# SMART TRAFFIC MANAGEMENT SYSTEM

**A PROJECT REPORT**

***Submitted by***

**ARYAN OMKAR ASHAR-19BAI10094**
**SHREYA SHETYE-19BAI10028**
**SOUMYA RAJADHYAKSHA: -19BAI10120**
**PRANAV SHARMA: - 19BAI10154**

*in partial fulfillment for the award of the degree*
*of*

**BACHELOR OF TECHNOLOGY**
*in*
**COMPUTER SCIENCE AND ENGINEERING**

*Specialization in*

***Artificial intelligence and machine learning***

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**

**VIT BHOPAL UNIVERSITY**

**KOTHRIKALAN, SEHORE**

**MADHYA PRADESH - 466114**

OCTOBER 2020.

**VIT BHOPAL UNIVERSITY,KOTHRIKALAN, SEHORE MADHYA PRADESH – 466114**

## BONAFIDE CERTIFICATE

Certified that this project report titled **“SMART TRAFFIC CONTROL SYSTEM”** is the bonafide work of “**Aryan Omkar Ashar(19BAI10094), Shreya Shetye (19BAI10028), Soumya Rajadhyaksha(19BAI10120) and Pranav Sharma(19BAI10154)”** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported here does not form part of any other project / research work on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

|  |  |
|---|---|
| **PROGRAM CHAIR** | **PROJECT GUIDE** |
| Dr. S. Sountharrajan | Dr. Manas Kumar Mishra |
| School of AI &ML division | School of AI &ML division |
| VIT BHOPAL UNIVERSITY | VIT BHOPAL UNIVERSITY |

The Project Exhibition I Examination is held on 30/10/2020

# ACKNOWLEDGEMENT

First and foremost, I would like to thank the Lord Almighty for His presence and immense blessings throughout the project work.

I wish to express my heartfelt gratitude to Dr. Manas Kumar Mishra, Head of the Department, School of Computing Science and Engineering for much of his valuable support encouragement in carrying out this work.

I would like to thank my internal guide Dr. Manas Kumar Mishra, for continually guiding and actively participating in my project, giving valuable suggestions to complete the project work.

I would like to thank all the technical and teaching staff of the School of Computing Science and Engineering, who extended directly or indirectly all support.

Last, but not the least, I am deeply indebted to my parents who have been the greatest support while I worked day and night for the project to make it a success.

-

## LIST OF ABBREVIATIONS

AI – Artificial Intelligence
ML- Machine Learning
API- Application Programming Interface
OpenCV- Open Source Computer Vision Library
STM-Smart Traffic Management

## LIST OF FIGURES

## LIST OF TABLES AND GRAPHS

## ABSTRACT

Traffic management has been an issue and has created havoc since a lot of time and has caused many problems in India. Efficient traffic management is the goal of the road industry and is very important and needed in today’s world so that each and every congestion can be managed smoothly creating spaces and making all the people reach their destination on time and also that the people complete their voyage safely. When vehicles are fully or partially stopped for an epoch of time it is known as a traffic jam or a traffic congestion. Traffic congestion is a condition on road networks and crossroads that occurs as use magnifies, and is described by slower speeds of on-road vehicles, longer voyage times, and augmented conveyance queuing. The project basically focuses on smart traffic management using automated systems and python programming is implemented in this project and we have also used Microsoft Azure and Custom Vision API as the external free software.

In this paper we propose a smart traffic control system which manages the congested traffic and allows smooth flow of traffic without any human intervention in between. The model that we have proposed also segregates the number of vehicles on the road into 4 categories that is car, bus, truck and motorcycle. The model also allows the emergency vehicles like ambulances and fire brigades to pass smoothly and gives priority to the lane having the emergency vehicle so that the patient does not suffer any serious consequences due to traffic.

A web camera is placed on all the lanes that is at all the sides in a traffic signal. The camera continuously captures the images of the lanes. In our project we have integrated a total of four

lanes. The model first checks for emergency vehicles if it is present in any of the lane and then allows the lane with maximum number of cars to go green. The model divides into the four types of vehicles which can also be used as an important information to use for the government. The model also can be used to detect the number of walkers on the road and thus the information can be of great use to the government. The model we have developed can learn from its mistakes and thus is a self-learning model.

# TABLE OF CONTENTS

# 1. INTRODUCTION

## 1.1: Introduction

Traffic congestion or jams as we normally refer to it is a very common condition we all experience on the roads of India specially during the rush hours. This unnecessarily increases the wait time of each vehicle and also increases the journey time of a road trip. To avoid this congestion, automated traffic signals and supervision is important. Setting timers for every signal is the best way to get traffic into control, but there are instances wherein there are emergency vehicles in one of the lanes which cannot make their way out due to the timed signals. For cases like these we need smart recognition systems which prioritizes the lane having either the most congestion or the one having an emergency vehicle. These systems will also be timed, but will undergo changes depending on the situation.

The project will mainly focus on the use of image analytics to implementing traffic control for image processing for proper management of traffic in urban areas using cameras and sensors. The project is made intelligent enough to prioritize specific lanes on daily basis.

## 1.2: Motivation for the work

We have been hearing a lot about police van and ambulances being late to reach their destination, while saying this we need to consider the amount of traffic that we face everyday. The number of accidents that are being reported at a daily basis is also a factor which made us think about creating a system which will reduce the amount of confusion on roads and will also help the traffic to move smoothly. Considering the rise in automobile production, handling the traffic manually is a tedious task to do. This system will reduce the burden of the traffic police and make them available in places of genuine need. In 2014, 54% of the total global population was urban residents. The prediction was a growth of nearly 2% each year until 2020 leading to more pressure on the transportation system of cities. Additionally, the high cost of accommodation in business districts lead to urban employees living far away from their place of work/education and therefore having to commute back and forth between their place of residence and their place of work.

## 1.3: About Introduction to the project including techniques

All of us, at least ones in our life time, come across the problem of road congestion. It might be either when we are going to attend a function ourselves our might have seen people trying to get through the traffic as soon as possible. To avoid situations like these, we have thought of implementing our knowledge of AI which will help people to have a safe journey and be on time.

1. The camera takes a picture of every lane in the junction in each iteration to check for emergency vehicles.
2. If none are found, then the first image is processed to find the approximate number of cars in the lane at the instant the picture was taken.

3. According to the number of cars, the time of the signal is adjusted so not waste time unnecessarily.
4. The cycle goes on through every signal at the junction.
5. In the case that an emergency vehicle is detected in the image, a short buffer time is given to the current signal and then the priority is shifted to the lane with the emergency vehicle.
6. After this, the cycle continues normally where it initially left off.

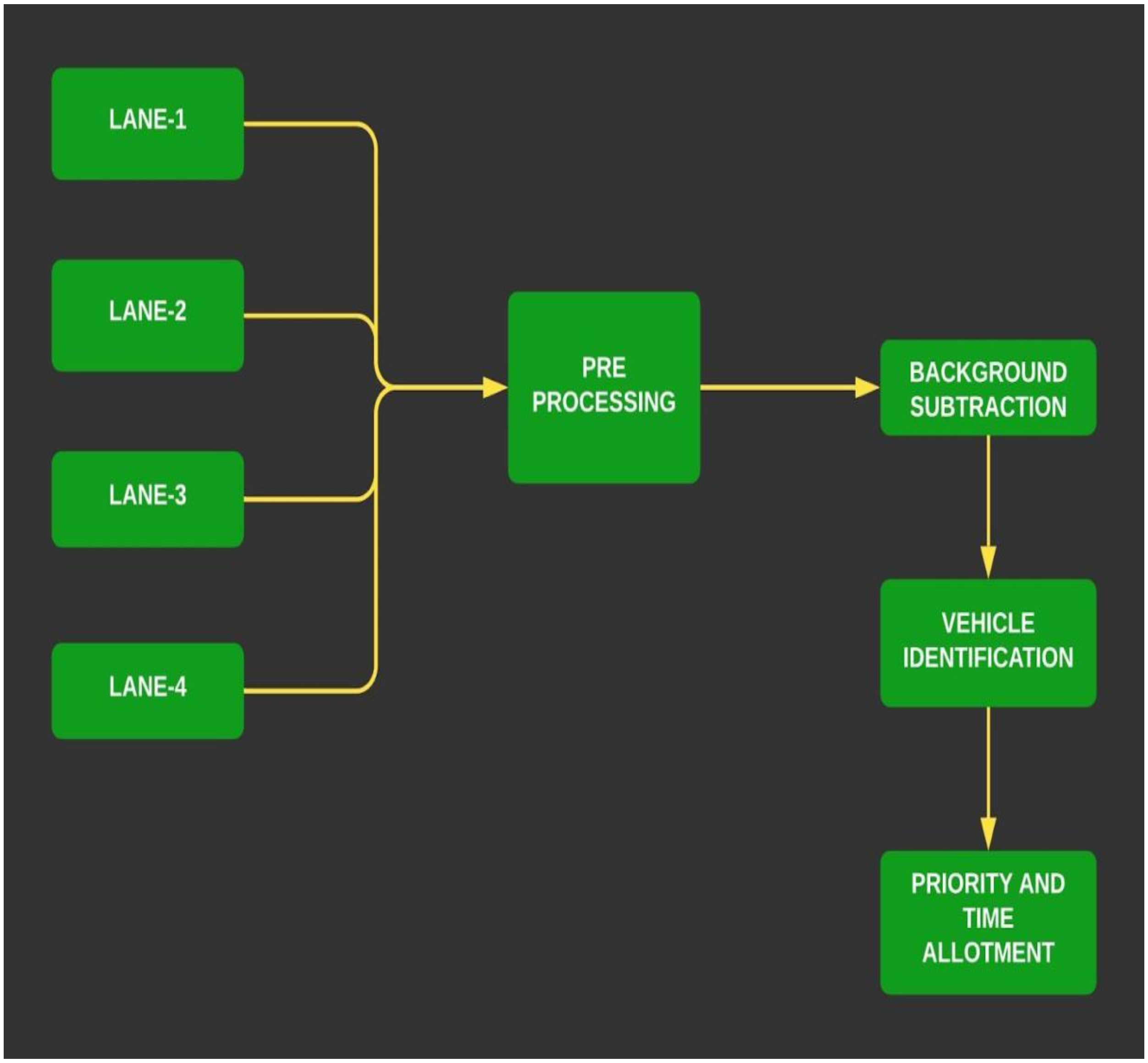


**Fig. 1.3.1 METHODOLOGY**

### 1.4: Problem Statement

- The pre-existing systems were first developed using sensors, but sensors made the implementation of the project a bit complicated.
- Our projects were implemented using image processing.
- The techniques used were vehicle counting, colour thresholding and background subtraction.

## 1.5: Objective of the work

- This project will detects the emergency vehicles namely ambulance, fire brigade van and police vehicles and gives them priority by switching their traffic signal to green.
- The idea is to create an artificially intelligent system which helps the emergency vehicles to reach its destination without any hustle and protect lives and provide security to the people of the country.
- It does not require manpower and is a very efficient system which allots signal time based on the data provided to it by the real time images.

## 1.6: Organization of the thesis

1. In section2, we discus about the research papers we referred to before starting with the project. There are some algorithms mentioned which already exist, the core are of the project and some of the new algorithms we used to make our project a success.
2. In section3, is bout system analysis we have listed some of the disadvantages in the existing systems.
3. In section4, we have given proper description about the modules that we have created in the project. The implementation and working have been described individually.
4. In section5, we have mentioned about the project and its details using tables and graphs.
5. The next and the very last section of our project has all the advantages, limitations and the future uses of this project.

## 1.7: Summary

All of the past traffic controlling systems have given some very promising outcomes, but have always consumed and wasted a lot of time by giving the green signal to empty lanes. The proposed procedure keeps away from this issue.  We have effectively actualized a calculation for a genuine -time picture handling-based traffic regulator. Endless supply of image identification calculations, it was surmised that Microsoft Assure is the most proficient method for this project.

# 2. LITERATURE REVIEW

## 2.1 Introduction

1.1. Traffic congestion is a temporary condition on networks that happens as utility increases, and is characterized by slower speeds, longer trip times, and increased queuing.

1.2. At the point when the volume of traffic is high and so heterogenous that the interaction between vehicles slows down the speed of traffic, traffic congestion is the result.

1.3. As the need approaches the constraint of a road (or of the intersections along the road), traffic congestion sets in. When vehicles are fully stopped for a particular period of time, it is colloquially known as a traffic jam. Below are some of the research papers which previously existed in this smart traffic recognition system

## 2.2 Core area of the project

Our project revolves around the idea of creating such a system which will reduce the time taken during road trips and will give preference of any of the emergency vehicles on road. This will be done using image processing, where images will be compared with each other after every cycle which will help us locate an emergency vehicle or the lane having most cars. Microsoft Azure and Custom Vision API is used to train and test our own new model that s designed in order to make the project efficient and is designed to distinguish between the different types of vehicles and also to identify people on road. Python programming is used to make this system/project.

## 2.3 Existing Algorithms

2.4Algorithm1

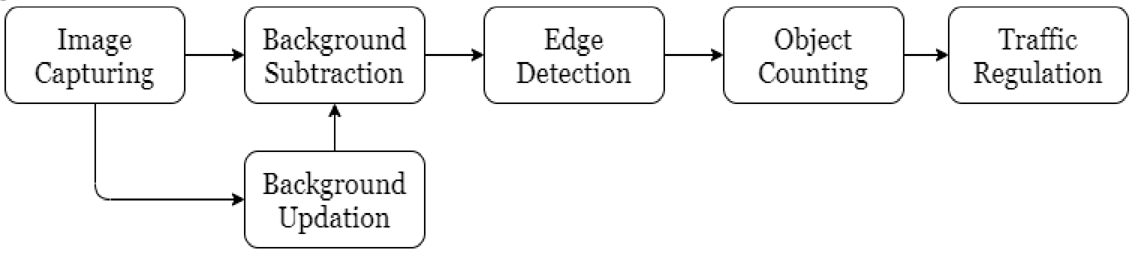


**Fig. 2.1.1 ALGORITHM 1**

## 2.5Algorithm2

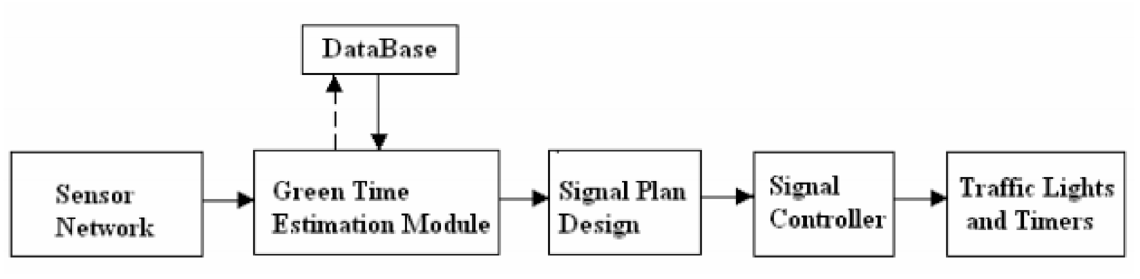


**Fig. 2.1.2 ALGORITHM 2**

## 2.6Algorithm3

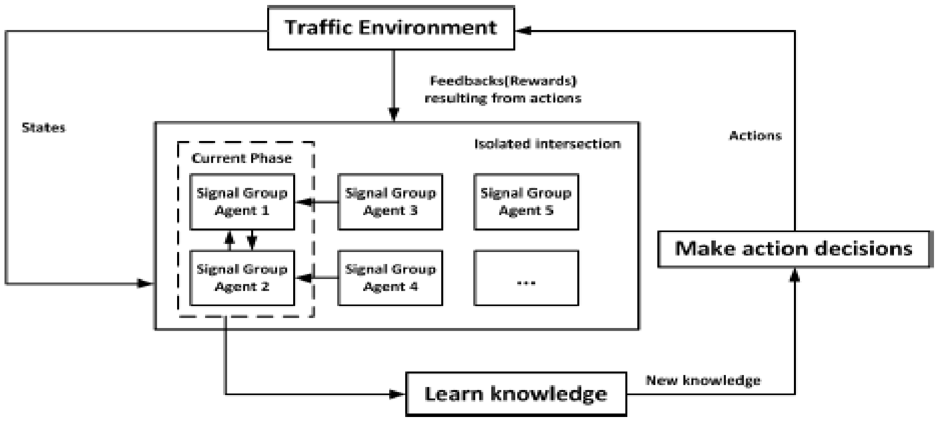


**Fig. 2.1.3 ALGORITHM 3**

## 2.7Algorithm4

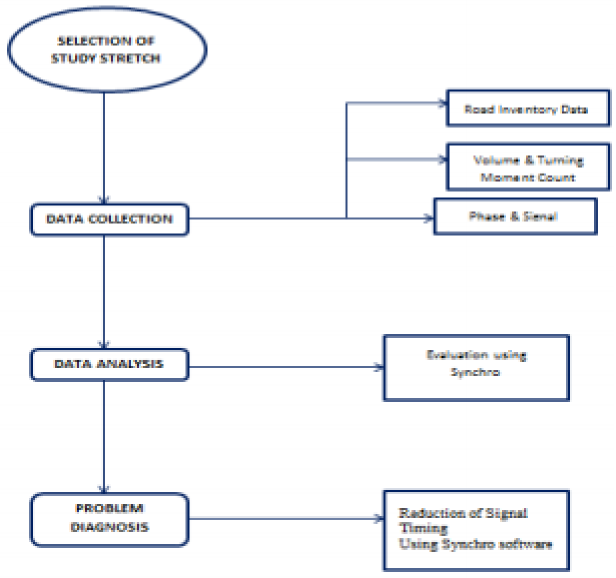

SELECTION OF STUDY STRETCH
DATA COLLECTION
Road Inventory Data
Volume & Turning Moment Count
Phase & Signal
DATA ANALYSIS
Evaluation using Synchro
PROBLEM DIAGNOSIS
Reduction of Signal Timing Using Synchro software

**Fig. 2.1.4 ALGORITHM 4**

2.8Algorithm5

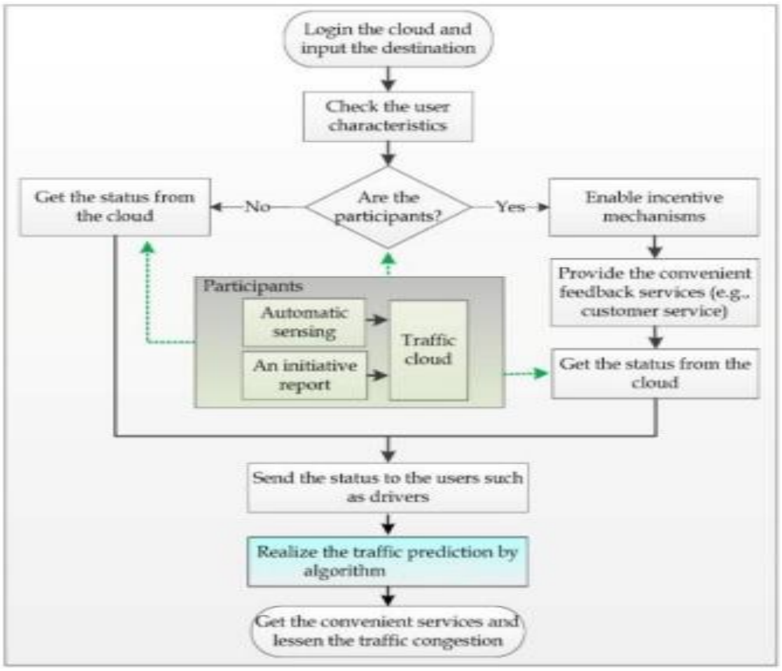

Login the cloud and input the destination
Check the user characteristics
Are the participants?
No
Yes
Get the status from the cloud
Enable incentive mechanisms
Provide the convenient feedback services (e.g., customer service)
Participants
Automatic sensing
An initiative report
Traffic cloud
Get the status from the cloud
Send the status to the users such as drivers
Realize the traffic prediction by algorithm
Get the convenient services and lessen the traffic congestion

**Fig. 2.1.5 ALGORITHM 5**

## 2.4 Any other method used in the project:

The project uses a free software like Microsoft Azure and Custom Vision API that is used to train the model to an accuracy of 90%. Microsoft Azure is a cloud computing software that is used to build and train models. Also to train the model to detect an emergency vehicle we have extracted images from various sites and have put them into the model in order to enhance the accuracy of the model and make it more efficient and reliable.

## 2.5 Observations from literature survey

2.6 Stevanovic, C. Kergaye, J. Stevanovic, in “Evaluating robustness of signal timings for varying traffic flows”. In their research they perform calculations to evaluate an efficient system for optimizing signal timings in varying traffic conditions. The results of this research show that despite the fact that improving signal timings for higher traffic requests is problematic, this technique is in a way better than streamlining signals for lower traffic requests and ought to be utilized when adequate traffic information is not accessible.

2.7 Naren Athmaraman and Srivathsan Soundararajan "Adaptive Predictive Traffic Timer Control Algorithm". They make use of dynamic queue length estimation for optimization of traffic light controllers. A web based planning update strategy for efficient traffic flow presents an upgraded signal coordination calculation.

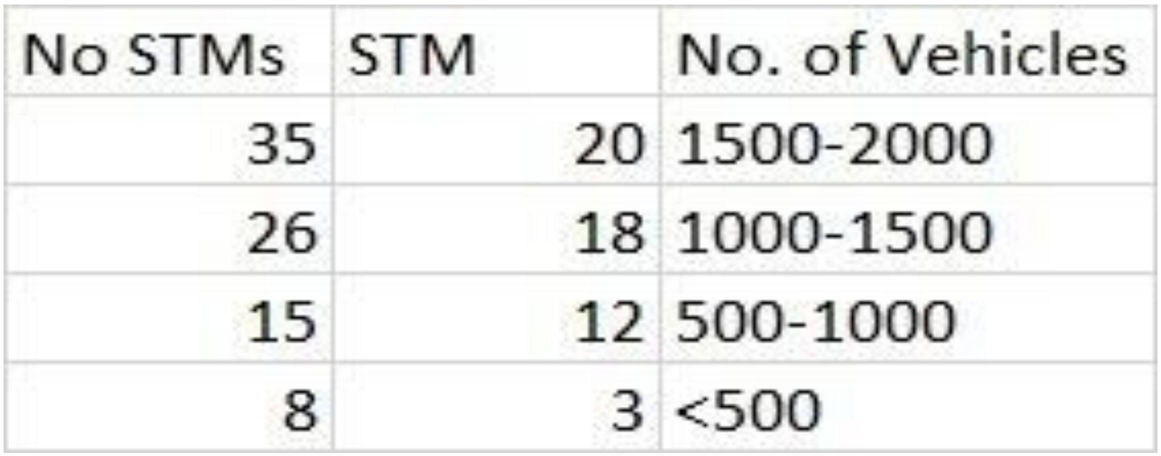
2.8 Junchen Jin and Xiaoliang Ma in their research paper on "A group-based traffic signal control with adaptive learning ability "They make use of group-based control i.e. an advanced traffic signal strategy with the potential of dynamically creating phase sequences at an intersection. Simulation results show that the proposed versatile group-based control system beats the upgraded GBVA control system basically on account of its real-time adaptive learning capacity in response to the changes of traffic demand.

2.9 Ishant Sharma and Dr. Pardeep K. Gupta in "Study of automatic traffic signal system for Chandigarh". They proposed to replace existing traffic signals with a framework that observes the traffic flow automatically in traffic signals and sensors are fixed in so that the time feed is made dynamic and automatic by processing the live detection.

2.10 Pavan Kumar and Dr. M. Kamala kumara in "A Novel Application of Adaptive Traffic Control System for India". They studied adaptive traffic control systems with VANET, centered on reliable traffic prediction approaches and varied sorts of adaptive control algorithms and additionally projected a mobile crowd sensing technology to support dynamic route selections for drivers to avoid congestion. Recommended crowdsourcing will be one of the most effective choices for adaptive traffic control systems for India.

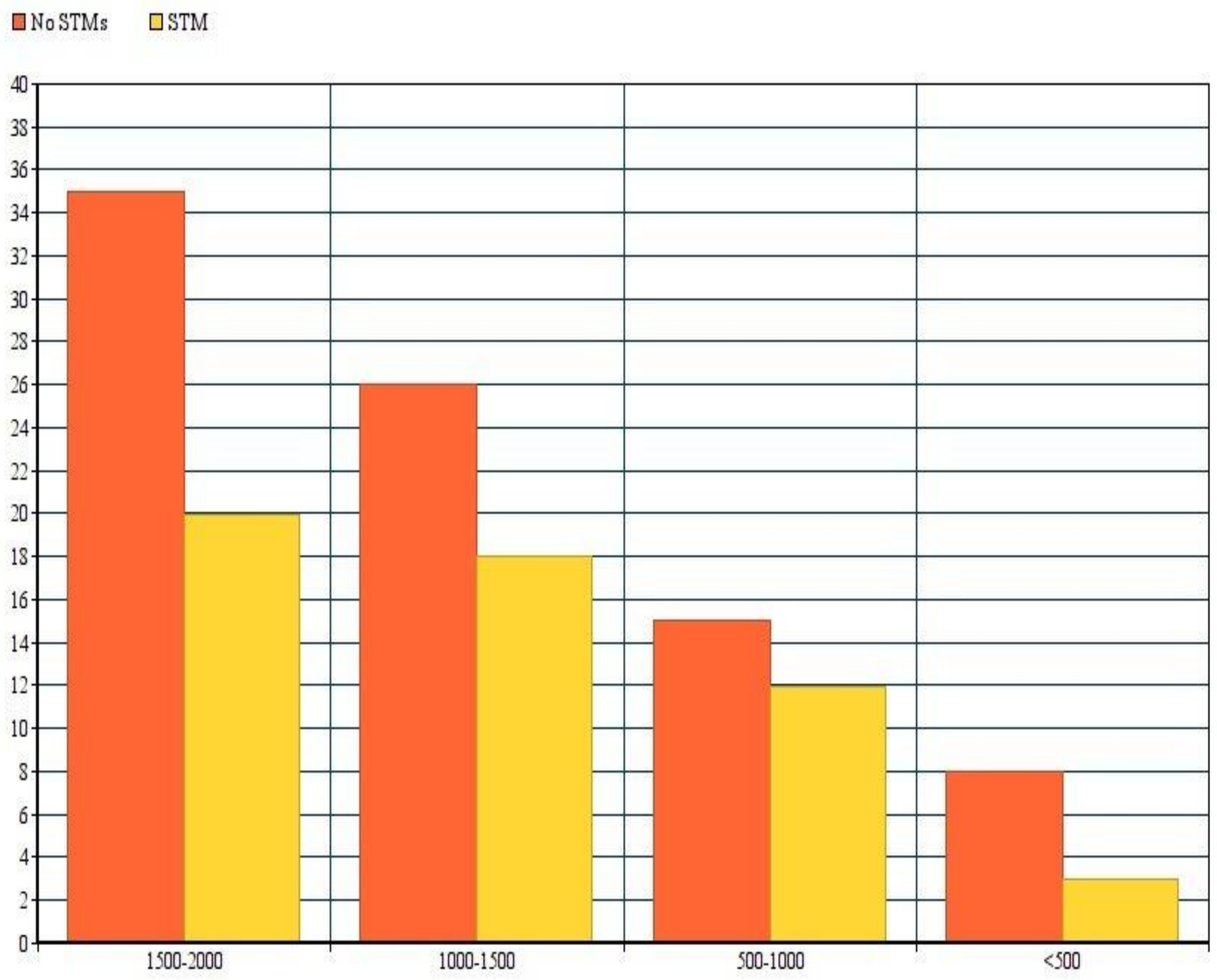
## 2.6 Summary

After going through plenty of papers and analysing each algorithm used thoroughly, we, with the guidance of our internal guide, Dr. Manas Kumar Mishra, have successfully been able to develop this model which is efficient and has scope in the future too. Using many learning resources and research papers by many eminent scholars we have designed our algorithms and models and thus the smart traffic management system is efficient, accurate and easy to implement and affordable to use.



| No STMs | STM | No. of Vehicles |
|---|---|---|
| 35 | 20 | 1500-2000 |
| 26 | 18 | 1000-1500 |
| 15 | 12 | 500-1000 |
| 8 | 3 | <500 |

**TABLE 2.6.1 COMPARISION BETWEEN STMs V/S NO STMs**



**GRAPH 2.6.1 COMPARISION BETWEEN STMs V/S NO STMs**

# 3. SYSTEM ANALYSIS

## 3.1 DISADVANTAGES/LIMITATIONS IN EXISTING SYSTEM

Indian Traffic Controllers are basically pre-timed where the time of each lane to have a particular is fixed. For better and smooth flow of traffic on roads, manpower is being used in order to control the signals. The traffic police are used to control the flow of traffic currently in some situations. The main drawback of this system controlled by the traffic police is that the system is not smart enough to deal with the traffic congestion. The traffic police official can either block a road for more amount of time or let the vehicles on another road pass by i.e. the decision making may not be smart enough and it entirely depends on the official's decision.

In a four-lane traffic signal, lanes are given a green signal for equal amounts of time. Thus, the traffic light allows the vehicles of all lanes to pass in a particular pre-defined sequence. This unnecessarily extends the waiting time of a particular lane even if it has a low number of vehicles in it.
The pre-existing systems were first developed using sensors, but sensors made the implementation of the project a bit complicated. The previous projects were implemented using image processing. The techniques used earlier were vehicle counting, color thresholding and background subtraction. This technique turned the image to black and white and minimized the noise to make the picture clearer. The cars were identified using black pixels in the otherwise white background. A cluster of these black pixels were identified as a car. The problem with this system what that if a bunch of cars were packed close together or behind one another if would be difficult to identify a particular car or the number of cars.

### 3.1.1 Disadvantages of Existing System:-

1. Traffic congestion
2. No means to detect traffic congestion
3. Number of accidents are more
4. It cannot be remotely controlled
5. It requires more manpower
6. It is less economical

## 3.2 PROPOSED WORK AND ITS ADVANTAGES

In this project, we are going to control the traffic light systems based on the number of cars at a particular lane. The signal will open and an appropriate time will be allotted to the lane based on the number of cars at the lane at the particular instant. Thus, time will not be wasted by opening a signal for an empty lane which may delay the other lanes as well. The number of cars and crowd at the lane at the instant will determine the amount of time the signal is open for the particular lane.

Our project also identifies the vehicles according to their different categories such as car, bus, truck and motorcycle. This can help to take a survey of which type of vehicles are more frequent at a particular lane. This data can be useful for optimizing the traffic further. Our project also helps to recognize people in a lane. This data can be used to plan zebra crossings at places with a greater number of people or frequent visits by people as well as various other uses.
The model also recognizes an emergency vehicle such as an ambulance or fire brigade at any lane before any lane is given a green signal. Then it allows the emergency vehicles to pass smoothly and gives priority to the lane having the emergency vehicle so that it can pass smoothly without any interruptions. This will ensure that the emergency vehicle can pass safely without causing any accidents or further increasing the current traffic.

## <u>3.2.1 Advantages of Proposed System</u>

1. Minimizes number of accidents.
2. Reduces fuel cost and saves time.
3. Low budget.
4. Easy implementation and maintenance.
5. Remotely controllable.
6. Minimizes hassle and cost of commuting.

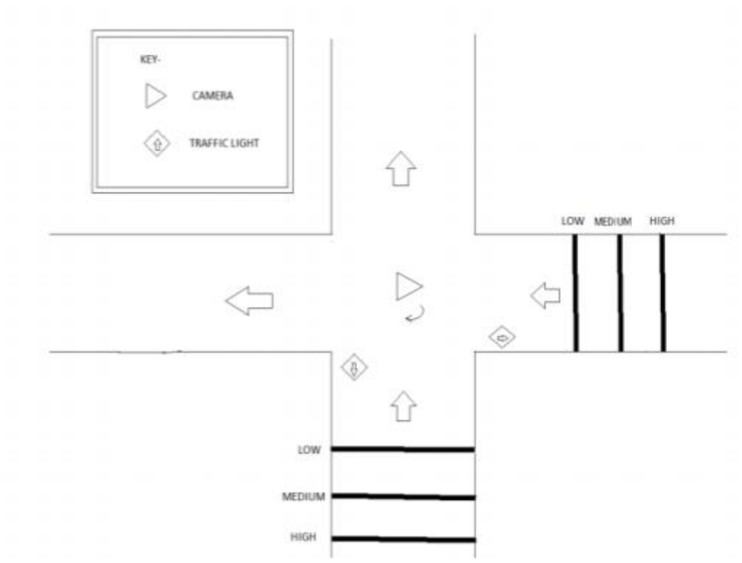


**FIG. 3.2.1 PROPOSED SYSTEM ARCHITECTURE**

# 4. SYSTEM IMPLEMENTATION

## 4.1 Project MODULE: -

Project is the basic and one of the most essential modules out of the many modules that is used in the making of a smart traffic control system. Project module revolves around the basic working of the smart traffic control system that includes accepting of all the images and processing it to remove the noise from the images. Project module will include a total of 4 different libraries that are important from the python standard library. The modules that are imported are namely mathplotlib.pyplot which is imported as plt. draw_bbox is imported from cvlib.object_detection and cvlib is imported as cv Matplotlib.pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.

NumPy stands for Numerical Python. The main use of the numpy library in python programming is to work with arrays. It also has functions of linear algebra, fourier transform, and matrices. This library in python provides a high-performance multidimensional array and basic tools to compute with and manipulate these arrays. The cv library is a part of the OpenCV module in python programming. OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Cvlib is imported from cv library in the OpenCV module. Cvlib.object_detection is used to detect objects in an image. This is a debug function to draw a rectangle around the bounding box returned by get window extent of an artist, to test whether the artist is returning the correct bbox. It is a debug function to draw a rectangle around the bounding box returned by an artist's Artist.get_window_extent to test whether the artist is returning the correct bbox.

**WORKING/IMPLEMENTATION**

The main function is defined inside the program module using the def keyword. There is a variable known as im which reads the different images using cv2.imread function. Two variables known as bbox and label are defined. The conf variable is initialized which is used to store the common objects in the image. It is defined by the help of the cv.detect_common_objects(im) function that detects common objects in the image

We find the total number of cars, trucks and bus in the image that we first imported by counting the labels with the name of car, truck and bus respectively using label.count() function. We print the number of cars, buses, motorcycles and truck in the image by print keyword  A list L is defined and the list L stores the total number of cars+ trucks+ buses+ motorcycles in the image the we have imported. The sum of numbers of all types of vehicles is stored in variable m and is appended into the list The same process is repeated for all the four images that signifies the four lanes in our project. The sum of the types of vehicles which is the total vehicles in each image is stored in the list. At the end we print L and return the list L to the statement that calls the function.

## 4.2 <u>Algos MODULE</u>

The Algos module consistes of the algorithm that is designed to build the project. The Algos module is used to compute the number of maximum cars and the minimum cars,second maximum filled lane and many different functions of the program. It is one of the most important module in the working of the program. We import the calling from the emmain module and assisgn elist = emain(). There are a total of 15 different functions defined in this module which are used to perform different tasks and the tasks of each function is stated in the working of the module. Each and every function in the module returns an integer value when it is called.  This module does not require any external libraries for the module to work efficiently and thus we do not import any external libraries.

**WORKING**

<u>GetMaxCarIndex() FUNCTION:-</u>The function GetMaxCarIndex is defined by the help of the def keyword. The function takes List from the previous module as the parameter<u>GetMaxCarIndex() FUNCTION:-</u>The function GetMaxCarIndex is defined by the help of the def keyword. The function takes List from the previous module as the parameter. Initially we define the Max variable to List[0] containing the number of cars present in image 1 and then we compare it with other images using the if loop and find the image with the maximum number of cars. And at the end we return the image number containing the most number of cars by returning IndexVal.

<u>GetSMaxCarIndex():-</u> The function GetSMaxCarIndex is defined by the help of the def keyword and takes the list as a parameter. Using the if loop, we compare two images to find the image with the second maximum number of cars. If the first image has more cars than the second, then the first image will be set to max and the second image will be set to second max. If not, the values will be set the other way round. Now, we run the for loop in the range (2,4) to compare the pre-found image having maximum and minimum cars, to another image of similar kind. If the new image is greater than the image set to max, then the old image will be get to secondmax and the new image will now be set to the max parameter.Else if the new image has comparatively less cars than that  of the secondmax image, the new image will be set to secondmax and the image two will become the max image. After running the loops, the value of secondmax will be returned.

<u>GetMinCarIndex(List):-</u>This function is almost similar as the first function but the only difference is that this one is used to find the minimum number of cars from all the 4 images.A variable named IndexVal is set to 0 to store the Minimum car index value from the four images.  Initially we define the Min variable to List[0] containing the number of cars present in image 1 and then we compare it with other images using the if loop and find the image with the maximum number of cars.  And at the end we return the image number containing the least number of cars by returning IndexVal.

<u>GetTMaxCarIndex(List):-</u>This function is defined with the help of the def keyword and takes List as the parameter. This function is used to find the image with the third most number of cars in a

laneWe first initialize min containing the minimum car numbers and second min containing the second minimum car number and run a loop from 2 to 4 and check for the image having number of cars lesser than List[min] which will be the image with third maximum number of cars

GetEMaxCarIndex(List):-The function is defined by the help of def keyword with list as the parameter. It takes a list and segregates the vehicle into emergency and non-emergency based on 0 and 1 in the list in which numbers of cars are stored.We have a list eslist and keep checking for i=1 in the list which signifies a presence of emergency vehicle . We keep it to 0 and return i(the position in the list). If there is no emergency vehicle the second part of function works where we allow the lane with maximum cars to go first

Opposite(Val):- Function Opposite() has been defined using keyword def. Using Val as a parameter, returns the number of cars going in the opposite direction The function returns the value using (Val+2)%4

Left(Val):- This function  is also defined using the def keyword and function returns (Val+1)%4 considering Val as a parameter.

Right(Val):- Function Right() is also defined by the keyword def and the function returns (Val+3)%4 using Val as a parameter.

Sum(List):-The function Sum() is defined using the def keyword and using List as a parameter from the previous module and is used to sum up all the images and returns the total number of cars using  List[0]+List[1]+List[2] and return it.
MaxVal():- The MaxVal() function is defined by the def keyword which uses List as the parameter. A variable named Max is set to 0 to store the Maximum value of the cars in the images. A for loop is executed wherein a new image is compared to the image which was set to max. If the number of cars in the new image is greater than the old image, then the new image is set to max. The max image is then returned.

Ones(List):- The Ones(List) function is defined by the def keyword. We first initialize the count variable to zero and execute a for loop in the range of 0 to 4, in which the value of count is increased as the loop proceeds.

Zeroes(List):-The function Zeroes() is defined by the def keyword which uses List as its parameter. Variable count is initialized to zero and a for loop is executed in the range of 0 to 4. The List parameter is compared to 0 and the count is increased every time the loop is executed.

NthZero(List):- The function NthZero() is defined using the keyword def and parameter List.A for loop is executed wherein a List[i] is checked to be 0 and if the value is 0 we return i(list position) else we return 0 by setting it to the value 0.

Min(A,B):- The function Min(A,B) is defined using the keyword def and A and B parameters are passed. An if loop is executed where the condition int(A)>int(B) is checked, if the condition is true then B is returned or else the parameter A is returned.

NonZero(List):- The function NonZero(List) is defined using the def keyword where the parameter List is initialized. A for loop is executed in the range 0 to 4 in which an if loop is executed where condition List[i]! =0 is checked. If the condition is true then the value of i is returned.

## 4.3 OutputDemo3 MODULE

The major function of the module OutputDemo3 is to display the output of the project. It integrates all the modules to produce a single output and display it. We use the turtle module in python to display the output. Turtle basically consists of a drawing board and a hypothetical turtle that moves as per your commands and draws the figure on the screen. Turtle is basically a graphical library that is used to create images and different drawings on the screen. From the turtle library we import screen module which provides us the area where the graphical designing is done and the signals are made accordingly.

The output contains three traffic signals and each has three traffic lights namely red, yellow and green which change according to the traffic conditions. The output also contains a live feed that gives a count of the cars. The signals are drawn by the help of the turtle module The algorithm to work the signals is defined in the different modules that we integrate in this smart traffic control project. This module of OutputDemo3 is only used to create the traffic signals which consist of a rectangle, black pole and each traffic signal has 3 circles for red, yellow and green. Initially we set all the traffic signals to yellow and after the image is inputted we work on giving the proper traffic signal colors

**WORKING**

HeadText() function: First to give a head start we define a function known as HeadText using the def keyword. The colour of the drawing is set to black using the color() function in turtle module and we set the style to “courier” and font size to 14. The speed() function in turtle library is used to set the speed of drawing and the speed is set to 1000. The penup() function in turtle is used to draw a continuous line without leaving tracks and dotted lines. The position for drawing the traffic signal is set using the setposition() function in which we specify the height and the width. Text is printed below the traffic signal in “Courier” style using the write() function and it is printed in the center. The same process is repeated for drawing all the four traffic signals.

Base() function: This function is used to draw the base of the traffic signals. The base is a rectangle with length of 1 unit and breadth of 2 units positioned at (-200 + (i*100), -100) points. The color of the base is grey. This function also prints some text in color ‘black’, font ‘Courier’ and text size ‘14’. The printed text is ‘Total Cars’, ‘Passing Cars’ and ‘Time’ at positions (-300, -157), (-309, -177) and (-277, -197) respectively.

Pole() function: This function is used to draw a pole for the traffic signals. The pole is a rectangle with length 9 units and breadth 1 unit. The coordinates of the pole are (-200 + (i*100), -15). The color of the pole is also grey.

Back() function: The use of this function is to make a rectangle with length 7.6 units and breadth 2.5 units. This structure is the head of the traffic signal on which all the traffic lights are present. The color of this rectangle is grey as well. The coordinates are ((-200 + i*100), 150).

Red() function: The function takes a parameter ‘num’. This function has three turtles named pen1, pen2 and pen3. In this case pen 1 makes a ‘red’ coloured ‘circle’ of radius ‘2 units’ at coordinates (-200 + (i*100), 200) with a speed of ‘100’. The turtles pen 2 and pen 3 make a white coloured circle of radius 2 units at coordinates (-200 + (i*100), 150) and (-200 + (i*100), 100) respectively with a speed of 100.

Yellow() function: The function takes a parameter ‘num’. This function also has three turtles named pen1, pen2 and pen3. In this case pen 2 makes a ‘yellow’ coloured ‘circle’ of radius ‘2 units’ at coordinates (-200 + (i*100), 150) with a speed of ‘100’. The turtles pen 1 and pen 3 make a white coloured circle of radius 2 units at coordinates (-200 + (i*100), 200) and (-200 + (i*100), 100) respectively with a speed of 100.

Green() function: The function takes a parameter ‘num’. This function also has three turtles named pen1, pen2 and pen3. In this case pen 3 makes a ‘green’ coloured ‘circle’ of radius ‘2 units’ at coordinates (-200 + (i*100), 100) with a speed of ‘100’. The turtles pen 1 and pen 2 make a white coloured circle of radius 2 units at coordinates (-200 + (i*100), 200) and (-200 + (i*100), 150) respectively with a speed of 100.

After the circles are drawn the turtle pen 3 is now used to make a white coloured square of length 1 unit at coordinate (-200 + (i*100), -147) with a speed of 100. Now the turtle turtle is used to write ‘TCars’ with text color black, font Courier, size 14 and speed 1000 at coordinate (-200 + (i*100), -157). Now the turtle pen 3 is used again to make a white coloured square of length 1 unit at coordinate (-200 + (i*100), -167) with a speed of 100.

The turtle turtle is used to write ‘PCars’ with text color black, font Courier, size 14 and speed 1000 at coordinate (-200 + (i*100), -177). The turtle pen 3 is used again to make a white coloured square of length 1 unit at coordinate (-200 + (i*100), -187) with a speed of 100. The turtle turtle is used to write ‘Time’ with text color black, font Courier, size 14 and speed 1000 at coordinate (-200 + (i*100), -197).

Reset() function: This function is used to call the Yellow() function with the parameters 1, 2, 3 and 4 respectively.

Now A screen of 1000 * 1000 units is set up and all the functions are called in order Base(), Pole(), Back(), HeadText() and Reset().

## 4.4 Calci MODULE

In this module we calculate the time interval between each cycle (i.e. between each change of signal). We have done so by firstly importing the math algorithm to the code which helps in calculations. We have then assumed some values for normal, slightly congested, moderately congested and highly congested flow of trafficBy defining the IntervalSlab() function we have decided the time interval for each type of traffic flow. We first import the math algorithm which is important for the calculations in the code.

**WORKING**

The flow of traffic control is considered NORMAL when the number of vehicles in each lane are between 0 to 10.
The flow of traffic is considered as slightly congested when the number of cars in each lane are between 30 and 50.
The traffic in the lane is considered to be moderately congested when the number of vehicles in each lane are between 50 and 70
There is high congestion in the traffic lane if the number of vehicles on the lane are between 70 to 100.
The number of vehicles in each lane are determined by the previous module . The function IntervalSlab() is defined using the keyword def in which a parameter Val is passed. A if loop is initiated wherein, if Val>0 and Val<=10 then the time interval between each signal will be of 20s. If Val>10 and Val<=40 the time interval will be 40s and if the Val is more than 40, the time interval between the change of signal will be 60s.

## 4.5 Colors MODULE DESIGN AND WORKING

Colors module is used to assign colors to the variables. A class is made by the help of the class keyword. The colors are assigned by the help of the codes of the colors. The HEADER variable is assigned the PINK color by the help of the code HEADER='\033[95m'. The OKBLUE variable is assigned to LIGHT BLUE color by the help of the code OKBLUE='\033[94m'. The OKGREEN variable is assigned to light GREEN color by the help of the code OKGREEN='\033[92m. The WARNING variable is assigned to YELLOW color by the help of the code WARNING='\033[93m

The FAIL variable is assigned to LIGHT RED color by the help of the code OKGREEN='\033[91m. The ENDC variable is assigned to reset colors by the help of the code ENDC='\033[0m. The BOLD variable is assigned to bold the color by the help of the code BOLD='\033[1m The UNDERLINE variable is assigned to underline the text by the help of the code OKGREEN='\033[4m

## 4.6 Calling MODULE

The use of this function to make a list which shows if there is an emergency vehicle in the given lane. The list emergency[] appends 1 if there is an emergency vehicle and 0 if it is not present. We import the main() function from the onnxruntime module and also the json module is imported for reading the json file that we get from the main() function. This module takes images as an input and then searches for emergency vehicles in that image using the main function().

**WORKING**

Firstly it imports the main function from the onnxruntime module and also imports json module. Next we create a function named emmain(). Then it takes four images as an input of the four lanes respectively and performs search for emergency vehicles using main function. If there is an emergency vehicle on the image then we append 1 to the emergency list or else we append 0. At last we return the emergency list and the function emmain().

## 4.7 SlabTrafficControl Module

This function is used to allot time slots to the traffic signals of four lanes. It is also used to allot red and green signals as per the traffic conditions. It looks for the traffic conditions and checks for the number of cars present in each lane and then turns the signals red or green accordingly.
The different conditions defined in this module are normal traffic, congested traffic 1 and congested traffic 2.

**WORKING**

If there are 0-10 cars in a particular lane then the time allotted is 20 seconds and it is called normal traffic. If there are 10-40 cars in a given lane then the time allotted is 40 seconds and it is called congested traffic 1. And lastly if there more than 40 cars in the given lane then the time allotted is 60 seconds and it is called congested traffic 2.

The modules imported are algos, time, Colors, OutputDemo3, turtle and calci. The output of this function is that which lane gets which signal and for how much time respectively.
NormalTrafficS(): This function takes the List[] as an input which contains the total number of cars in each lane. The Gtime i.e the time for which it is given a green signal, for normal condition is set to 20 seconds. Now it checks if there are any cars present in that lane and if there are present then it turns the signal green for that lane and red for all the other lanes. After the given time, the green signal is changed to yellow.
CongestedTrafficS(): This function also takes the List as an input which contains the number of cars in each lane. It creates a list named CheckList with four elements all set to zero initially and a variable LowCount initially set to zero.

The first case checks if the number of cars in the List is greater than zero and less than 3, if so, then it turns the signal for that lane to green. If this condition is satisfied, CheckList value for that lane is changed to 1 and the LowCount is also incremented by 1. Other signals are turned to red for that duration of time. After the given time, the green signal is changed to first yellow and then red.
MaxIndex is set as the maximum value of list using GetEMaxCarIndex. Next is the condition when only one lane is left. It checks if there are three zeroes in the List. The variable Req is set as the non-zero value of the list and the time given to clear the traffic is the maximum value between the number of cars*2 and 10. Passable cars are the minimum of number of cars and Gtime/2. This signal is turned green while others are turned red. After the given time, the green signal is changed to yellow. Then the number of cars in the List is made zero and the value at the index of the lane left in the CheckList is changed to 1.

The next condition is for the case when there is only one lane left to be changed in the CheckList. It checks if there are three ones in the CheckList and accordingly decides
that the lane with value 0 should be given a preference to turn to green and others are set to red.

If the value of the MaxIndex has been changed to 1 in the CheckList, then the next step is to set the maximum index of the List to the second maximum index and if the value of the second maximum index too, has changed to 1 then the maximum index of the List is changed to the third maximum index.

The normal body is now brought into execution if the above conditions do not apply and a specific time to the lane in the order is allotted. The Gtime i.e the time for which it is given a green signal, for normal condition is set to the minimum value between IntervalSlab(List[MaxIndex]) and List[MaxIndex] * 2 (maximum number of cars * 2). The Gtime is decided as the maximum between the Gtime calculated in the previous step and 10 seconds. Now it turns that signal to green for that lane and red for all the other lanes. After the given time, the green signal is changed to yellow. The List is updated by minimizing the number of passing cars and the MaxIndex is updated to the next maximum value with the function GetEMaxCarIndex.

## 4.8 <u>Main MODULE</u>

We have imported all the important algorithms from different files that will be needed for the calculation and display of the traffic signals. At the beginning of the main code we import some algorithms from different files. First, we import the random file to the main code. Following which, we import functions GetMaxCarIndex, NormalTrafficVar, CongestedTrafficVar, Turtle,Screen, Base, Back, Pole, HeadText, Reset, NormalTrafficS, CongestedTrafficS, NormalTrafficT, CongestedTrafficT, main and emmain from the files Algos, TrafficControlVar, turtle, OutputDemo3, SlabTrafficControl, TrafficControl, project and calling respectively.

**WORKING**

We have assumed that there are on an average 20 cars crossing the intersection within a time span of 40 seconds. Considering the assumption, we assign the variable NormVal to 10.

From the images taken at junction 1, 2, 3 and 4 we take the input values for further calculations. Assuming that there are 20 cars crossing the intersection in 40sec, we assign the variable NormVal to 10.
The parameter List is assigned to the main() function which stores all the major calculations of the program. The elements of the List are multiplied by a random integer between 1 and 4 to provide varying results in every execution. MaxIndex is set as the maximum value of list using GetMaxCarIndex. In the main function, a for loop is executed which returns the number of cars at each junction/lane. A screen is set up to display the output using the turtle module. When the max cars are registered, another if loop is executed which informs us which lane should follow which form of execution i.e. NormalTraffic or CongestedTraffic based on the maximum number of cars among the four lanes. Finally, the signals are reset and the program ends.

## 4.9 Object_detection MODULE AND WORKING

This is the module that has been exported from Microsoft Azure. The basic function of this module is to detect for an emergency vehicle, in our case, ambulance and fire brigade van. As we train our model by feeding it with images that we need to detect, it writes a code that runs behind the scenes and detects our vehicle. The object_detection module contains code provided to us by azure. This module imports the numpy and the math libraries from python. This module has a class ObjectDetection which takes object variable as an input and has several functions defined in it. The functions defined are , __init__(self, labels, prob_threshold=0.75, max_detections = 20), logistic(self,x),non_maximum_suppression(self,boxes,class_probs, max_detections), extract_bb(self,prediction_output,anchors),predict_image(self,image), preprocess(self, image), def predict(self, preprocessed_inputs), postprocess(self, prediction_outputs). This module returns the probability percentage of the vehicle that it has detected and gives how much confidence it has on the image detected. We can also change the probability of detection as we want. In our case it is 75 percent. This module is then called in the final onnxruntime_predict module.

## 4.10 Onnxruntime_Predict MODULE AND WORKING

This module is also directly exported from Microsoft Azure. This is the final working of the previous module and this module itself. This module exports the os, sys, onnxruntime, onnx, numpy libraries from python. It also exports Image and ImageDraw from the PIL library of python. It then exports the objectdetection class from OBJECT_DETECTION module and tempfile. This is a model created and exported as onnx because it is compatible with windows operating system. The primary steps are resizing the image to a desired width for high resolution and then giving that image as the input. There is also a text file which contains the tags that are used in the image while detection, in our case it is ‘emergency’. There is a class defined in this module named ONNXRuntimeObjectDetection which takes the class ObjectDetection as a parameter which means it takes all its functions as parameters. The functions defined in this class are __init__(self, model_filename, labels) and predict(self, pre-processed_image). The last function in this module that is main() takes the image file name as a parameter. This function integrates both the modules and provides a tagged image as an output.

# 5. PERFORMANCE ANALYSIS

## 5.1 INTRODUCTION

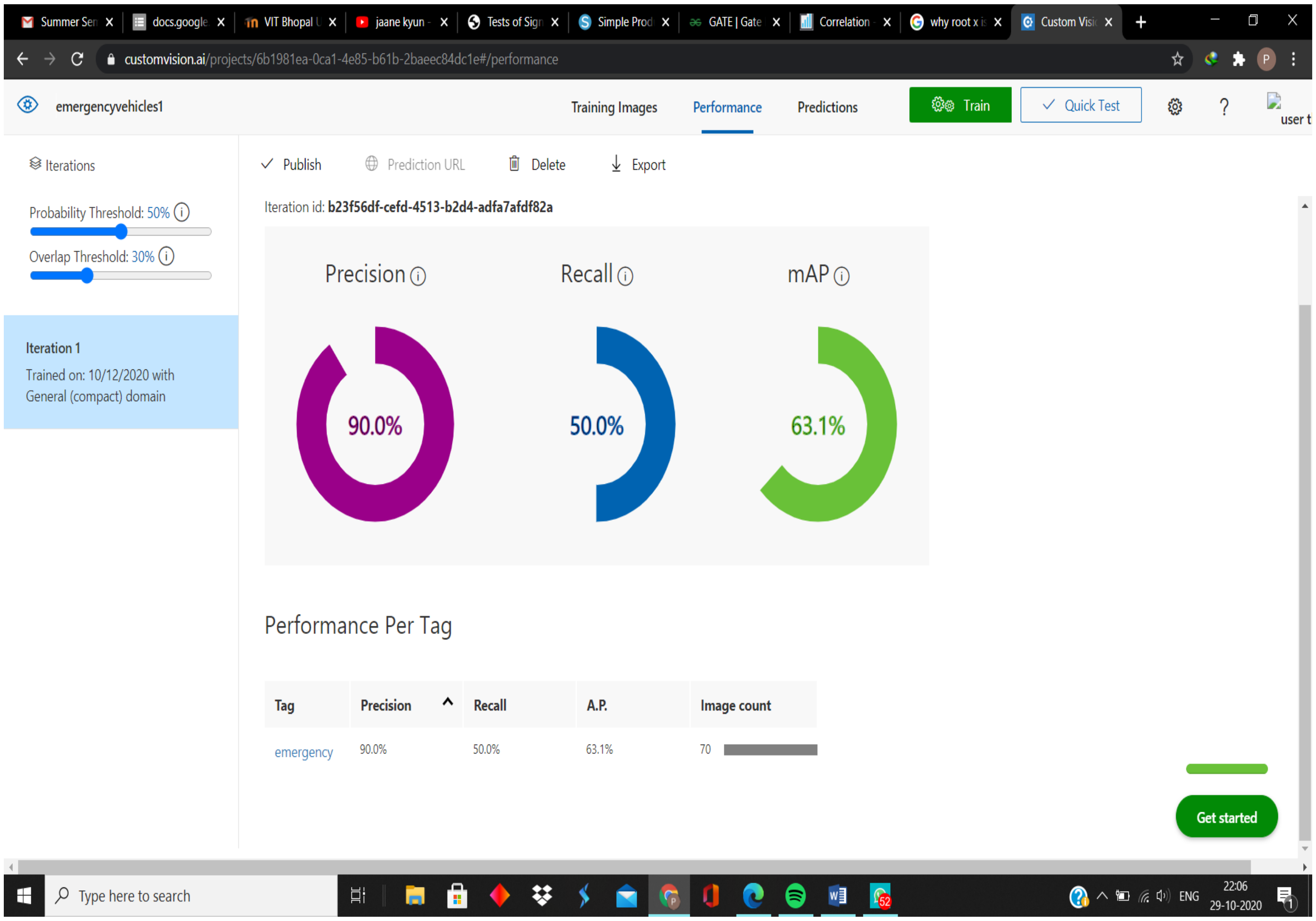
Performance analysis is the technique of studying or comparing the performance of a specific situation in contrast to the aim and yet executed. We will discuss about the efficiency of our smart traffic control system in this section. Our system is capable of car, truck, motorcycle, person and emergency vehicle individually on a busy Indian road. We will also discuss about the model developed in azure for the detection of emergency vehicles. Our model detects for an ambulance and fire brigade van as of now thus fulfilling maximum needs. The model also detects and can count the number of people walking on the road and some on the data can be of the great help to the government. The performance analysis is discussed further.

## 5.2 PERFORMANCE MEASURES

The performance measure is the process of collecting and analyzing data, thus giving an appropriate measure or count of how efficient a system is. In our case, the system detects and differentiates between cars, trucks, motorcycles, emergency vehicles and persons on a fairly busy Indian road. The system faces certain difficulties when people don’t wear helmets while riding a motorcycle, then it detects them as person instead of motorcycles. Speaking of the azure modules they detect emergency vehicles with a precision of 90 percent, recall of 50 percent and a mean average precision of 63.1 percent. the azure model was fed with 70 images.

## 5.3 PERFORMANCE ANALYSIS

There are a several factors that affect performance analysis. When people don’t wear helmets while riding a motorcycle then sometimes it detects motorcycle as persons thus reducing the signal time and also the overall performance measure. The training model in azure was fed with 70 images giving a precision of 90 percent, this can be increased by feeding it with more images and also increasing the overall performance.



Probability Threshold: 50%

Overlap Threshold: 30%

Iteration 1
Trained on: 10/12/2020 with General (compact) domain

Iteration id: b23f56df-cefd-4513-b2d4-adfa7afdf82a

Precision 90.0%

Recall 50.0%

mAP 63.1%

Performance Per Tag

| Tag | Precision | Recall | A.P. | Image count |
|---|---|---|---|---|
| emergency | 90.0% | 50.0% | 63.1% | 70 |

Fig. 5.3.1 AZURE TRAINING MODEL

## 5.4 CONCLUSION

The overall performance analysis suggests that the above system is ready for implementation on the Indian roads or any roads and it can be proved as a feasible solution considering both performance and cost. The model can and is capable of increasing its performance measure based on the current results. Thus, the model we have developed is a SELF LEARNING MODEL. The model is ready to come in use and make the city into a smart city with less human intervention and less cost factor.

# 6 FUTURE ENHANCEMENTS

## 6.1 INTRODUCTION

The smart traffic control system can be of great help on the roads of India and abroad and it is a step closer to the smart cities to which India is aiming to achieve. The smart traffic control system does not involve any human interaction and thus the task of humans can be reduced to making it more comfortable for humans and more accurate as compared to humans. It will be error free and thus the model is efficient to be installed on roads. This model is easy to install and also keeping in mind the cost factor involved it is easily affordable and very less hardware tools are used in the smooth functioning of this system. Overall, the future scope of this model in the implementation sector is very high as India needs such a type of control system where there is ample of traffic and the people are stuck in a congestion for lots of time.

Smart Cities are the agenda on which all the state governments are working very fast and our model is a perfect ingredient in the making of smart city. High accuracy, efficient traffic management system and less human interaction makes it very feasible for the human race and also helps us to reduce time and reach the destination quickly and without any chaos and hinderances. Thus, the smart traffic recognition system is a great help in the development of India.

## 6.2 LIMITATIONS

The model that we have made has an accuracy of 70% which means that the model is 90% accurate only in detecting an ambulance/fire brigade. Also, the system will allow any emergency vehicle to pass through without checking that if it is actually on its duty or not. The emergency vehicles that are in a rush have a buzzer on the top which blinks and makes a noise but the system we have made does not check the buzzer light is on/off. Even if the light is off the model would allow the emergency vehicle to pass through it which is unfair to other vehicles in other lanes. Another limitation in this is that cycles and bikes are both detected as motorcycles and there is no separate classification for the same.

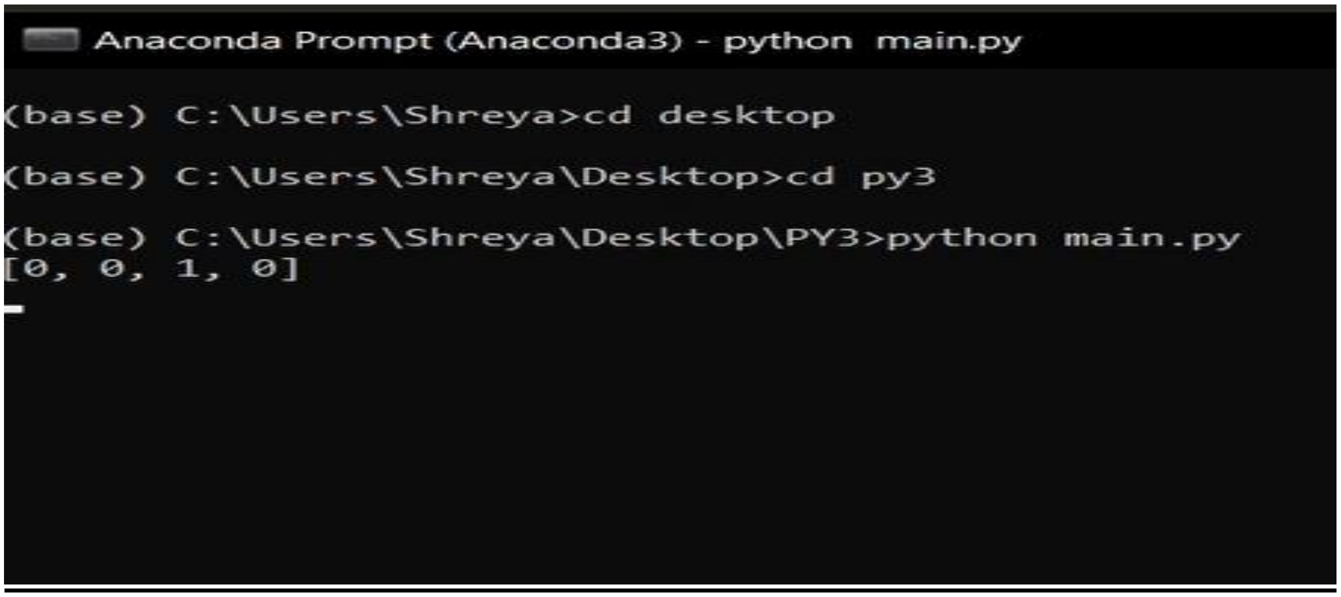
## 6.3 FUTURE ENHANCEMENTS

- The model that we have made of the smart traffic control system has an accuracy of 90%. The model is trained using 35 ambulance and 35 fire brigade images which gives us an accuracy of 90%. Thus, we can work on increasing the accuracy of the model by giving more images in order to train the model to be very precise and exact.
- The model that we have prepared is just for one traffic signal. We can in future enhance the smart traffic management system and establish a wireless connection between multiple signals. This means that if an emergency vehicle is given a priority at one signal, the next signal should be informed that an emergency vehicle is incoming and thus the vehicle should be quickly allowed to pass.

- The smart traffic management system segregates the vehicles into 4 parts namely cars, buses, trucks and motorcycles. A cycle and a bike both are detected in the motorcycle section. We can work on segregating and different detections of both the cycle and the bike.
- In India there is a major problem that the riders in the country do not follow the safety measures and rules designed by the government. People on motorcycles drive at a very high speed and without helmets. The model can be trained in such a way that if the camera catches any bike rider without a helmet, the number plate will be scanned and the biker can be fairly penalized in such a way.
- The smart traffic management system can also be designed to capture the speed of the vehicles passing by the signal and if the speed is too fast the number plate of the vehicle can be scanned and the vehicle owner can be penalised for the same.
- The model can also guide an emergency vehicle to the right path quickly which means that on detection of an emergency vehicle it will inform the hospital/fire spot that an ambulance/fire brigade is incoming and the people present on the spot can make their preparations in that way and also the roads can be cleared so that they reach their spot quickly.

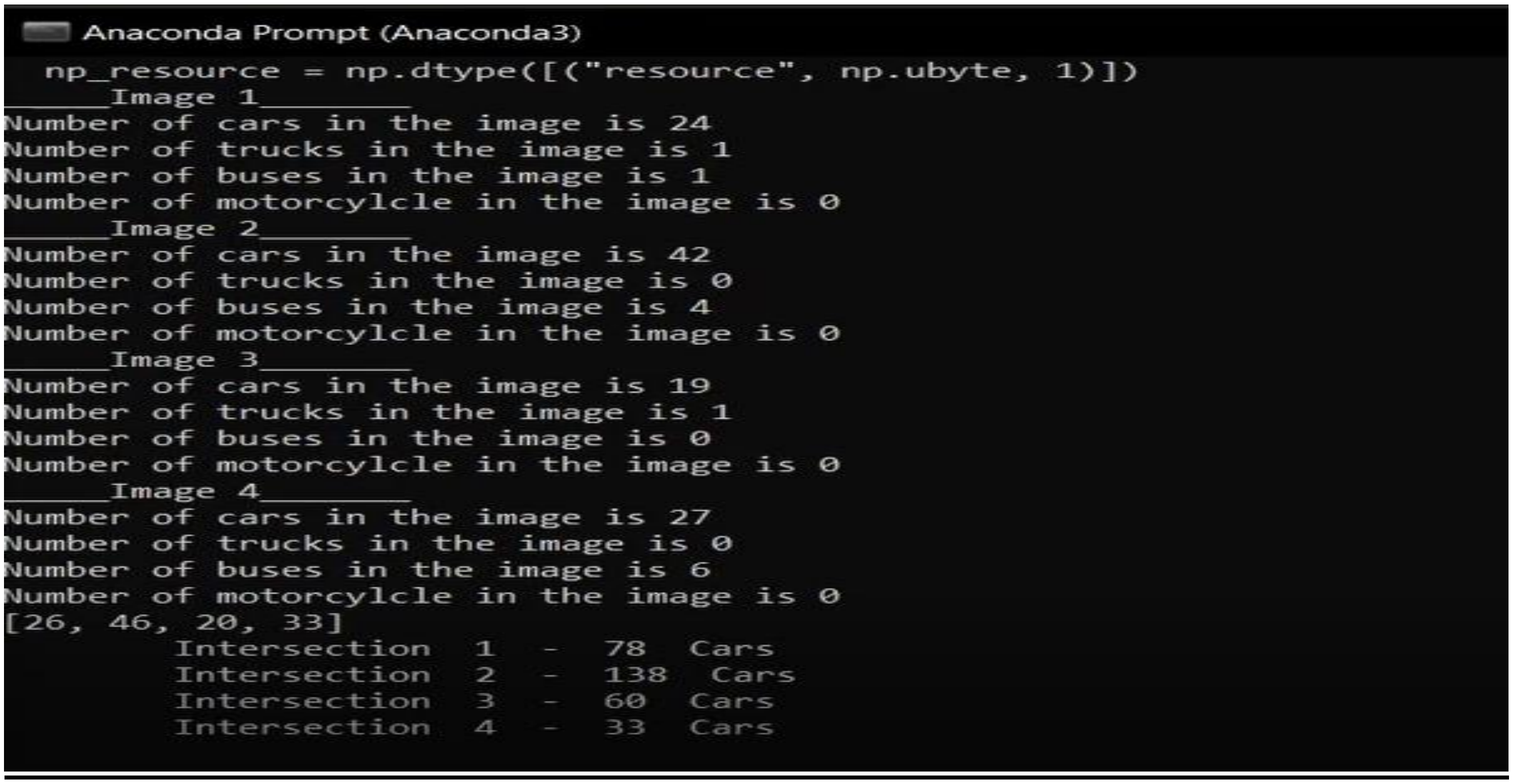
# 7. OUTPUT

## SOME OUTPUT IMAGES



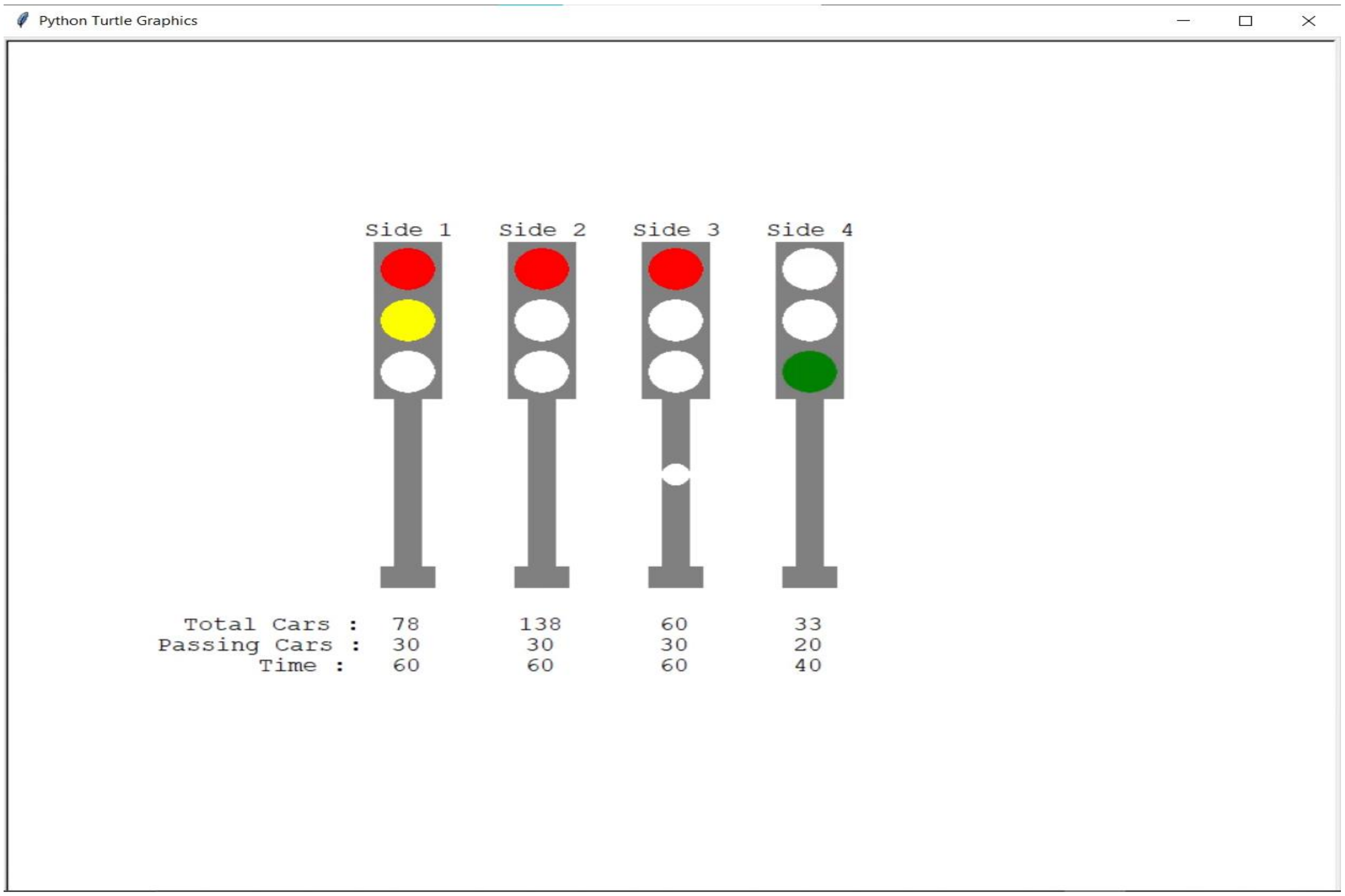
```
Anaconda Prompt (Anaconda3) - python  main.py

(base) C:\Users\Shreya>cd desktop

(base) C:\Users\Shreya\Desktop>cd py3

(base) C:\Users\Shreya\Desktop\PY3>python main.py
[0, 0, 1, 0]
_
```

**Fig. 7.1.1 MAIN MODULE OUTPUT**



```
  np_resource = np.dtype([("resource", np.ubyte, 1)])
______Image 1_________
Number of cars in the image is 24
Number of trucks in the image is 1
Number of buses in the image is 1
Number of motorcylcle in the image is 0
______Image 2_________
Number of cars in the image is 42
Number of trucks in the image is 0
Number of buses in the image is 4
Number of motorcylcle in the image is 0
______Image 3_________
Number of cars in the image is 19
Number of trucks in the image is 1
Number of buses in the image is 0
Number of motorcylcle in the image is 0
______Image 4_________
Number of cars in the image is 27
Number of trucks in the image is 0
Number of buses in the image is 6
Number of motorcylcle in the image is 0
[26, 46, 20, 33]
        Intersection  1  -  78  Cars
        Intersection  2  -  138  Cars
        Intersection  3  -  60  Cars
        Intersection  4  -  33  Cars
```

**Fig. 7.2 PROJECT MODULE OUTPUT**



```
   Total Cars :   78       138      60       33
 Passing Cars :   30       30       30       20
         Time :   60       60       60       40
```

**Fig. 7.3 FINAL OUTPUT OF THE SYSTEM FROM VIDEO**

# 8. APPENDIX A-ABOUT MICROSOFT AZURE

Azure is a cloud computing service provided by Microsoft. It can be used for various purposes and has many facilities to provide. In this project we use the CUSTOM VISION API provided as service of azure to train a model and detect emergency vehicles. We fed our model with several photos of ambulance and fire brigade vans in India and showed it how to detect them as emergency vehicles. Basically azure does all the behind the scenes work that is it develops a code as we train our model.

This code was further exported from azure and integrated with our project in order to detect the presence of emergency vehicles. The lane with emergency vehicles is then given preference over other lanes. The modules onnxruntime_predict.py() and object_detection.py() are exported from azure and they detect the presence of emergency vehicles using image detection. The tag provided by us to ambulance and fire brigade vans is “emergency”. We can also set the threshold value for detection and in our case it is set to 75 percent. Thus these both modules detect the presence of emergency vehicles and return their presence in the image.

Modules Onnxruntime_predict and Object_detection are used to train the model in Microsoft Azure and are exported from Microsoft Azure. We have ourselves trained the module using Azure Services and our model is a self-training model with 90% accuracy.

# 9. **REFERENCES**

1. Stevanovic, Aleksandar & Kergaye, Cameron & Stevanovic, Jelka. (2011) “Evaluating Robustness of Signal Timings for Varying Traffic Flows”. Transportation Research Record: Journal of the Transportation Research Board. 2259. 141-150. 10.3141/2259-13.

2. Athmaraman, Naren & Soundararajan, Srivathsan. (2005). “Adaptive Predictive Traffic Timer Control Algorithm”.

3. Jin, Junchen & Kalaie, Ahmad. (2017). “A group-based traffic signal control with adaptive learning ability”. Engineering Applications of Artificial Intelligence. 65. 282–293. 10.1016/j.engappai.2017.07.022.

4. Sharma, Ishant & Gupta, Pardeep. (2017). STUDY OF AUTOMATIC TRAFFIC SIGNAL SYSTEM FOR CHANDIGARH. 10.13140/RG.2.2.28390.83526.

5. Emin Basic, Yasser De Jesus and Ashrafur Rahman, "Adaptive Signal Control Technology: Current Practice and Comparison", *Intelligent Transportation Society of Connecticut Student Research Project*, 2012.

6. Vadrevu S. V. S. R. Pavan Kumar and Dr. M. Kamala kumara, "A Novel Application of Adaptive Traffic Control System for India", *International Journal of Science Engineering and Technology Research (IJSETR)*, vol. 5, no. 7, July 2016.

7. Viswanathan, V. and Santhanam, V. (2013). Traffic signal control using wireless sensor networks. 2nd International Conference on Advances in Electrical and Electronics Engineering (ICAEE'2013).