DAYANANDA SAGAR UNIVERSITY

KUDLU GATE, BANGALORE – 560068

Bachelor of Technology
in
COMPUTER SCIENCE AND ENGINEERING

Major Project Phase-II Report

SMART TRAFFIC-SIGNAL MANAGEMENT SYSTEM

By

Ravi Ranjan- ENG19CS0251.
Ravikeerthi Shetty- ENG19CS0252.
Shakti Sidheswar Mishra- ENG19CS0291.
Sharath H N- ENG19CS0293.

Under the supervision of

Prof. Pooja Shree H R
Assistant Professor and Computer Science & Engineering

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY,
BANGALORE

(2022-2023)

DAYANANDA SAGAR UNIVERSITY

School of Engineering
Department of Computer Science & Engineering
Kudlu Gate, Bangalore – 560068
Karnataka, India

CERTIFICATE

This is to certify that the Phase-II project work titled **"SMART TRAFFIC-SIGNAL MANAGEMENT SYSTEM"** is carried out by **Ravi Ranjan (ENG19CS0251), Ravikeerthi Shetty (ENG19CS0252), Shakti Sideswar Mishra (ENG19CS0291), Sharath H N (ENG19CS0293),** bonafide students of Bachelor of Technology in Computer Science and Engineering at the School of Engineering, Dayananda Sagar University, Bangalore in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Engineering, during the year **2022-2023**.

Prof. Pooja Shree H R	**Dr. Girisha G S**	**Dr. Udaya Kumar Reddy K R**
Assistant Professor Dept. of CS&E, School of Engineering Dayananda Sagar University	Chairman CSE School of Engineering Dayananda Sagar University	Dean School of Engineering Dayananda Sagar University
Date:	Date:	Date:

Name of the Examiner **Signature of Examiner**

1.

2

DECLARATION

We, **Shakti Sidheswar Mishra(ENG19CS0291), Ravi Ranjan(ENG19CS0251), Ravikeerthi Shetty(ENG19CS0252), Sharath H N(ENG19CS0293),** are students of seventh semester B. Tech in **Computer Science and Engineering**, at School of Engineering, **Dayananda Sagar University**, hereby declare that the Major Project Stage-II titled **"Smart Traffic Signal Management System"** has been carried out by us and submitted in partial fulfilment for the award of degree in **Bachelor of Technology in Computer Science and Engineering** during the academic year **2022-2023**.

Student **Signature**

Name1: Ravi Ranjan (ENG19CS0251)

Name2: Ravikeerthi Shetty (ENG19CS0252)

Name3: Shakti Sidheswar Mishra (ENG19CS0291)

Name4: Sharath H N(ENG19CS0293)

Place: Bangalore

Date:

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to School of Engineering & Technology, Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor's degree in this institution.

We would like to thank ***Dr. Udaya Kumar Reddy K R, Dean, School of Engineering & Technology, Dayananda Sagar University*** *for his constant encouragement and expert advice.*

It is a matter of immense pleasure to express our sincere thanks to ***Dr. Girisha G S, Department Chairman****,* ***Computer Science and Engineering****,* ***Dayananda Sagar University,*** *for providing right academic guidance that made our task possible.*

We would like to thank our guide ***Pooja Shree H R, Assistant Professor****,* ***Dept. of Computer Science and Engineering****,* ***Dayananda Sagar University****, for sparing his/her valuable time to extend help in every step of our project work, which paved the way for smooth progress and fruitful culmination of the project.*

We would like to thank our ***Project Coordinator Dr. Meenakshi Malhotra*** *and* ***Pramod Nayak*** *and all the staff members of Computer Science and Engineering for their support.*

We are also grateful to our family and friends who provided us with every requirement throughout the course.

We would like to thank one and all who directly or indirectly helped us in the Project work.

TABLE OF CONTENTS

Page

NOMENCLATURE USED

YOLO	You Only Look Once
OpenCV	Open-Source Computer Vision Library
RFID	Radio Frequency Identification
OCR	Optical Character Recognition

List of Figures

ABSTRACT

Traffic congestion is a major problem in many cities of India along with other countries. Failure of signals, poor law enforcement and bad traffic management has lead to traffic congestion. **One of the major problems with Indian cities is that the existing infrastructure cannot be expanded more, and thus the only option available is better management of the traffic.** Traffic congestion has a negative impact on economy, the environment and the overall quality of life. Hence it is high time to effectively manage the traffic congestion problem. There are various methods available for traffic management such as video data analysis, infrared sensors, inductive loop detection, wireless sensor network, etc. All these methods are effective methods of smart traffic management. But the problem with these systems is that the installation time, the cost incurred for the installation and maintenance of the system is very high. **This new technology which will require less time for installation with lesser costs as compared to other methods of traffic congestion management. Use of this new technology will lead to reduced traffic congestion**. Bottlenecks will be detected early and hence early preventive measures can be taken thus saving time and money of the driver.

CHAPTER 1 INTRODUCTION

1.1 INTRODUCTION

In this project, we will develop a model which will able to avoid congestion in traffic and smoother flow of traffic. The model is based on avoiding congestion in traffic and smoother the flow of traffic, which will result in decrease in the accident rates per year and indirectly will also reduce the traffic violation. Congestion detection also enables adaptive control, which causes dynamic adjustments to systems including traffic lights, on-ramp signaling, and bus rapid transit lanes.

We used image processing technique caused by using this we can easily calculated density of traffic present on road. The system will detect vehicles through images instead of using electronic sensors embedded in the pavement. A camera will be installed alongside the traffic light. It will capture image sequences. Image processing is a better technique to control the state change of the traffic light. It will need machine Learning Algorithm to predict the time when all vehicle will be identified by help of image processing. We will use a library named OpenCV which provides a real-time optimized Computer Vision library, tools, and hardware and an algorithm called YOLO is an algorithm that uses neural networks to provide real-time object detection therefore mainly used in the image processing to handle the detection of the vehicles.

Lack of efficient Traffic management to replace the faulty and inefficient manual system leading to congestion and for critical emergency vehicles, which leads to lots of wastage of time. The model is based on avoiding congestion in traffic and smoother the flow of traffic, which will result in decrease in the accident rates per year and indirectly will also reduce the traffic violation. We are using image processing techniques for calculating the density of traffic present on road. It will need machine Learning Algorithm for prediction of the time when all vehicle will be identified by help of image processing.

1.2 OBJECTIVE

The objective is to control traffic congestion in an optimal manner and optimize the duration of green and red lights at intersections. Rather than using a fixed timer for green and red signals, timing should depend on vehicle density. If traffic is heavy in one direction, the green signal should be on for a longer duration. This should eliminate inefficiencies at intersections and reduce the costs of commuting and pollution. Most of the traffic management systems available today are not automated and prone to human errors. Traffic monitoring in many countries is the biggest concern, due to overcrowding, increasing daily commuters, globalization, and shortened or narrow roads. Improper architecture in the signaling system and avoidance of traffic rules are affecting the traffic inflow and leading to congestion.

Intelligent automated systems are taking over traditional operational methods after the technology explosion. An intelligent traffic management system provides an advantage by offering safe public transportation

1.3 SCOPE

This helps to optimize the green signal times, and traffic is cleared at a much faster rate than a static system, thus reducing the unwanted delays, congestion, and waiting time, which in turn will reduce the fuel consumption and pollution. It will also help us to reduce the number of sensors that are using in larger amount in every traffic signal.

CHAPTER 2 PROBLEM DEFINITION

Traffic Jam is a major problem in many cities of India and across the other countries. Failure of signal systems, poor law enforcement and bad traffic management has started resulting in traffic congestion and many major and minor accidents. One of the major problems with Indian cities is that the existing systems cannot be expanded more due to houses, shops and other facilities, and thus the only option available is better management of the whole traffic system.

Traffic congestion has a negative impact on economy, the environment and the overall quality of life. Hence it is high time to effectively manage the traffic congestion problem. Due to increasing number of vehicles traffic jams are becoming a common scenario in the whole country as well as in the world. These frequent traffic jams at major junctions kill a lot of man hours. Thus, it creates a need for an efficient traffic management system.

In today's life we have to face many problems, one of which being traffic congestion and it's becoming more serious day after day. Conventional traffic system does not have proper monitoring system and often requires manual handing at traffic junction. This not only causes mental stress in passengers but also lot of fuel goes wasted due to delay at traffic junction. This requires development of a system to handle traffic in a smart way.

There are numerous methods which are available for traffic engagement such as video data analysis, infrared sensors, inductive loop detection, wireless sensor network, etc. All these management methods are effective methods of smart traffic management. But the problem with these systems is that the installation and execution time, the cost incurred for the installation and maintenance of the system is very high and expensive.

With the increasing number of vehicles in urban areas, many road networks are facing problems with the capacity drop of roads and the corresponding Level of Service. Many traffic-related issues occur because of traffic control systems on intersections that use

fixed signal timers. They repeat the same phase sequence and its duration with no changes. Increased demand for road capacity also increases the need for new solutions for traffic control that can be found in the field of Intelligent Transport Systems.

This new technology which will require less time for installation with lesser costs as compared to other methods of traffic congestion management. Use of this new technology will lead to reduced traffic congestion. Bottlenecks or errors will be detected early and hence early preventive measures can be taken thus saving time and money of the driver.

CHAPTER 3 LITERATURE REVIEW

1. Vandana Niranjan, Indu, Anam Firdous / Smart Density Based Traffic Light System published on 2020, International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) Amity University.

In this, traffic is sense using digital IR Sensors and IR Sensors detect vehicles further based on the signal reflected from them. Sensors placed adjacent to the road to control the traffic density by changing traffic signal appropriately. All IR Sensors are interfaced with Arduino Uno and it reads data from IR Sensors. Traffic Signal for the system is designed using LEDs and each signal consist two LEDs for each lane.

Using this system development at traffic junction we need not to worry about handing the traffic manually and also consumes less time as compared to the conventional traffic system. We harness solar power from solar panel and this is used to build prototype working model of smart traffic signal which automatically adjusts its timing based on traffic direction.

2. Dr. Vikram Bali, Ms. Sonali Mathur, Dr. Vishnu Sharma / Smart Traffic Management System using IoT Enabled Technology published on 2020, International Conference on Advances in Computing, Communication Control and Networking (ICACCCN).

The system gives emergency vehicles the benefit of green corridor and reach destination on time. The RFID reader scans the RFID tag applied on the ambulance and updates the upcoming traffic light to switch to green and displays a message to vehicles ahead of ambulance to provide a “Green Corridor” by shifting other lanes. This system comprises whole traffic system and functions for emergency situations in order to reach the destination without any delay.

The proposed system is based on the implementation of Infrared Technology, visual sensing, RFIDs/radar system. All these together can help in controlling the traffic in more efficient manner whenever any ambulance or some emergency vehicle comes closer. The

STMS eliminates the delay time faced by the emergency response vehicles on to the vehicle by displaying the arrival message of vehicle on LCD screen to the vehicles moving ahead and indicates them to shift to other lanes on the road.

3. Mr. V.S. Gadakh, Avhad Vaishali2, Dherange Rutika, Darade Sonali, Shinde Pooja / Smart Traffic Light Control System with Automatic vehicle Speed Breaker published on 2021, International Research Journal of Engineering and Technology (IRJET).

When signal timing changes by delay is provided with the help of microcontroller, barrier on zebra crossing will open or close to allow vehicles to pass. When the signal is red the interfaced barrier gate closes and a buzzer notifies the closing of gate, thereby blocking the traffic but when the signal is green the same barrier opens and allows a proper flow of vehicles to avoid traffic jam. We provided time delay for all signal to change and buzzer action by using PIC microcontroller. In front of the barrier gate a stop line is drawn and with the help of another IR sensor, the vehicle is tracked whenever it crosses the stop line. PIC microcontroller is used for signal timing change as well as street light control according to LDR.

For smart traffic light control they are using LDR. When signal timing changes by delay is provided with the help of microcontroller, barrier on zebra crossing will open or close to allow vehicles to pass. When the signal is red the interfaced barrier gate closes and a buzzer notifies the closing of gate, thereby blocking the traffic but when the signal is green the same barrier opens.

4. Mr. Sunil S. Sonawane, Bhagyashree Kenchannvar, Kajal Kumbhar, Pranjali Salunkhe published on 2022, Ijraset Journal for Research in Applied Science and Engineering technology.

This paper proposes a control system supported image processing including video processing within which traffic signals change accordingly the density of traffic and it will make use of Arduino UNO board for Traffic Lights, Emergency Vehicles and Barrier. A video camera and traffic lights are interfaced with Arduino UNO. The video is processed and Arduino enables the traffic lights to vary when required. Along with this,

barrier at zebra crossing and emergency vehicle passing are the best concept for today's smart city.

5. Islam Mohammad Albatish and Samy S. Abu Nasir/ Modelling & Controlling Traffic Light System published on 2019, International Conference on Promising electronics technology.

Edge Detection Technique was used in order to find the traffic density and PLC was in order to decrease heavy traffic. Intelligent light control system was also used. In Reference 2, The evaluated prototype was applicable, efficient and effective. The dynamic cycle time also reduced congestions. The evaluated prototype was applicable, efficient and effective. The dynamic cycle time also reduced congestions.

CHAPTER 4 PROJECT DESCRIPTION

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. As shown in figure, a reference 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 will be 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.

4.1 SYSTEM DESIGN

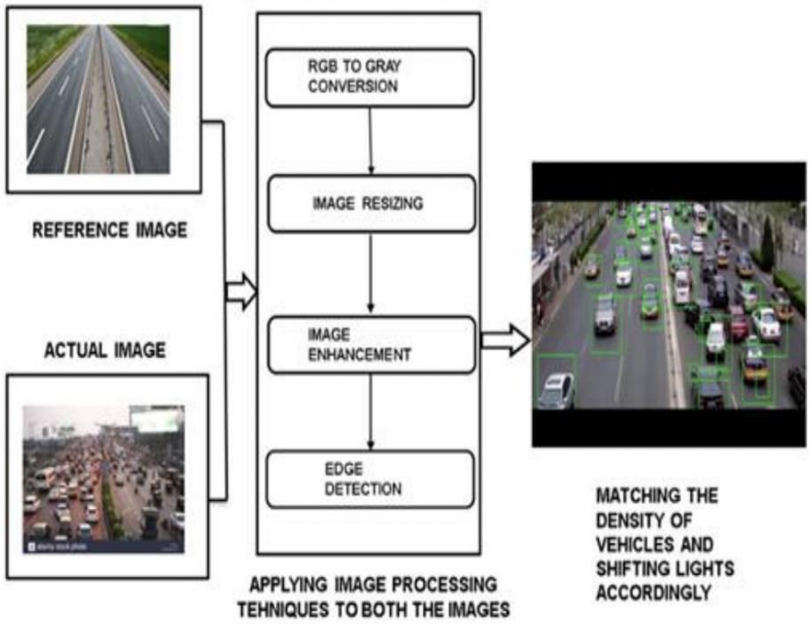


Fig 4.1 System Design

The initial design planned for Smart Traffic Signal Management System is to use OpenCV which is a most widely used tools for computer vision and image processing tasks and YOLO is an algorithm that uses neural networks to provide real-time object detection. This algorithm is popular because of its speed and accuracy. It has been used in various applications to detect traffic signals, people, parking meters, and animals. At first reference image to be used for inspecting the whole algorithm and the system and actual image has to be inspected thoroughly for correct detection of the vehicles where vehicles would be separated according to density of vehicles and whether it is 2-wheeler or 4-wheeler.

4.2 ASSSUMPTION AND DEPENDECIES

Traffic congestion is a major problem in many cities, and the fixed-cycle light signal controllers are not resolving the high waiting time in the intersection. We see often a policeman managing the movements instead of the traffic light. Video is a powerful medium for conveying information and data is plentiful wherever there are cameras. From dash, body, and traffic cams, to YouTube and other social media sites, there is no shortage of video data. Congestion detection also enables adaptive control, which causes dynamic adjustments to systems including traffic lights, on-ramp signaling, and bus rapid transit lanes.

The goal of this work is to improve intelligent transport systems by developing a Self-adaptive algorithm to control road traffic based on deep Learning and this This new system facilitates the movement of cars in intersections, resulting in reducing congestion, less CO2 emissions, etc.

CHAPTER 5 REQUIREMENTS

5.1 Functional Requirements

Yolo: - We are using customized YOLO (You Look Only Once) model for object detection in order to detect vehicles.

CV Algorithm: - This is a base class for all more or less complex algorithms in OpenCV, especially for classes of algorithms, for which there can be multiple implementations.

Image angle: -Image should be of high quality with some angle so that all vehicles should visible for range of 100m.

5.2 Non-Functional Requirement

- Performance
- Reliability
- Availability
- Maintainability
- Accuracy

5.3 System/Software Requirement

- Python programming Language
- pyGame
- Different Tool kits and libraries
- Minimum 8GB RAM
- Windows 10 or Mac OS

CHAPTER 6 ARCHITECTURE

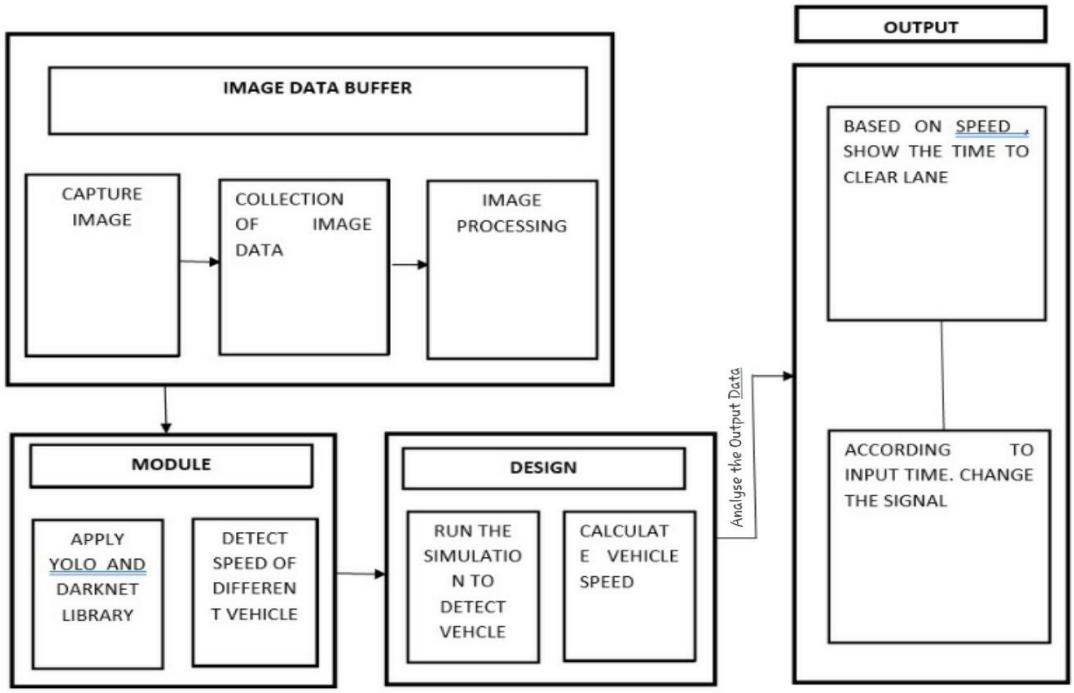


Fig 6.1 Design Architecture

The model is based on avoiding congestion in traffic and smoother the flow of traffic, which will result in decrease in the accident rates per year and indirectly will also reduce the traffic violation. Machine learning and image processing techniques are the main tools used for the implementation of the project. The design planned for Smart Traffic Signal Management System is to use OpenCV which is a most widely used tools for computer vision and image processing tasks and YOLO is an algorithm that uses neural networks to provide real-time object detection. This algorithm is popular because of its speed and accuracy. And The Simulation is programmed to analyse and optimize traffic systems effectively.

We are using image processing techniques for calculating the density of traffic present on road. It will need machine Learning Algorithm for prediction of the time when all vehicle will be identified by help of image processing.

CHAPTER 7 METHODOLOGY

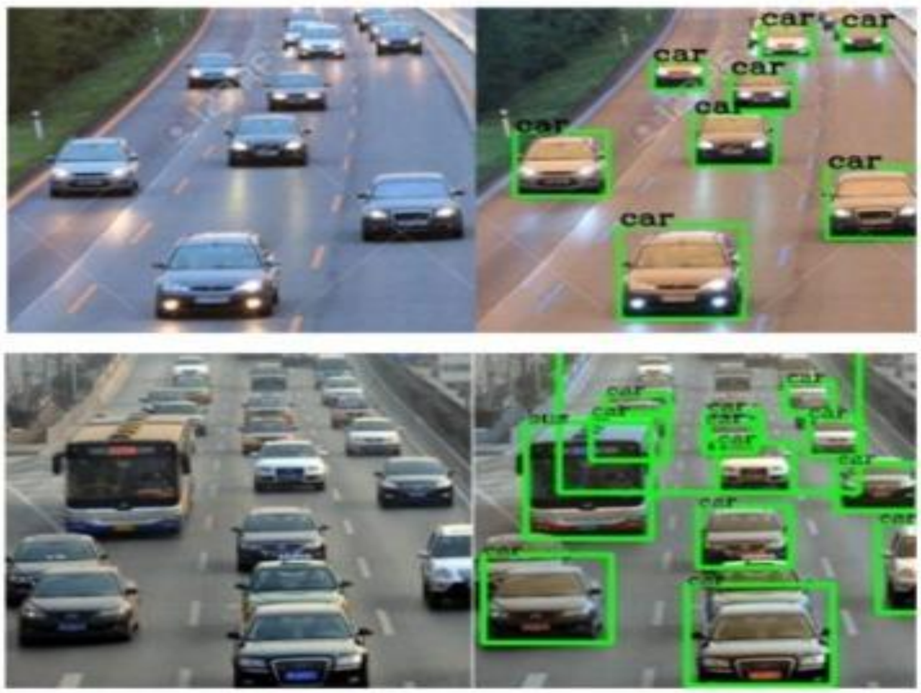


Fig 7.1 Vehicle Image Detection

The 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. The initial design planned for Smart Traffic Signal Management System is to use OpenCV which is a most widely used tools for computer vision and image processing tasks and YOLO is an algorithm that uses neural networks to provide real-time object detection. The number of vehicles of each class, such as car, bike, bus, and truck will be 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. And reference image to be used for inspecting the whole algorithm and the system and actual image has to be inspected thoroughly.

It will need machine Learning Algorithm for prediction of the time when all vehicle will be identified by help of image processing.

We will use a library and an algorithm mainly in the image processing to handle the detection of the vehicles.

1. OpenCV- Open-Source Computer Vision Library

OpenCV provides a real-time optimized Computer Vision library, tools, and hardware. It also supports model execution for Machine Learning (ML) and Image Processing. It uses the images from the CCTV cameras at traffic junctions for real-time traffic density calculation by detecting the vehicles at the signal and setting the green signal time accordingly. It is basically an object detection technique which is used in order to detect the number of vehicles for each direction.

2. YOLO- You Only Look Once

YOLO is an algorithm that uses neural networks to provide real-time object detection. It works by dividing the image into N grids, each having an equal dimensional region. Each of these N grids is responsible for the detection and localization of the object it contains. It has been used in various applications to detect traffic signs, people, parking meters, and animals.

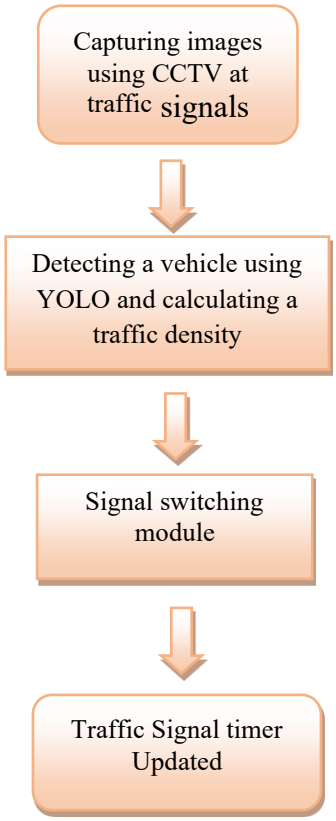
3. Darknet Library

Darknet is a convolutional neural network framework that act as a backbone for the YOLOv3 Object detection. It is written in C language. It also supports CPU and GPU computation. It is a fast and highly accurate framework for real time object detection.

The Simulation is programmed to analyze and optimize traffic systems effectively. The main reason behind simulating traffic is to generate data without going directly for the real world. Instead of testing new ideas on how to manage traffic systems in the real world or to collect the data, a model run on software can be used to predict traffic flow which helps to accelerate the optimization and data gathering of traffic systems. Simulation is a much cheaper and faster alternative to real-world testing.

CHAPTER 8 IMPLEMENTATION

Our proposed methodology was implemented using OpenCV and YOLO. We used camera for capturing image and OpenCV and YOLO for object detection. This study suggested an adaptive traffic signal management system which captures images of all the various lane present at the traffic signal junction and by using object detection and image processing controls the traffic effectively. And at lastly, it will give time as an output. This image was used as input to the YOLO object detection algorithm.



The YOLO, which offers the actual accuracy and the processing velocity and speed, is utilized for real-time vehicle detection. Vehicle detection will be trained using the "You Only Look Once (YOLO)" model, which is capable of detecting a wide range of vehicle types, including automobiles, bikes, buses, trucks, rickshaws, and more. You Only Look Once is the technique which provides us the real-time object detection using neural network technology. Total number of vehicles in each class has to be counted for determining the congestion density. The traffic-congestion density, along with few variables, which can be used by the traffic-signal switching process to set the green signal timer for each long road.

The Traffic light Signal Algorithm will adjust the green signal timer and update the red signal timers according to the traffic density determined using the YOLO method. According to the timers, it alternates between the signals.

CHAPTER 9 EXPERIMENTATION

9.1 Software Development

In this project, we have developed code for better enhancement for traffic management system by firstly identifying different Types of vehicles for determining their different speed and then after detection we implemented simulation code for both vehicles and Traffic signal, So that if one lane of vehicles will get clear automatically, it will change the signal colour and then allow other lane vehicles to move. By this way, as soon as traffic will get clear another lane vehicle will able to go quickly.

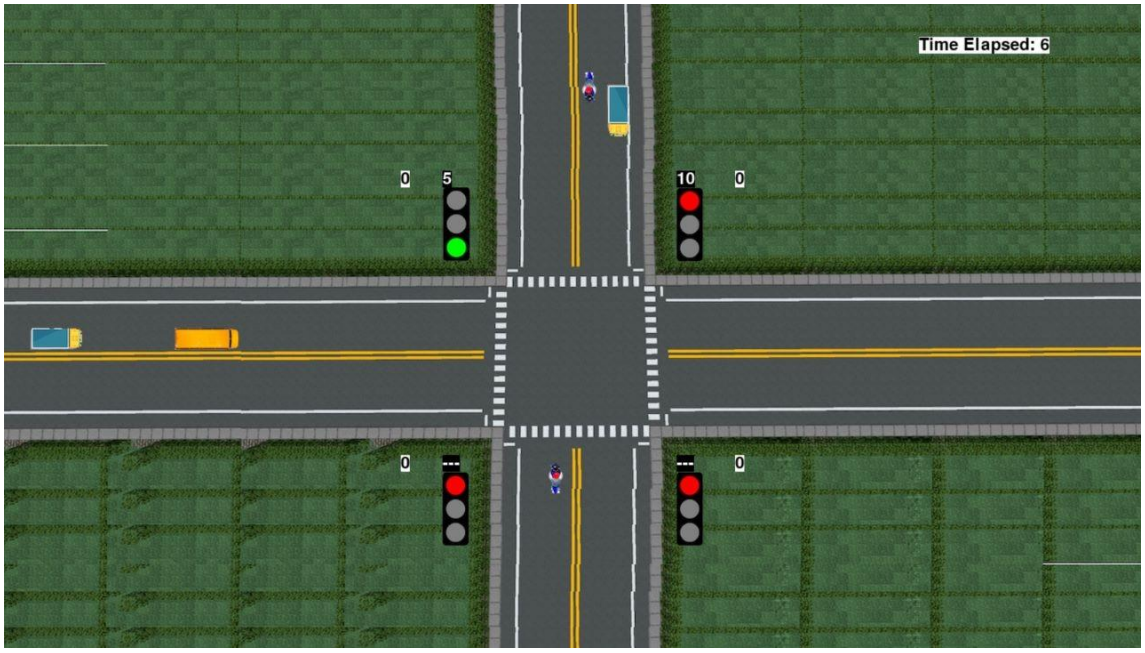
Features added:

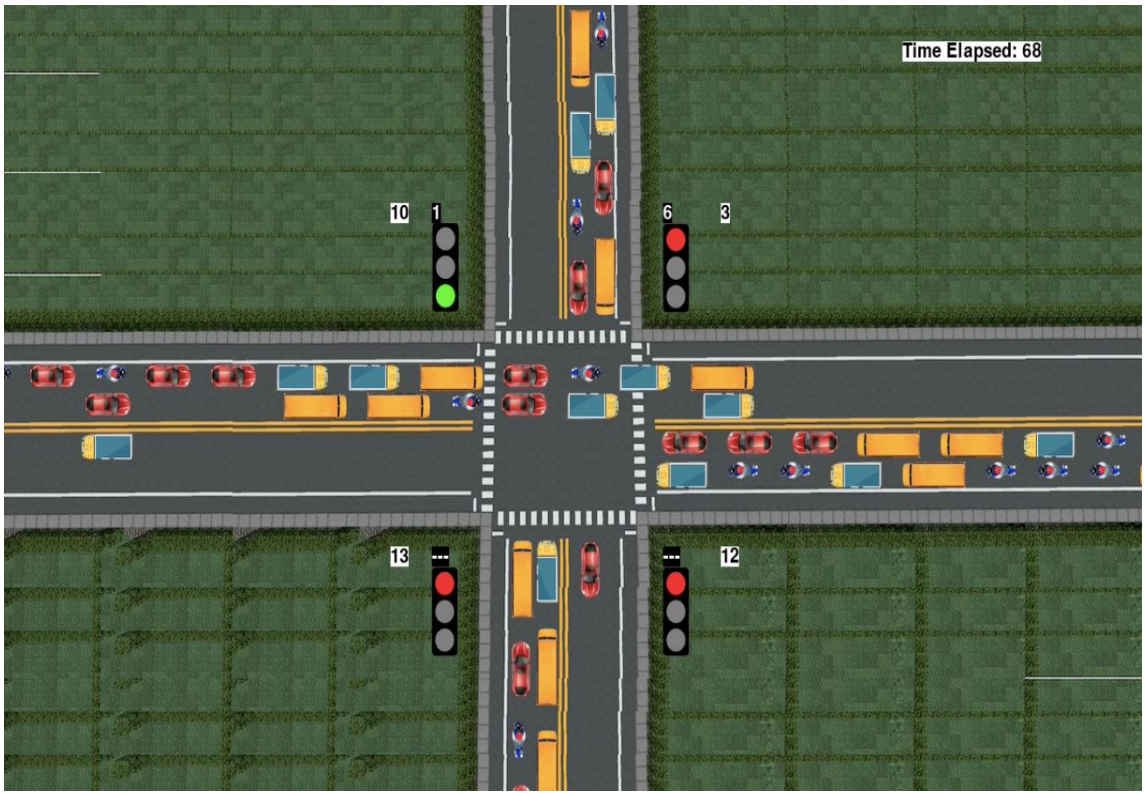
- Vehicle counts - The number of vehicles that have crossed the intersection is displayed adjacent to each traffic signal for each direction.
- Time Elapsed - The time elapsed since the start of the simulation is displayed at the top right.
- Simulation Duration - The total duration of simulation or end time can be set. The simulation will quit automatically when the time elapsed is equal to the simulation duration.
- Printing timers on Terminal - The timer values of all signals are printed on the Terminal every second. This is useful for effective debugging while customizing the simulation according to your application.
- Printing cumulative statistics - Statistics like direction-wise vehicle counts, total vehicles, and simulation time are printed on the Terminal at the end of the simulation

CHAPTER 10 RESULTS

10.1 Result

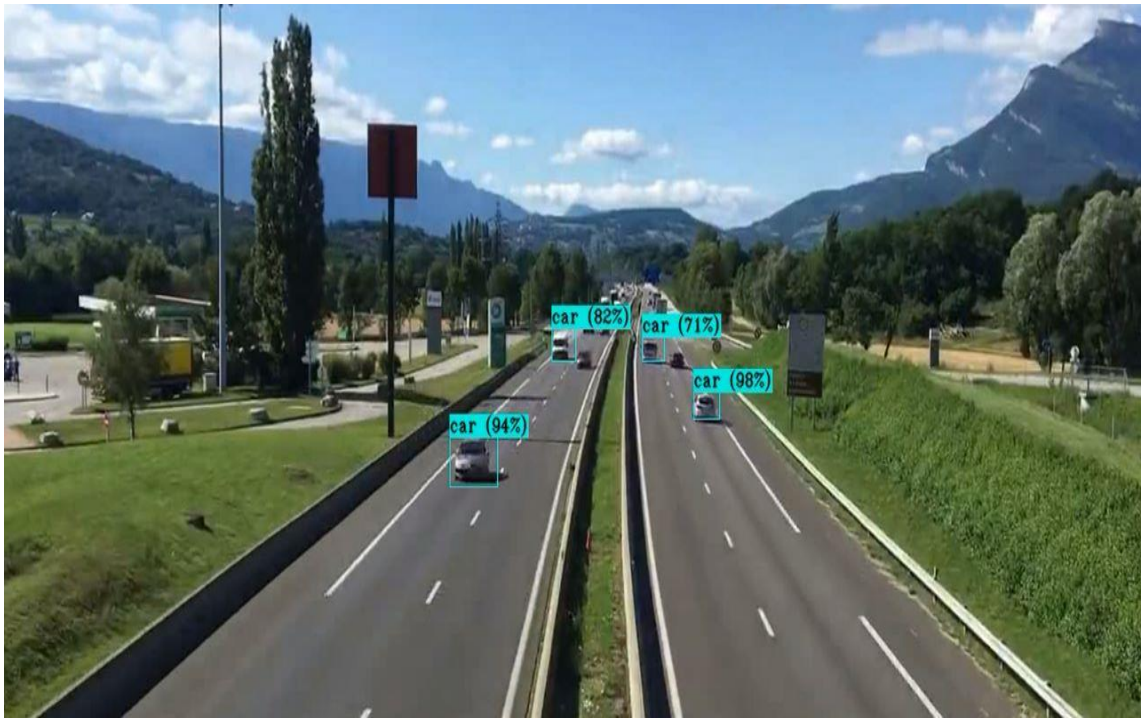
Simulation-

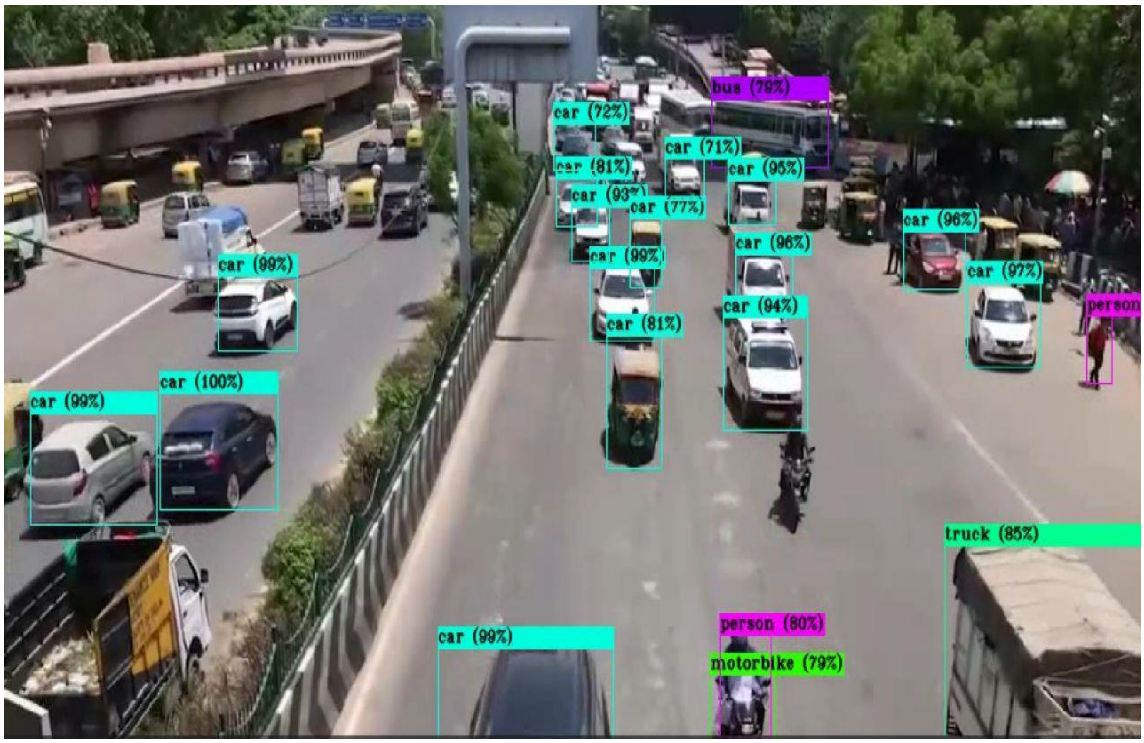




Vehicle Detection-

The YOLO, which offers the actual accuracy and the processing velocity and speed, is utilized for real-time vehicle detection. Vehicle detection will be trained using the “You Only Look Once (YOLO)” model, which is capable of detecting a wide range of vehicle types, including automobiles, bikes, buses, trucks, rickshaws, and more. You Only Look Once is the technique which provides us the real-time object detection using neural network technology.





10.2 Discussion of Results

So, after Implementation of code, we will be able to identify different types of vehicles and also will able to count number of vehicles in lane. Then after this, we will be getting simulation for all vehicles according to signal changes. if signal will be green then only, they will be able to move else in case of red signal it won't move, Then As soon as one lane will get clear other lane green signal will glow according to number of vehicles. So, by this way we will be able to make our traffic management system efficient and it will lead to save lot of time.

The primary aim of the project is to enhance the congestion management system through creating the congestion-control algorithm that makes use of image processing and the YOLO algorithm. This new system will make it easier and more affordable to relieve traffic congestion.

CHAPTER 11 CONCLUSION AND FUTURE WORK

11.1 CONCLUSION

In all urban areas, with the modernization and reduction of industries, traffic has become the main cause for air pollution. Both in Winter as in summer (ozone) pollution is present. In all cities the main effort is to control and monitor the pollution level trough a wide network of measurement stations for a real time representation in order to activate traffic reduction measurements. The model is based on avoiding congestion in traffic and smoother the flow of traffic, which will result in decrease in the accident rates per year and indirectly will also reduce the traffic violation. Lack of efficient Traffic management to replace the faulty and inefficient manual system leading to congestion and for critical emergency vehicles, which leads to lots of wastage of time. Congestion detection also enables adaptive control, which causes dynamic adjustments to systems including traffic lights, on-ramp signaling, and bus rapid transit lanes.

11.2 SCOPE FOR FUTURE WORK

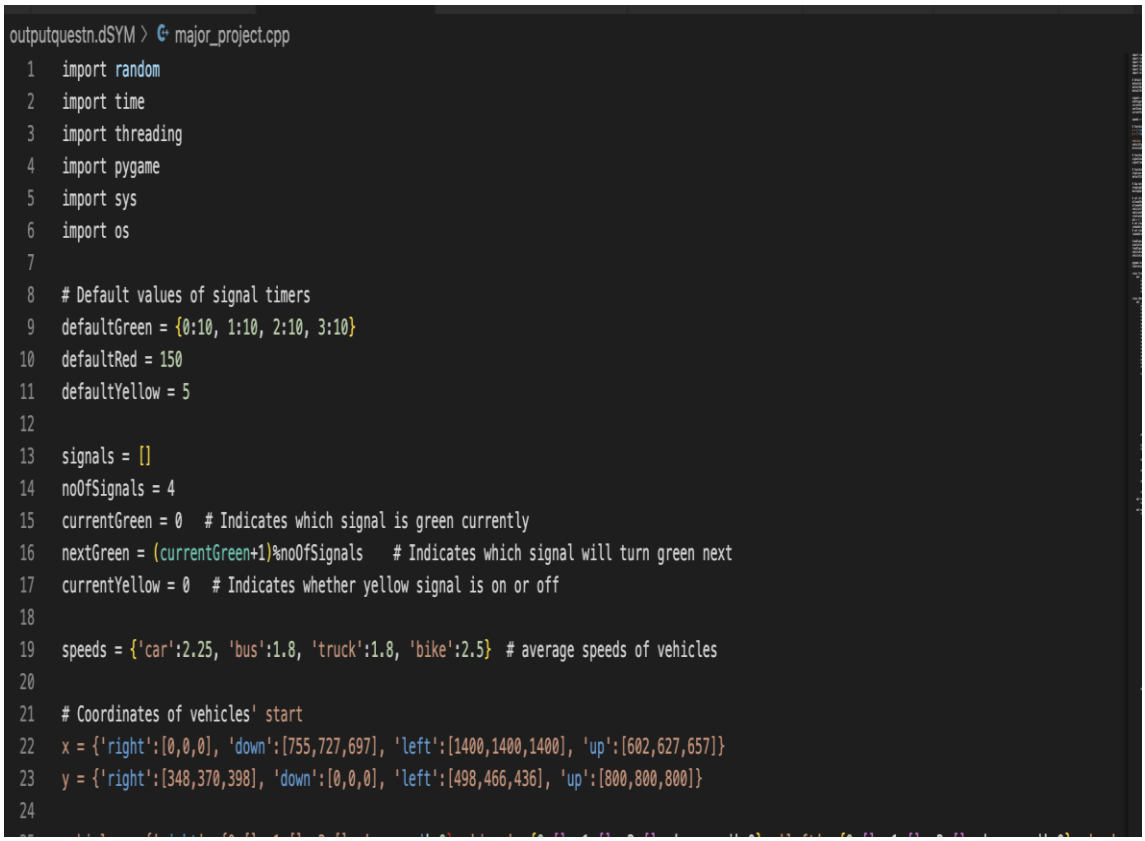
The transportation industry is going through a resurgence. Everything is increasing: traffic, congestion, the percentage of connected cars on the road – even the number of pedestrians. Transportation agencies need to respond to these changes to improve public safety and make travel easier for drivers and pedestrians. Las Vegas was able to listen to vehicles with connected technology to learn about emergency braking conditions and traction control issues. This capability proved especially valuable during inclement weather: the city was able to improve safety conditions by identifying where vehicles were having traction control issues (i.e., hydroplaning). Sensors could be arranged properly in the roads so that they won't get affected by the rain or due to any kind of water pump problems.

REFERENCES

1. Vandana Niranjan, Indu, Anam Firdous, “Smart Density Based Traffic Light System”, “International Conference on Reliability, Infocom Technologies and Optimization”, ICRITO, 2020.
2. Dr. Vikram Bali, Ms. Sonali Mathur, Dr. Vishnu Sharma, “Smart Traffic Management System using IoT Enabled Technology”, “2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)”, 2020.
3. Mr. V.S. Gadakh, Avhad Vaishali2, Dherange Rutika, Darade Sonali, Shinde Pooja, “Smart Traffic Light Control System with Automatic vehicle Speed Breaker”, “International Research Journal of Engineering and Technology (IRJET)”, 2021.
4. Mr. Sunil S. Sonawane, Bhagyashree Kenchannvar, Kajal Kumbhar, Pranjali Salunkhe”, “Ijraset Journal for research in applied science and engineering technology”, 2022
5. Islam Mohammad Albatish and Samy S. Abu Nasi, “Modelling & Controlling Traffic Light System”, “International Conference on Promising electronics technology”, “15th International Conference on Electronics, Computer and Computation (ICECCO)”, 2019.

SAMPLE CODE

Imported Library and Define Coordinates of vehicles



outputquestn.dSYM 〉 major_project.cpp

```
import random
import time
import threading
import pygame
import sys
import os

# Default values of signal timers
defaultGreen = {0:10, 1:10, 2:10, 3:10}
defaultRed = 150
defaultYellow = 5

signals = []
noOfSignals = 4
currentGreen = 0   # Indicates which signal is green currently
nextGreen = (currentGreen+1)%noOfSignals    # Indicates which signal will turn green next
currentYellow = 0   # Indicates whether yellow signal is on or off

speeds = {'car':2.25, 'bus':1.8, 'truck':1.8, 'bike':2.5}  # average speeds of vehicles

# Coordinates of vehicles' start
x = {'right':[0,0,0], 'down':[755,727,697], 'left':[1400,1400,1400], 'up':[602,627,657]}
y = {'right':[348,370,398], 'down':[0,0,0], 'left':[498,466,436], 'up':[800,800,800]}

```

Set Different Vehicle Types and initialize their coordinates

```
42  allowedVehicleTypes = {'car': True, 'bus': True, 'truck': True, 'bike': True}
43  allowedVehicleTypesList = []
44  vehiclesTurned = {'right': {1:[], 2:[]}, 'down': {1:[], 2:[]}, 'left': {1:[], 2:[]}, 'up': {1:[], 2:[]}}
45  vehiclesNotTurned = {'right': {1:[], 2:[]}, 'down': {1:[], 2:[]}, 'left': {1:[], 2:[]}, 'up': {1:[], 2:[]}}
46  rotationAngle = 3
47  mid = {'right': {'x':705, 'y':445}, 'down': {'x':695, 'y':450}, 'left': {'x':695, 'y':425}, 'up': {'x':695, 'y':400}}
48  # set random or default green signal time here
49  randomGreenSignalTimer = True
50  # set random green signal time range here
51  randomGreenSignalTimerRange = [10,20]
52
53  timeElapsed = 0
54  simulationTime = 300
55  timeElapsedCoods = (1100,50)
56  vehicleCountTexts = ["0", "0", "0", "0"]
57  vehicleCountCoods = [(480,210),(880,210),(880,550),(480,550)]
58
59  pygame.init()
60  simulation = pygame.sprite.Group()
61
62  class TrafficSignal:
63      def _init_(self, red, yellow, green):
64          self.red = red
65          self.yellow = yellow
66          self.green = green
```

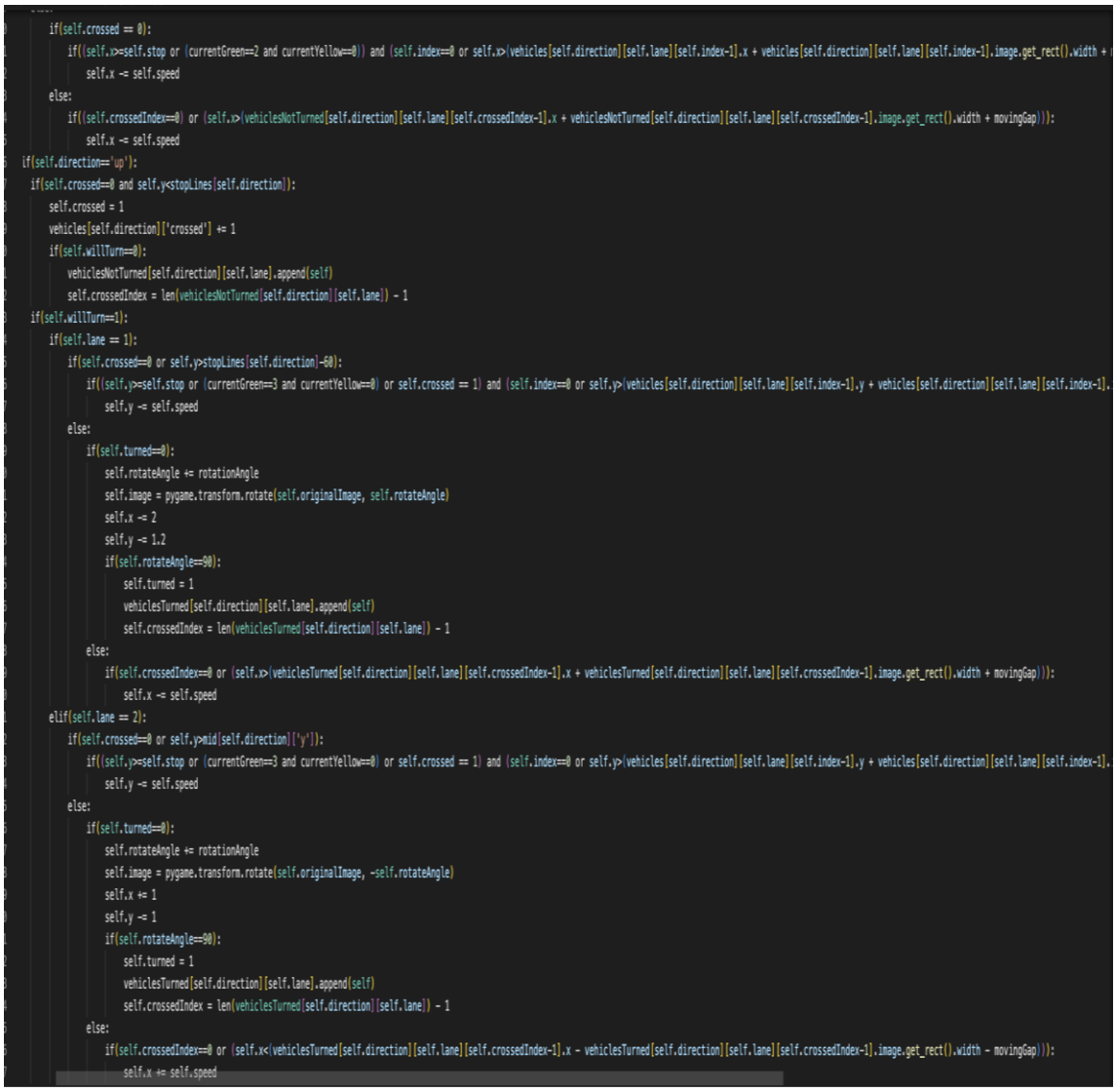
Setting new starting and Stopping Coordinates

```
        if(direction=='right'):
            temp = self.image.get_rect().width + stoppingGap
            x[direction][lane] -= temp
        elif(direction=='left'):
            temp = self.image.get_rect().width + stoppingGap
            x[direction][lane] += temp
        elif(direction=='down'):
            temp = self.image.get_rect().height + stoppingGap
            y[direction][lane] -= temp
        elif(direction=='up'):
            temp = self.image.get_rect().height + stoppingGap
            y[direction][lane] += temp
        simulation.add(self)

    def render(self, screen):
        screen.blit(self.image, (self.x, self.y))

    def move(self):
        if(self.direction=='right'):
            if(self.crossed==0 and self.x+self.image.get_rect().width>stopLines[self.direction]):
                self.crossed = 1
                vehicles[self.direction]['crossed'] += 1
                if(self.willTurn==0):
                    vehiclesNotTurned[self.direction][self.lane].append(self)
                    self.crossedIndex = len(vehiclesNotTurned[self.direction][self.lane]) - 1
            if(self.willTurn==1):
```

Defining Different Angles for Moving Vehicles



```
        if(self.crossed == 0):
            if((self.x>=self.stop or (currentGreen==2 and currentYellow==0)) and (self.index==0 or self.x>(vehicles[self.direction][self.lane][self.index-1].x + vehicles[self.direction][self.lane][self.index-1].image.get_rect().width +
                self.x -= self.speed
        else:
            if((self.crossedIndex==0) or (self.x>(vehiclesNotTurned[self.direction][self.lane][self.crossedIndex-1].x + vehiclesNotTurned[self.direction][self.lane][self.crossedIndex-1].image.get_rect().width + movingGap))):
                self.x -= self.speed
    if(self.direction=='up'):
        if(self.crossed==0 and self.y<stopLines[self.direction]):
            self.crossed = 1
            vehicles[self.direction]['crossed'] += 1
            if(self.willTurn==0):
                vehiclesNotTurned[self.direction][self.lane].append(self)
                self.crossedIndex = len(vehiclesNotTurned[self.direction][self.lane]) - 1
        if(self.willTurn==1):
            if(self.lane == 1):
                if(self.crossed==0 or self.y>stopLines[self.direction]-60):
                    if((self.y>=self.stop or (currentGreen==3 and currentYellow==0) or self.crossed == 1) and (self.index==0 or self.y>(vehicles[self.direction][self.lane][self.index-1].y + vehicles[self.direction][self.lane][self.index-1].
                        self.y -= self.speed
                else:
                    if(self.turned==0):
                        self.rotateAngle += rotationAngle
                        self.image = pygame.transform.rotate(self.originalImage, self.rotateAngle)
                        self.x -= 2
                        self.y -= 1.2
                        if(self.rotateAngle==90):
                            self.turned = 1
                            vehiclesTurned[self.direction][self.lane].append(self)
                            self.crossedIndex = len(vehiclesTurned[self.direction][self.lane]) - 1
                    else:
                        if(self.crossedIndex==0 or (self.x>(vehiclesTurned[self.direction][self.lane][self.crossedIndex-1].x + vehiclesTurned[self.direction][self.lane][self.crossedIndex-1].image.get_rect().width + movingGap))):
                            self.x -= self.speed
            elif(self.lane == 2):
                if(self.crossed==0 or self.y>mid[self.direction]['y']):
                    if((self.y>=self.stop or (currentGreen==3 and currentYellow==0) or self.crossed == 1) and (self.index==0 or self.y>(vehicles[self.direction][self.lane][self.index-1].y + vehicles[self.direction][self.lane][self.index-1].
                        self.y -= self.speed
                else:
                    if(self.turned==0):
                        self.rotateAngle += rotationAngle
                        self.image = pygame.transform.rotate(self.originalImage, -self.rotateAngle)
                        self.x += 1
                        self.y -= 1
                        if(self.rotateAngle==90):
                            self.turned = 1
                            vehiclesTurned[self.direction][self.lane].append(self)
                            self.crossedIndex = len(vehiclesTurned[self.direction][self.lane]) - 1
                    else:
                        if(self.crossedIndex==0 or (self.x<(vehiclesTurned[self.direction][self.lane][self.crossedIndex-1].x - vehiclesTurned[self.direction][self.lane][self.crossedIndex-1].image.get_rect().width - movingGap))):
                            self.x += self.speed
```

Initialize Signal with default Values and Printing the Signal Timer

```
def initialize():
    minTime = randomGreenSignalTimerRange[0]
    maxTime = randomGreenSignalTimerRange[1]
    if(randomGreenSignalTimer):
        ts1 = TrafficSignal(0, defaultYellow, random.randint(minTime,maxTime))
        signals.append(ts1)
        ts2 = TrafficSignal(ts1.red+ts1.yellow+ts1.green, defaultYellow, random.randint(minTime,maxTime))
        signals.append(ts2)
        ts3 = TrafficSignal(defaultRed, defaultYellow, random.randint(minTime,maxTime))
        signals.append(ts3)
        ts4 = TrafficSignal(defaultRed, defaultYellow, random.randint(minTime,maxTime))
        signals.append(ts4)
    else:
        ts1 = TrafficSignal(0, defaultYellow, defaultGreen[0])
        signals.append(ts1)
        ts2 = TrafficSignal(ts1.yellow+ts1.green, defaultYellow, defaultGreen[1])
        signals.append(ts2)
        ts3 = TrafficSignal(defaultRed, defaultYellow, defaultGreen[2])
        signals.append(ts3)
        ts4 = TrafficSignal(defaultRed, defaultYellow, defaultGreen[3])
        signals.append(ts4)
    repeat()

# Print the signal timers on cmd
def printStatus():
    for i in range(0, 4):
        if(signals[i] != None):
            if(i==currentGreen):
                if(currentYellow==0):
                    print(" GREEN TS",i+1,"-> r:",signals[i].red," y:",signals[i].yellow," g:",signals[i].green)
                else:
                    print("YELLOW TS",i+1,"-> r:",signals[i].red," y:",signals[i].yellow," g:",signals[i].green)
            else:
                print("   RED TS",i+1,"-> r:",signals[i].red," y:",signals[i].yellow," g:",signals[i].green)
    print()
```

Displaying Simulation as a output with Time elapsed for each passing of vehicles

```
496         screen.blit(background,(0,0))   # display background in simulation
497         for i in range(0,noOfSignals):  # display signal and set timer according to current status: green, yello, or red
498             if(i==currentGreen):
499                 if(currentYellow==1):
500                     signals[i].signalText = signals[i].yellow
501                     screen.blit(yellowSignal, signalCoods[i])
502                 else:
503                     signals[i].signalText = signals[i].green
504                     screen.blit(greenSignal, signalCoods[i])
505             else:
506                 if(signals[i].red<=10):
507                     signals[i].signalText = signals[i].red
508                 else:
509                     signals[i].signalText = "---"
510                 screen.blit(redSignal, signalCoods[i])
511         signalTexts = ["","","",""]
512 
513         # display signal timer
514         for i in range(0,noOfSignals):
515             signalTexts[i] = font.render(str(signals[i].signalText), True, white, black)
516             screen.blit(signalTexts[i],signalTimerCoods[i])
517 
518         # display vehicle count
519         for i in range(0,noOfSignals):
520             displayText = vehicles[directionNumbers[i]]['crossed']
521             vehicleCountTexts[i] = font.render(str(displayText), True, black, white)
522             screen.blit(vehicleCountTexts[i],vehicleCountCoods[i])
523 
524         # display time elapsed
525         timeElapsedText = font.render(("Time Elapsed: "+str(timeElapsed)), True, black, white)
526         screen.blit(timeElapsedText,timeElapsedCoods)
527 
528         # display the vehicles
529         for vehicle in simulation:
530             screen.blit(vehicle.image, [vehicle.x, vehicle.y])
531             vehicle.move()
532         pygame.display.update()
```