**[ Image & Video Processing, Analysis of Traffic Signal Area to**

**Find Emergency Vehicles Using Deep Learning Models ]**

**Submitted**

**By**

P. Lakshmi Vignesh [BU21EECE0100515]

Moda SriRangaManjula [BU21ECE0100104]

Lalam Jithendra [BU21EECE010487]

**Under the Guidance of**

**(Sanhita manna, Assistant professor)**

**(Duration: 06/07/2024 to 13/03/2025)**



**Department of Electrical, Electronics and Communication Engineering**

**GITAM School of Technology**

**GITAM**

**(DEEMED TO BE UNIVERSITY)**

**(Estd. u/s 3 of the UGC act 1956)**

**NH 207, Nagadenehalli, Doddaballapur taluk, Bengaluru-561203 Karnataka, INDIA.**

**DECLARATION**

**I/We declare that the project work contained in this report is original and it has been done by me under the guidance of my project guide.**

**Name:**

**Date: Signature of the Student**

**Department of Electrical, Electronics and Communication Engineering [14 Bold]**

**GITAM School of Technology, Bengaluru-561203**

**A logo with text on it

AI-generated content may be incorrect.**

**CERTIFICATE**

**This is to certify that (Student Name) bearing (Regd. No.:) has satisfactorily completed Mini Project Entitled in partial fulfillment of the requirements as prescribed by University for VIIIth semester, Bachelor of Technology in “Electrical, Electronics and Communication Engineering” and submitted this report during the academic year 2024-2025.**

**[Signature of the Guide] [Signature of HOD]**

**Table of contents**

[**Chapter 1: Introduction 1**](#_heading=h.gjdgxs)

[1.1 Overview of the problem statement 1](#_heading=h.30j0zll)

[1.2 Objectives and goals 1](#_heading=h.1fob9te)

[**Chapter 2 : Literature Review 2**](#_heading=h.3znysh7)

[**Chapter 3 : Strategic Analysis and Problem Definition 3**](#_heading=h.2et92p0)

[3.1 SWOT Analysis 3](#_heading=h.tyjcwt)

[3.2 Project Plan - GANTT Chart 3](#_heading=h.1t3h5sf)

[3.3 Refinement of problem statement 3](#_heading=h.2s8eyo1)

[**Chapter 4 : Methodology 4**](#_heading=h.17dp8vu)

[4.1 Description of the approach 4](#_heading=h.3rdcrjn)

[4.2 Tools and techniques utilized 4](#_heading=h.26in1rg)

[4.3 Design considerations 4](#_heading=h.lnxbz9)

[**Chapter 5 : Implementation 5**](#_heading=h.1ksv4uv)

[5.1 Description of how the project was executed 5](#_heading=h.44sinio)

[5.2 Challenges faced and solutions implemented 5](#_heading=h.2jxsxqh)

[**Chapter 6:Results 6**](#_heading=h.z337ya)

[6.1 outcomes 6](#_heading=h.3j2qqm3)

[6.2 Interpretation of results 6](#_heading=h.1y810tw)

[6.3 Comparison with existing literature or technologies 6](#_heading=h.2xcytpi)

[**Chapter 7: Conclusion 7**](#_heading=h.1ci93xb)

[**Chapter 8 : Future Work 8**](#_heading=h.2bn6wsx)

[Here write Suggestions for further research or development Potential improvements or extensions 8](#_heading=h.qsh70q)

[**References 9**](#_heading=h.1pxezwc)

# Chapter 1: Introduction

# Overview of the problem statement :

This problem statement revolves around the need to develop an advanced system that can automatically monitor, process, and analyze images and videos from traffic signal areas. By applying image and video processing, deep learning, and AI to analyze the vehicles to identify the high priority vehicles and to prioritize them at traffic junctions. Traffic congestion in urban areas has become one of the most pressing challenges

in modern cities. With an increasing number of vehicles on the road, traffic signals often cause delays, leading to severe bottlenecks at junctions. While traffic management systems are designed to regulate vehicle movement efficiently, they lack the ability to prioritize emergency vehicles like ambulances and fire engines, which need immediate clearance to save lives. These emergency vehicles frequently get stuck in traffic, losing valuable response time, which can be critical in life-or-death situations.

Currently, most traffic control systems operate on fixed signal timings or use sensor-based systems that adjust signals based on overall traffic flow. However, these methods do not differentiate between emergency vehicles and regular traffic, resulting in unnecessary delays for ambulances and fire engines. In many cases, traffic police manually intervene to clear the way, which is neither efficient nor scalable in large cities with multiple junctions.

The need for an automated, AI-based traffic management system that can identify high-priority vehicles in real time and dynamically adjust signal timings is crucial. By leveraging image processing, deep learning, and artificial intelligence (AI), this project aims to develop an intelligent traffic control system that can:

1. Automatically detect emergency vehicles from traffic camera feeds.
2. Analyse real-time traffic density to optimize signal timings dynamically.
3. Prioritize emergency vehicles by adjusting traffic lights accordingly.
4. Improve overall traffic flow, reducing congestion for both regular and emergency vehicles.

This system would significantly reduce response times for ambulances and fire engines, ensuring they reach their destination without unnecessary delays. It would also minimize manual intervention and contribute to the development of smart, AI-driven urban traffic management systems.

## 

## Objectives and goals

**Objective:**

* This project involves developing a machine learning model that utilizes object detection techniques to analyze vehicle density across traffic lanes. An algorithm will be designed to display identified high priority vehicles at traffic junctions, enabling reduced emergency rate and accurate predictions.
* The objective of our project is to collect datasets best suited for segmentation, relabel and reannotate them with masked coordinates to support instance segmentation.
* To train a new object detection model with added instance segmentation to improve the previous model.

**Main Goals**

* To identify emergency vehicles and classify them as high-priority vehicles separate from the rest of the vehicles.
* To identify the lane of emergency vehicles and communicate their presence to traffic signal junctions for prompt action.

**Additional Goals**

* To reduce the emergencies for people by prioritizing the high priority vehicles.
* To reduce the waiting time of high-priority vehicles at traffic junctions.

# Chapter 2 : Literature Review

**2.1 Advanced Traffic Signal Control System for Emergency Vehicles:** Published in

December 2022, this study was authored by Sunil M, V Yashaswini Naidu, Vignesh R, Vishwas P, and Amitha S. It utilizes IoT and Machine Learning technologies to enhance traffic signal control for emergency vehicles. The primary drawbacks highlighted in this study include issues with the reliability of detection, challenges in integrating with existing traffic infrastructure, and concerns related to privacy.

**Link:** [Advanced Traffic Signal Control System for Emergency Vehicles](https://kssem.edu.in/images/naac_2022_CRITERIA3_1707279223256.pdf)

**2.2 AI-Based Emergency Vehicles Detecting and Traffic Controlling System**: This research, published in March 2024 by Aditya Pawar and Yogesh Sutar, explores the use of AI, object detection, and video processing for emergency vehicle detection and traffic control. The study identifies several drawbacks, such as the reliability and safety of AI, security and privacy concerns, and issues related to scalability and maintenance.

**Link:**[AI-Based Emergency Vehicles Detecting and Traffic Controlling System](https://ijnrd.org/papers/IJNRD2403303.pdf)

**2.3** **Traffic Management for Emergency Vehicle Priority Based on Visual Sensing**: Authored by Kapileswar Nellore and Gerhard P. Hancke and published in August 2016, this study focuses on using object detection and image processing to prioritize traffic management for emergency vehicles. The drawbacks noted in this research include the need for MAC protocol enhancement, challenges in scalability and deployment, and high energy consumption.

**Link:** [Traffic Management for Emergency Vehicle Priority Based on Visual Sensing](https://www.mdpi.com/1424-8220/16/11/1892)

**2.4 Efficient Dynamic Traffic Control System Using Wireless Sensor Networks:** This study was published in June 2014 and authored by R. Bharadwaj, J. Deepak, M. Baranitharan, and V. Vaidehi. The research focuses on the use of embedded systems to develop a dynamic traffic control system utilizing wireless sensor networks. The study identifies several drawbacks, including potential privacy concerns, data latency and processing delays, and challenges in identifying emergency vehicles.

**Link:**[Efficient Dynamic Traffic Control System Using Wireless Sensor Networks](https://www.researchgate.net/publication/271473076_Efficient_dynamic_traffic_control_system_using_wireless_sensor_networks)

# Chapter 3 : Strategic Analysis and Problem Definition

## 3.1 SWOT Analysis:

### 

### 

### 3.2 Project Plan - GANTT Chart



# Chapter 4 : Methodology

## 4.1 Description of the approach

This study follows a structured two-stage methodology designed to build an effective and efficient Vehicle classification system, leveraging advanced data preprocessing and machine learning techniques.

**Stage I: Data Collection and Preprocessing**

* In the first stage of the methodology, data collection and preprocessing are critical for developing an effective traffic signal control system that prioritizes high-priority vehicles. This stage involves several key steps
* Video Footage Acquisition: The initial step is to gather video footage from various traffic intersections. This footage is essential for analyzing vehicle movements and densities, which will inform the training of deep learning models. The data collected should represent diverse traffic conditions to ensure the model's robustness in real-world scenarios.
* Annotation of Data: Once the video data is collected, it must be annotated to identify different vehicle types, including high-priority vehicles like ambulances and fire engines. This annotation process is crucial as it provides the labeled data necessary for training the deep learning models. Accurate labeling ensures that the model learns to recognize and differentiate between various vehicle categories effectively.
* Data Preprocessing: After annotation, the data undergoes preprocessing to enhance its quality and usability. This may include resizing video frames, normalizing pixel values, and augmenting the dataset to increase its diversity. Data augmentation techniques, such as rotation, flipping, and color adjustments, can help improve the model's ability to generalize across different conditions.
* Segmentation of Vehicle Types: In addition to general preprocessing, instance segmentation techniques are applied to classify each vehicle type distinctly. This step is vital for improving detection accuracy, as it allows the model to recognize not just the presence of a vehicle but also its specific type, such as bikes, cars, ambulances, and fire engines. Enhanced classification capabilities are essential for prioritizing high-priority vehicles at traffic signals .
* Preparation for Model Training: Finally, the preprocessed and annotated data is organized and split into training, validation, and test sets. This division is necessary to evaluate the model's performance accurately and ensure that it can generalize well to unseen data. The training set is used to teach the model, while the validation and test sets help assess its accuracy and reliability in real time applications.

**Stage II: Model Development and Training**

* In Stage II of the methodology, the focus shifts to model development and training, which are essential for creating an effective traffic signal control system that prioritizes high-priority vehicles. This stage encompasses several critical steps:
* Model Selection: The first step involves selecting an appropriate deep learning architecture for the task. In this case, the YOLOv11n model was chosen due to its efficiency in real-time object detection. This model is particularly suitable for identifying high-priority vehicles like ambulances and fire engines in dynamic traffic environments.
* Training the Model: The selected model is then trained using the preprocessed and annotated dataset from Stage I. The training process

involves feeding the model with images and their corresponding labels, allowing it to learn to recognize different vehicle types. The training is conducted over multiple epochs, with the model iteratively adjusting its parameters to minimize detection errors.

* Validation and Testing: After training, the model's performance is validated against a separate validation dataset. This step is crucial to ensure that the model generalizes well to new, unseen data. The validation process helps in fine-tuning the model's parameters and improving its accuracy. The final YOLOv11n model achieved over 73% detection accuracy, indicating its moderate reliability for real-time traffic management .
* Instance Segmentation: To enhance the model's capabilities, instance segmentation techniques are employed. This allows the model not only to detect vehicles but also to classify them into specific categories, such as bikes, cars, ambulances, and fire engines. This classification is vital for prioritizing high-priority vehicles at traffic signals, as it enables the system to respond appropriately based on the vehicle type detected.
* Performance Evaluation: The model's performance is continuously evaluated using various metrics, such as precision, recall, and F1-score. These metrics provide insights into how well the model is performing in detecting and classifying vehicles. The goal is to achieve high precision in detecting ambulances and fire engines, ensuring that these vehicles receive priority at traffic signals.
* Integration with Traffic Signal Control Systems: Finally, the trained model is integrated with traffic signal control systems. This integration allows for dynamic adjustments to signal timings based on real-time traffic conditions and the presence of high-priority vehicles. The system aims to optimize traffic flow and reduce waiting times for emergency vehicles, ultimately improving overall traffic management.

### 

**4.2 Tools and Techniques Utilized**

To develop an efficient AI-powered traffic management system, several advanced tools and techniques have been utilized. These components work together to enable real-time vehicle detection, classification, and traffic flow optimization. Below is a detailed explanation of each tool and technique:

**1. Deep Learning Models:**

**YOLOv11 for Vehicle Detection**

* **Model Overview**: YOLOv11 (You Only Look Once, version 11) is a state-of-the-art object detection model designed for high-speed and high-accuracy real-time processing.
* **Usage in Project**: This model is employed to detect and classify vehicles (e.g., cars, buses, ambulances, fire engines) from live video feeds captured at traffic signals.
* **Why YOLOv11?**
  + **Real-time processing**: YOLO-based models process images in a single pass, making them ideal for real-time applications.
  + **High accuracy**: The model achieves a high mean Average Precision (mAP), ensuring reliable vehicle identification.
  + **Robustness**: Can detect objects in varying lighting and weather conditions, essential for outdoor traffic environments.

**2. Image Processing**

**OpenCV for Real-time Video Analysis**

* **Purpose**: OpenCV (Open Source Computer Vision Library) is used to preprocess and enhance video frames for better model performance.
* **Key Functions Utilized**:
  + **Frame Extraction**: Capturing individual frames from real-time video feeds.
  + **Image Preprocessing**: Applying techniques such as grayscale conversion, noise reduction, and contrast enhancement to improve model accuracy.
  + **Vehicle Tracking**: Using object tracking algorithms to monitor vehicle movement between frames.
  + **Lane and Region Detection**: Identifying lanes and specific regions of interest (e.g., emergency vehicle lanes).

**3. Programming Languages**

**Python for Model Development and Integration**

* **Why Python?**
  + **Extensive AI/ML Libraries**: Python supports frameworks like TensorFlow, PyTorch, and OpenCV, making it ideal for deep learning applications.
  + **Easy Integration**: Python allows seamless integration with databases, web applications, and IoT devices.
  + **Rapid Prototyping**: Python’s simplicity and extensive libraries accelerate development and testing.

**4. AI Frameworks**

**TensorFlow and PyTorch for Model Training**

* **TensorFlow**
  + Used for model deployment and inference optimization.
  + TensorFlow Lite (TFLite) is employed for lightweight AI model execution on edge devices.
  + TensorFlow’s hardware acceleration ensures faster processing on GPUs and TPUs.
* **PyTorch**
  + Used for initial model training due to its dynamic computation graph, making debugging and modifications easier.
  + Preferred for research and experimentation before final deployment.
* **Framework Selection**
  + The project leverages **PyTorch** for model development and **TensorFlow** for optimized deployment, ensuring the best of both worlds in terms of flexibility and efficiency.

**5. Database Management**

* **Purpose**: A database is required to store detected vehicle details, timestamps, and movement history for further analysis and decision-making.
* **Stored Data**:
  + **Vehicle ID**: Unique identifier for each detected vehicle.
  + **Timestamp**: Entry and exit times at a particular junction.
  + **Vehicle Type**: Classification into emergency vehicles, private cars, public transport, etc.
  + **Lane Information**: To track which lane has more congestion.
  + **Emergency Alerts**: Data on detected ambulances or fire trucks for prioritization.

By combining state-of-the-art deep learning models, efficient image processing techniques, robust AI frameworks, and a scalable database, the system effectively detects, classifies, and optimizes traffic in real-time. This architecture not only ensures smooth traffic flow but also prioritizes emergency vehicles, making urban roads more efficient and safer.

#### 4.3 Design considerations:

1. **Model Accuracy**
   * Uses **YOLOv11** for high-precision vehicle detection, ensuring correct classification of emergency and regular vehicles.Enhances accuracy with **image preprocessing**
2. **Real-Time Processing Speed**
   * Optimized for **fast inference** using **GPU acceleration, edge computing, and TensorRT**.
3. **Scalability**
   * Modular architecture allows easy expansion to multiple intersections.
4. **System Integration with Traffic Signals**
   * Uses **IoT-based adaptive signal control** to dynamically adjust signal timings based on real-time vehicle detection.

# Chapter 5 : Implementation

## 5.1 Description of how the project was executed:

1.**Project Planning**: The initial phase includes defining the project objectives and milestones. This involves outlining the specific goals for each review stage, ensuring that the project remains on track and meets its intended outcomes.

**2.Data Collection and Preprocessing:** The next step involves gathering video data from traffic environments. This data is then preprocessed to enhance its quality and prepare it for model training. Preprocessing may include resizing images, normalizing pixel values, and annotating vehicle types.

* The collected images were labeled and annotated using the CVAT.ai platform. We manually marked the vehicles and classified them based on their type (ambulance, fire engine, etc.) to prepare the dataset for training the object detection model.



3.**Preprocessing**:

* Before feeding the data into the model, we applied several preprocessing steps. The images were resized to 640 x 640 pixels for uniformity. We also applied auto-orientation to correct any discrepancies in the image orientation.
* For two of the models, we converted all images to greyscale to assess performance without color information.

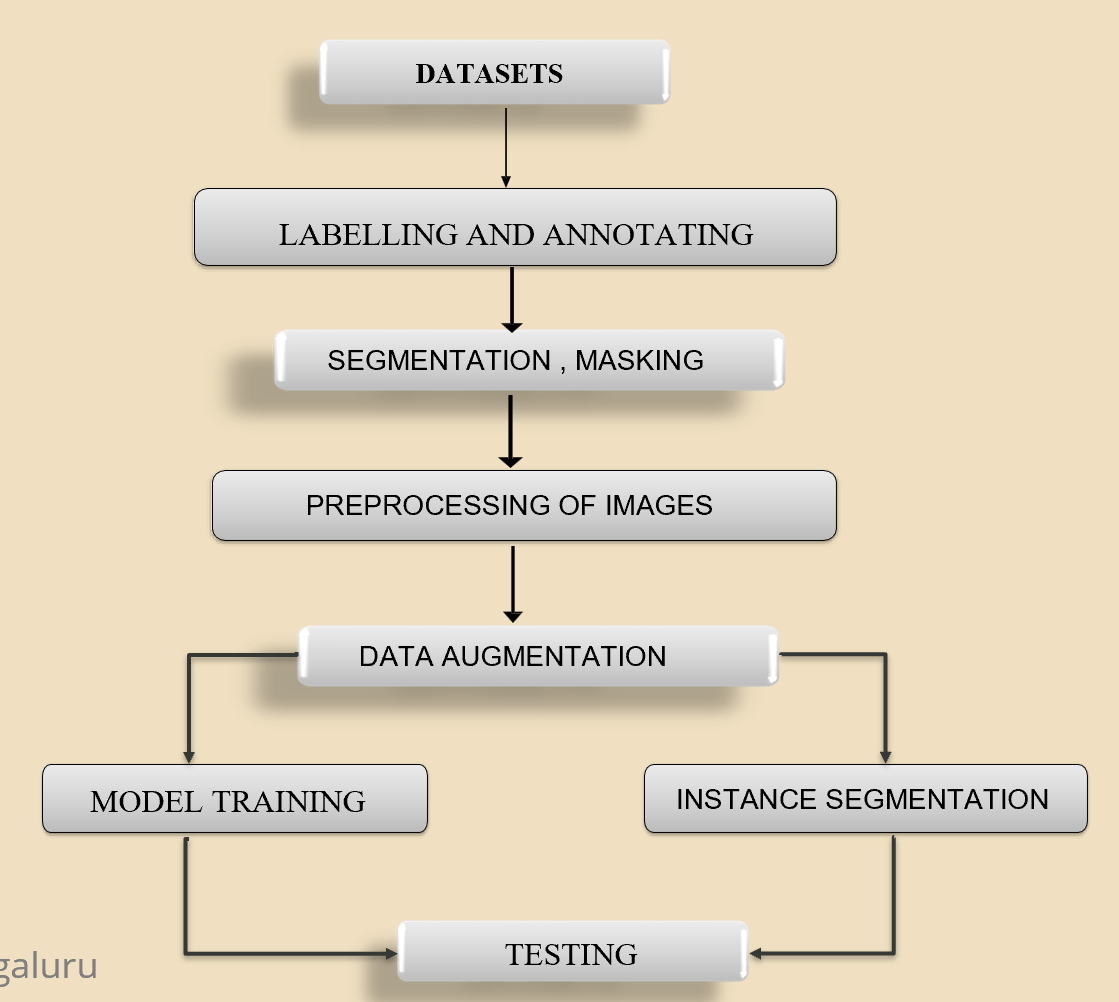
**4.Model Development:** In this stage, a suitable deep learning model, such as YOLOv11n, is selectefor the task. The model is designed to detect and classify various vehicle types, including high-priority vehicles like ambulances and fire engines.

1. **Model Selection**:

o We selected YOLOv11 for object detection, which provides multiple options such as augmentation, segmentation, posing, and classification. However, we focused specifically on the object detection model due to the need for real-time vehicle identification.

1. **Training**:

o Initially, we attempted training with YOLOv11m, but due to system compatibility issues, we switched to YOLOv11n. After trying different numbers of epochs (iterations), we found that training with 50 epochs provided the best balance between accuracy and performance.

1. **Testing and Validation**:
   * + The model was tested in real-world conditions, with a focus on detecting ambulances, fire engines, and other vehicles. Testing included scenarios with varied lighting, blurring, noise, and low visibility.
     + The system was validated against defined use cases and test cases, confirming that the model met accuracy and real-time performance requirements.

### Challenges faced and solutions implemented:

**1.Difficulty in Distinguishing Emergency Vehicles from Normal Traffic:**

* **Challenge:**
* Emergency vehicles (ambulances, fire trucks, and police cars) often have similar shapes, colors, and sizes as regular vehicles, making it difficult for traditional object detection models to differentiate them.
* Factors like low visibility, occlusion by other vehicles, and nighttime conditions further reduced the accuracy of emergency vehicle detection.
* **Solution:**
* **Instance Segmentation for Precise Classification**: Instead of relying solely on object detection (bounding boxes), instance segmentation was implemented using a modified YOLOv11 model with a segmentation head.
* **Dataset Augmentation for Emergency Vehicles**: Collected a diverse dataset of emergency vehicles under different conditions (day/night, different angles, partially occurred.

**2. Processing Real-Time Video Efficiently**

**Challenge:**

* Analyzing high-resolution video feeds (1080p/4K) at 30-60 FPS in real-time requires significant computational power.
* High latency **and memory bottlenecks** were observed, especially during peak traffic hours.

**Solution:**

* **Optimized Batch Processing**
* Implemented **batch-wise frame processing**, where frames were processed in small groups instead of one-by-one, reducing computational overhead.
* Used **asynchronous execution** to **separate frame capture, preprocessing, inference, and decision-making** into parallel threads.
* **Model Compression & Acceleration**
* Converted the **YOLOv11 model to TensorFlow Lite (TFLite)** for efficient execution on edge devices.
* Used **TensorRT optimization** to accelerate inference on **NVIDIA Jetson and GPUs**.
* Applied **pruning and quantization** to reduce model size without sacrificing accuracy.
* **Region of Interest (ROI) Processing**
* Instead of analyzing the entire video frame, the system **focused only on traffic lanes, pedestrian crossings, and intersections** using ROI extraction.
* Reduced unnecessary computations by skipping static background areas.

# Chapter 6:Results

## 6.1 outcomes

### 6.2 Interpretation of results

### 

#### 6.3 Comparison with existing literature or technologies

# Chapter 7: Conclusion

Here write Suggestions for further research or development and Potential improvements or extensions

# 

# Chapter 8 : Future Work

#### Here write Suggestions for further research or development Potential improvements or extensions

#### 

# References