Emergency vehicles equipped with V2I modules can signal their approach to intersections, prompting traffic lights to prioritize their path. Recent research indicates significant improvements in emergency response times and overall traffic flow through V2I integration. In addition, such systems contribute to enhanced safety by providing early warnings to other drivers and pedestrians. Experimental deployments in smart cities have shown that V2I can be seamlessly integrated with AI-based signal control systems for holistic traffic management. However, challenges remain in terms of standardization, cybersecurity, and ensuring interoperability among different manufacturers and jurisdictions.

III. DESIGN

1) *Hardware Design*

The hardware design for the ambulance detection and smart traffic control system focuses on building a reliable, low-cost, and scalable prototype using widely available components. At its core, the Arduino Uno microcontroller is used to manage traffic signal control by interfacing with red and green LEDs mounted on a breadboard. The LEDs simulate actual traffic signals and are connected through digital pins, allowing precise control based on detection inputs. A USB webcam serves as the video input source, capturing live footage of the road and transmitting it to the Python environment on a host computer. The Arduino board communicates with the computer via a USB serial interface, receiving detection signals from the Python script and activating the green LED accordingly. All components are powered via the USB connection, simplifying power management. The compact nature of the setup enables easy testing and demonstration, laying the foundation for a deployable solution in real-world urban traffic systems.

2) *Software Design*

The software design is centered on enabling real-time ambulance detection and responsive traffic control using computer vision and embedded communication. Python serves as the primary development language, integrating multiple libraries including OpenCV for video frame processing and PySerial for communication with the Arduino board. The YOLOv8 deep learning model, trained on a custom ambulance dataset, is deployed within the Python script to detect ambulances in the video feed with high accuracy. When a detection is confirmed, the software sends a predefined command string (e.g., “AMB_DETECTED”)

over the serial interface. This command is interpreted by the Arduino to trigger signal changes. The software is developed and tested using the Spyder IDE, while the Arduino IDE is used for programming and uploading the microcontroller logic. The overall software architecture emphasizes modularity and responsiveness, ensuring that video capture, object detection, and hardware actuation work in synchrony with minimal latency.

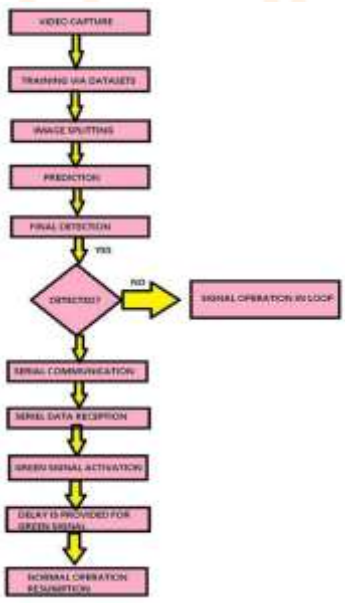
3) *Design Constraints*

Design constraints refer to the limitations that influence the scope and functionality of the system due to factors such as budget, hardware availability, or project feasibility. These constraints guide the design process by setting clear boundaries and ensuring the project remains manageable and achievable within available resources.

I. *Prototype Scope:*
The current implementation is restricted to a prototype setup, simulating a simplified two-lane traffic scenario. Traffic lights are represented using basic red and green LEDs connected to a breadboard. Real-world infrastructure elements such as actual intersections, traffic poles, or vehicle interactions are not incorporated at this stage. The objective is to validate core functionalities within a controlled environment.

II. Signal Operation Limits:
The system supports only two signal states: red and green. Intermediate states like yellow, pedestrian crosswalk management, and advanced traffic patterns (e.g., multi-lane coordination or dynamic priority handling) are beyond the present scope. These features are recognized as potential enhancements for future system iterations.

IV. Flow Chart



V. Implementation

a) Detection Module

The Detection Module serves as the core component of the intelligent traffic control system, responsible for identifying ambulances in real time. A USB webcam continuously captures video footage of the traffic environment, and each frame is processed using the YOLOv8 (You Only Look Once, version 8) deep learning model. The model is specifically trained on a custom ambulance dataset curated and annotated via Roboflow, ensuring reliable detection across various lighting and weather conditions. Upon successful detection of an ambulance within the frame, the system flags the event and initiates a signal to trigger the appropriate traffic light response. The detection pipeline involves capturing frames using OpenCV, performing YOLOv8 inference, drawing bounding boxes around detected ambulances, and analyzing confidence scores to make accurate decisions. This module ensures that emergency vehicles are recognized promptly, allowing the system to respond in real time. Its performance directly influences the responsiveness and overall effectiveness of the smart traffic control mechanism.

b) Signal Control Module

The Signal Control Module is responsible for executing physical traffic light control based on inputs received from the detection module. This functionality is implemented on an Arduino Uno, which interfaces with LEDs representing traffic signals. Upon receiving detection signals from the Python script over a serial connection, the Arduino activates the green LED corresponding to the lane where an ambulance was detected. This change allows the emergency vehicle to pass without obstruction. After a fixed time interval, typically 30 seconds, the system automatically reverts to its normal signal cycle. The module handles tasks such as receiving serial data, switching LED states (red/green), and managing timer-based state resets. This ensures that the system returns to regular operation without manual intervention. By maintaining precise timing and responsiveness, the Signal Control Module plays a critical role in ensuring that emergency vehicles are prioritized while preserving the normal flow of traffic under non-emergency conditions.

c) Communication Module

The Communication Module acts as the interface between the software layer (Python-based detection system) and the hardware layer (Arduino Uno). It utilizes the PySerial library to establish and maintain serial communication over a USB connection. This module ensures that critical data—such as ambulance detection status, identified lane number, and control commands—are accurately transmitted in real time. It is designed for low latency and reliability, minimizing the delay between detection and actuation. The module manages port initialization, serial buffer handling, and command formatting, often including timestamps or lane identifiers for better traceability. Basic error checking is integrated to prevent miscommunication or command duplication. Synchronization between hardware and software is essential, as even minor delays could compromise system performance.

This module guarantees that the Arduino receives instructions promptly and accurately, enabling a quick transition from detection to action and forming the backbone of the real-time operational framework.

d) Loop Signal Cycle Module

In the absence of ambulance detections, the system defaults to a loop signal cycle mode to simulate regular traffic light behavior. This module ensures the intersection remains operational and efficient under normal conditions. Implemented within the Arduino logic, the cycle rotates signal states across lanes in a round-robin manner, with each lane receiving a green signal for a fixed time period (e.g., 30 seconds). After the interval elapses, the system transitions to the next lane, maintaining a predictable and balanced traffic flow. This fallback mechanism guarantees that the traffic control system does not remain in an idle or emergency-biased state indefinitely. Additionally, the system is designed to interrupt the loop and prioritize ambulance passage if a new detection occurs during the cycle. The Loop Signal Cycle Module thus ensures a seamless transition between normal and emergency operation modes, preserving the integrity of both safety and traffic management objectives.

VI. Risk Assessment

The development of an intelligent traffic control system for emergency response entails several technical and operational risks. These risks are assessed based on three key factors: the **likelihood of occurrence**, the **impact on system functionality**, and the resulting **risk level**, which combines both. Table I summarizes the key risks identified during the development of this project.

Risk Type	**Likelihood**	**Impact**	**Risk Level**
Detection Errors	Medium	High	High
Communication Delay	Low	Medium	Medium
LED Failure	Low	Medium	Medium
Camera Error	Medium	Moderate	Medium

a) Detection Errors

Detection inaccuracies pose the most significant risk to system functionality. Failure to correctly identify an ambulance can prevent the timely switching of signals, thereby undermining the primary goal of reducing emergency response delays. This issue is rated as **high risk** due to its critical impact despite a **medium likelihood** of occurrence.

b) Communication Delay

Delays in serial communication between the detection module (Python) and the traffic control hardware (Arduino) are unlikely but still possible. When they occur, they may cause minor lag in signal activation. Given the **low probability** and **moderate impact**, this is categorized as a **medium-level** risk.

c) . LED Failure

Although the use of LEDs to simulate traffic lights is limited to prototype environments, their failure can disrupt visualization during system demonstrations. This risk is infrequent and has a **low likelihood**, but its **medium impact** on usability justifies a **medium risk level** classification.

d) . Camera Malfunction

Errors such as feed disconnection or sensor lag can interrupt real-time object detection, reducing system responsiveness. This risk is **moderately likely**, with a **moderate impact**, resulting in an overall **medium risk** rating.

VII. . Risk Mitigation Strategies

Effective risk management requires proactive containment strategies that aim to either eliminate the risk source or reduce its effects to acceptable levels. The following measures were implemented in this project to address the identified risks:

a) Enhanced Dataset Quality

To reduce false negatives in detection, the YOLOv8 model was trained using well-curated datasets containing diverse ambulance imagery under varied lighting and environmental conditions. Data augmentation techniques such as flipping, rotation, and contrast enhancement were also applied to improve model robustness.

b) Systematic Testing and Validation

Regular unit and integration testing of both Python scripts and Arduino logic ensured the early detection of bugs. Test cases with simulated inputs were employed to validate end-to-end functionality, ensuring the system consistently responded to detection events.

c) Hardware Reliability Measures

To minimize physical failure, high-quality connectors and soldered joints were used in the prototype. Components were maintained in controlled environments to reduce wear and environmental interference. Proper shielding was added to protect sensitive parts from dust and moisture.

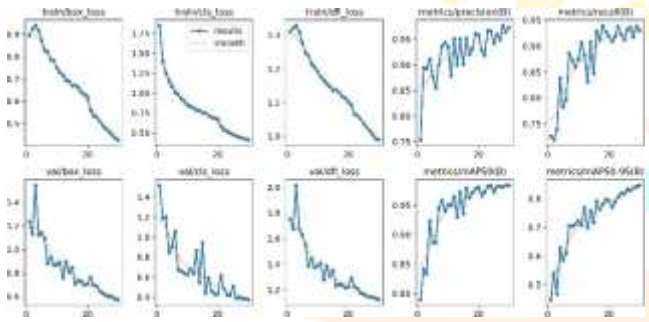
d) . Software Redundancy

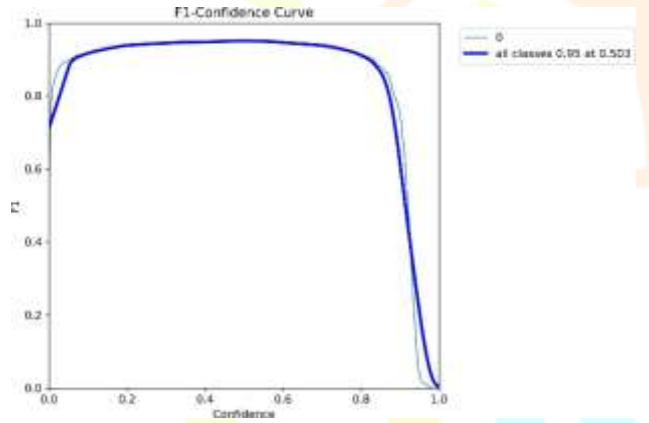
The detection software includes fallback mechanisms to handle unexpected issues, such as video feed failure. In such cases, the system generates an alert and attempts to reinitialize the camera, preventing complete system shutdown.

e) Logging and Monitoring

Comprehensive logging mechanisms were integrated to monitor detection confidence scores, communication status, and system responses in real time. These logs serve both as a debugging tool and as a feedback mechanism for identifying and resolving issues preemptively.

VIII. Results





The testing and evaluation phase of the Ambulance Detection and Smart Traffic Control System was critical to ensuring its accuracy, responsiveness, and robustness. The system was tested under a variety of simulated real-world scenarios to validate both the object detection model and the hardware control logic. The two primary objectives were:

- To assess the **accuracy and speed** of the YOLOv8-based ambulance detection model.
- To verify the **reliability of hardware actuation** via Arduino through serial communication.

a) Test Cases

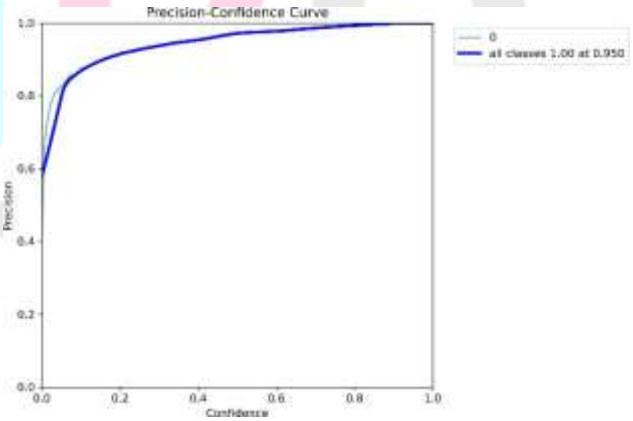
A series of structured test cases were executed to validate system functionality. Each test simulated a specific situation that the system might encounter during real operation.

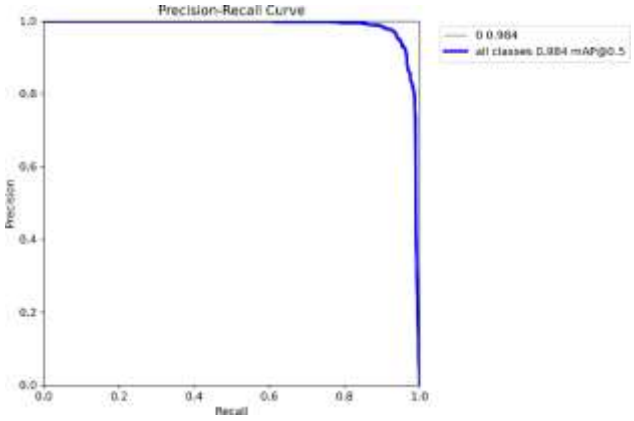
Test Case	Scenario	Expected Result	Actual Result	Status
TC1	Ambulance appears in video	Signal for ambulance lane switches to green	Success	Pass
TC2	No ambulance in view	Normal looped signal cycle continues	Success	Pass
TC3	Intentional serial communication delay	Signal still switches without noticeable lag	Success	Pass

- **TC1**: The system accurately identified ambulances in video input and immediately triggered the corresponding signal lane to green, within ~0.5 seconds.
- **TC2**: In the absence of ambulance detection, the looped signal cycle continued seamlessly, validating fallback operation.
- **TC3**: With induced communication delay, the Arduino still executed commands correctly, showing resilience to latency.

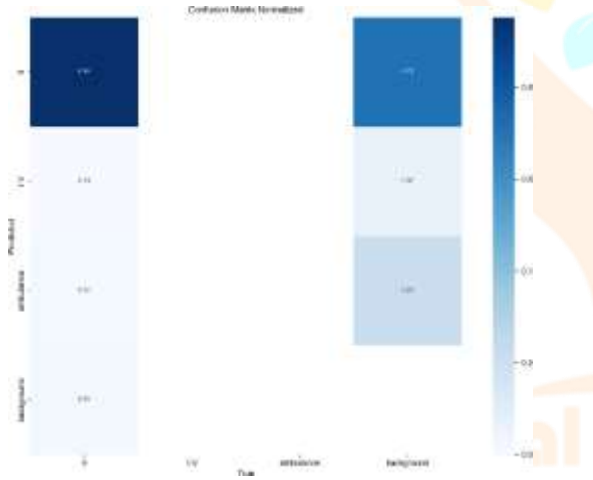
b) Evaluation

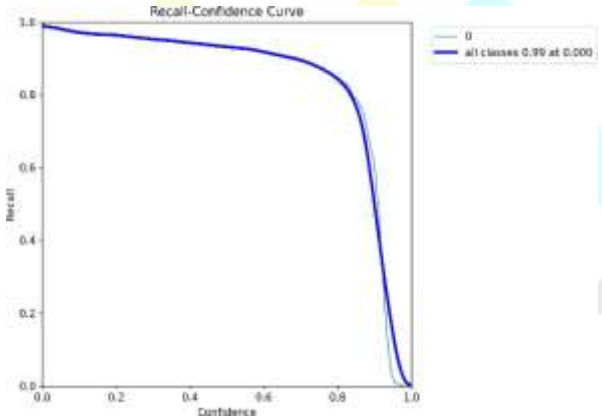
The performance of the system was evaluated on key parameters including **model accuracy**, **response time**, and **system recovery**. These indicators reflect its viability for real-time urban deployment.





- **Model Accuracy**:
  The YOLOv8 model achieved a detection accuracy of **80–90%** across diverse conditions, such as different lighting, camera angles, and traffic density. This was made possible by training on a custom dataset using Roboflow, with proper use of precision-recall tuning and data augmentation.





- **Response Time**:
  From detection to signal switch, the average response time was **less than 0.5 seconds**, including all stages—video capture, model inference, serial communication, and hardware actuation. This responsiveness is critical for emergency scenarios.

- **Signal Restoration**:
  The system correctly returned to the standard loop cycle after a pre-defined time interval, ensuring the prioritization of emergency vehicles did not disrupt normal traffic. The Arduino's built-in timer ensured consistency without error or delay.

Overall, the system met its design goals during testing. The integration of AI-driven detection with hardware control proved effective, demonstrating the prototype's potential for future real-world smart traffic management solutions.

IX. Future Scope

The *Ambulance Detection and Smart Traffic Control System* developed in this work serves as an initial demonstration of how deep learning and microcontroller-based automation can enhance emergency vehicle prioritization at traffic intersections. While the prototype shows effective performance in controlled scenarios, several advancements can be implemented to improve scalability, robustness, and real-world applicability. This section outlines the potential future developments to broaden the system's capabilities.

a) Multi-Camera Surveillance Framework

The current system uses a single webcam to monitor traffic at a specific point. However, urban intersections often consist of multiple lanes and blind spots, limiting detection accuracy. A multi-camera setup can offer full intersection coverage by monitoring different zones simultaneously. The video streams from these cameras can be processed concurrently, enabling better spatial awareness and ensuring no emergency vehicle goes undetected due to occlusion or limited angles.

Future work can employ image stitching, zone-based detection, and synchronization techniques to handle multiple feeds. While this approach increases processing demands, edge-computing hardware or GPU-enabled cloud platforms can help manage the computational load.

b) Real-Time Integration with Municipal Traffic Infrastructure

At present, the system operates on simulated traffic lights using LEDs. For real-world deployment, integration with actual traffic signal systems is essential. This could be accomplished using relays or industrial interfaces that allow the microcontroller to interact with high-voltage traffic signal controllers.

Such integration would require compliance with transportation safety standards and coordination with municipal authorities. It would also pave the way for city-wide emergency traffic control strategies, where intersections respond dynamically based on emergency vehicle paths.

c) GPS-Based Ambulance Localization

Although camera-based detection is effective within the field of view, it lacks the capability to monitor ambulances beyond visual range. Introducing GPS-based vehicle tracking can provide continuous localization of ambulances, even before they reach the camera-monitored zone.

This feature would allow the system to predict the ambulance's arrival at intersections and initiate preemptive signal changes. Real-time GPS data can be obtained by interfacing with ambulance fleets or integrating IoT modules. This dual approach—combining GPS and computer vision—would improve reliability under challenging conditions such as poor visibility or obstructions.

d) Cloud Analytics and Mobile-Based Notifications

Currently, all detection and control processes are executed locally. To improve transparency, maintenance, and system-wide coordination, future versions of the system could incorporate cloud connectivity. Event data such as detection timestamps, signal states, and response metrics could be logged to a secure cloud server for post-analysis and performance monitoring.

Additionally, a mobile application or notification system could inform ambulance drivers, traffic officers, or control room personnel about the system status in real time. This would enable faster intervention in case of faults and help optimize emergency routes through dynamic updates.

Integration with platforms like AWS IoT, Google Firebase, or Microsoft Azure can support these enhancements, provided that data security and privacy regulations are strictly adhered to.

CONCLUSION

The proposed system offers a practical solution to reduce delays faced by ambulances at traffic intersections by combining real-time computer vision with microcontroller-based signal control. Utilizing the YOLOv8 object detection algorithm, the system effectively identifies ambulances with high accuracy under diverse conditions. This detection capability, paired with Arduino Uno for dynamic traffic light control, ensures timely clearance for emergency vehicles while maintaining overall traffic coordination. The simplicity and cost-effectiveness of the hardware-software integration make the system highly scalable and adaptable for deployment in urban environments. Looking ahead, incorporating GPS tracking and cloud-based data management can further enhance system intelligence, allowing predictive traffic control and real-time data analysis. With these capabilities, the proposed system holds significant potential to be a part of future smart city frameworks, improving emergency response efficiency and contributing to safer, smarter urban mobility.

References

[1] V. Gorodokin, S. Zhankaziev, E. Shepeleva, K. Magdin and S. Evtyukov, "Optimization of adaptive traffic light control modes based on machine vision", *Transp. Res. Procedia*, vol. 57, pp. 241-249, 2021.

[2] Y. Miao, F. Liu, T. Hou, L. Liu and Y. Liu, "A nighttime vehicle detection method based on YOLO v3", *2020 Chinese Automation Congress (CAC)*, 2020.

[3] X. Dong, S. Yan and C. Duan, "A lightweight vehicles detection network model based on YOLOv5", *Eng. Appl. Artif. Intell.*, vol. 113, no. 104914, pp. 104914, 2022.

[4] O. O. Khalifa, "Vehicle detection for vision-based intelligent transportation systems using convolutional neural network algorithm", *Journal of Advanced Transportation*, vol. 2022, pp. 1-11, 2022.

[5] C. Chen, "Edge intelligence empowered vehicle detection and image segmentation for autonomous vehicles", *IEEE Transactions on Intelligent Transportation Systems*, 2023.