TrafficSync

Project submitted to

Shri Ramdeobaba College of Engineering & Management,

Nagpur in partial fulfillment of requirement for the award

of degree of

Bachelor of Engineering

In

COMPUTER SCIENCE AND ENGINEERING

(ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

By

Ms. Aanchal Pahuja

Ms. Anushka Mahule

Ms. Rishika Batra

Mr. Manvendra Chauhan

Mr. Vivek Ghidoday

Guide

Prof. Pranali Dandekar

Computer Science and Engineering
Shri Ramdeobaba College of Engineering & Management, Nagpur

440013

(An Autonomous Institute affiliated to Rashtrasant Tukdoji Maharaj Nagpur University Nagpur)

April 2024

SHRI RAMDEOBABA COLLEGE OF ENGINEERING & MANAGEMENT, NAGPUR

(An Autonomous Institute affiliated to Rashtrasant Tukdoji Maharaj
Nagpur University Nagpur)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project on **"TrafficSync"** is a bonafide work of

1. **Ms. Aanchal Pahuja**
2. **Ms. Anushka Mahule**
3. **Ms. Rishika Batra**
4. **Mr. Manvendra Chauhan**
5. **Mr. Vivek Ghidoday**

submitted to the Rashtrasant Tukdoji Maharaj Nagpur University, Nagpur in partial fulfillment of the award of a Degree of Bachelor of Engineering, in Computer Science and Engineering (Artificial Intelligence and Machine Learning). It has been carried out at the Department Computer Science and Engineering, Shri Ramdeobaba College of Engineering and Management, Nagpur during the academic year 2023-24.

Date:

Place: Nagpur

Prof. Pranali Dandekar
Project Guide

Dr. Suresh Balpande
H.O.D
(Department of Artificial Intelligence & Machine Learning)

DECLARATION

I, hereby declare that the project titled **"TrafficSync"** submitted herein, has been carried out in the Department of Computer Science and Engineering of Shri Ramdeobaba College of Engineering & Management, Nagpur. The work is original and has not been submitted earlier as a whole or part for the award of any degree / diploma at this or any other institution / University

Date:
Place: Nagpur

Ms. Aanchal Pahuja
(Roll no.: 01)

Ms. Anushka Mahule
(Roll no.: 04)

Ms. Rishika Batra
(Roll no.: 17)

Mr. Manvendra Chauhan
(Roll no.: 41)

Mr. Vivek Ghidoday
(Roll no.: 63)

ACKNOWLEDGEMENT

We would like to express our deep and sincere gratitude to our guide **Prof. Pranali Dandekar,** Professor of Computer Science and Engineering Department, RCOEM, for giving us the opportunity to work on this project and providing valuable guidance throughout the project. It was a great privilege and honor to work under her guidance. We are extremely grateful for the experience we had in this project with her.

We express our sincere gratitude to Dr. Suresh Balpande**,** Head of the Department of Computer Science Department, RCOEM for his guidance. Talent wins games, but teamwork and intelligence win championships. We would like to take this opportunity to express our deep gratitude to all those who extended their support and guided us to complete this project.

Ms. Aanchal Pahuja **Ms. Anushka Mahule**

Ms. Rishika Batra **Mr. Manvendra Chauhan**

Mr. Vivek Ghidoday

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Abstract:

In order to improve road safety and urban traffic flow, this project proposes a dynamic traffic management system. Important characteristics include decreasing delays and optimizing efficiency, as well as adjusting signal timing based on emergency vehicle priority and real-time congestion. Testing the system's responsiveness and capacity to effectively handle traffic in emergency situations is made possible via emergency vehicle simulation. In order to identify congestion hotspots, congestion density analysis algorithms take into account factors including vehicle count, traffic speed, and historical data. The system can seamlessly recover from disruptions or restarts thanks to a persistent state storage method that guarantees traffic-related parameters are preserved. When combined, these developments produce a traffic network that is safer, more effective, and more responsive.

1.2 Introduction:

In today's metropolitan areas, traffic congestion and delays caused by emergency vehicles are major concerns. By installing a dynamic traffic control system that improves safety and streamlines TrafficSync, our project tackles these problems. One of the main benefits is the ability to modify signal timings in response to current traffic conditions and emergency vehicle priority, which reduces delays and increases traffic efficiency.

Emergency vehicle simulations are incorporated into the design to evaluate the system's response and enhance management in emergency situations. In order to pinpoint congestion hotspots, algorithms that analyze traffic density at intersections take into account variables including vehicle count, speed, and historical data. Persistent state storage is also used to guarantee that the data and configurations of the system may be restored in the event of unplanned stops or restarts.

The goal of this project is to improve road safety and the general quality of life for both emergency responders and commuters by concentrating on these important regions and building a smarter, more efficient urban traffic network.

1.3 Motivation:

- **ECONOMIC IMPACT**
  Traffic congestion results in lost productivity due to longer commutes, higher logistics costs for businesses, and reduced economic activity in congested areas. These factors lead to decreased productivity, increased operating expenses, and inhibited economic growth, impacting both individuals and communities.

- **PUBLIC HEALTH IMPACT**
  Traffic congestion exacerbates air pollution, emitting harmful pollutants such as carbon monoxide and particulate matter. Prolonged exposure to these pollutants increases the risk of respiratory illnesses, cardiovascular diseases, and other health issues, particularly among vulnerable populations. This degraded air quality negatively impacts public health, leading to increased healthcare costs and reduced overall well-being within affected communities.

- **QUALITY OF LIFE IMPACT**
  Traffic congestion significantly impacts quality of life by increasing stress and frustration due to longer commutes and wasted time stuck in traffic jams. This leads to decreased opportunities for leisure activities and personal development, affecting overall well-being and satisfaction with daily life.

(1.3.1 - Traffic Congestion)

1.4 Objectives:

- **Signal Timing Adjustment**:
  Creating a dynamic signal timing system entails modifying traffic signal phases in real-time in response to the amount of traffic and the presence of emergency vehicles. The system may adjust signal timings, such as phase durations and sequences, to optimize traffic movement and shorten wait times at crossings by continually monitoring traffic flow through sensors and cameras. Furthermore, the system minimizes response times by giving emergency vehicles priority and modifying signals to give them a clear road.

- **Emergency Vehicle Simulation**:
  Simulating emergency vehicle scenarios allows the system to test its ability to handle emergency situations effectively. This process evaluates how well the system prioritizes emergency vehicles by adjusting signal timings to ensure they can move swiftly and safely through traffic. By optimizing pathways and studying outcomes from these simulations, the system can identify areas for improvement and better prepare for real-world emergency events.

- **Congestion Density Analysis:**
  Gathering and analyzing real-time traffic data, including vehicle count, velocity, and daily patterns, is necessary to analyze congestion density. This information is used to detect high-traffic regions and assess the degree of congestion at various crossings. Through an analysis of past patterns, the system is able to forecast future patterns of traffic congestion and modify traffic management tactics accordingly, focusing on congestion hotspots for more effective interventions.

1.5 Problem Definition:

TrafficSync revolutionizes urban mobility with its integration of cutting-edge technologies like object identification, computer vision, deep learning, and image processing. It finds hotspots for congestion, dynamically reroutes cars, and improves traffic flow in real-time for more comfortable journeys. TrafficSync prioritizes emergency vehicles and provides accurate lane detection to guarantee timely delivery of vital services. It improves traffic safety and law enforcement with its ability to recognize and classify vehicles. Through the seamless integration of cutting-edge technology with urban infrastructure, TrafficSync offers cities worldwide safer and more effective transportation networks.

CHAPTER 2

LITERATURE SURVEY

Sr. No.	Title - Year of Publish	Journal	Methodology	Research Gap	Conclusion
1.	A framework for dynamic smart traffic light management system	Bharati Vidyapeeth's Institute of Computer Applications and Management 2021	The methodology involves evaluating congestion using Image Processing with a practical focus on employing high-resolution cameras and Fog computing. For Smart Traffic Light (STL) implementation, cameras on each road side capture images to assess vehicle counts, allowing timely signaling of green lights and promoting coordinated traffic flow.	Limitations arise in developing countries with limited technical expertise for implementing Image Processing and Fog technologies. While STL offers automated traffic management, achieving comprehensive solutions often demands integration of various advanced capabilities on different levels.	STL offers a cost-effective solution for intersection congestion, but implementing it in developing countries with limited technical expertise poses challenges. Satisfactory traffic management requires integrated solutions on various levels.

2.	Analysis of Traffic Congestion Impacts of Urban Road Network under Indian Condition	IOP Conference Series: Materials Science and Engineering	The research provides an in-depth analysis of traffic congestion, emphasizing its negative effects on the economy, health, and the environment. It underscores the significance of congestion indices such as Travel Time Index (TTI), Buffer Index (BTI), and Planning Time Index (PTI) in evaluating the serviceability and overall performance of the urban roadway.	Implementation of High Parking charge, Increasing pedestrian facilities, still without strict traffic law all of these factors cannot be implemented With this current study.Congestion cannot be eliminated totally due to the increase in population as well as vehicle, specially two wheelers and passenger cars.	According to the vehicle composition and lane distribution, there is the variation of travel time and congestion indices. Buffer Time Index displays adequate outcome in mixed traffic conditions among the other indices.
3.	Smart traffic light control system using image processing	Annual Conference on Computer Science and Engineering Technology (AC2SET) 2020 IOP Conf. Series: Materials Science and Engineering 1088 (2021) 012021	The system captures high-res lane images, processes in MATLAB for empty and occupied lane matching, and allocates time based on traffic density through feature-based matching, resulting in shorter green signals for less congested lanes.	Challenges include technical expertise and hardware dependencies, potential delays in real-time processing, sensitivity to rapid lane changes, privacy concerns, and integration complexities.	The paper introduces an adaptive traffic light system using image processing for signal analysis and time optimization. Real-time image detection prioritizes green signals for congested lanes, utilizing efficient Canny edge extraction

4.	Traffic Prediction for Intelligent Transportation System using Machine Learning	3rd International Conference on Emerging Technologies in Computer Engineering: Machine Learning and Internet of Things (ICETCE-2020), 07-08 February 2020, (IEEE Conference Record 48199	The travel time is the essential aspect in ITS. Highlighted the challenges in accurately forecasting traffic flow due to the immense volume of available transportation data and proposed the use of machine learning, genetic, soft computing, and deep learning algorithms to analyze big data with reduced complexity.The study also presents the implementation and results of different machine learning algorithms, focusing on decision tree, support vector machines (SVM), and random forest, with an emphasis on their accuracy, precision, recall, and time taken.It also includes the ROC curve for the decision tree algorithm	It's notable that deep networks trained during this method have dangerous performance. So didn't incorporate the deep learning models in the work. Also the dataset developed doesn't have many features so it won't be a sensible decision to use the deep learning and genetic algorithms.	Performance of the models obtained through different machine learning algorithms calculated with the help of Accuracy, Precision, Recall and Time Taken. Out of all the algorithms, Random forest gave the best accuracy, precision and recall scores but SVM gave the least time taken.

CHAPTER 3

THEORETICAL BACKGROUND

3.1 Overview of YOLO and Transfer Learning:

A) YOLO:

YOLO (You Only Look Once) is a popular real-time object detection system that revolutionized the field of computer vision when it was introduced by Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi in 2016. The main idea behind YOLO is to detect objects in images or video frames by framing the object detection task as a single regression problem, rather than the traditional two-step approach of first generating region proposals and then classifying those regions. This single-step regression approach makes YOLO extremely fast and efficient, making it suitable for real-time applications.

Here's a brief overview of how YOLO works:

- **Divide and Conquer:** YOLO divides the input image into a grid of cells. Each cell is responsible for predicting bounding boxes and class probabilities for the objects contained within it.

- **Predictions at Once:** YOLO makes predictions for bounding boxes and class probabilities directly from the full image in a single pass through the neural network. This is in contrast to other object detection systems that require multiple passes or separate processing for different regions of the image.

- **Bounding Box Prediction:** For each grid cell, YOLO predicts a fixed number of bounding boxes (usually a few, e.g., 2 or 3). Each bounding box consists of 5 elements: (x, y, w, h, confidence). (x, y) represents the center of the box relative to the cell, and (w, h) represent the width and height of the box relative to the whole image. The confidence score indicates the likelihood of the box containing an object and how accurate the box is.

- **Class Prediction:** YOLO also predicts the probability of each class for each bounding box. This allows YOLO to detect multiple objects in a single grid cell and classify them simultaneously.

- **Non-Max Suppression:** After predicting bounding boxes and class probabilities, YOLO applies non-maximum suppression to eliminate redundant and overlapping bounding boxes, keeping only the most confident ones.

- **Output:** The final output of YOLO is a set of bounding boxes along with their associated class labels and confidence scores, which represent the detected objects in the input image.

YOLO has undergone several iterations since its original release, with improvements in accuracy, speed, and robustness. Variants like YOLOv2, YOLOv3, and YOLOv4 have been introduced, each with enhancements over the previous versions. These improvements include better feature extraction, more advanced network architectures, and techniques to handle small objects and improve generalization. YOLO remains one of the most widely used object detection systems in both academia and industry due to its speed and accuracy.

B) Transfer Learning:

Transfer learning is a machine learning technique where a model trained on one task is repurposed or fine-tuned for a different but related task. Instead of starting the learning process from scratch, transfer learning leverages the knowledge gained from solving one problem and applies it to a different, but often related, problem. This approach is particularly useful when the target task has a small amount of labeled data available, as it allows the model to benefit from the large amount of data used to train the source task.

Here's an overview of how transfer learning typically works:

- **Pre-trained Model:** Transfer learning starts with a pre-trained model that has been trained on a large dataset for a related task. This pre-trained model has already learned useful features and representations from the data.

- **Task Adaptation:** The pre-trained model is then adapted or fine-tuned to the new target task using a smaller dataset. The final layers of the pre-trained model are often replaced or retrained, while the earlier layers are frozen or fine-tuned with a lower learning rate. This allows the model to retain the learned representations from the source task while adapting to the specifics of the target task.

- **Training:** The adapted model is then trained on the target task using the new dataset. The training process may involve adjusting the model's parameters to minimize a task-specific loss function, typically using techniques such as gradient descent.

- **Evaluation:** Finally, the performance of the adapted model is evaluated on a separate validation or test dataset to assess its effectiveness for the target task.

Transfer learning offers several advantages:

- **Reduced Data Requirement:** Transfer learning can be effective even when there is limited labeled data available for the target task, as it leverages knowledge from the source task.

- **Faster Training:** By starting with a pre-trained model, transfer learning can significantly reduce the time and computational resources required to train a model for the target task.

- **Improved Generalization:** Transfer learning can lead to models that generalize better to new tasks or datasets, as they have learned rich feature representations from the source task.

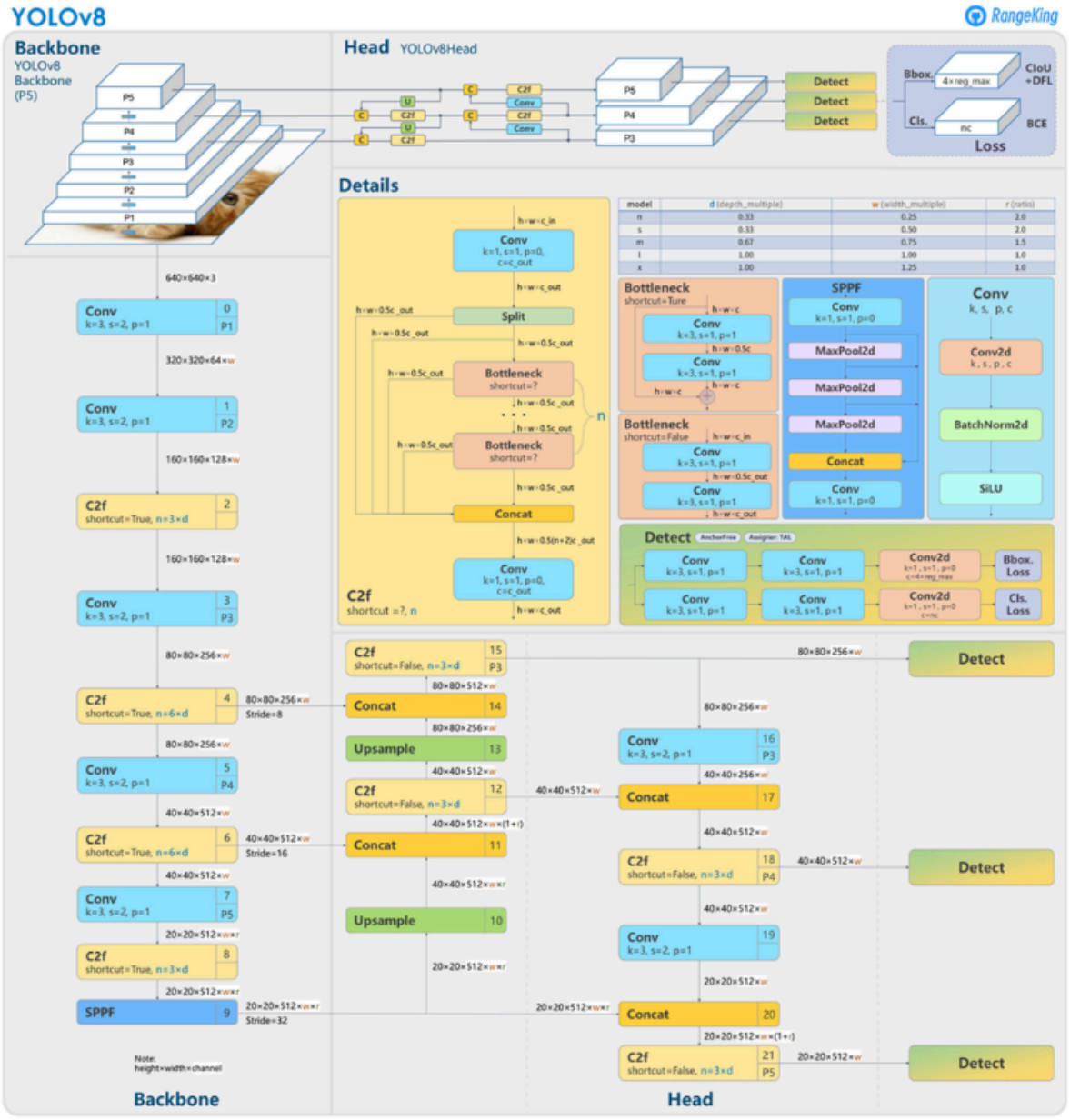
- **Domain Adaptation:** Transfer learning can also be useful for adapting models trained on the data from one domain to perform well on data from a different domain.

Transfer learning is widely used across various domains, including computer vision, natural language processing, and speech recognition. Pre-trained models like BERT, ResNet, and GPT have become popular choices for transfer learning in their respective domains, serving as starting points for building models for a wide range of tasks.

3.2 Pre-trained Model YOLO-V8:

YOLO (You Only Look Once) object detection and image segmentation model developed by Ultralytics. YOLO is a popular real-time object detection system that has seen several versions, each improving upon the previous one. YOLOv8, like its predecessors, is known for its speed and accuracy in object detection tasks. The fundamental idea behind YOLO is to divide an image into a grid and simultaneously predict bounding boxes and class probabilities for each grid cell. This differs from traditional object detection methods that involve sliding a window or region proposal network over an image. It can be trained on large datasets and is capable of running on a variety of hardware platforms, from CPUs to GPUs.

Ultralytics YOLOv8 is the latest version of



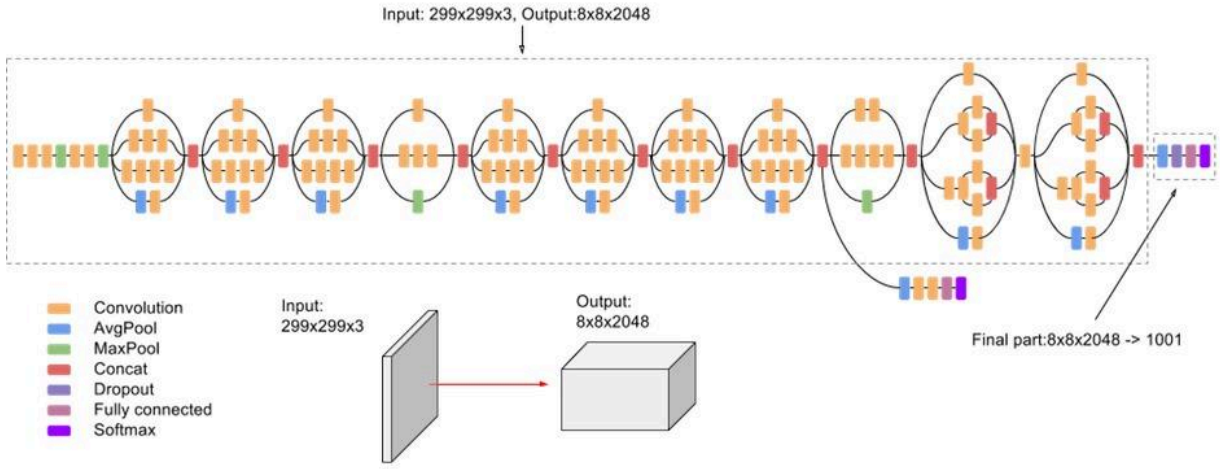
(3.2.1 - YOLOv8 Architecture)

YOLOv8 stands out from its predecessors primarily due to its utilization of an anchor-free model. Unlike previous versions, YOLOv8 predicts the center of an object directly instead of calculating the offset from a predefined anchor box. This anchor-free approach results in fewer box predictions, thereby accelerating the Non-Maximum Suppression (NMS) process. NMS is a crucial post-processing step that sifts through candidate detections after inference, and the reduction in box predictions helps streamline this complex procedure.

3.3 Pre-trained Model Inception-V3:

Inception v3 is a convolutional neural network architecture that was introduced by Google in 2015. It is a deep learning model designed primarily for image classification tasks. Inception v3 builds upon its predecessors, incorporating advancements in neural network design to improve both accuracy and efficiency.

One of the notable features of Inception v3 is its use of inception modules, which are composed of parallel convolutional layers of different filter sizes. These modules allow the network to capture features at multiple scales, enhancing its ability to recognize complex patterns in images. Additionally, Inception v3 incorporates other techniques such as batch normalization and factorized convolutions to accelerate training and improve generalization.



(3.3.1 - Inception V3 Architecture)

Inception v3 has been pre-trained on large-scale image datasets such as ImageNet, where it achieved state-of-the-art performance in image classification tasks. Its pre-trained weights can be used as a starting point for transfer learning, making it a popular choice for various computer vision applications. Overall, Inception v3 represents a significant advancement in deep learning architecture, combining accuracy, efficiency, and versatility for image understanding tasks.

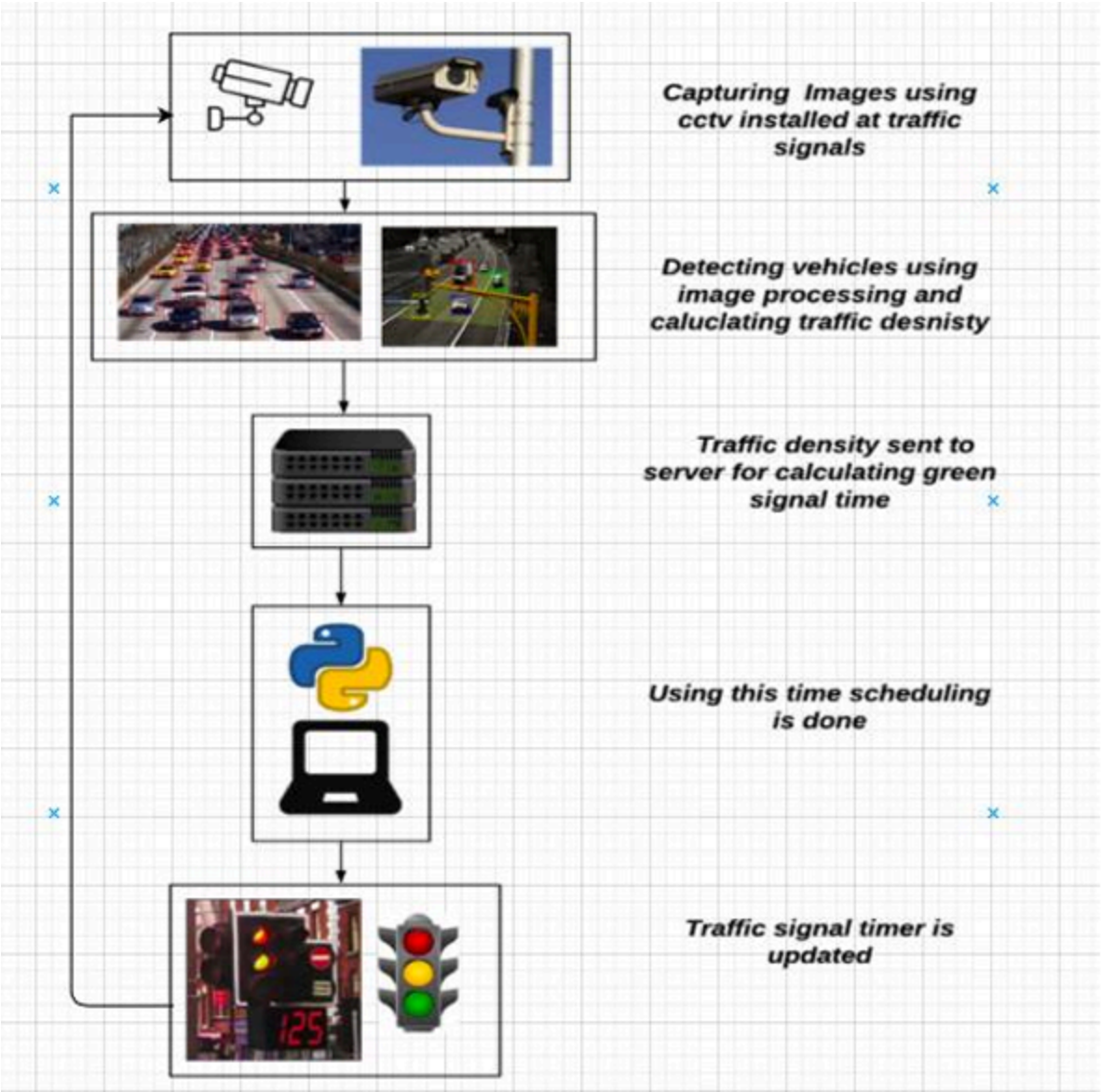
CHAPTER 4
IMPLEMENTATION

4.1 Dataset Description:

The dataset is made up of a number of images showing different kinds of cars, bikes, trucks, emergency vehicles that were taken in different urban and suburban environments. Annotations on each image designate whether the vehicle is classified as an emergency vehicle or not, which makes it easier to do classification jobs that need to separate typical commuter vehicles from emergency response vehicles like ambulances, police cars, and fire trucks. The collection also offers useful information for counting and estimating vehicle densities, which is necessary for dynamically modifying signal timings and efficiently handling traffic congestion. This information makes it possible to construct intelligent systems that can optimize traffic flow and minimize congestion by assessing the density and count of vehicles in various scenarios and making real-time adjustments to traffic management methods and signal timing.

4.2 Workflow:

Our proposed system takes an image from the CCTV cameras at traffic junctions as input for real time traffic density calculation using image processing and object detection. This system can be broken down into 3 modules: Vehicle Detection module, Signal Switching Algorithm, and Simulation module. As shown in the figure below, this image is passed on to the vehicle detection algorithm, which uses YOLO. The number of vehicles of each class, such as car, bike, bus, and truck, is detected, which is to calculate the density of traffic. The signal switching algorithm uses this density, among some other factors, to set the green signal timer for each lane. The red signal times are updated accordingly. The green signal time is restricted to a maximum and minimum value in order to avoid starvation of a particular lane. A simulation is also developed to demonstrate the system's effectiveness and compare it with the existing static system.



(4.2.1 - Proposed Workflow of our system)

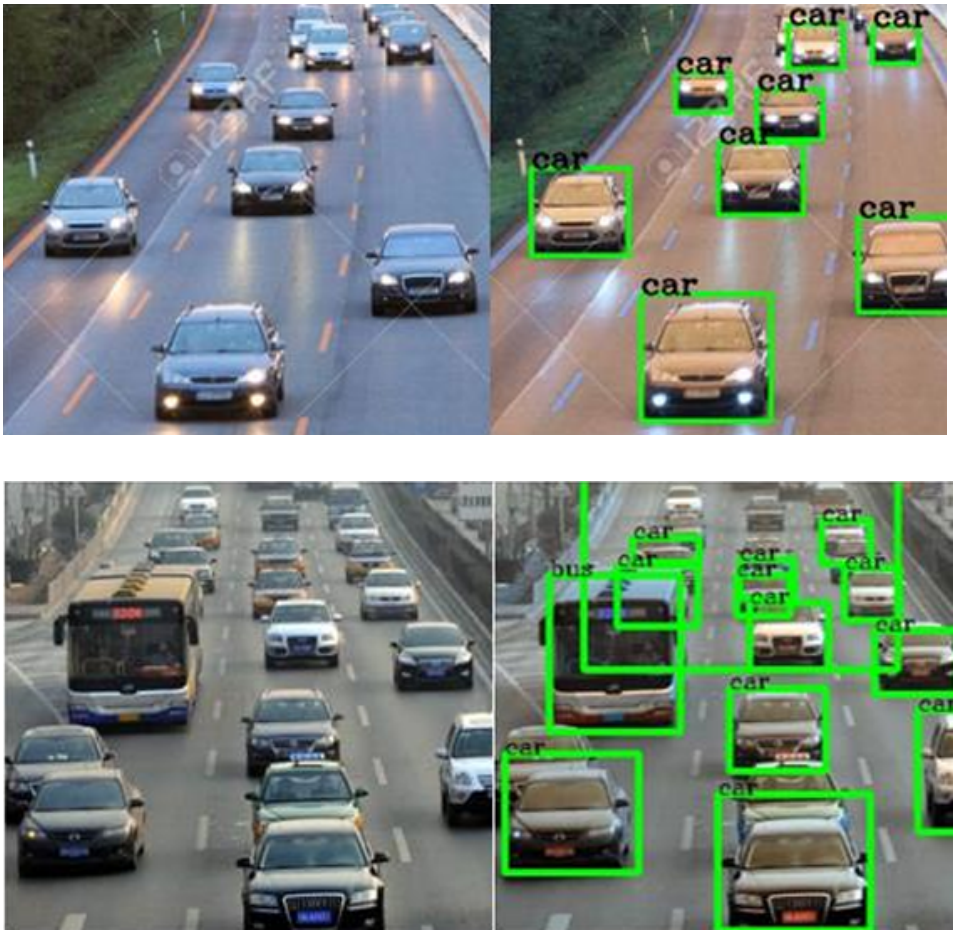
4.3 Vehicle Detection:

The proposed approach employs YOLO, or "You only look once," for vehicle detection, which offers the required processing speed and accuracy. For the purpose of vehicle detection, a special YOLO model was trained, and it can identify several classes of vehicles, including cars, bikes, heavy vehicles (trucks and buses) as well as rickshaws.
Using the graphical image annotation application LabelIMG, images were scraped from Google and manually labeled by using LabelIMG to create the dataset for the model's training.

Subsequently, pre-trained weights acquired from the YOLO website were used to train the model. The.cfg file that was utilized for training was modified to conform to our model's specifications. By adjusting the 'classes' variable, the number of output neurons in the final layer was adjusted to match the number of classes the model is expected to identify. This equated to four in our system: car, bike, bus/truck, and rickshaw. It is also necessary to adjust the number of filters using the formula 5*(5+number of classes), which in our case equals 45. The model was trained until the loss was greatly reduced and no longer appeared to be decreasing after these configuration adjustments.
This marked the end of the training, and the weights were now updated according to our requirements.

The OpenCV library was then used to detect vehicles using these weights that had been imported into the code. The lowest level of confidence needed for a successful detection is determined by a threshold. Following model loading and image feeding, the model outputs results in JSON format, or key-value pairs, where labels are keys and their corresponding coordinates and confidence are values. Once more, using the labels and coordinates that are provided, OpenCV may be used to draw bounding boxes on the images provided.

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



(4.3.1 - Vehicle Detection using Bounding Boxes)

4.4 Calculating Vehicle Count:

Let:

- N be the total number of detected bounding boxes representing vehicles.
- T be the confidence threshold for successful detection.

The formula for counting the number of vehicles is defined as follows:

$$\text{Number of Vehicles} = \sum_{i=1}^{N} 1\{C_i \geq T\}$$

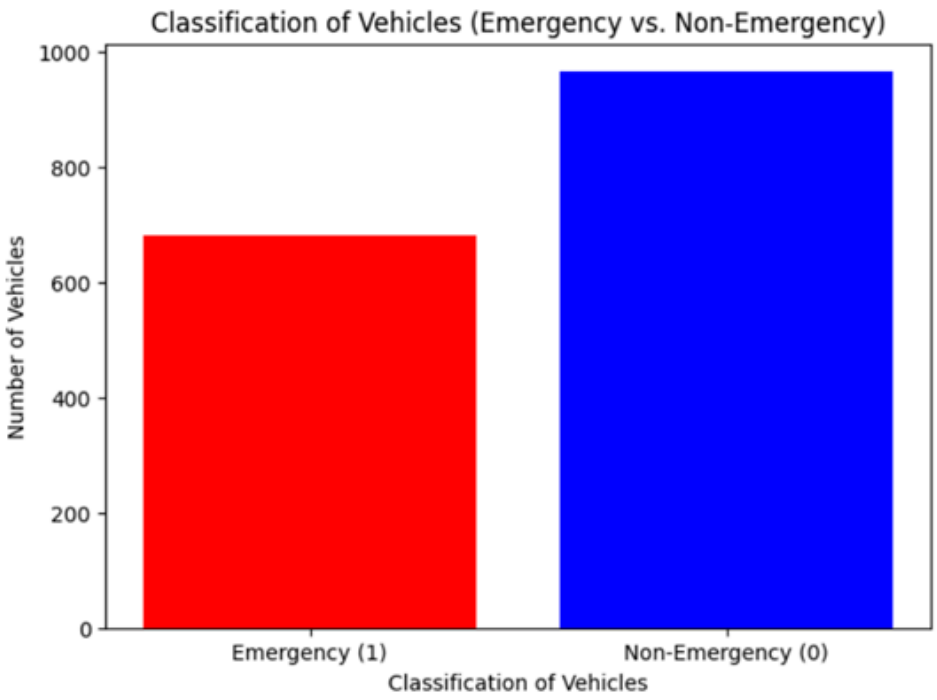
Where:

- C_i represents the confidence score associated with the ith detected bounding box.
- The indicator function $1\{C_i \geq T\}$ evaluates to 1 if the confidence score of the ith detected bounding box is greater than or equal to the threshold T, indicating a successful detection. Otherwise, it evaluates to 0.

4.5 Vehicle Classification:

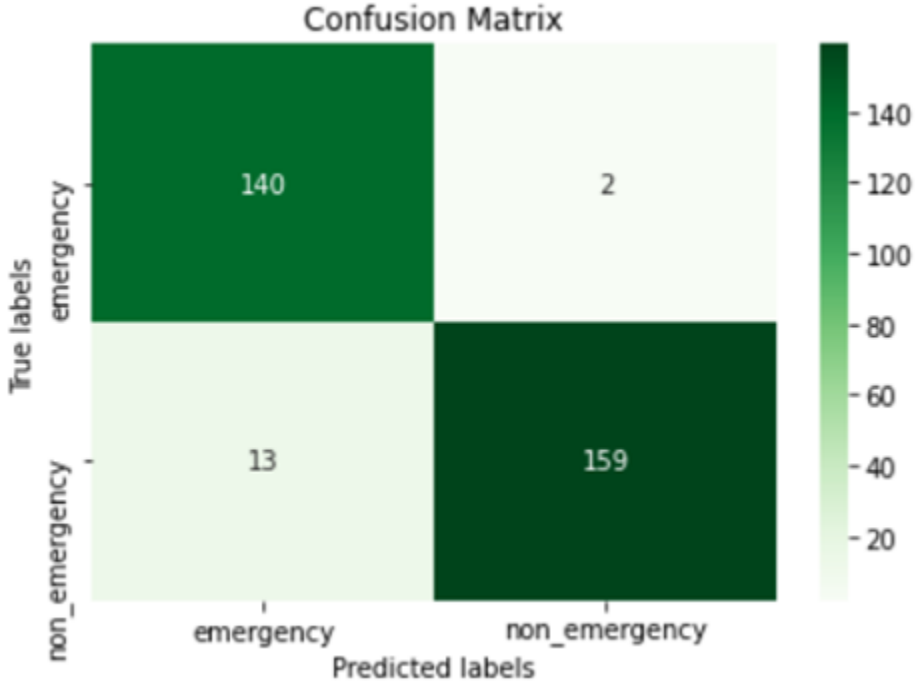
Based on the characteristics and features shown in the images, vehicles are classified into emergency and non-emergency classes. Deep learning techniques are used to achieve this classification, mostly with the help of a pre-trained Inception v3 model. A collection of vehicle images, each with an emergency or non-emergency label applied, is used to train the algorithm. Through training, the model gains the ability to identify patterns and characteristics common to both non-emergency and emergency vehicles, including cars, bikes, buses, and rickshaws, as well as emergency vehicles like ambulances, police cars, and fire trucks.

After being trained, the model can reliably classify newly captured images of vehicles into the relevant category, offering insightful information for a range of uses including urban planning, traffic management, and emergency response optimization.



(4.5.1 - Classification of Vehicles into Emergency & Non-emergency)

Below is the utilization of confusion_matrix function from sklearn.metrics to calculate the confusion matrix based on the true and predicted labels.Then usage of seaborn is done in order to visualize the confusion matrix as a heatmap, annotating each cell with the count of observations.



(4.5.2 - Confusion Matrix)

4.6 Signal Timing Adjustment:

The vehicle detecting module's traffic density is utilized by the Signal Switching Algorithm to set the green signal timer and update the red signal timers of other signals. In accordance with the timers, it also alternates between the signals cyclically. As mentioned in the previous section, the algorithm's input is the data on the vehicles that were found using the detection module. This is in JSON format, where the values are the coordinates and confidence, and the label of the object detected is the key. Following that, this data is analyzed to determine the number of vehicles there are overall in each class.
After this, the green signal time for the signal is calculated and assigned to it, and the red signal times of other signals are adjusted accordingly. The algorithm can be scaled up or down to any number of signals at an intersection.

The following factors were considered while developing the algorithm:
1. The processing time of the algorithm to calculate traffic density and then the green light duration – this decides at what time the image needs to be acquired.
2. Number of lanes.
3. Total count of vehicles of each class like cars, trucks, motorcycles, etc.
4. Traffic density calculated using the above factors.
5. Time added due to lag each vehicle suffers during start-up and the non-linear increase in lag suffered by the vehicles which are at the back.
6. The average speed of each class of vehicle when the green light starts i.e. the average time required to cross the signal by each class of vehicle.
7. The minimum and maximum time limit for the green light duration - to prevent starvation

Working of the algorithm:

When the algorithm is first run, the default time is set for the first signal of the first cycle and the times for all other signals of the first cycle and all signals of the subsequent cycles are set by the algorithm. A separate thread is started which handles the detection of vehicles for each direction and the main thread handles the timer of the current signal. When the green light timer of the current signal (or the red light timer of the next green signal) reaches 0 seconds, the detection threads take the snapshot of the next direction. The result is then parsed and the timer of the next green signal is set. All this happens in the background while the main thread is counting down the timer of the current green signal. This allows the assignment of the timer to be seamless and hence prevents any lag. Once the green timer of the current signal becomes zero, the next signal becomes green for the amount of time set by the algorithm.

The image is captured when the time of the signal that is to turn green next is 0 seconds. This gives the system a total of 5 seconds (equal to value of yellow signal timer) to process the image, to detect the number of vehicles of each class present in the image, calculate the green signal time, and accordingly set the times of this signal as well as the red signal time of the next signal. To find the optimum green signal time based on the number of vehicles of each class at a signal, the average speeds of vehicles at startup and their acceleration times were used, from which an estimate of the average time each class of vehicle takes to cross an intersection was found. The green signal time is then calculated using the formula below.

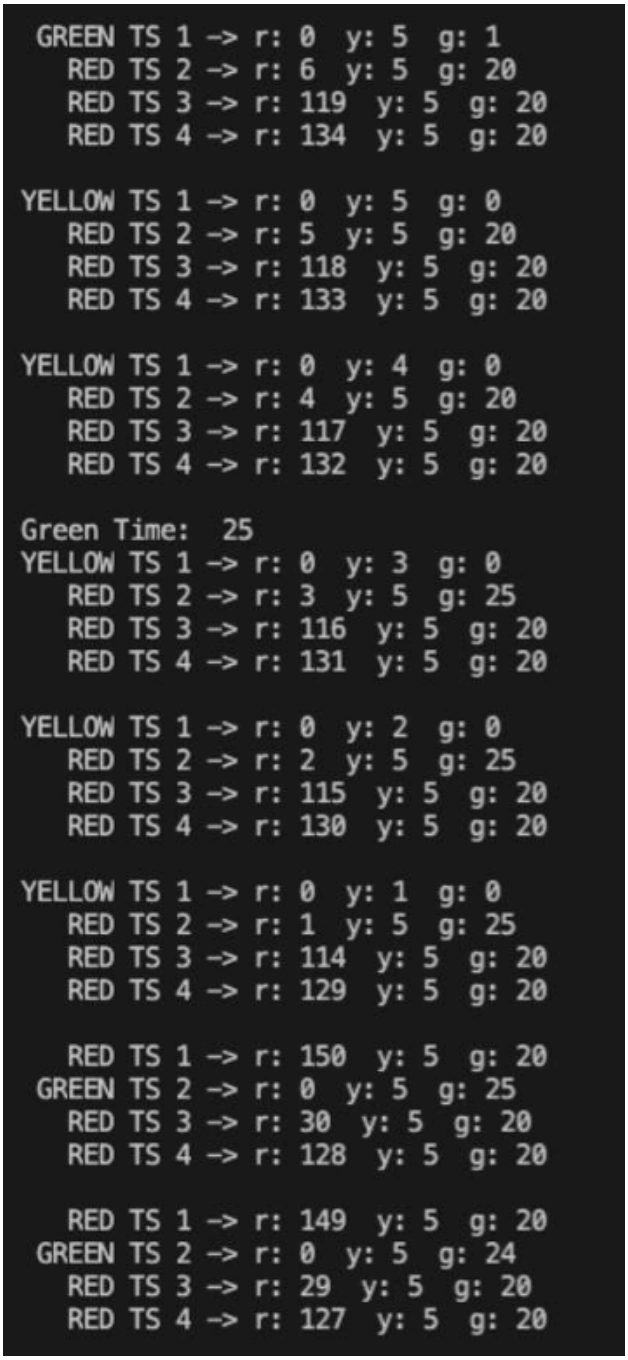
$$GST = \frac{\sum_{vehicleClass} (NoOfVehicles_{vehicleClass} * AverageTime_{vehicleClass})}{(NoOfLanes + 1)}$$

where:

● GST is green signal time
● noOfVehiclesOfClass is the number of vehicles of each class of vehicle at the signal as detected by the vehicle detection module,
● averageTimeOfClass is the average time the vehicles of that class take to cross an intersection, and
● noOfLanes is the number of lanes at the intersection.

The average time each class of vehicle takes to cross an intersection can be set according to the location, i.e., region-wise, city-wise, locality-wise, or even intersection-wise based on the characteristics of the intersection, to make traffic management more effective. Data from the respective transport authorities can be analyzed for this.

The signals switch in a cyclic fashion and **not** according to the densest direction first. This is in accordance with the current system where the signals turn green one after the other in a fixed pattern and does not need the people to alter their ways or cause any confusion. The order of signals is also the same as the current system, and the yellow signals have been accounted for as well.
Order of signals: Red → Green → Yellow → Red



(4.6.1 - Output of Signal Switching Algorithm)

After a complete cycle, again, TS 1 changes from green to yellow. As the yellow timer counts down, the results of the vehicle detection algorithm are processed and a green time of 25 seconds is calculated for TS 2. As this value is more than the minimum green time and less than maximum green time, the green signal time of TS 2 is set to 25 seconds. When the yellow time of TS 1 reaches 0, TS 1 turns red and TS 2 turns green, and the countdown continues. The red signal time of TS 3 is also updated as the sum of yellow and green times of TS 2 which is 5+25=30.

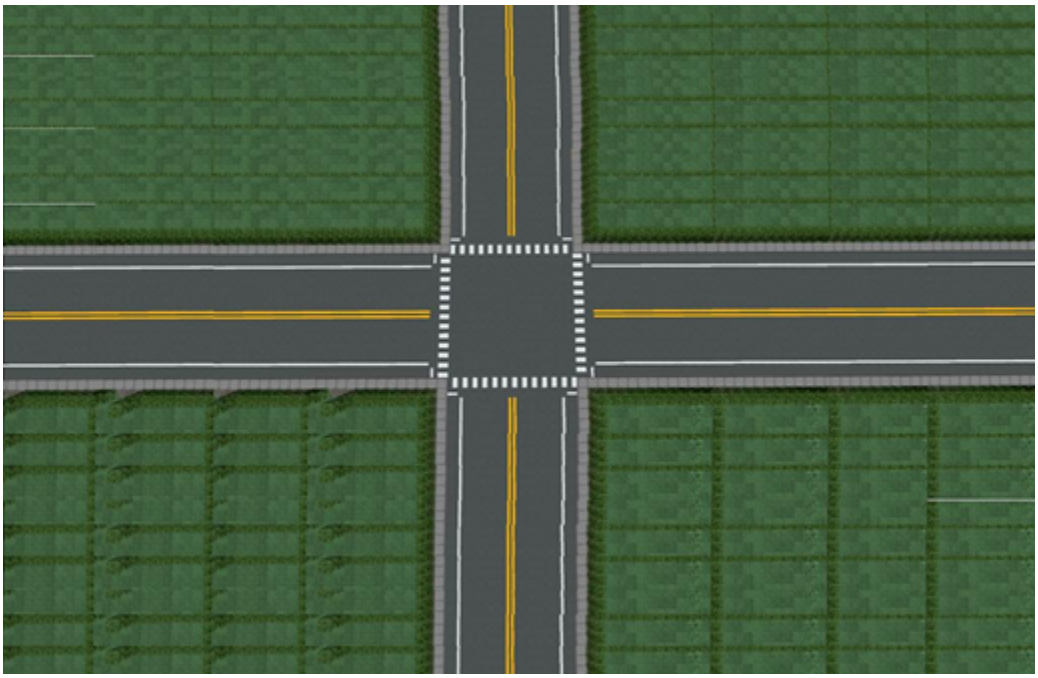
Chapter 5
RESULT ANALYSIS

5.1 Simulation:

A simulation was developed from scratch using Pygame to simulate real-life traffic. It assists in visualizing the system and comparing it with the existing static system. It contains a 4-way intersection with 4 traffic signals. Each signal has a timer on top of it, which shows the time remaining for the signal to switch from green to yellow, yellow to red, or red to green. Each signal also has the number of vehicles that have crossed the intersection displayed beside it. Vehicles such as cars, bikes, buses, trucks, and rickshaws come in from all directions. In order to make the simulation more realistic, some of the vehicles in the rightmost lane turn to cross the intersection. Whether a vehicle will turn or not is also set using random numbers when the vehicle is generated. It also contains a timer that displays the time elapsed since the start of the simulation.

Key steps in development of simulation:

1. Took an image of a 4-way intersection as background.



(5.1.1 - 4-way intersection of lanes)

2. Gather the top-view images of car, bike, bus, truck, and rickshaw.

3. Resize them.

(5.1.2 - Categories of Vehicles)

4. Rotated them for display along different directions.

5. Gathered images of traffic signals - red, yellow, and green.

6. Code: For rendering the appropriate image of the signal depending on whether it is red, green, or yellow.

7. Code: For displaying the current signal time i.e. the time left for a green signal to turn yellow or a

red signal to turn green or a yellow signal to turn red. The green time of the signals is set according to the algorithm, by taking into consideration the number of vehicles at the signal. The red signal times of the other signals are updated accordingly.

8. Generation of vehicles according to direction, lane, vehicle class, and whether it will turn or not all set by random variables. Distribution of vehicles among the 4 directions can be controlled. A new vehicle is generated and added to the simulation after every 0.75 seconds.

9. Code: For how the vehicles move, each class of vehicle has different speed, there is a gap between 2 vehicles, if a car is following a bus, then its speed is reduced so that id does not crash into the bus.

10. Code: For how they react to traffic signals i.e. stop for yellow and red, move for green. If they have passed the stop line, then continue to move if the signal turns yellow.

11. Code: For displaying the number of vehicles that have crossed the signal.

12. Code: For displaying the time elapsed since the start of the simulation.

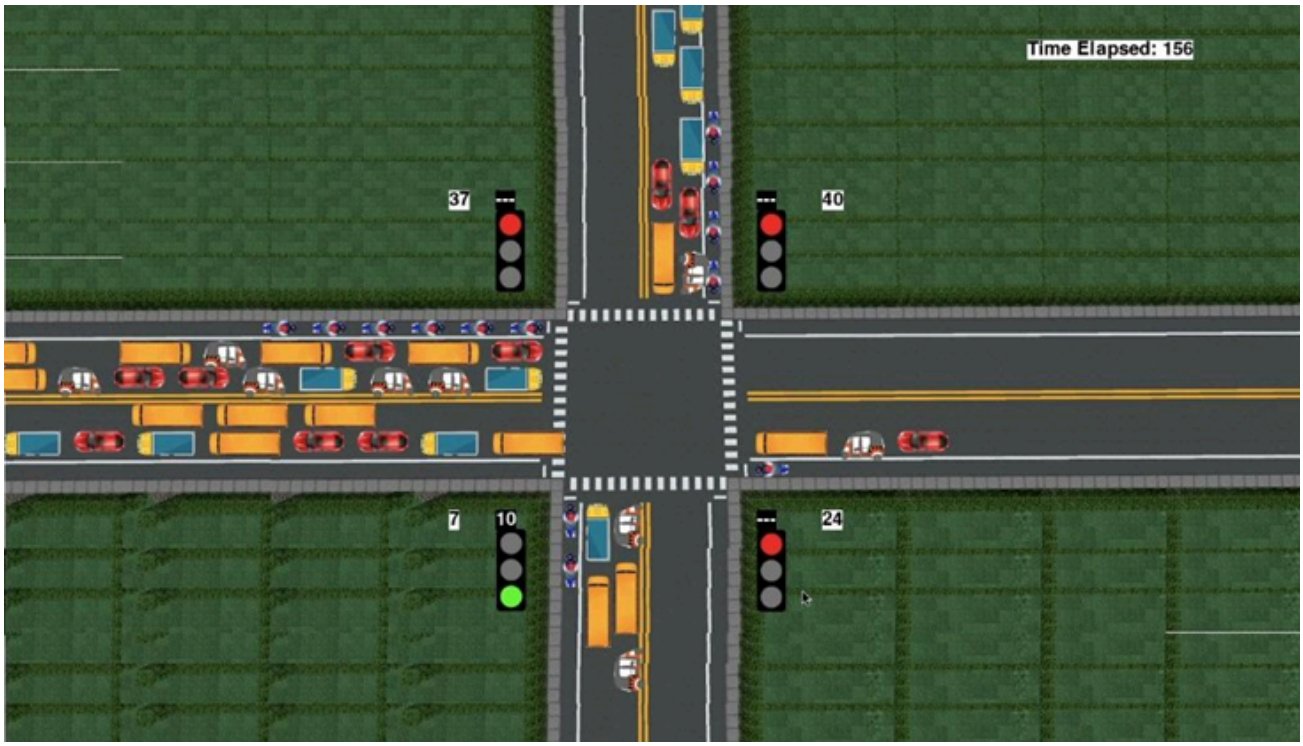
13. Code: For updating the time elapsed as simulation progresses and exiting when the time elapsed equals the desired simulation time, then printing the data that will be used for comparison and analysis.

14. To make the simulation closer to reality, even though there are just 2 lanes in the image, we add another lane to the left of this which has only bikes, which is generally the case in many cities.

15. Vehicles turning and crossing the intersection in the simulation to make it more realistic.

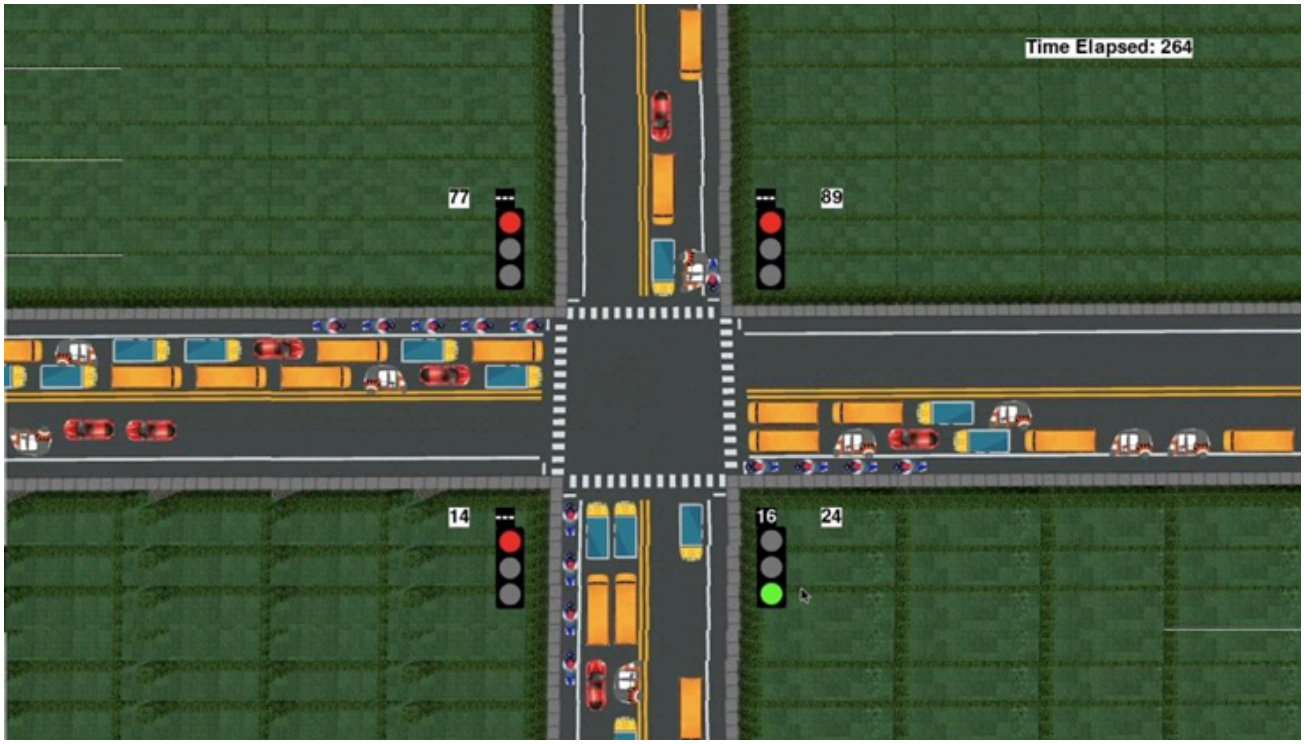
5.2 Results Analysis:

Following are some images of the final simulation:



(5.2.1 - Case 1)

Simulation showing green time of signal for vehicles moving up set to 10 seconds according to 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 this case.



(5.2.2 - Case 2)

Simulation showing green time of signal for vehicles moving left set to 24 seconds according to the vehicles in that direction.

CHAPTER 6
CONCLUSION

6.1 Summary:

The traffic optimization project aims to revolutionize urban transportation by implementing a dynamic and intelligent system for managing traffic flow at intersections. By leveraging advanced technologies such as image recognition, real-time vehicle tracking, and smart signal control algorithms, the project seeks to address the challenges of congestion, safety, and efficiency in urban mobility.

Through the deployment of this innovative system, the project anticipates significant improvements in traffic flow, safety, and resource utilization. By prioritizing emergency vehicles, optimizing signal timings, and utilizing real-time data for decision-making, the project aims to create a more seamless and sustainable transportation network.

Furthermore, the project emphasizes scalability and adaptability, envisioning a solution that can be implemented across various urban settings and seamlessly integrated with existing infrastructure. By harnessing the power of data-driven insights and intelligent automation, the project sets out to not only alleviate current traffic woes but also pave the way for a more connected, efficient, and resilient urban transportation ecosystem.

In conclusion, the traffic optimization project represents a bold step towards transforming the way we move within cities, promising tangible benefits in terms of reduced congestion, enhanced safety, and improved overall quality of life for urban residents and commuters.

6.2 Future Scope:

1. Secure Overall Traffic System:
Enhance security measures to prevent signal hacking, including the implementation of robust encryption protocols, intrusion detection systems, and regular security audits to safeguard against unauthorized access and tampering.

2. GreenWave Generation:
Implement predictive algorithms to optimize traffic signal timings, ensuring drivers receive green signals at each intersection based on real-time traffic flow predictions.

3. Automatic detection of speed limit Violation:
Implement automatic detection of speed limit violations to calculate motorists' speeds, issue warnings via audio/video interfaces, and bill penalties monthly to vehicle owners.

4. Preventing Road Accidents:
Implementing a system which can capture the images of accidents through CCTV cameras and send signals to nearby hospitals so that the injured people can be provided with necessary medical assistance & their lives can be saved.

CHAPTER 7

REFERENCES

1. Ali, H., & Hayawi, D. M. J. (2024). An Adaptive Traffic Light Control System Based On Artificial Intelligence. *Journal of Education for Pure Science-University of Thi-Qar*, *14*(1).
2. Kunekar, P., Narule, Y., Mahajan, R., Mandlapure, S., Mehendale, E., & Meshram, Y. (2024). Traffic Management System Using YOLO Algorithm. *Engineering Proceedings*, *59*(1), 210.
3. Alharbi, A., Halikias, G., Sen, A. A. A., & Yamin, M. (2021). A framework for dynamic smart traffic light management system. International Journal of Information Technology, 13, 1769-1776.
4. Meng, B. C. C., Damanhuri, N. S., & Othman, N. A. (2021, February). Smart traffic light control system using image processing. In IOP Conference Series: Materials Science and Engineering (Vol. 1088, No. 1, p. 012021). IOP Publishing.
5. Ariffin, W. N. S. F. W., Keat, C. S., Prasath, T., Suriyan, L., Nore, N. A. M., Hashim, N. B. M., & Zain, A. S. M. (2021, May). Real-time dynamic traffic light control system with emergency vehicle priority. In *Journal of Physics: Conference Series* (Vol. 1878, No. 1, p. 012063). IOP Publishing.
6. Meena, G., Sharma, D., & Mahrishi, M. (2020, February). Traffic prediction for intelligent transportation system using machine learning. In 2020 3rd International Conference on Emerging Technologies in Computer Engineering: Machine Learning and Internet of Things (ICETCE) (pp. 145-148). IEEE.
7. Samal, S. R., Kumar, P. G., Santhosh, J. C., & Santhakumar, M. (2020, December). Analysis of traffic congestion impacts of urban road network under Indian condition. In IOP conference series: materials science and engineering (Vol. 1006, No. 1, p. 012002). IOP Publishing.
8. De Oliveira, L. F. P., Manera, L. T., & Da Luz, P. D. G. (2020). Development of a smart traffic light control system with real-time monitoring. IEEE Internet of Things Journal, 8(5), 3384-3393.
9. Lingani, G. M., Rawat, D. B., & Garuba, M. (2019, January). Smart traffic management system using deep learning for smart city applications. In 2019 IEEE 9th annual computing and communication workshop and conference (CCWC) (pp. 0101-0106). IEEE.
10. Natafgi, M. B., Osman, M., Haidar, A. S., & Hamandi, L. (2018, November). Smart traffic light system using machine learning. In 2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET) (pp. 1-6). IEEE.