

Intelligent Traffic Light System using Digital Image Processing



By

GC Safdar Shaukat

GC Saad Ali Khan

GC Yousaf Ejaz

FC Saddam Abdus Salam

Submitted to the Faculty of Electrical (Telecomm) Engineering

National University of Sciences and Technology, Islamabad in partial fulfillment

For the requirements of a B.E. Degree in Electrical (Telecomm) Engineering

June 2017

ABSTRACT

Traffic is one of the major concern for almost every country in the world nowadays. As the urban area traffic congestion increases, its control has become a huge problem. The existing way to control traffic signals is by using a timer or manually or simply using electronic sensors to detect vehicles, so there is a vital need for the introduction of advanced traffic control algorithms to deal with this ever increasing problem. We will be making a system to control traffic lights using digital image processing techniques. It will use images to compute the density of vehicles. We will have cameras mounted on the traffic lights that will produce a sequence of images which will then be analyzed for vehicle detection using different digital image processing techniques. Then according to the conditions of traffic on the road the traffic lights will be controlled.

DECLARATION

We hereby declare that none of the content of our work presented in the thesis has been submitted for some other award of qualification or degree either in this or anywhere else in another institution.

CERTIFICATE FOR CORRECTNESS AND APPROVAL

It is hereby certified that the form of the report and the contents of the project “**Intelligent Traffic Light System using Digital Image Processing**” being submitted by the syndicate of students

1. GC Safdar Shaukat
2. GC Saad Ali Khan
3. GC Yousaf Ejaz
4. FC Saddam Abdus Salam

Have been written well and found satisfactory as per the requirements of the B.E. Degree in Electrical (Telecom) Engineering.

APPROVED BY

Col Dr. Imran Touqir

Electrical Engineering Department
Military College of Signals (NUST)

DATED: 13 June 2017

DEDICATION

Almighty Allah,
Family, relatives and friends for their support
And faculty for their help.

ACKNOWLEDGEMENTS

We thank Allah Almighty for bestowing his countless blessing upon us without which nothing could have been achieved.

We are also most thankful and grateful to our parents especially our mothers who bore with us in times of difficulty and hardships and were such a support. Without their consistent support and encouragement, we would never have achieved the targets successfully.

We would like to specially thank to our supervisor, Col Dr. Imran Touqir for his admirable support, guidance, motivation and critical reviews throughout the course of our project. We would also like to thank the faculty for being there for us whenever we needed help of any kind.

Table of Contents

1	INTRODUCTION	2
1.1	Background	2
1.1.1	Traffic controlling systems	2
1.1.2	Digital Image Processing	2
1.1.3	Microcontroller	3
1.1.4	Problems with existing technologies	4
1.2	Problem statement	5
1.3	Proposed solution	5
2	LITERATURE REVIEW	8
2.1	Introduction to MATLAB	8
2.2	Traffic Light System	8
2.3	Scope of work.....	9
2.4	Image Acquisition:	9
2.5	RGB to gray conversion.....	9
2.6	Image enhancement.....	10
2.6.1	Image enhancement techniques	11
2.7	Edge detection	11
2.7.1	Edge detection techniques.....	12
2.8	Image Matching.....	13
3	SYSTEM REQUIREMENT SPECIFICATION.....	15
3.1	Components.....	15
3.1.1	Hardware.....	15
3.1.2	Software	17
4	SYSTEM DESIGN SPECIFICATION	19

4.1	Working Points.....	19
4.2	Working Process	19
4.3	Software Process	20
4.3.1	Methodology.....	21
4.4	Hardware	22
4.4.1	Image sensors.....	22
4.4.2	Computer.....	22
4.4.3	Platform.....	22
4.4.4	Hardware Design	23
4.5	Objectives.....	24
4.5.1	General Objectives.....	24
4.5.2	Academic Objectives	24
4.6	Special Skills Required	24
4.7	Work Done	24
4.7.1	Software:.....	24
4.7.2	Working of Software programs:	26
4.7.3	Hardware:.....	30
4.8	Visual Indications.....	32
4.9	MATLAB indications:	33
4.10	Deliverables.....	35
5	Applications.....	37
5.1	Traffic Assistance.....	37
5.2	Emergency Conditions	37
5.3	Reduction of Manpower.....	37
5.4	Wastage of Time	37

5.5	Security System.....	37
6	FUTURE WORK.....	39
7	CONCLUSION	41
8	RESOURCES REQUIRED	43
9	REFERENCES.....	45
10	BIBLIOGRAPHY	47
11	GLOSSARY	49

TABLE OF FIGURES:

Figure 1: Microcontroller.....	3
Figure 2: Microcontroller II.....	3
Figure 3: RGB to Gray.....	10
Figure 4: Image Enhancement	11
Figure 5: Edge Detection	12
Figure 6: Web Camera	15
Figure 7: Microcontroller.....	15
Figure 8: Interfacing Cables.....	16
Figure 9: Signal Unit.....	16
Figure 10: Cars.....	16
Figure 11: Matlab.....	17
Figure 12: Working Process.....	19
Figure 13: Software Process	21
Figure 14: Hardware Process	23
Figure 15: Camera 1.....	26
Figure 16: Camera 2.....	26
Figure 17: Camera 3.....	26
Figure 18: Camera 4.....	26
Figure 19: Command Window.....	27
Figure 20: Camera 2.....	27
Figure 21: Camera 1.....	27
Figure 22: Camera 4.....	28
Figure 23: Camera 3.....	28
Figure 24: Command Window.....	28
Figure 25: Camera 2.....	29
Figure 26: Camera 1.....	29
Figure 27: Command Window.....	29
Figure 28: Camera 4.....	29
Figure 29: Camera 3.....	29
Figure 30: Empty Road.....	30

Figure 31: Crowded Road.....	31
Figure 32: Ambulance.....	32
Figure 33: Final Hardware	35

TABLE OF TABLES:

Table 1: Hardware LCD Display	33
Table 2: Matlab Command Window Display	34

CHAPTER: 1
INTRODUCTION

1 INTRODUCTION

1.1 Background

Due to increasing population in urban areas traffic congestion becomes an increasing problem. This congestion of traffic causes wastage of time, fuel consumption and many environmental concerns. Road accident is another main problem in modern world.

Solution to these traffic problems lies in abiding by the traffic laws, rules and traffic signs along road. Traffic rules and regulations govern vehicles and directs them how to drive and behave on road. Road signs along the road are erected to provide instructions to the drivers.

1.1.1 Traffic controlling systems

1.1.1.1 *Manual controlling*

Manual controlling is the method of controlling traffic with the help of traffic police or the traffic wardens. Traffic wardens will use aids such as sign boards, sign lights or whistle to control the traffic lights. This requires more manpower and hectic duty as we would need at least one warden standing at every junction in the city to regulate the traffic flow. Due to the shortage of the traffic wardens traffic cannot be controlled manually.

1.1.1.2 *Automatic controlling*

Timers and electrical sensors are used in automatic controlling. A predefined value is set at the timers in the traditional light systems. Electrical sensors are usually embedded at the sides of the roads or the pavement and they detect the vehicles on the road and produce signal for the timer according to the density of the road. This technique has not been an efficient method as it is has a high tendency to be effected by noise and it is expensive.

1.1.2 Digital Image Processing

Digital image processing is in itself a complete subject. In short it is the use of computer algorithms to perform image processing on digital images. Digital image processing is better than analog image processing in many ways. Wider range of algorithms can be applied to the input data and problems such as signal distortion and build up noise can be avoided.

In image Processing we enhance the raw form of images received from different sources. Deals with the issues related to image presentation its data compression and involves many other complex operations.

1.1.3 Microcontroller

A microcontroller is a single integrated circuit consisting a processor core, memory, and programmable input/output peripherals designed for embedded applications. Microcontrollers are used in automatically controlled products, devices and embedded systems. Here is how a microcontroller looks like;

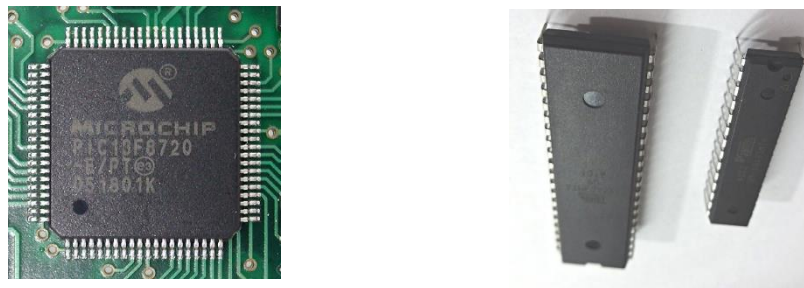


Figure 1: Microcontroller

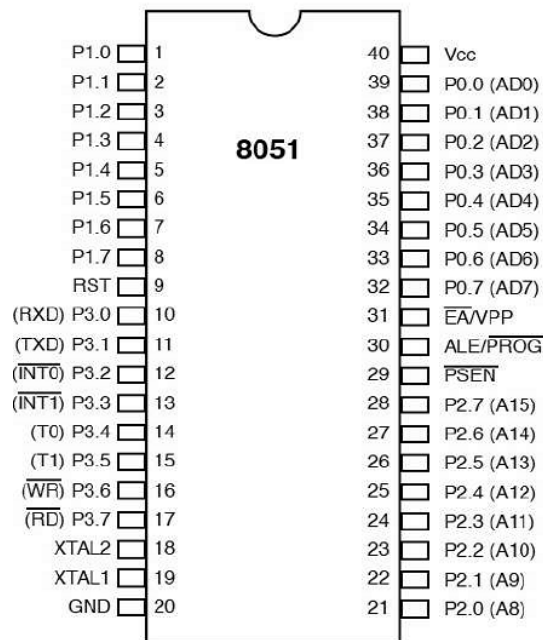


Figure 2: Microcontroller II

1.1.4 Problems with existing technologies

Some of the latest and mostly adopted technologies are.

1.1.4.1 *Infrared Detectors*

IR sensors are mainly of two types:

- Active: They operate by emitting energy from either LED or a laser diode.
- Passive: They detect energy emitted by objects in the line of sight.

Their disadvantages are they can be sensitive to changing weather conditions and ambient light. The nature of the detecting material and other materials used in construction of different systems.

1.1.4.2 *Ultrasonic detectors*

Japan is widely using ultrasonic sensors in traffic applications however not adopted in USA as they are focused on inductive loop detectors. There are two types of ultrasonic sensors available, presence-only and speed measuring. Their advantages are that they are all weather resistant and can be fixed or mounted above the road. The disadvantages are:

- Should be mounted in a manner that it is facing down.
- There is difficulty in identifying lane-straddling traffic and Traffic which is traveling side by side.
- Also it has susceptibility to high wind speed.

1.1.4.3 *Piezoelectric*

Currently being tested in Virginia, since they are very accurate detectors, but they lack in detecting the presence of static vehicle, unless if its wheels are stopped on the detector. It consists of a long strip of piezoelectric material in some protective casing.

The disadvantage to use this technology is they must be embedded in the pavement i.e. permanent installations. Whenever the roadway is repaved or some hole occurs, the sensor might get damage and need to replace.

1.1.4.4 *Inductive loop detectors*

This is the most widely used technology in USA .It mainly consists of: a loop, loop extension cable and a detector. Every time the inductance of the loop changes when a vehicle passes or stays over the loop, results in detection to be signaled in the control box.

Their disadvantages are that they are very sensitive to the installation process and we need to reinstall them every time a road is repaved.

Some other technologies are:

- Microwave radar
- PSDA(Passive Acoustic Detector Arrays)
- Photoelectric detectors
- Magnetic detectors
- Acceleration detectors

1.2 Problem statement

In Pakistan, the traditional system is used for controlling traffic lights, we can either see wardens at places or the simple timer traffic lights. There are many limitations in these methods which have led to the ever increasing traffic congestion problems. Timing is not based on the vehicle density due to which we have the following drawbacks;

1. Heavy traffic jams in urban cities.
2. Wastage of time as the green light remains ON for empty road.
3. More man power required.
4. Inductors or sensors used are sensitive to the installation process, must be reinstalled every time road is repaved or pothole appears. They are expensive and have high tendency to be affected by noise and weather conditions.

1.3 Proposed solution

We propose the use of image processing for controlling the traffic lights. The vehicles will be detected by cameras i.e. the images will be taken instead of using the electrical sensors to know the density of traffic on the road. Cameras will be mounted on the traffic lights and real time images will be taken. This is a better technique for controlling the change in

state of the traffic light. This technique can reduce the traffic congestion problems to the minimum and also avoid time being wasted by a green light on an empty road. Moreover image processing is a more practical and realistic approach towards solving this problem as it takes the actual traffic images.

CHAPTER: 2
LITERATURE REVIEW

2 LITERATURE REVIEW

2.1 Introduction to MATLAB

MATLAB stands for Matrix laboratory and it's a high-performance modern programming language. MATLAB was initially designed to give access to matrix software made by LINPACK and EISPACK projects.

Strengths of MATLAB are:

- Base on C program, easy to write, plot, run.
- Widely adopt, easy to find functions to do difficult tasks such as sorting, numerical analysis.
- Extremely powerful when you deal with arrays.
- Can build GUI, normal people don't need to see code behind your program.

Some limitations of MATLAB are:

- Slow compared to C, but we can integrate C and MATLAB code together to speed things up

2.2 Traffic Light System

A traffic light system is a standard electronic device system which directs the pedestrians and traffic at a junction or intersection by displaying red, yellow and green colored signals. An addition, it also displays pedestrians signs that helps them crossing the roads. Traffic signals are placed in order to assure safe drive and ride.

Nowadays most traffic signals will have the following components or part:

- Main display with red, yellow and green lights.
- Traffic signal cabinet containing the traffic signal controller and Vehicle Detection Systems, either
- Inductive loops or sensors

2.3 Scope of work

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 of both empty and crowded road. The image sequence will then be analyzed and compared using digital image processing (RGB to Gray conversion) for vehicle detection using edge detection technique, and according to traffic conditions on the road traffic light can be controlled.

2.4 Image Acquisition:

For any image at any point say f the amplitude of image is intensity of the image also the gray level of image at that point. To form a digital image we have to convert those x and y values into some finite discrete values. However in order to process through computers we need to convert analogue image to digital image. Digital images comprised of discrete and finite elements we called pixels.

Image Scaling and resizing:

Whenever you scale or resize any image from one pixel to another we need image resizing in order to increase or decrease the total number of pixels. Every camera has its own resolution and so we need to resize the image so when we exquisite a picture from one camera it has resolution issues with other camera since it has its own camera specifications. So it is important to do image scaling or resizing to make the resolution similar for all applications.

2.5 RGB to gray conversion

Humans perceive color through wavelength-sensitive sensory cells called cones. The cones are sensitive to green, red and blue light. While any image displayed on any electronic device like computers, mobiles or digital cameras they are the combination of these three colors and by varying their amount. This is the RGB color model. In all RGB encodings, red, green and blue are the first, second and third values respectively i-e (255,255,255).

Whereas in grayscale or grayscale image the value of the image is stored in a single sample which means it carries only intensity information. Such images are also known as black and white and are mostly made up of shades of gray varying from black to white weakest

to strongest respectively. Let say, white is (255,255,255), black is (0, 0, 0) and medium gray is (127,127,127). The higher the numbers, the lighter the gray.

To convert from RGB to gray we take the average: $(R+G+B)/3$. However most commonly way is the weighted average, e.g.: $0.3R + 0.59G + 0.11B$. Below is the RGB to Gray conversion table:



















Pure Red (255,0,0)		Equivalent Gray (76,76,76)	
Pure Green (0,255,0)		Equivalent Gray (150,150,150)	
Pure Blue (0,0,255)		Equivalent Gray (29,29,29)	
Cyan (0,255,255)		Equivalent Gray (179,179,179)	
Magenta (255,0,255)		Equivalent Gray (105,105,105)	
Yellow (255,255,0)		Equivalent Gray (226,226,226)	
Brown (158,85,54)		Equivalent Gray (103,103,103)	
Olive (155,160,52)		Equivalent Gray (146,146,146)	
Purple (100,0,150)		Equivalent Gray (47,47,47)	

Figure 3: RGB to Gray

2.6 Image enhancement

Digital images are adjusted in the process of image enhancement so that the results are more suitable for further analysis and display. Different image enhancement techniques allows the users to improve the SNR ratio and modify the features of an image. Following operations are carried out in image enhancement:

- De-blurring of an image
- Color management of the device independently
- Transformation of an image
- Conversion of an image



Figure 4: Image Enhancement

2.6.1 Image enhancement techniques

Following are some of the most widely used techniques of image enhancement;

- Median Filtering
- Histogram equalization
- Power law transformation

2.7 Edge detection

One of the fundamental tools of digital image processing is edge detection. The main aim of this tool is to identify the points in a digital image at which the image brightness changes sharply or, more formally, has discontinuities. Step detection is a similar tool which is used in finding the discontinuities in ID signals. The aim of detecting sharp changes in image brightness is to seize important details. The image discontinues and brightness are mostly correspond to;

- Non uniformity in depth,
- Non uniformity in the orientation of surface,
- Material properties changes.

In the ideal case, we will get a set of curved images as a result of applying edge detection to an image that indicates the boundaries of the objects, the boundaries of surface markings as well as curves that correspond to discontinuities in surface orientation. Hence, this technique allows us to filter out the useful and relevant information, while preserving the important structural properties of an image as shown in figure 7.



Figure 5: Edge Detection

2.7.1 Edge detection techniques

Following are the most widely used edge detection techniques:

- Sobel operator:
- Prewitt's operator
- Canny

The Canny Edge Detector is one of the most widely used image processing tools detecting edges in a very robust manner. Canny edge detection technique is based on three basic objectives.

1. Low error rate
2. Edge point should be well localized
3. Single edge point response

2.8 Image Matching

Image matching is a technique used correlation that relates two images here we are using a different method for image matching. There is a reference image as well as crowded image upon comparing them with the real time image we gets the matching results. Image matching has too many advantages related decision making and is one of the nest techniques for algorithms besides some disadvantages regarding pixels based matching.

Percentage of matching is expressed as

$$*Percentage\ matching = Number\ of\ pixels\ matched\ successfully / Total\ no.of\ pixels*$$

CHAPTER: 3
SYSTEM REQUIREMENT
SPECIFICATION

3 SYSTEM REQUIREMENT SPECIFICATION

This part of the document contains information about the product, its features, objectives and constraints.

3.1 Components

3.1.1 Hardware

3.1.1.1 *Web Camera 2.0*

4 x Web Cameras are used for taking the image sequences at all four roads of the crossing. The camera will take images of the roads at different intervals and send the result to matlab where the images will be processed



Figure 6: Web Camera

3.1.1.2 *Microcontroller*

Microcontroller used here is Intel 8051, it's a single chip complex instructions set microcontroller use in embedded systems. The initial versions of its by Intel were famous in the 1980s and early 1990s and enhanced binary suitable derivative. This microcontroller 8051 is used to programme the timings of the traffic lights.



Figure 7: Microcontroller

3.1.1.3 *Interfacing Cables*

Hardware and software are interfaced through interfacing cables.



Figure 8: Interfacing Cables

3.1.1.4 *Signal Unit*

These signals have been used just to give a demonstration of the real scenario.

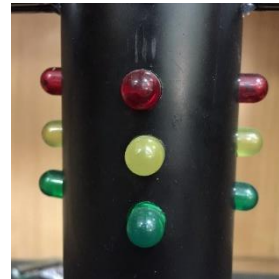


Figure 9: Signal Unit

3.1.1.5 *Cars*

Toy cars have been used for demonstration of the real case. Different cases have been tested using these toy cars.

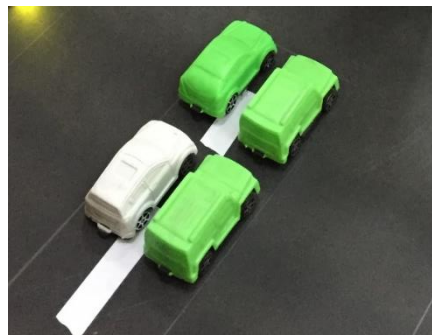


Figure 10: Cars

3.1.2 Software

3.1.2.1 *Matlab*

MATLAB stands for Matrix laboratory and it's a high-performance modern programming language. MATLAB was initially designed to give access to matrix software made by LINPACK and EISPACK projects.

We have used Matlab coding to make this project work in its true sense.

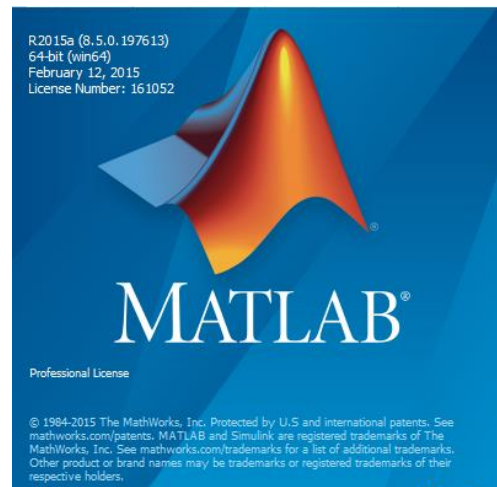


Figure 11: Matlab

3.1.2.2 *Keil*

Keil is a compiler in which we write and compiled machine language code. After compilation, this is converted into hex code and then burn for further processing. Keil also compiles C language code. The programs were made using online help as the 8051 microcontroller uses open source software.

CHAPTER: 4
SYSTEM DESIGN SPECIFICATION

4 SYSTEM DESIGN SPECIFICATION

4.1 Working Points

In the present work the designed system aims to achieve the following.

- Distinguish the presence and absence of vehicles in road images;
 1. When the road is empty turns the signal to red.
 2. When there is traffic on road the traffic light turn to go green and according to the density of traffic on road its duration will be adjusted accordingly.
- For emergency condition like some ambulance or any protocol car, it will be having priority over all the signals.

This project consist of:

- Hardware module
- Software programming
- Interfacing

4.2 Working Process

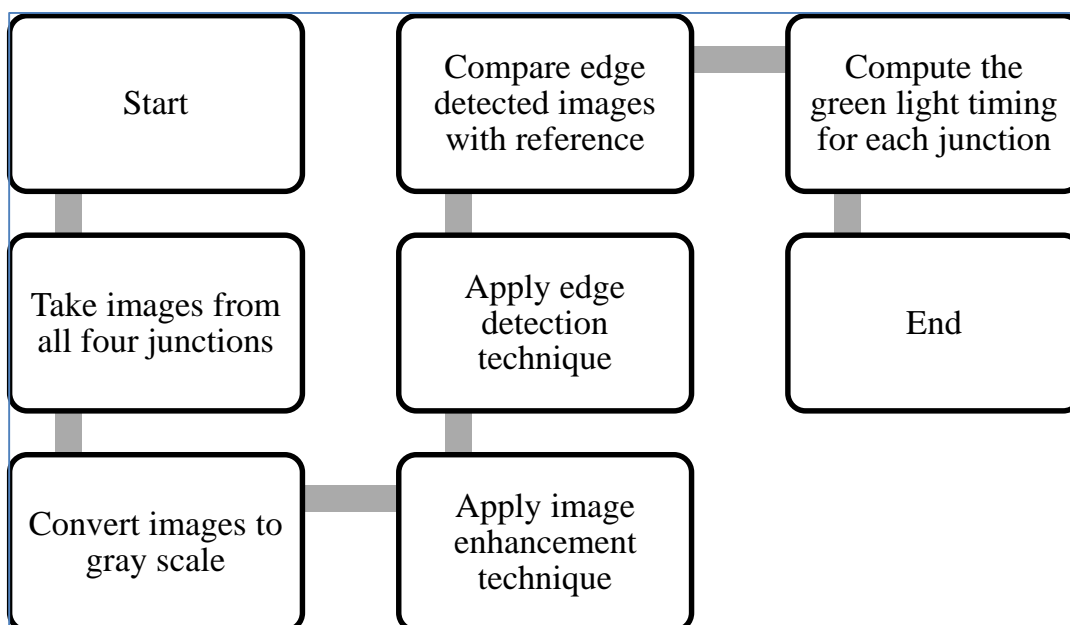


Figure 12: Working Process

This project mainly consists of VGA web cameras that are mounted over signals and act as a source input and directs images to processing device, which after processing all the images do image matching and sends the output to the microcontroller which directs signals accordingly in addition to the emergency conditions.

Following are the general working steps:

- The cameras will take the images from all four junctions.
- For monitoring of cameras there will be window in MATLAB.
- Firstly the empty road image is stored as reference image after this crowded road image is taken in each turn.
- These images will then converted into grayscale.
- Image enhancement will then be applied to reduce noise from the image.
- For the detection of the cars edge detection will be applied on the enhanced image.
- To compute the density on the road we then will compare the reference and crowded image.
- Compute the timing for green light on the basis of percentage matching for each junction.
- Pass these results to microcontroller to operate the signals.

4.3 Software Process

Following are the algorithms applied on an image. This will give an overview of how we will be able to solve the traffic congestion problems.

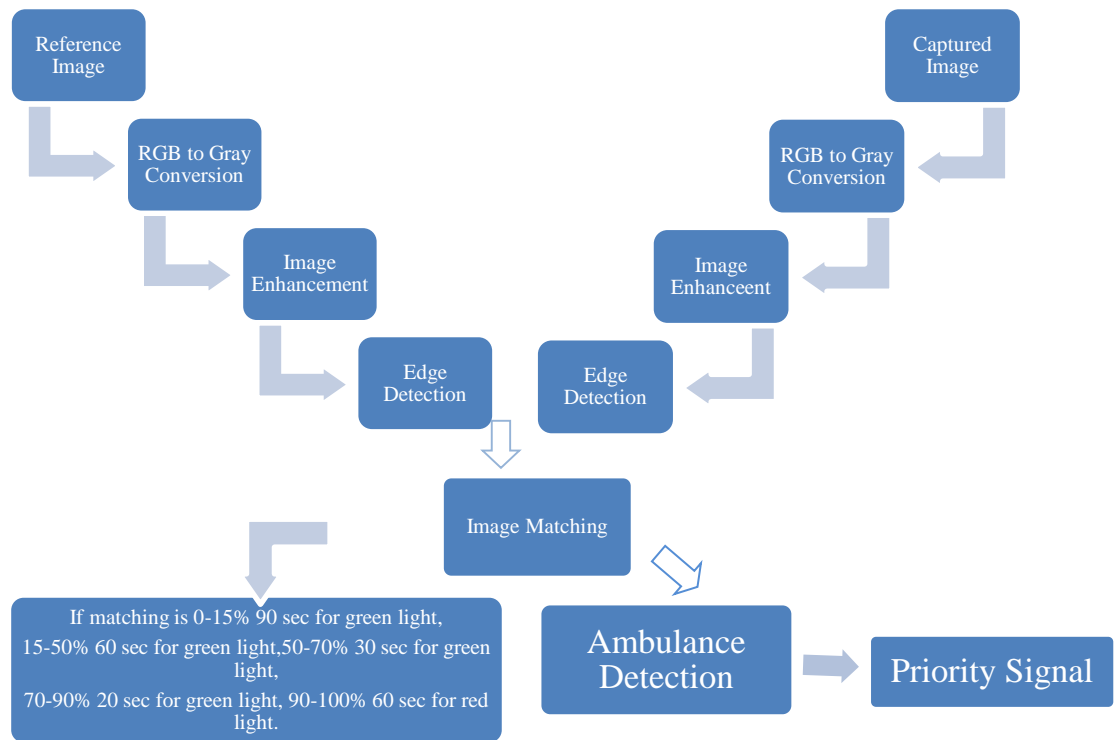


Figure 13: Software Process

4.3.1 Methodology

Following are the generalized main steps involved in the image processing:

- Image acquisition
- RGB to gray conversion
- Image enhancement
- Image matching using edge detection

We have divided our project into three phases which are as follows:

- **Phase1:**
 - Firstly image is taken by using camera.
 - Initially image of the empty road is captured as reference image.
 - This reference image is then converted to gray scale from RGB.
 - Then this Gray scale image is having image enhancement.

- Thereafter by using canny method we do edge detection on this reference image.
- **Phase2:**
 - Images of the crowded road are captured,
 - These sequence of images is then converted to gray scale.
 - Image enhancement is done on each of the captured gray image,
 - Thereafter by using canny method we do edge detection of these real time images of the road.
- **Phase3:**

Now we compare both the images and traffic lights can be controlled based on percentage of matching. However if there is any emergency condition like some ambulance or any protocol car then it will be having priority over all the signals.

On the basis of image matching we will control the traffic lights i-e :

- For matching 0-15% - green light will be on for 90 secs.
- For matching 15-30% - green light will be on for 60 secs.
- For matching 30-60% - green light will be on for 30 secs.
- For matching 60-90% - green light will be on for 20 secs.
- For matching 90-100% - green light will be on for 10 secs

4.4 Hardware

4.4.1 Image sensors

Instead of image detection sensors we have placed USB based web cameras.

4.4.2 Computer

As a central controlling unit we are using PC to perform various image processing tasks.

4.4.3 Platform

To demonstrate the working of our algorithm we have made platform consisting toy cars and Signal units using LEDs.

4.4.4 Hardware Design

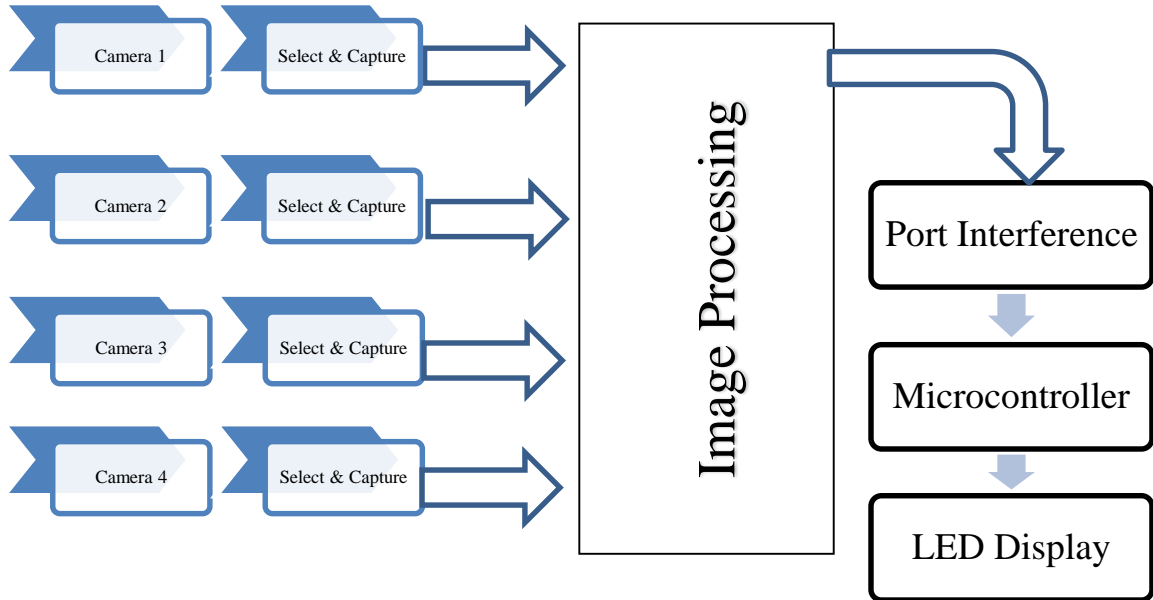


Figure 14: Hardware Process

4.4.4.1 Methodology

The hardware process is as follows:

- VGA Webcam i.e. USB based are installed on all directions,
- They will capture the image of both the empty and the crowded road,
- These images will then be processed by image processing,
- A microcontroller is further fed with the output through parallel port cable which is programmed as to turn on the specific LED for definite time

On the top of every signal we have placed camera to capture images of the particular side of road. When it captures the image it will be analyzed the traffic and matches it with reference image then it will be decided and respective signal will be awarded that time.

Emergency:

In case of any emergency only the authorized vehicles like ambulances, firefighters or police will be allowed to cross the signal. The signals will remain open till the time the protocol cars will cross the signal then resume the normal operation. However if there is case more than one emergency arises then they will be served on priority basis i-e first come will be served first and so on.

4.5 Objectives

4.5.1 General Objectives

Based on image processing develop an intelligent traffic light control system to calculate the traffic density on the road and set the timer accordingly.

4.5.2 Academic Objectives

Academic objectives of this project are:

- Development of an Intelligent Transportation System (ITS)
- To select the suitable image enhancement and edge detection techniques suitable for the vehicle detection & density calculations,
- To implement the image processing techniques & simulate the result.

4.6 Special Skills Required

The special skills required for this project is the understanding of:

- **MATLAB** Programming expertise
- C++ programming

4.7 Work Done

4.7.1 Software:

The software requirements of the project were completed through the

- MATLAB 2014b
- Keil μ Vision5

MATLAB: MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming environment. Here the purpose of using MATLAB is its program are easy to write, plot, run .Also Widely adopt, easy to find functions to do difficult tasks such as sorting, numerical analysis. Especially extremely powerful when you deal with arrays. MATLAB programming is the backbone here, it processes the image detects the cars and count them.

The objective to regulate the traffic according their density and give priority signal to the ambulance is achieve as:

Through cameras mounted over signals we took the input image which is in RGB .It processes in clockwise fashion and every signal will be given its chance on its turn. When there is the turn of signal then on every turn it takes the image and sends as input to the camera.

To process further it needs to be converted into GRAY scale, so we first convert it into grayscale, it's because we have to define the threshold. To convert R-G-B value to gray, we know that each pixel has different R-G-B parameter, consider each and obtain reflected single color. This gray scale image needs to be enhanced so that to remove the noise factor from the image. Image enhancement is required for more appropriate results. Edge detection is a set of mathematical methods which aim at identifying points at which the image brightness changes sharply, has discontinuities or noise. Value of brightness varies for different colors.

Now to detect the cars present on road we have various techniques are Sober Edge Detection, Prewitt Edge Detection and canny Edge Detection. Pixels are located in image which corresponds to boundaries of cars detected in the image, which results to binary image of edge pixels detected.

Keil μ Vision5: Keil is a compiler in which we write and compiled machine language code. After compilation, this is translated into hex code and then burn for further processing. Keil

also compiles C language code. The programs were made using online help as the 8051 microcontroller uses open source software.

4.7.2 Working of Software programs:

Programs attached in Annex A of the document.

After completing the code of our project we have performed some tests in real time depicting real life scenarios. Until now we have used the webcam of the laptop to depict one side of the road and do our experiments on one side. Below we will be showing different scenarios.

4.7.2.1 Empty Road

When the road is empty and there is no vehicle on the road the pixel change i-e we have used “obj_Present” is less than 100 which is a set threshold. By using edge detection technique we will know that the road is empty and the signal will remain red.

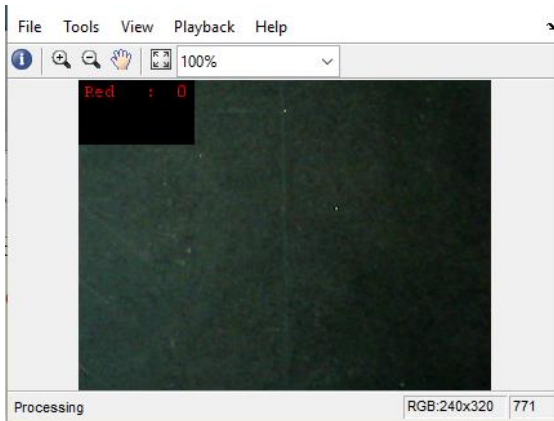


Figure 15: Camera 1

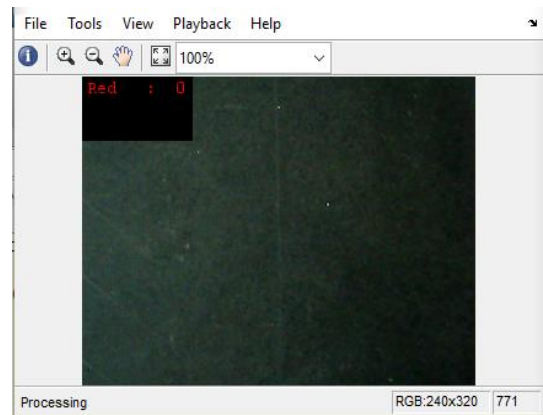


Figure 16: Camera 2

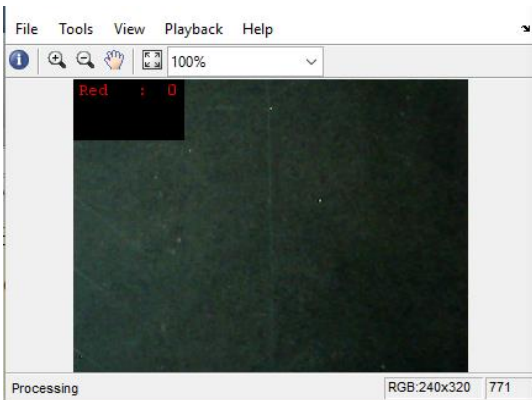


Figure 17: Camera 3

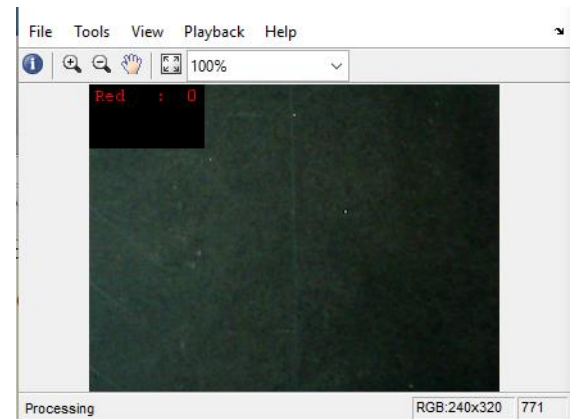


Figure 18: Camera 4

```

569     %% Clearing Memory
570 -    release(hVideoIn); % Release all memory and buffer used
571 -    release(hVideoIn1);

```

Command Window

New to MATLAB? See resources for [Getting Started](#).

```

objX =

     0     0     0

array =

S1ES2ES3E

```

Figure 19: Command Window

4.7.2.2 Crowded Road

The object size is also defined hence anything smaller than that will also not be detected as a vehicle or an object. Maximum time limit is set so that if there is still vehicles present on road it will switch to next signal. Now as we can see that as the density of on signal is more hence that side will be given more time and the other side will be given less time.

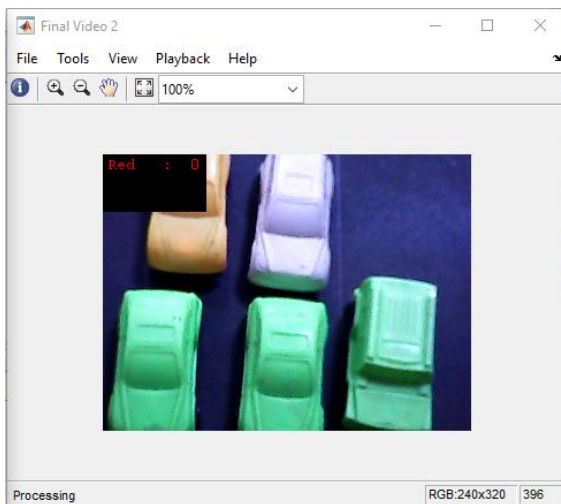


Figure 21: Camera 1

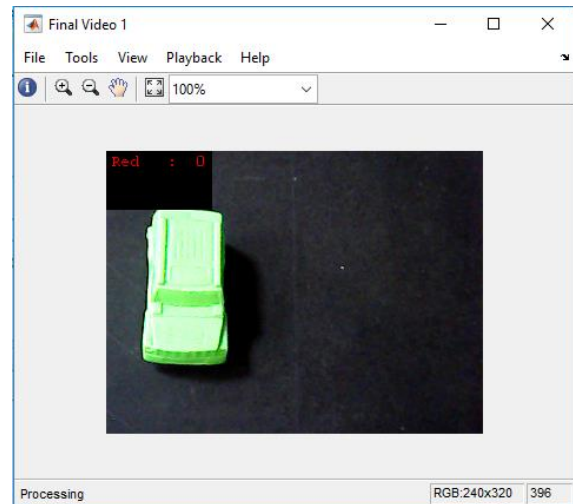


Figure 20: Camera 2

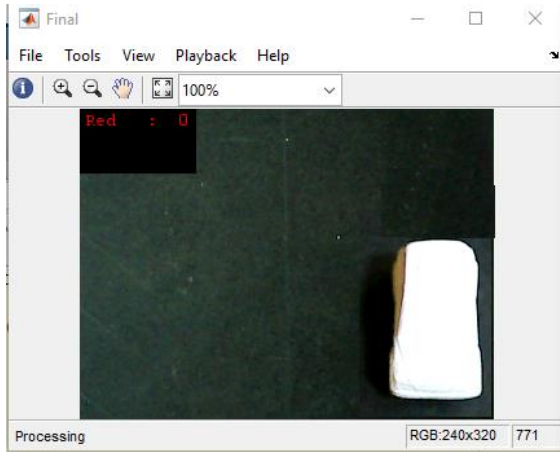


Figure 23: Camera 3

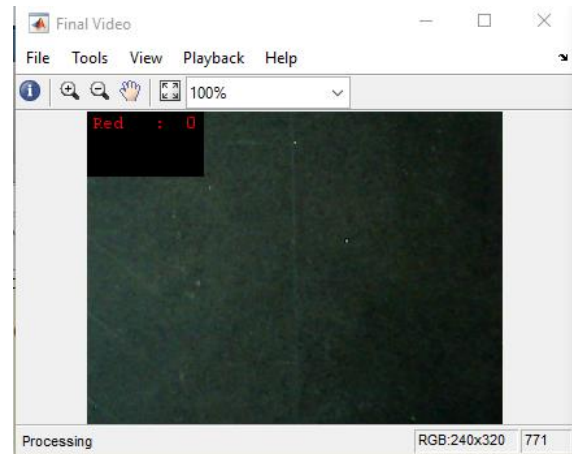


Figure 22: Camera 4

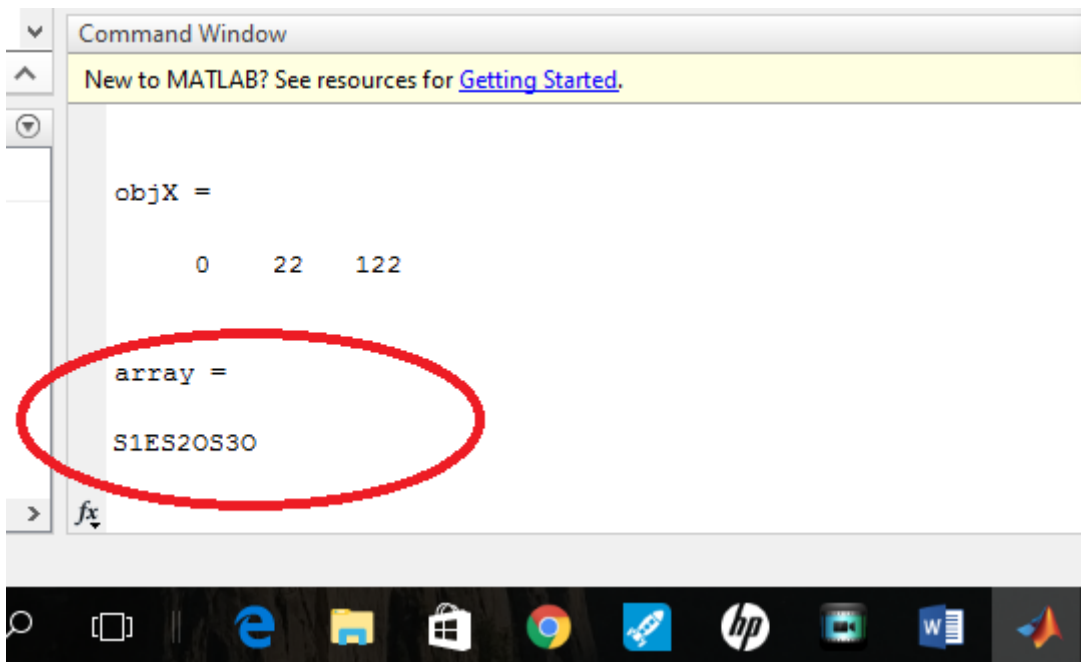


Figure 24: Command Window

4.7.2.3 Ambulance Detection

A special case for ambulance has been added in our project. The cameras will detect the ambulance by its light, in that case any protocol care will be given priority such as the police etc. as soon as the camers detect the light of the ambulance, it will treat it as an emergency case and turn the signal green for the ambulance to pass.

Now when a red colour ambulance comes it will be detected as a different entity and will be given priority over any other vehicle hence all other signals will turn red and the one which detected the ambulance will turn green.

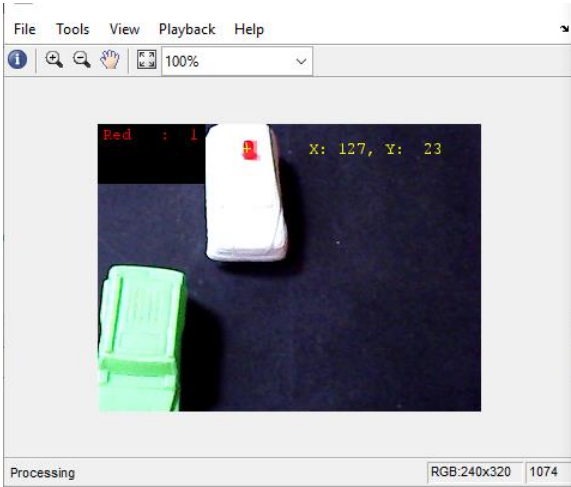


Figure 26: Camera 1

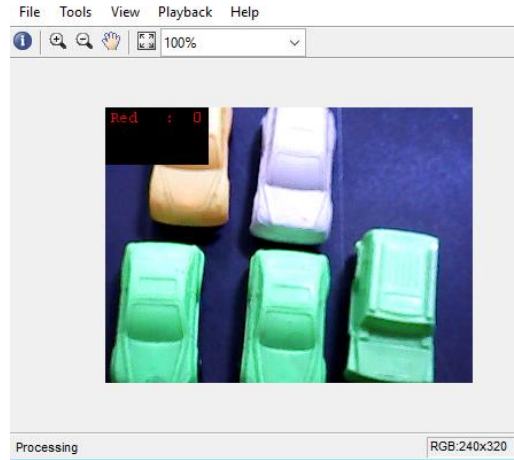


Figure 25: Camera 2

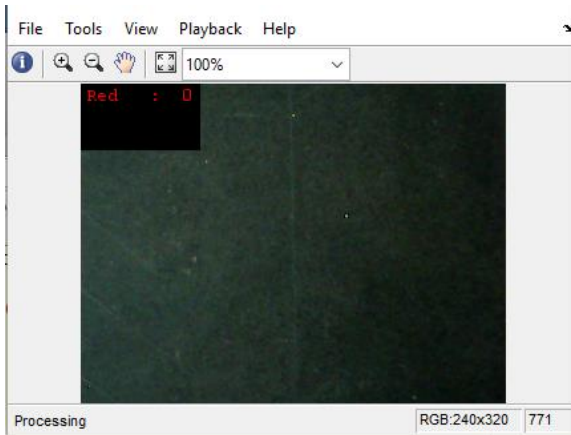


Figure 29: Camera 3

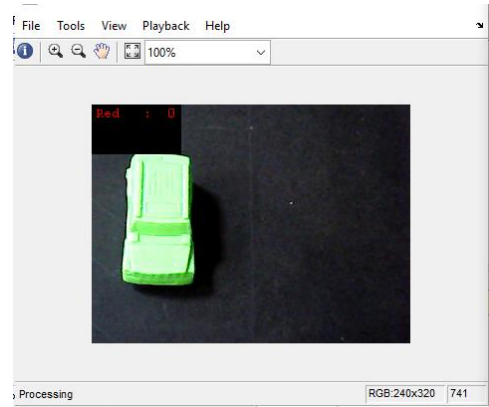
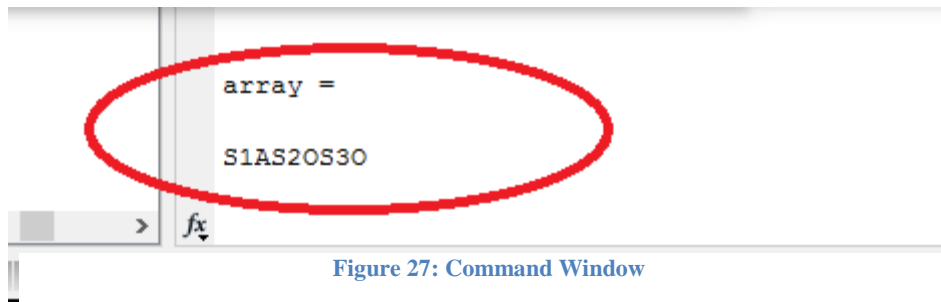


Figure 28: Camera 4



4.7.3 Hardware:

Hardware implementation of the project has been completed, which includes proper modelling of traffic junction depicting real time scenarios. In hardware model consists of:

- USB Based web cameras 2.0
- Traffic signal model
- Interfacing through serial port
- Cars
- Microcontroller 8051
- USB Extension cables

Here as above the three cases are discussed:

4.7.3.1 *Empty road*

When the road is empty and there is no vehicle on the road the pixel change then every will have its turn but turns back to red since detects no object.

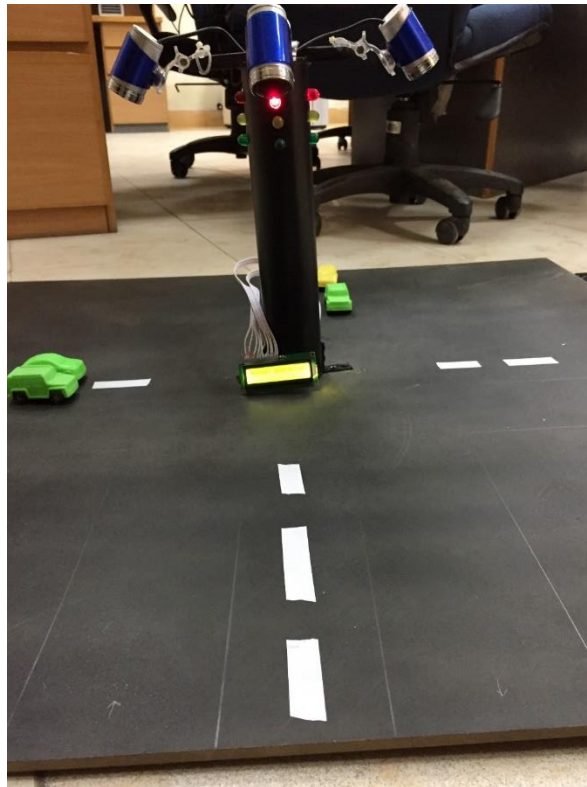


Figure 30: Empty Road

4.7.3.2 *Crowded Road*

The object size is also defined hence anything smaller than that will also not be detected as a vehicle or an object. Maximum time limit is set so that if there is still vehicles present on road it will switch to next signal. Now as we can see that as the density of on signal is more hence that side will be given more time and the other side will be given less time.

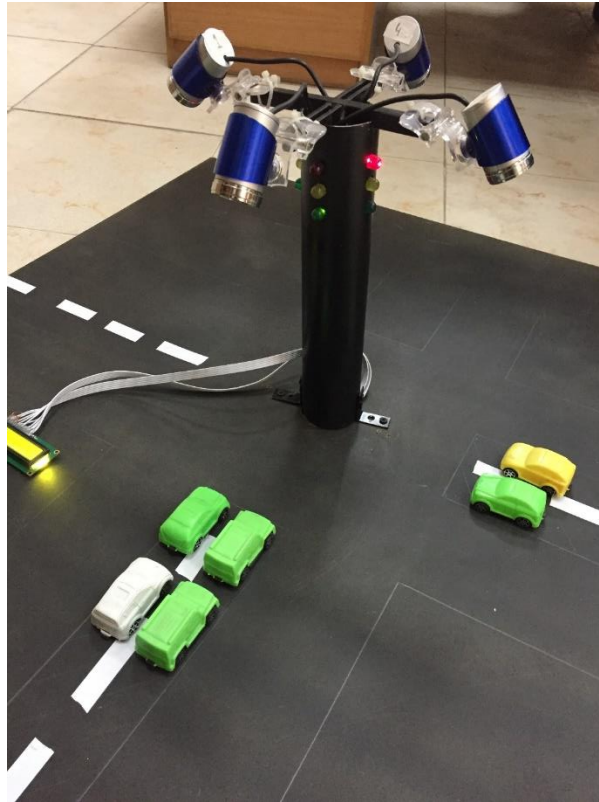


Figure 31: Crowded Road

4.7.3.3 *Emergency on Road*

A special case for ambulance has been added in our project. The cameras will detect the ambulance by its light, in that case any protocol car will be given priority such as the police etc. as soon as the cameras detect the light of the ambulance, it will treat it as an emergency case and turn the signal green for the ambulance to pass.

Now when a red color ambulance comes it will be detected as a different entity and will be given priority over any other vehicle hence all other signals will turn red and the one which detected the ambulance will turn green

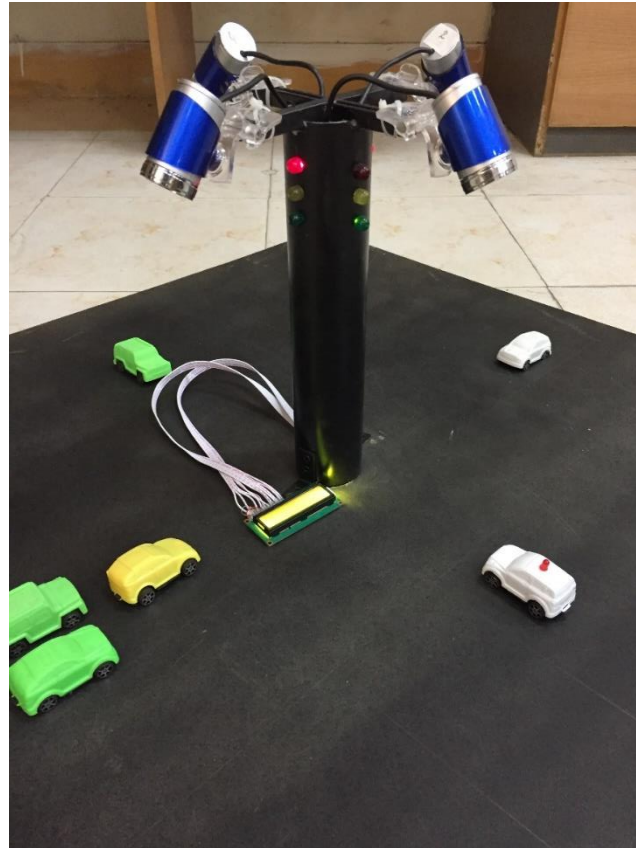



Figure 32: Ambulance

4.8 Visual Indications

S.NO.	Indication	Implication	Display
1.	Initializing	When the program is processing for the first time and	



		whenever it restarts.	
2.	Normal operation	When there is normal operation i-e road is empty or when there is traffic on road.	
3.	Ambulance detection	Whenever there is Ambulance or any other protocol car on any signal then it would be displayed along with the signal indication.	

Table 1: Hardware LCD Display

4.9 MATLAB indications:

S.No	Indication	Implication	Display
1.	Collecting data	When the program is initializing, collecting cameras input	

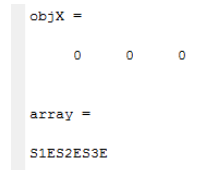
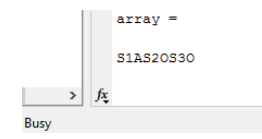
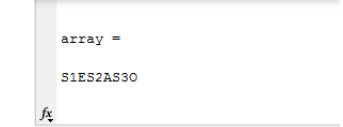
2.	Empty road	When there is no traffic on road then the command window of matlab will be continuously displaying the status of road i-e 'E' tells road is empty.	 <pre>objX = 0 0 0 array = S1ES2ES3E</pre>
3.	Crowded road	But when there is traffic either less or more the command window will show us the status of every road i-e 'O' tells there is traffic on road.	 <pre>array = S1A52OS3O</pre> <p>Busy</p>
4.	Ambulance	Whenever there is ambulance on any road it will be display as 'A' in command window	 <pre>array = S1ES2AS3O</pre>

Table 2: Matlab Command Window Display

4.10 Deliverables

Intelligent traffic light system based on image processing implemented on Matlab that directs the traffic by their images taken from camera and having special assistance to the ambulances.

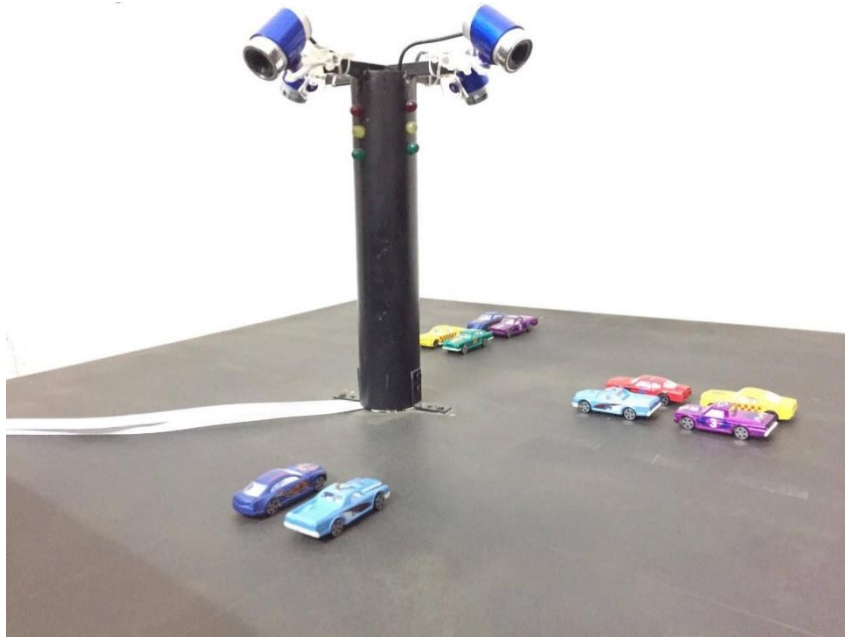


Figure 33: Final Hardware

CHAPTER: 5
APPLICATIONS

5 Applications

5.1 Traffic Assistance

Due to increasing number of vehicles leads to traffic jams mainly in urban cities. Image based traffic controlling will provide special traffic assistance to these overly crowded signals.

5.2 Emergency Conditions

Ambulances or protocol cars will be having special assistance in this scenario, as they will be having priority on all other vehicles.

5.3 Reduction of Manpower

Since we have pitiable strength of traffic police men we cannot control traffic manually in all areas of a city or town. So the better solution is to automate the signals.

5.4 Wastage of Time

Time being wasted due to green light remains ON for empty road and as the signals are in queue so both the pedestrians and vehicles have to wait for their turn.

5.5 Security System

This system will minimize the chances of accidents as the vehicles know that there is no loop hole to carry on and also these cameras can provide us the security.

CHAPTER: 6
FUTURE WORK

6 FUTURE WORK

In future, we can use Raspberry pi microcontroller and there will be no need to have separate OpenCV software. By using raspberry pi the traffic control room will be provided a view so in case if there is some issue with the car or there is need of giving more time to a signal in required area.

CHAPTER: 7
CONCLUSION

7 CONCLUSION

The study showed that image processing is a better technique to control the state change of the traffic light. Traffic management using image processing eliminates all the shortcomings of earlier standard systems used for controlling traffic. Excessive manpower is required in manual controlling, while timer is used in automatic controlling had a drawback of time being wasted by green light on a empty road. Image processing eliminates all these pitfalls. This technique is far more effective in traffic control. It diminishes the use of extra hardware devices like sensors, wireless routers, GSM modems, setup for a monitoring station etc. Presence of vehicles detection is consistent as we are using actual images of traffic here. Reality is visualized and hence, functionality is more effective and efficient than all techniques. Major advantage is the variation in signal time which control appropriate traffic density using Image matching. It achieves near perfect accuracy and performance of system is remarkable.

CHAPTER: 8

RESOURCES REQUIRED

8 RESOURCES REQUIRED

Ser. No	Item Required	Quantity	Estimate Cost (Rs)
1.	VGA webcam (Logitech C160)	4	8000
2.	USB extension cable	5	2500
3.	USB hub	2	1000
4.	Microcontroller	1	500
5.	Traffic Model with LED display	1	1500
6.	Wooden board (3x3)	1	1500
		Total	15000

CHAPTER: 9
REFERENCES

9 REFERENCES

- i. <http://www.slideshare.net/louiseantonio58/image-processing-based-intelligent-traffic-control-systemmatlab-gui>
- ii. <http://www.ijcse.com/docs/IJCSE11-02-01-031.pdf>
- iii. <http://www.rroj.com/open-access/density-based-traffic-signal-system.pdf>
- iv. <http://ijsetr.org/wp-content/uploads/2014/04/IJSETR-VOL-3-ISSUE-4-1010-1014.pdf>
- v. <http://www.had2know.com/technology/rgb-to-gray-scale-converter.html>
- vi. http://www.irdindia.in/journal_ijaeef/pdf/vol2_iss5/19.pdf
- vii. <https://theses.lib.vt.edu/theses/available/etd-43098-201311/unrestricted/APPENDIX-B2.PDF>
- viii. <http://ieeexplore.ieee.org/document/5941662>

CHAPTER: 10
BIBLIOGRAPHY

10 BIBLIOGRAPHY

- i. David Beymer, Philip McLauchlan, Benn Coifman, and Jitendra Malik, "A real-time computer vision system for measuring traffic parameters," IEEE Conf. On Computer Vision and Pattern Recognition, pp495 -501, 1997.
- ii. M. Fathy and M. Y. Siyal, "An image detection technique based on morphological edge detection and background differencing for real-time traffic analysis," Pattern Recognition Letters, vol. 16, pp. 1321-1330, Dec 1995.
- iii. "Traffic light control system simulation through vehicle detection using image processing" By Mac Michael B. Reyes and Dr Eliezer A. Albaccea
- iv. N. J. Ferrier, S. M. Rowe, A. Blake, "Real-time traffic monitoring," Proceedings of the Second IEEE Workshop on Applications of Computer Vision, pp.81 -88, 1994.
- v. V. Kastinaki, M. Zervakis, and K. Kalaitzakis, "A survey of video processing techniques for traffic applications," Image and Vision Computing, vol. 21, pp. 359-381, Apr 1 2003.

CHAPTER: 11

GLOSSARY

11 GLOSSARY

- **DIP:** Digital Image Processing
- **GUI:** Graphical User Interface
- **VGA:** Video Graphic Array
- **RGB:** Red, Green and Blue
- **LED:** Light Emitting Diode
- **KEIL:** Knowledge Engineering Integration Laboratory

APPENDIX A

MATLAB CODE

```
redThresh = 0.22; % Threshold for red detection
greenThresh = .9; % Threshold for green detection
blueThresh = 0.15; % Threshold for blue detection

% To construct a serial port object:

%   serial_com = serial('COM14', 'BaudRate', 9600);
%   fopen(serial_com);
%   fscanf(serial_com)
%   pause(2);
%   fprintf(serial_com,'1\n'); % Set timing 1 for Green 1
%   pause(2);
%   fscanf(serial_com)
%   pause(2);
%   fprintf(serial_com,'1\n'); % Set timing 2 for Green 2
%   pause(2);
%   fscanf(serial_com)
%   pause(2);
%   fprintf(serial_com,'1\n'); % Set timing 3 for Green 3
%   pause(2);
%   fscanf(serial_com)
%   pause(2);
%   fprintf(serial_com,'1\n'); % Set timing 4 for Green 4
%   pause(2);
%   fscanf(serial_com)
%   pause(2);
%   fclose(serial_com);

cam = webcam(2);
```

```

cam1 = webcam(3);
cam2 = webcam(4);
cam3 = webcam(5);
cam.Resolution= '320x240';
cam1.Resolution= '320x240';
cam2.Resolution= '320x240';
cam3.Resolution= '640x480';
% vidDevice=videoinput('winvideo',2,'YUY2_160x120');
% vidInfo = imaqhwinfo(vidDevice); % Acquire input video property
hblob = vision.BlobAnalysis('AreaOutputPort', false, ... % Set blob analysis handling
    'CentroidOutputPort', true, ...
    'BoundingBoxOutputPort', true, ...
    'MinimumBlobArea', 100, ...
    'MaximumBlobArea', 500, ...
    'MaximumCount', 10);
hshapeinsBox = vision.ShapeInserter('BorderColorSource', 'Input port', ... % Set box handling
    'Fill', true, ...
    'FillColorSource', 'Input port', ...
    'Opacity', 0.4);
htextinsRed = vision.TextInserter('Text', 'Red : %2d', ... % Set text for number of blobs
    'Location', [5 2], ...
    'Color', [1 0 0], ... // red color
    'Font', 'Courier New', ...
    'FontSize', 14);
htextinsGreen = vision.TextInserter('Text', 'Green : %2d', ... % Set text for number of blobs
    'Location', [5 18], ...
    'Color', [0 1 0], ... // green color
    'Font', 'Courier New', ...
    'FontSize', 14);

```

```

hTextinsBlue = vision.TextInserter('Text', 'Blue : %2d', ... % Set text for number of blobs
    'Location', [5 34], ...      'Color', [0 0 1], ... // blue color
    'Font', 'Courier New', ...
    'FontSize', 14);

hTextinsCent = vision.TextInserter('Text', '+ X:%4d, Y:%4d', ... % set text for centroid
    'LocationSource', 'Input port', ...
    'Color', [1 1 0], ... // yellow color
    'Font', 'Courier New', ...
    'FontSize', 14);

hVideoIn = vision.VideoPlayer('Name', 'Final Video', ... % Output video player
    'Position', [100 100 640+20 480+30]);

hVideoIn1 = vision.VideoPlayer('Name', 'Final Video 1', ... % Output video
player
    'Position', [100 100 640+20 480+30]);

hVideoIn2 = vision.VideoPlayer('Name', 'Final Video 2', ... % Output video player
    'Position', [100 100 640+20 480+30]);

hVideoIn3 = vision.VideoPlayer('Name', 'Final Video2', ... % Output video player
    'Position', [100 100 640+20 480+30]);

nFrame = 0; % Frame number initialization
arrayx=0;
arrayy=0;
logic_array=[];

%% Processing Loop
while(nFrame < 3000)
    rgbFrame = snapshot(cam);
    rgbFrame = flipdim(rgbFrame,2); % obtain the mirror image for displaying
    rgbFrame1 = snapshot(cam1);
    rgbFrame1 = flipdim(rgbFrame1,2); % obtain the mirror image for displaying
    rgbFrame2 = snapshot(cam2); % Acquire single frame

```



```

    rgbFrame2 = flipdim(rgbFrame2,2); % obtain the mirror image for displaying
    rgbFrame3 = snapshot(cam3); % Acquire single frame
    rgbFrame3 = flipdim(rgbFrame3,2); % obtain the mirror image for displaying

    diffFrameRed = imsubtract(rgbFrame(:,1), rgb2gray(rgbFrame)); % Get red component of the
image
    diffFrameRed = medfilt2(diffFrameRed, [3 3]); % Filter out the noise by using median filter

    binFrameRed = im2bw(diffFrameRed, redThresh); % Convert the image into binary image with
the red    objects as white

    rgb2_gray = rgb2gray(rgbFrame); % Get green component of the image
    diffFrameGreen = medfilt2(rgb2_gray, [3 3]); % Filter out the noise by using median filter

    binFrameGreen = im2bw(diffFrameGreen, greenThresh); % Convert the image into binary
image with the    green objects as white

    obj_present=sum(sum(binFrameGreen));

    diffFrameRed1 = imsubtract(rgbFrame1(:,1), rgb2gray(rgbFrame1)); % Get red component of
the image
    diffFrameRed1 = medfilt2(diffFrameRed1, [3 3]); % Filter out the noise by using median filter

    binFrameRed1 = im2bw(diffFrameRed1, redThresh); % Convert the image into binary image
with the red objects as white

    rgb2_gray1 = rgb2gray(rgbFrame1); % Get green component of the image
    diffFrameGreen1 = medfilt2(rgb2_gray1, [3 3]); % Filter out the noise by using median filter

    binFrameGreen1 = im2bw(diffFrameGreen1, greenThresh); % Convert the image into binary
image with the green objects as white

    obj_present1=sum(sum(binFrameGreen1));

    diffFrameRed2 = imsubtract(rgbFrame2(:,1), rgb2gray(rgbFrame2)); % Get red component of
the image
    diffFrameRed2 = medfilt2(diffFrameRed2, [3 3]); % Filter out the noise by using median filter

    binFrameRed2 = im2bw(diffFrameRed2, redThresh); % Convert the image into binary image
with the red objects as white

    rgb2_gray2 = rgb2gray(rgbFrame2); % Get green component of the image
    diffFrameGreen2 = medfilt2(rgb2_gray2, [3 3]); % Filter out the noise by using median filter

    binFrameGreen2 = im2bw(diffFrameGreen2, greenThresh); % Convert the image into binary
image with the green objects as white

    obj_present2=sum(sum(binFrameGreen2));

```

```

diffFrameRed3 = imsubtract(rgbFrame3(:, :, 1), rgb2gray(rgbFrame3)); % Get red component of
the image

diffFrameRed3 = medfilt2(diffFrameRed3, [3 3]); % Filter out the noise by using median filter

binFrameRed3 = im2bw(diffFrameRed3, redThresh); % Convert the image into binary image
with the red objects as white

rgb2_gray3 = rgb2gray(rgbFrame3); % Get green component of the image

diffFrameGreen3 = medfilt2(rgb2_gray3, [3 3]); % Filter out the noise by using median filter

binFrameGreen3 = im2bw(diffFrameGreen3, greenThresh); % Convert the image into binary
image with the green objects as white

obj_present3 = sum(sum(binFrameGreen3));

[centroidRed, bboxRed] = step(hblob, binFrameRed); % Get the centroids and bounding boxes of
the red blobs

centroidRed = uint16(centroidRed); % Convert the centroids into Integer for further steps

[centroidRed1, bboxRed1] = step(hblob, binFrameRed1); % Get the centroids and bounding boxes
of the red blobs

centroidRed1 = uint16(centroidRed1); % Convert the centroids into Integer for further steps

[centroidRed2, bboxRed2] = step(hblob, binFrameRed2); % Get the centroids and bounding
boxes of the red blobs

centroidRed2 = uint16(centroidRed2); % Convert the centroids into Integer for further steps

[centroidRed3, bboxRed3] = step(hblob, binFrameRed3); % Get the centroids and bounding boxes
of the red blobs

centroidRed3 = uint16(centroidRed3); % Convert the centroids into Integer for further steps

rgbFrame(1:50, 1:90, :) = 0; % put a black region on the output stream

rgbFrame = im2single(rgbFrame);

vidIn = step(hshapeinsBox, rgbFrame, bboxRed, single([1 0 0])); % Instert the red box

rgbFrame1(1:50, 1:90, :) = 0; % put a black region on the output stream

rgbFrame1 = im2single(rgbFrame1);

vidIn1 = step(hshapeinsBox, rgbFrame1, bboxRed1, single([1 0 0])); % Instert the red box

```

```

rgbFrame2(1:50,1:90,:) = 0; % put a black region on the output stream
rgbFrame2 = im2single(rgbFrame2);
vidIn2 = step(hshapeinsBox, rgbFrame2, bboxRed2, single([1 0 0])); % Instert the red box
rgbFrame3(1:50,1:90,:) = 0; % put a black region on the output stream
rgbFrame3 = im2single(rgbFrame3);
vidIn3 = step(hshapeinsBox, rgbFrame3, bboxRed3, single([1 0 0])); % Instert the red box

if length(bboxRed(:,1))> 0
%   array='Green_1xxxxxx';
end
if (obj_present <100 && strcmp(array,'Green_1xxxxxx')==1)
%   array='Green_1_clear';
end
if length(bboxRed1(:,1))> 0
%   array='Green_2xxxxxx';
end
if obj_present <100 && strcmp(array,'Green_2xxxxxx')==1
%   array='Green_2_clear';
end
if length(bboxRed2(:,1))> 0
%   array='Green_3xxxxxx';
end
if obj_present <100 && strcmp(array,'Green_3xxxxxx')==1
%   array='Green_3_clear';
end
if (strcmp(array,'Green_1_clear')==1 || strcmp(array,'Green_2_clear')==1 ||
strcmp(array,'Green_3_clear')==1)
%   array='Normal3xxxxxx';
end

```

```

        if length(bboxRed3(:,1))> 0
%       array='Green_4'
        elseif obj_present <100
%       array='Green_4_clear'
        else
%       array='Normal4xxxxxx'
        end
%% Logic for all inputs
if length(bboxRed(:,1))> 0
    disp('1_ambulance');
    logic_array(1,1)=1;
else
    logic_array(1,1)=0;
end
if obj_present >100
    disp('1_normal');
    logic_array(1,2)=1;
else
    logic_array(1,2)=0;
end
if obj_present <100
    logic_array(1,3)=1;
    disp('empty');
else
    logic_array(1,3)=0;
end
if length(bboxRed1(:,1))> 0
    disp('2_ambulance');
    logic_array(1,4)=1;

```

```

else
    logic_array(1,4)=0;
end
if obj_present1 >100
    disp('2_normal');
    logic_array(1,5)=1;
else
    logic_array(1,5)=0;
end
if obj_present1 <100
    logic_array(1,6)=1;
    disp('empty');
else
    logic_array(1,6)=0;
end
if length(bboxRed2(:,1))> 0
    disp('3_ambulance');
    logic_array(1,7)=1;
else
    logic_array(1,7)=0;
end
if obj_present2 >100
    disp('3_normal');
    logic_array(1,8)=1;
else
    logic_array(1,8)=0;
end
if obj_present2 <100
    logic_array(1,9)=1;

```

```

    disp('empty');
else
    logic_array(1,9)=0;
end
if length(bboxRed3(:,1))> %    disp('4_ambulance');
    logic_array(1,10)=1;
else
    logic_array(1,10)=0;
end
if obj_present >100
    disp('4_normal');
    logic_array(1,11)=1;
else
    logic_array(1,11)=0;
end
if obj_present <100
    logic_array(1,12)=1;
%    disp('empty');
    Else
    logic_array(1,12)=0;
end
%% for 4 Camera
if logic_array == [0 0 1 0 0 1 0 0 1 0 0 1]
    array='Nor';
end
if logic_array == [0 1 0 0 0 1 0 0 1 0 0 1]
    array='SIN';
end
if logic_array == [0 0 1 0 1 0 0 0 1 0 0 1]

```

```

    array='S2N';
end
if logic_array == [0 0 1 0 0 1 0 1 0 0 0 1]
    array='S3N';
end
if logic_array == [0 0 1 0 0 1 0 0 1 0 1 0]
    array='S4N';
end
if logic_array == [1 1 0 0 0 1 0 0 1 0 0 1]
    array='S1A';
end
if logic_array == [0 0 1 1 1 0 0 0 1 0 0 1]
    array='S2A';
end
if logic_array == [0 0 1 0 0 1 1 1 0 0 0 1]
    array='S3A';
end
if logic_array == [0 0 1 0 0 1 0 0 1 1 1 0]
    array='S4A';
end
if logic_array == [0 1 0 0 1 0 0 1 0 0 1 0]
    array='Nor';
end
if logic_array == [1 1 0 1 1 0 0 0 1 0 0 1]
    array='12A';
end
if logic_array == [0 1 0 0 1 0 1 1 0 1 1 0]
    array='34A';
end

```

```

if logic_array == [0 1 0 1 1 0 1 1 0 0 1 0]
    array='23A';
end

if logic_array == [1 1 0 0 1 0 0 1 0 1 1 0]
    array='14A';
end

if logic_array == [1 1 0 0 1 0 1 1 0 0 1 0]
    array='13A';
end

if logic_array == [0 1 0 1 1 0 0 1 0 1 1 0]
    array='24A';
end

if logic_array == [1 1 0 1 1 0 1 1 0 1 1 0]
    array='All';
end

% for 3 camera
if logic_array == [0 0 1 0 0 1 0 0 1]
    array=['S1E' 'S2E' 'S3E'];
end

if logic_array == [0 1 0 0 0 1 0 0 1]
    array=['S1O' 'S2E' 'S3E'];
end

if logic_array == [0 0 1 0 1 0 0 0 1]
    array=['S1E' 'S2O' 'S3E'];
end

if logic_array == [0 0 1 0 0 1 0 1 0]
    array=['S1E' 'S2E' 'S3O'];
end

if logic_array == [0 0 1 0 1 0 0 1 0]

```



```

    array=['S1E' 'S2O' 'S3O'];
end
if logic_array == [0 1 0 0 0 1 0 1 0]
    array=['S1O' 'S2E' 'S3O'];
end
if logic_array == [0 1 0 0 1 0 0 0 1]
    array=['S1O' 'S2O' 'S3E'];
end
if logic_array == [0 1 0 0 1 0 0 1 0]
    array=['S1O' 'S2O' 'S3O'];
end
if logic_array == [1 1 0 0 0 1 0 0 1]
    array=['S1A' 'S2E' 'S3E'];
end
if logic_array == [0 0 1 1 1 0 0 0 1]
    array=['S1E' 'S2A' 'S3E'];
end
if logic_array == [0 0 1 0 0 1 1 1 0]
    array=['S1E' 'S2E' 'S3A'];
end
if logic_array == [1 1 0 1 1 0 0 0 1]
    array=['S1A' 'S2A' 'S3E'];
end
if logic_array == [1 1 0 0 0 1 1 1 0]
    array=['S1A' 'S2E' 'S3A'];
end
if logic_array == [0 0 1 1 1 0 1 1 0]
    array=['S1E' 'S2A' 'S3A'];
end

```

```

if logic_array == [1 1 0 1 1 0 1 1 0]
    array=['S1A' 'S2A' 'S3A'];
end
if logic_array == [1 1 0 0 1 0 0 1 0]
    array=['S1A' 'S2O' 'S3O'];
end
if logic_array == [0 1 0 1 1 1 0 0 1 0]
    array=['S1O' 'S2A' 'S3O'];
end
if logic_array == [0 1 0 0 1 0 1 1 0]
    array=['S1O' 'S2O' 'S3A'];
end
if logic_array == [1 1 0 1 1 0 0 1 0]
    array=['S1A' 'S2A' 'S3O'];
end
if logic_array == [1 1 0 0 1 0 1 1 0]
    array=['S1A' 'S2O' 'S3A'];
end
if logic_array == [0 1 0 1 1 0 1 1 0]
    array=['S1O' 'S2A' 'S3A'];
end
if logic_array == [1 1 0 1 1 0 1 1 0]
    array=['S1A' 'S2A' 'S3A'];
end
if logic_array == [0 0 1 0 1 0 1 1 0]
    array=['S1E' 'S2O' 'S3A'];
end
if logic_array == [1 1 0 0 0 1 0 1 0]
    array=['S1A' 'S2E' 'S3O'];

```

```

end
if logic_array == [0 1 0 1 1 0 0 0 1]
    array=['S1O' 'S2A' 'S3E'];
end
if logic_array == [1 1 0 0 1 0 0 0 1]
    array=['S1A' 'S2O' 'S3E'];
end
if logic_array == [0 0 1 1 1 0 0 1 0]
    array=['S1E' 'S2A' 'S3O'];
end
if logic_array == [0 1 0 0 0 1 1 1 0]
    array=['S1O' 'S2E' 'S3A'];
end
if logic_array == [0 0 1 1 1 0 0 1 0]
    array=['S1E' 'S2A' 'S3O'];
end
if logic_array == [0 1 0 0 0 1 1 1 0]
    array=['S1O' 'S2E' 'S3A'];
end
for object = 1:1:length(bboxRed(:,1)) % Write the corresponding centroids for red
    centXRed = centroidRed(object,1); centYRed = centroidRed(object,2);
    vidIn = step(htextinsCent, vidIn, [centXRed centYRed], [centXRed-6 centYRed-9]);
end
for object1 = 1:1:length(bboxRed1(:,1)) % Write the corresponding centroids for red
    centXRed1 = centroidRed1(object1,1); centYRed1 = centroidRed1(object1,2);
    vidIn1 = step(htextinsCent, vidIn1, [centXRed1 centYRed1], [centXRed1-6 centYRed1-9]);
end
for object2 = 1:1:length(bboxRed2(:,1)) % Write the corresponding centroids for red
    centXRed2 = centroidRed2(object2,1);

```

```

centYRed2 = centroidRed2(object2,2);

vidIn2 = step(htextinsCent, vidIn2, [centXRed2 centYRed2], [centXRed2-6 centYRed2-9]);

end

for object3 = 1:1:length(bboxRed3(:,1)) % Write the corresponding centroids for red
centXRed3 = centroidRed3(object3,1); centYRed3 = centroidRed3(object3,2);

    vidIn3 = step(htextinsCent, vidIn3, [centXRed3 centYRed3], [centXRed3-6 centYRed3-9]);

    end

    vidIn = step(htextinsRed, vidIn, uint8(length(bboxRed(:,1)))); % Count the number of red blobs
vidIn1 = step(htextinsRed, vidIn1, uint8(length(bboxRed1(:,1)))); % Count the number of red blobs
vidIn2 = step(htextinsRed, vidIn2, uint8(length(bboxRed2(:,1)))); % Count the number of red blobs

    vidIn3 = step(htextinsRed, vidIn3, uint8(length(bboxRed3(:,1)))); % Count the number of red
blobs

    step(hVideoIn, vidIn); % Output video stream

    step(hVideoIn1, vidIn1); % Output video stream

    step(hVideoIn2, vidIn2); % Output video stream

    step(hVideoIn3, vidIn3); % Output video stream

objX=[ obj_present/100 obj_present1/100 obj_present2/100];

objX=round(objX)

logic_array;

array

nFrame = nFrame+1;

if(strcmp(array,arrayx)==0)

arrayx=array

    serial_com = serial('COM5', 'BaudRate', 9600);

    fopen(serial_com);

    fprintf(serial_com, ['M' arrayx '\n']);

% pause(2)

fclose(serial_com);

end

```

```
end
%% Clearing Memory
release(hVideoIn); % Release all memory and buffer used
release(hVideoIn1);
release(hVideoIn2);
release(hVideoIn3);
fclose(serial_com);
clear all
close all
clc
```

C++ Code:

```
#include <AT89x51.h>

#define red1 P0_0

#define yellow1 P0_1

#define green1 P0_2

#define red2 P0_3

#define yellow2 P0_4

#define green2 P0_5

#define red3 P0_6

#define yellow3 P0_7

#define green3 P2_7

#define red4 P2_6

#define yellow4 P2_5

#define green4 P2_4

#define Max 30

#define AD 5000

char array[9] = 0;

char s1 = 'O';

char s2 = 'O';

char s3 = 'O';

char s4 = 'O';

char s_counter = 0;

char rx_ok = 0;

char rx_counter = 0;

int counter = 0;

char sec = 0;

bit flag = 0;

char s_buf = 0;

void comm_isr(void) interrupt 4
```

```

{array[rx_counter] = SBUF;}

void delay(unsigned int i)
{unsigned int j = 0;
unsigned int k = 0;
for (j = 0; j < i; j++)
{for (k = 0; k <= 125; k++)
{;}}}

void check_ambulance(void)
{clear();
    if(s1 == 'A')
        {print_lcd("Ambulance is On ");
            print_lcd(" Signal 1");
                red1 = 1;

                green1 = 0;
                red2 = 0;
                green2 = 1;
                red3 = 0;
                green3 = 1;
                red4 = 0;
                green4 = 1;
                delay(AD);
                red1 = 0;
                red2 = 0;
                red3 = 0;
                red4 = 0;
                green1 = 1;
                green2 = 1;
                green3 = 1;
                green4 = 1;
        }
}

```

```

        s1 = 'O';}
if(s2 == 'A')
{print_lcd("Ambulance is On ");
  print_lcd(" Signal 2");
  red1 = 0;
  green1 = 1;
  red2 = 1;
  green2 = 0;
  red3 = 0;
  green3 = 1;
  red4 = 0;
  green4 = 1;
  delay(AD);
  red1 = 0;
  red2 = 0;
  red3 = 0;
  red4 = 0;
  green1 = 1;
  green2 = 1;
  green3 = 1;
  green4 = 1;
  s2 = 'O';}
if(s3 == 'A')
{print_lcd("Ambulance is On ");
  print_lcd(" Signal 3");
  red1 = 0;
  green1 = 1;
  red2 = 0;

```



```

        green2 = 1;
        red3 = 1;
        green3 = 0;
        red4 = 0;
        green4 = 1;
        delay(AD);
        red1 = 0;
        red2 = 0;
        red3 = 0;
        red4 = 0;
        green1 = 1;
        green2 = 1;
        green3 = 1;
        green4 = 1;
        s3 = 'O';}
if(s4 == 'A')
{print_lcd("Ambulance is On ");
  print_lcd(" Signal 4");
  red1 = 0;
  green1 = 1;
  red2 = 0;
  green2 = 1;
  red3 = 0;
  green3 = 1;
  red4 = 1;
  green4 = 0;
  delay(AD);
  red1 = 0;
  red2 = 0;

```

```

        red3 = 0;
        red4 = 0;
        green1 = 1;
        green2 = 1;
        green3 = 1;
        green4 = 1;
        s4 = 'O';}
print_lcd("Normal Operation");}
void main(void)
{red1 = 0;
    red2 = 0;
    red3 = 0;
    red4 = 0;
    delay(1000);
    red1 = 1;
    yellow1 = 0;
    delay(1000);
    yellow1 = 1;
    green1 = 0;
    if(s1 == 'E'){delay(1000);}
    s_counter = 0;
    while(s1 == 'O')
    {s_counter++;
        delay(1000);
        check_ambulance();
        red1 = 1;
        green1 = 0;
        if(s_counter == Max)
        {break;}}
}

```

```
green1 = 1;
yellow1 = 0;
delay(1000);
yellow1 = 1;
red1 = 0;
delay(1000);
red2 = 1;
yellow2 = 0;
delay(1000);
yellow2 = 1;
green2 = 0;
//delay(10000);
if(s2 == 'E'){delay(1000);}
s_counter = 0;
while(s2 == 'O')
{
    s_counter++;
    delay(1000);
    check_ambulance();
    red2 = 1;
    green2 = 0;
    if(s_counter == Max)
    {break;}}
green2 = 1;
yellow2 = 0;
delay(1000);
yellow2 = 1;
red2 = 0;
delay(1000);
red3 = 1;
```

```

yellow3 = 0;
delay(1000);
yellow3 = 1;
green3 = 0;
//delay(10000);
if(s3 == 'E'){delay(1000);}
s_counter = 0;
while(s3 == 'O')
{
s_counter++;
    delay(1000);
    check_ambulance();
    red3 = 1;
    green3 = 0;
    if(s_counter == Max)
        {break;}
}
green3 = 1;
yellow3 = 0;
delay(1000);
yellow3 = 1;
red3 = 0;
delay(1000);
red4 = 1;
yellow4 = 0;
delay(1000);
yellow4 = 1;
green4 = 0;
//delay(10000);
if(s4 == 'E'){delay(1000);}
s_counter = 0;

```

```
while(s4 == 'O')
{
  s_counter++;
  delay(1000);
  check_ambulance();
  green4 = 0;
  if(s_counter == Max)
  {
    break;
  }
  green4 = 1;
  yellow4 = 0;
  delay(1000);
  yellow4 = 1;
  red4 = 0;
  delay(1000);
}
```