# TRAFFIC DENSITY CONTROL USING IMAGE PROCESSING

*by*


| Name | Roll No. | Registration No: |
|---|---|---|
| Ankita Sarkar | **Ece-2014/087** | **141170110203** of 2014-2015 |
| Ricktika Biswas | **Ece-2014/076** | **141170110251** of 2014-2015 |
| Soumya Roy | **Ece-2014/040** | **141170110285** of 2014-2015 |
| Sayon Gupta | **Ece-2014/084** | **141170110272** of 2014-2015 |


*A comprehensive project report has been submitted in partial fulfillment of the requirements for the degree of*

**Bachelor of Technology**

*in*

**ELECTRONICS & COMMUNICATION ENGINEERING**

*Under the supervision of*

**Dr. Soham Sarkar**

Professor


**Department of Electronics & Communication Engineering**

**RCC INSTITUTE OF INFORMATION TECHNOLOGY**

**Affiliated to Maulana Abul Kalam Azad University of Technology, WestBengal**

**CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700015**


**MAY,2018**

# CERTIFICATE OF APPROVAL


This is to certify that the project titled "**TRAFFIC DENSITY CONTROL USING IMAGE PROCESSING**" carried out by

| Name | Roll No. | Registration No: |
|---|---|---|
| **Ankita Sarkar** | **ECE2014/087** | **141170110203** of 2014-2015 |
| **Ricktika Biswas** | **ECE2014/076** | **141170110251** of 2014-2015 |
| **Soumya Roy** | **ECE2014/040** | **141170110285** of 2014-2015 |
| **Sayon Gupta** | **ECE2014/084** | **141170110272** of 2014-2015 |

for the partial fulfillment of the requirements for B.Tech degree in **Electronics and Communication Engineering** from **Maulana Abul Kalam Azad University of Technology, West Bengal**is absolutely based on his own work under the supervision of Mr. **Soham Sarkar**. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

.....

**Dr. Abhishek Basu**

Head of the Department(ECE)

RCC Institute of Information Technology

.....

**Dr. Soham Sarkar**

Professor, Dept. of ECE

RCC Institute of Information Technology

# DECLARATION


"We Do hereby declare that this submission is our own work conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute and that, to the best of our knowledge and belief, it contains no material previously written by another neither person nor material (data, theoretical analysis, figures, and text) which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text."

.....

**Ankita Sarkar**

Registration No: **141170110203** OF 2014-2015

Roll No: **ECE2014/087**

.....

**Ricktika Biswas**

Registration No: **141170110251** OF 2014-2015

Roll No: **ECE2014/076**

.....

**Soumya Roy**

Registration No: **141170110285** OF 2014-2015

Roll No: **ECE2014/040**

.....

**Sayon Gupta**

Registration No: **141170110272** OF 2014-2015

Roll No: **ECE2014/084**

Date:

Place:

# CERTIFICATE of ACCEPTANCE

This is to certify that the project titled "**TRAFFIC DENSITY CONTROL USING IMAGE PROCESSING**" carried out by

| Name | Roll No. | Registration No: |
|---|---|---|
| **Ankita Sarkar** | **ECE2014/087** | **141170110203** of 2014-2015 |
| **Ricktika Biswas** | **ECE2014/076** | **141170110251** of 2014-2015 |
| **Soumya Roy** | **ECE2014/040** | **141170110285** of 2014-2015 |
| **Sayon Gupta** | **ECE2014/084** | **141170110272** of 2014-2015 |

is hereby recommended to be accepted for the partial fulfilment of the requirements for B.Tech degree in **Electronics and Communication Engineering** from **Maulana Abul Kalam Azad University of Technology, West Bengal**

**Name of the Examiner Signature with Date**

1. ……

2.……

3.……

4. ……

# ABSTRACT

In this paper we propose a method for determining traffic congestion on roads using image processing techniques and a model for controlling traffic signals based on information received from images of roads taken by camera. We extract traffic density which corresponds to total area occupied by vehicles on the road in terms of total amount of pixels in a frame and calculating number of vehicles. By calculating the number of vehicles, we detect the traffic congestion and hence we determine the traffic signal delay, happening according to the congestion. We set two parameters as output, variable traffic cycle and weighted time for each road based on traffic density and control traffic lights in a sequential manner.

# CONTENTS

# LIST OF FIGURES

# 1.Introduction:

Most of the city traffic is controlled by sensors and cameras shall be installed in big highways and streets. But existence of a system for detecting the size of traffic automatically will be felt. Such systems can allow to extract information from the bigger traffic issue and helps us decide to improve the traffic policy. The paper aims to render automate control system for traffic on highways and streets. The system using image processing has been implemented where upon it entailed the following results: 1) Density 2) Streets and roads in order to census counted three cars 3) monitor off roads 4) Detect the occurrence of accidents and violations occurred as well as motion detection car is a dangerous spiral. Scientists and other researchers suggested other different ways. Technically, this system is based on computers and cameras. The project components include: (A) hardware model (B) software model.

In recent years, traffic congestion has become a significant problem. Early solutions attempted to lay more pavement to avoid congestion, but adding more lanes is becoming less and less feasible. Contemporary solutions emphasize better information and control to use the existing infrastructure more efficiently. The quest for better traffic information, and thus, an increasing reliance on traffic surveillance, has resulted in a need for better vehicle detection such as wide-area detectors; while the high costs and safety risks associated with lane closures has directed the search towards non-invasive detectors mounted beyond the edge of the pavement. One promising approach is vehicle tracking via image processing, which can yield traditional traffic parameters such as flow and velocity, as well as new parameters such as lane changes and vehicle trajectories. Because the vehicle tracks, or trajectories, are measured over a length of roadway, rather than at a single point, it is possible to measure true density instead of simply recording detector occupancy. In fact, by averaging trajectories over space and time, the traditional traffic parameters are more stable than corresponding measurements from point detectors, which can only average overtime.

The additional information from the vehicle trajectories could lead to improved incident detection, both by detecting stopped vehicles within the camera's field of view and by identifying lane change manoeuvres or acceleration/deceleration patterns that are indicative of incidents beyond the camera's field of view. The trajectory data could also be used to automate previously labour-intensive traffic studies, such as examining vehicle manoeuvres in weaving sections or bottlenecks. The vehicle tracking system can produce individual vehicle data (e.g. spacing, headway, velocity, acceleration), which could lead to better traffic flow modelling and an improved understanding of driver behaviour. The system can extract vehicle signatures and match observations of the same vehicle at multiple detector stations (Huang and Russell, 1998). This signature matching can be used to measure true link travel time and thus, quantify conditions between widely spaced detectors rather than assuming that local conditions are representative of the entire link. To be an effective traffic surveillance tool, whether by mimicking loop detectors or actually tracking vehicles, a image processing system (VIPS) should meet several stringent requirements:

1. Automatic segmentation of each vehicle from the background and from other vehicles so that all vehicles are detected.

2. Correctly detect all types of road vehicles like motorcycles, passenger cars, buses, construction equipment, trucks, etc.

3. Function under a wide range of traffic conditions like light traffic, congestion, varying speeds in different lanes.

4. Function under a wide variety of lighting conditions like sunny, overcast, twilight, night, rainy, etc.

5. Operate in real-time.

Even though a number of commercial VIPS for monitoring traffic have been introduced to the market, many of these criteria still cannot be met.

# 2.About image Processing:

## 2.1 What is Image Processing?

**Image processing** is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from it. It is a type of signal dispensation in which input is image, like frame or photograph and output may be image or characteristics associated with that image. Usually **Image Processing** system includes treating images as two-dimensional signals while applying already set signal processing methods to them.

It is among rapidly growing technologies today, with its applications in various aspects of a business. Image Processing forms core research area within engineering and computer science disciplines too.

*Image processing basically includes the following three steps.*

- Importing the image with optical scanner or by digital photography.
- Analyzing and manipulating the image which includes data compression and image enhancement and spotting patterns that are not to human eyes like satellite photographs.
- Output is the last stage in which result can be altered image or report that is based on image analysis.

## 2.2 Purpose of Image processing:

The purpose of image processing is divided into 5 groups. They are:

1. Visualization - Observe the objects that are not visible.
2. Image sharpening and restoration - To create a better image.
3. Image retrieval - Seek for the image of interest.

4. Measurement of pattern – Measures various objects in an image.
5. Image Recognition – Distinguish the objects in an image.

## 2.3 Types of image processing:

The two types of **methods used for Image Processing** are **Analog and Digital** Image Processing. Analog or visual techniques of image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. The image processing is not just confined to area that has to be studied but on knowledge of analyst. Association is another important tool in image processing through visual techniques. So analysts apply a combination of personal knowledge and collateral data to image processing.

Digital Processing techniques help in manipulation of the digital images by using computers. As raw data from imaging sensors from satellite platform contains deficiencies. To get over such flaws and to get originality of information, it has to undergo various phases of processing. The three general phases that all types of data have to undergo while using digital technique are Pre- processing, enhancement and display, information extraction.

# 3.Various working strategies of this project:

## 3.1 VEHICALE TRACKING STRATEGIES:

Multi-object tracking and data association have received considerable attention in the computer vision field and much of the background work has been in non-transportation applications.
From the computer vision literature, the different tracking approaches for data can be classified as follows.

### i. Model based tracking:

Three-dimensional model-based vehicle tracking systems have previously been investigated by several research groups, the most prominent being the groups at Karlsruhe (Koller et al., 1993) and at the University of Reading (Baker and Sullivan, 1992; Sullivan, 1992). The emphasis is on recovering trajectories and models with high accuracy for a small number of vehicles. The most serious weakness of this approach is the reliance on detailed geometric object models. It is unrealistic to expect to be able to have detailed models for all vehicles that could be found on the roadway.

### ii. Region based tracking

In this approach, the VIPS identifies a connected region in the image, a `blob', associated with each vehicle and then tracks it over time using a cross-correlation measure. Typically,

the process is initialized by the background subtraction technique. A Kalman filter-based adaptive background model (Karmann and Brandt, 1990; Kilger, 1992) allows the background estimate to evolve as the weather and time of day affect lighting conditions. Foreground objects (vehicles) are detected by subtracting the incoming image from the current background estimate, looking for pixels where this difference image is above some threshold and then finding connected components.

### iii. Active contour-based tracking

A dual to the region-based approach is tracking based on active contour models, or snakes. The basic idea is to have a representation of the bounding contour of the object and keep dynamically updating it. The advantage of having a contour-based representation instead of a region-based representation is reduced computational complexity.

## 3.2 FEATURE BASED TRACKING ALGORITHM:

This section presents our vehicle tracking system, which includes: camera calibration, feature detection, feature tracking, and feature grouping modules. First, the camera calibration is conducted once, offline, for a given location and then, the other modules are run continuously online in real-time.

### i. Offline camera definition

Before running the tracking and grouping system, the user specifies camera-specific parameters offline. These parameters include:

1. line correspondences for a projective mapping, or homography, as explained below;
2. a detection region near the image bottom and an exit region near the image top, and
3. multiple fiducial points for camera stabilization.

Since most road surfaces are flat, the grouper exploits an assumption that vehicle motion is

parallel to the road plane. To describe the road plane, the user simply specifies four or more line or point correspondences between the image of the road (i.e. the image plane) and a separate ‘world' road plane, as shown in Fig. In other words, the user must know the relative distance in world coordinates between four points visible in the image plane. Based on this offline step, our system computes a projective transform, or homography, H, between the image coordinates (x,y) and world coordinates (X,Y).

### ii. On-line tracking and grouping

A block diagram for our vehicle tracking and grouping system is shown in Fig. First, the raw camera image is stabilized by tracking manually chosen fiducial points to subpixel accuracy and subtracting their motion from the entire image. Second, the stabilized image is sent to a detection module, which locates corner features in a detection zone at the bottom of the image. In our detection module, ``corner'' features are defined as regions in the gray level intensity image where brightness varies in more than one direction. Fig. (A) shows some example corners detected by the system.

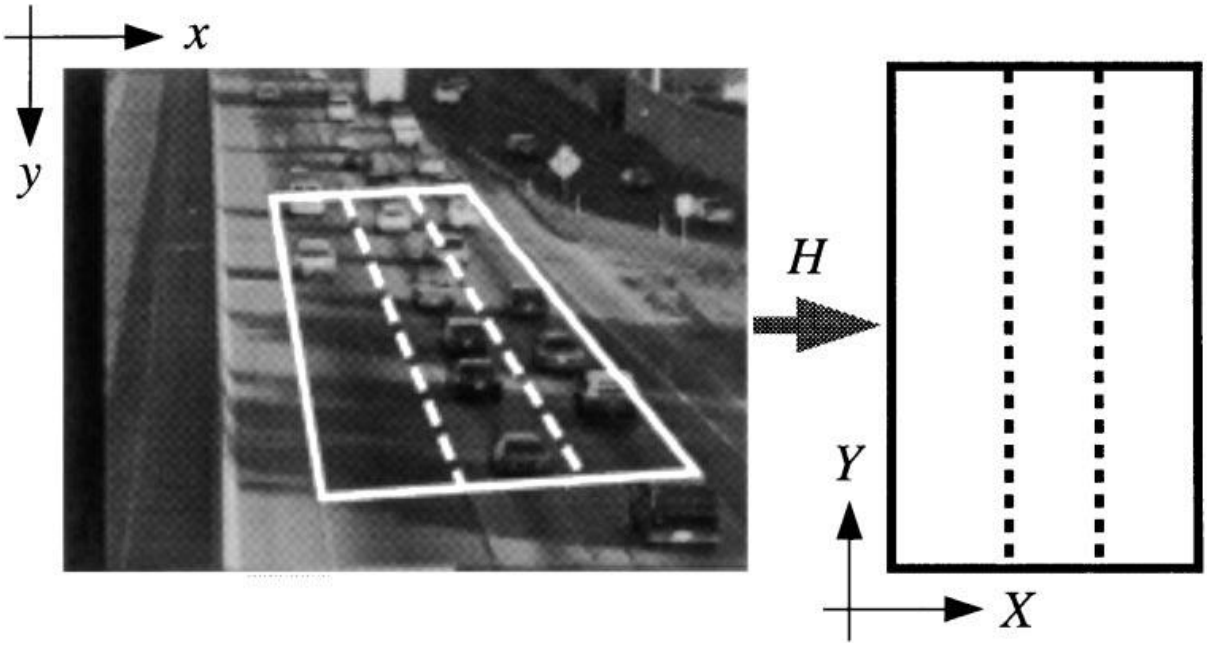


Fig3.1. A projective transform, H, or homography is used to map from image coordinates, (x,y), to world coordinates,(X,Y).


Fig3.2. (A) Sample corner features identified by the tracker. (B) Sample feature tracks from the tracker. (C) Sample feature groups from the grouper.

## 3.3 MEASURING TRAFFIC PARAMETERS USING WIDE AREA DETECTORS:

Traditional traffic parameters such as flow, occupancy and velocity are usually defined with respect to point detectors and they are only averaged over time. The vehicle tracker extracts

vehicle trajectories over a significant distance and is a true wide-area detector. Using the traditional point-based definitions would mean discarding information available from the detector.
Instead, it is possible to measure the generalized traffic parameters over the entire tracking region and thus, average over time and space.

For the real-time system, the generalized parameters are measured for each lane using the entire tracking region, L, and user defined sample period, T. The definitions are robust to lane changes. Eg vehicle 3 leaving the lane and vehicle 5 entering the lane .
Refining traditional metrics is only one aspect of the vehicle tracker. The vehicle trajectories allow for new approaches to measuring traffic and ultimately, better automated surveillance.
The time space diagram for vehicle trajectories from a single lane as a shock wave passes through the detection region [solid lines indicate the vehicle tracker output and dashed lines show manually generated ground truth (Coifman, 1997)]. Note that horizontal trajectories correspond to stopped vehicles and the tracker did not lose them; also notice that the tracker followed almost all of the vehicles even though the traffic momentarily reached jam density and most of the vehicles were partially occluded. This figure only shows the longitudinal position along the roadway, the vehicle tracker also extracts the lateral position. Thus, it is a trivial matter to detect lane changes within the surveillance region.

After the tracker has extracted the vehicle trajectories, it is possible to send this data over a low bandwidth communications link for scene reconstruction and automated surveillance at a remote location. It is also possible to extract a vehicle signature for vehicle reidentification at a downstream site and measure section data such as travel time and O/D patterns (see Huang and Russell, 1998); thus, making it possible to quantify conditions between detectors even if there are no observable effects at the individual detector stations.

## 3.4 USING GUI TECHNIQUE

A graphical user interface (GUI) is a graphical display in one or more windows containing controls, called components, which enable a user to perform interactive tasks. The user of

the GUI does not have to create a script or type commands at the command line to accomplish tasks, the user of GUI need not understand the details of how the tasks are performed. GUI components can include menus, toolbars, push buttons, radio button, list boxes etc. GUI using matlab tools can perform any type of computation, read and write data files communicate with other GUIs and display data as tables or plots.

The GUI contains-

- An axes component
- A pop-up menu listing three data sets that correspond to MATLAB

Typically, the GUI wait for n end user to manipulate a control, and then respond to each user action in turn. Each control, and the GUI itself has one or more call-backs, named for the fact that they "call back" to MATLAB to ask it to do things. A particular user action such as pressing screen button, or passing the cursor over a component, triggers the execution of each call back. The GUI then responds to these events.

The GUI helps to plot the results and show in a plotted format and hence the congestion can be easily calculated, according to which the delays can be implemented in the traffic control signal, to reduce the congestion.

# 4.Aim of the project.

Many techniques have been developed using image processing during the last four or five decades. Most of the methods are developed for enhancing images obtained from unmanned space probes, spacecrafts and military flights. Image Processing Systems are becoming widely popular due to easy availability of powerful personal computers, large memory devices, graphics software and many more.

Image processing involves issues related to image representation techniques and various complex operations, which can be carried out on the image data. The operations that come under image processing are image enhancement operations such as sharpening, blurring, brightening , edge enhancement. Traffic density of lanes is calculated using image processing which we will be dealing in this project. This is image captured by digital camera . we have chosen image processing for calculation of traffic density as cameras are very much cheaper than other devices such as sensors.

Signal synchronization automatically changes when traffic density is detected at the junction. Traffic congestion is a serious problem in many major cities around the world and has become a nightmare for travellers in these cities. The conventional semaphore system is based on the fixed time concept assigned to each side of the joint that cannot be varied according to the variable traffic density. The assigned bonding times are fixed. Sometimes a higher traffic density on one side of the junction requires a longer green time compared to the standard allotted time. The image captured in the traffic signal is processed and converted into grayscale image, then its threshold is calculated on the basis of which the contour has been drawn to calculate the number of vehicles present in the image.

Making use of the above-mentioned virtues of image processing we propose a technique that can be used for traffic control. we will further discuss the algorithm , and mention how we use the following technique in our project.

The following points indicate the aim for this project, i.e. this project will help in:

- Reducing traffic congestion.
- Reducing unwanted long-time delay.
- It keeps a track of the vehicles.
- Need to develop the system without developing the infrastructure.

# 5.Block Diagram of the project

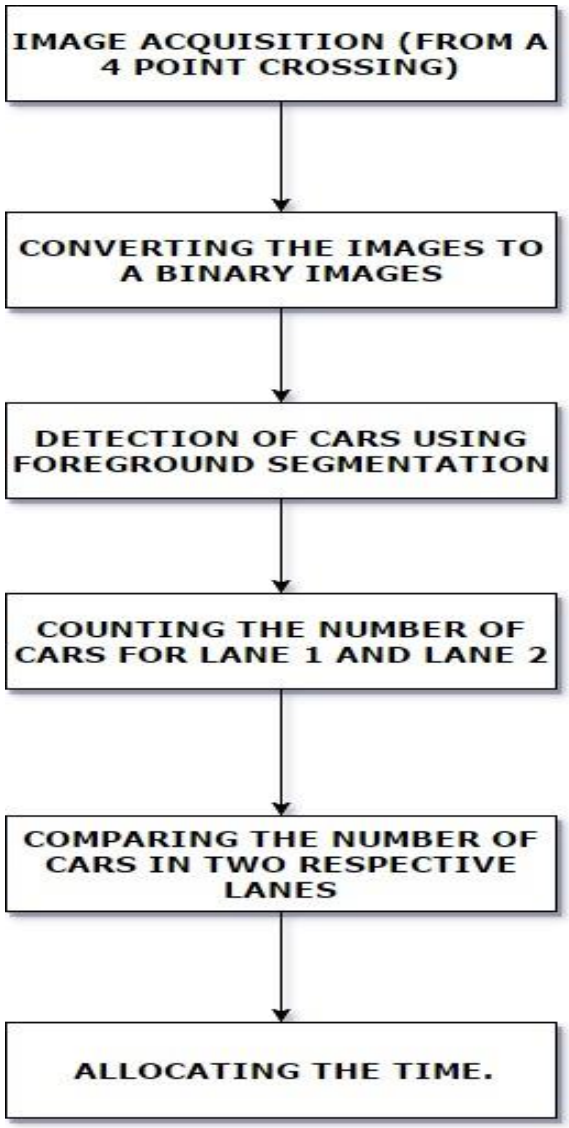



Fig 5.

## 5.1. Image acquisition

Image is generally a 2D function f(x,y) where x and y are the coordinates. The amplitude of image at any point say f is called intensity of image. It is also called the gray level of image at that point. The image is a fundus taken from stare database to drive data base. The image of the retina for processing and to check the condition of the person. Here f is the intensity of the image. We need to convert the analog image to digital image to process through the digital computer. Each digital image composed of finite element and finite elements is called a pixel. We change x and y values from final discrete values to form a digital image.

## 5.2. Conversion of image to binary image

A binary image is a digital image that has only two possible values for each pixel. Typically, the two colors used for a binary image are black and white. The color used for the object(s) in the image is the foreground color while the rest of the image is the background color. In the document-scanning industry, this is often referred to as "bi-tonal".

Binary images are also called *bi-level* or *two-level*. This means that each pixel is stored as a single bit—i.e., a 0 or 1. The names *black-and-white*, *B&W*, monochrome or monochromatic are often used for this concept, but may also designate any images that have only one sample per pixel, such as grayscale images.

Binary images are produced from color images by segmentation. Segmentation is the process of assigning each pixel in the source image to two or more classes. If there are more than two classes then the usual result is several binary images. The simplest form of segmentation is probably Otsu's method which assigns pixels to foreground or background based on greyscale intensity. Another method is the watershed algorithm. Edge detection also often creates a binary image with some pixels assigned to edge pixels and is also a first step in further segmentation.

## 5.3. Fore ground segmentation

In this step, it identifies the pixels in the frame. Foreground detection compares the image frame with the background model and identify candidate foreground pixels from the frame. Commonly- used approach for foreground detection is to check whether the pixel is significantly different from the corresponding background estimate.

## 5.4. Car Tracking

It occurs through the gaussian mixture model.

In order to give a better understanding of the algorithm used for background subtraction the following steps were adopted to achieve the desired results:

1. Firstly, we compare each input pixels to the mean 'μ' of the associated components. If the value of a pixel is close enough to a chosen component's mean, then that component is considered as the matched component. In order to be a matched component, the difference between the pixel and mean must be less than compared to the component's standard deviation scaled by factor D in the algorithm.

2. Secondly, update the Gaussian weight, mean and standard deviation (variance) to reflect the new obtained pixel value. In relation to non-matched components the weights 'w' decreases whereas the mean and standard deviation stay the same. It is dependent upon the learning component 'p' in relation to how fast they change.

3. Thirdly, here we identify which components are parts of the background model. To do this a threshold value is applied to the component weights 'w'.

4. Fourthly, in the final step we determine the foreground pixels. Here the pixels that are identified as foreground don’t match with any components determined to be the background.

## 5.5. Time allocation

Finally depending on the count of the car, there will be a threshold limit. If the threshold limit exceeds, the timing of the green signal will be increased and vice versa.

# 6.Implementation Algorithm:

The block diagram of the project was discussed in previous chapter. The algorithm behind the block diagram consists of following steps

1. the captured image of the four sides of the four-point crossing, (which is two sides of lane 1 and 2 side of lane 2) is loaded.
2. then the captured images are converted into binary images.
3. Cars are detected for lane 1 and the numbers of cars are detected from both sides of the lane.
4. For lane 2, the same procedure is being repeated.
5. If the number of cars in lane 1 is greater than that of the lane 2, then the signal of lane 1 will be green and the signal for lane 2 will be red for 90 seconds. After 90 seconds both the signal will be yellow for 10 seconds and then the reverse signal will be set 60 seconds.
6. If the number of cars in lane 2 is greater than lane 1, then same process will occur vice-verily for lane 2 as shown in point no.5.
7. If the number of cars in lane 1 is equal to that of lane 2 then, both the signals will set timer of 60 seconds.

# 7.Code of the program:

```
function varargout = final(varargin)
% FINAL MATLAB code for final.fig
%      FINAL, by itself, creates a new FINAL or raises the existing
%      singleton*.
%
%      H = FINAL returns the handle to a new FINAL or the handle to
%      the existing singleton*.
%
%      FINAL('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in FINAL.M with the given input arguments.
%
%      FINAL('Property','Value',...) creates a new FINAL or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before final_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to final_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help final

% Last Modified by GUIDE v2.5 08-May-2018 16:06:46

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
```

```
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @final_OpeningFcn, ...
                   'gui_OutputFcn',  @final_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before final is made visible.
function final_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to final (see VARARGIN)

% Choose default command line output for final
handles.output = hObject;

% Update handles structure
```

```
guidata(hObject, handles);

% UIWAIT makes final wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = final_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



function lane1_Callback(hObject, eventdata, handles)
% hObject    handle to lane1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of lane1 as text
%        str2double(get(hObject,'String')) returns contents of lane1 as a double


% --- Executes during object creation, after setting all properties.
function lane1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to lane1 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function car_num1_Callback(hObject, eventdata, handles)
% hObject    handle to car_num1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of car_num1 as text
%        str2double(get(hObject,'String')) returns contents of car_num1 as a double


% --- Executes during object creation, after setting all properties.
function car_num1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to car_num1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
```

```
function numcars1_Callback(hObject, eventdata, handles)
% hObject    handle to numcars1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of numcars1 as text
%        str2double(get(hObject,'String')) returns contents of numcars1 as a double


% --- Executes during object creation, after setting all properties.
function numcars1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to numcars1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function car_num2_Callback(hObject, eventdata, handles)
% hObject    handle to car_num2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of car_num2 as text
%        str2double(get(hObject,'String')) returns contents of car_num2 as a double


% --- Executes during object creation, after setting all properties.
function car_num2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to car_num2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function numcars2_Callback(hObject, eventdata, handles)
% hObject    handle to numcars2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of numcars2 as text
%        str2double(get(hObject,'String')) returns contents of numcars2 as a double


% --- Executes during object creation, after setting all properties.
function numcars2_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to numcars2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in browse.
function browse_Callback(hObject, eventdata, handles)
% hObject    handle to browse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%here we get the image of lane1
global im im2
[path,user_cance]=imgetfile();
if user_cance
   msgbox(sprintf('error'),'error','error');
   return
end
im=imread(path);
im=im2double(im);%convert image to double
im2=im; %for back up process :)
axes(handles.axes1);
imshow(im2);
```

```
%detecting no. of cars
a1=im2bw(im2);
se = strel('square', 3);
filteredForeground = imopen(a1, se);
figure; imshow(filteredForeground); title('Clean Foreground');
blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...
    'AreaOutputPort', false, 'CentroidOutputPort', false, ...
    'MinimumBlobArea', 150);
bbox = step(blobAnalysis, filteredForeground);
result = insertShape(im2, 'Rectangle', bbox, 'Color', 'green');
numCars = size(bbox, 1);
result = insertText(result, [10 10], numCars, 'BoxOpacity', 1, ...
    'FontSize', 14);
figure; imshow(result); title('Detected Cars');

global im1 im3
[path,user_cance]=imgetfile();
if user_cance
    msgbox(sprintf('error'),'error','error');
    return
end
im1=imread(path);
im1=im2double(im1);%convert image to double
im3=im1; %for back up process :)
axes(handles.axes2);
imshow(im3);

%detecting no. of cars
a2=im2bw(im3);
```

```
se = strel('square', 3);
filteredForeground = imopen(a2, se);
figure; imshow(filteredForeground); title('Clean Foreground');
blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...
    'AreaOutputPort', false, 'CentroidOutputPort', false, ...
    'MinimumBlobArea', 150);
bbox = step(blobAnalysis, filteredForeground);
result = insertShape(im3, 'Rectangle', bbox, 'Color', 'green');
numCars1 = size(bbox, 1);
result = insertText(result, [10 10], numCars1, 'BoxOpacity', 1, ...
    'FontSize', 14);
figure; imshow(result); title('Detected Cars');

%Total number of detected car in lane1
tot1=numCars+numCars1;

numcars1=getappdata(0,'numcars1');
numcars1=num2str(numcars1);
set(handles.numcars1,'string',tot1);



%here we get the image for lane2
global im4 im5
[path,user_cance]=imgetfile();
if user_cance
    msgbox(sprintf('error'),'error','error');
    return
end
im4=imread(path);
im4=im2double(im4);%convert image to double
```

```
im5=im4; %for back up process :)
axes(handles.axes3);
imshow(im5);

%detecting no. of cars
a3=im2bw(im5);
se = strel('square', 3);
filteredForeground = imopen(a3, se);
figure; imshow(filteredForeground); title('Clean Foreground');
blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...
    'AreaOutputPort', false, 'CentroidOutputPort', false, ...
    'MinimumBlobArea', 150);
bbox = step(blobAnalysis, filteredForeground);
result = insertShape(im5, 'Rectangle', bbox, 'Color', 'green');
numCars2 = size(bbox, 1);
result = insertText(result, [10 10], numCars2, 'BoxOpacity', 1, ...
    'FontSize', 14);
figure; imshow(result); title('Detected Cars');

global im6 im7
[path,user_cance]=imgetfile();
if user_cance
    msgbox(sprintf('error'),'error','error');
    return
end
im6=imread(path);
im6=im2double(im6);%convert image to double
im7=im6; %for back up process :)
axes(handles.axes4);
imshow(im7);
```

```
%detecting no. of cars
a4=im2bw(im7);
se = strel('square', 3);
filteredForeground = imopen(a4, se);
figure; imshow(filteredForeground); title('Clean Foreground');
blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...
    'AreaOutputPort', false, 'CentroidOutputPort', false, ...
    'MinimumBlobArea', 150);
bbox = step(blobAnalysis, filteredForeground);
result = insertShape(im7, 'Rectangle', bbox, 'Color', 'green');
numCars3 = size(bbox, 1);
result = insertText(result, [10 10], numCars3, 'BoxOpacity', 1, ...
    'FontSize', 14);
figure; imshow(result); title('Detected Cars');

%Total number of detected car in lane1
tot2=numCars2+numCars3;

numcars2=getappdata(0,'numcars2');
numcars2=num2str(numcars2);
set(handles.numcars2,'string',tot2);

%Comparing the number of cars in two lane
b=get(handles.numcars1,'String');
b=str2double(b);

c=get(handles.numcars2,'String');
c=str2double(c);
```

```
if b>c
  set(handles.result1,'BackgroundColor','green');
    set(handles.result2,'BackgroundColor','red');
sec=90;
    set(handles.timer,'string',num2str(sec));
    while sec>0
        pause(1);
        sec=sec-1;
        set(handles.timer,'string',num2str(sec));
    end
set(handles.result1,'BackgroundColor','yellow');
set(handles.result2,'BackgroundColor','yellow');
sec=10;
    set(handles.timer,'string',num2str(sec));
    while sec>0
        pause(1);
        sec=sec-1;
        set(handles.timer,'string',num2str(sec));
    end
    set(handles.result2,'BackgroundColor','green');
    set(handles.result1,'BackgroundColor','red');
     sec=60;
    set(handles.timer,'string',num2str(sec));
    while sec>0
        pause(1);
        sec=sec-1;
        set(handles.timer,'string',num2str(sec));
    end
else if b<c
        set(handles.result2,'BackgroundColor','green');
```

```
    set(handles.result1,'BackgroundColor','red');
    sec=90;
    set(handles.timer,'string',num2str(sec));
    while sec>0
        pause(1);
        sec=sec-1;
        set(handles.timer,'string',num2str(sec));
    end
set(handles.result1,'BackgroundColor','yellow');
set(handles.result2,'BackgroundColor','yellow');
sec=10;
    set(handles.timer,'string',num2str(sec));
    while sec>0
        pause(1);
        sec=sec-1;
        set(handles.timer,'string',num2str(sec));
    end
    set(handles.result1,'BackgroundColor','green');
    set(handles.result2,'BackgroundColor','red');
     sec=60;
    set(handles.timer,'string',num2str(sec));
    while sec>0
        pause(1);
        sec=sec-1;
        set(handles.timer,'string',num2str(sec));
    end
    else
       set(handles.result2,'BackgroundColor','green');
    set(handles.result1,'BackgroundColor','red');
    sec=60;
```

```
    set(handles.timer,'string',num2str(sec));
    while sec>0
        pause(1);
        sec=sec-1;
        set(handles.timer,'string',num2str(sec));
    end
set(handles.result1,'BackgroundColor','yellow');
set(handles.result2,'BackgroundColor','yellow');
sec=10;
    set(handles.timer,'string',num2str(sec));
    while sec>0
        pause(1);
        sec=sec-1;
        set(handles.timer,'string',num2str(sec));
    end
    set(handles.result1,'BackgroundColor','green');
    set(handles.result2,'BackgroundColor','red');
     sec=60;
    set(handles.timer,'string',num2str(sec));
    while sec>0
        pause(1);
        sec=sec-1;
        set(handles.timer,'string',num2str(sec));
    end
    end
 end
```

```
function res_Callback(hObject, eventdata, handles)
% hObject    handle to res (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of res as text
%        str2double(get(hObject,'String')) returns contents of res as a double


% --- Executes during object creation, after setting all properties.
function res_CreateFcn(hObject, eventdata, handles)
% hObject    handle to res (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function result1_Callback(hObject, eventdata, handles)
% hObject    handle to result1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of result1 as text
%        str2double(get(hObject,'String')) returns contents of result1 as a double


% --- Executes during object creation, after setting all properties.
function result1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to result1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function result2_Callback(hObject, eventdata, handles)
% hObject    handle to result2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of result2 as text
%        str2double(get(hObject,'String')) returns contents of result2 as a double


% --- Executes during object creation, after setting all properties.
```

```
function result2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to result2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit12 as text
%        str2double(get(hObject,'String')) returns contents of edit12 as a double


% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function timer_Callback(hObject, eventdata, handles)
% hObject    handle to timer (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of timer as text
%        str2double(get(hObject,'String')) returns contents of timer as a double


% --- Executes during object creation, after setting all properties.
function timer_CreateFcn(hObject, eventdata, handles)
% hObject    handle to timer (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
```

# 8.Outputs of The Following Code:

Fig-8.1 the GUI interface is ready to take the input manually.

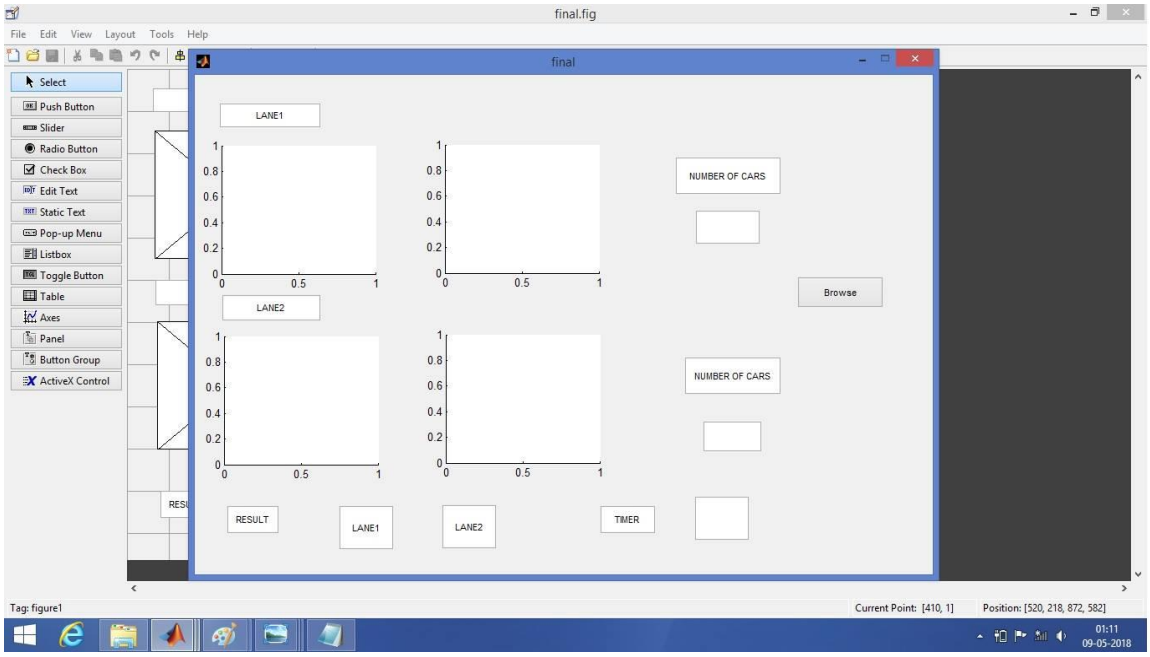



Fig-8.2 the  number of detected cars is shown on the upper left corner.

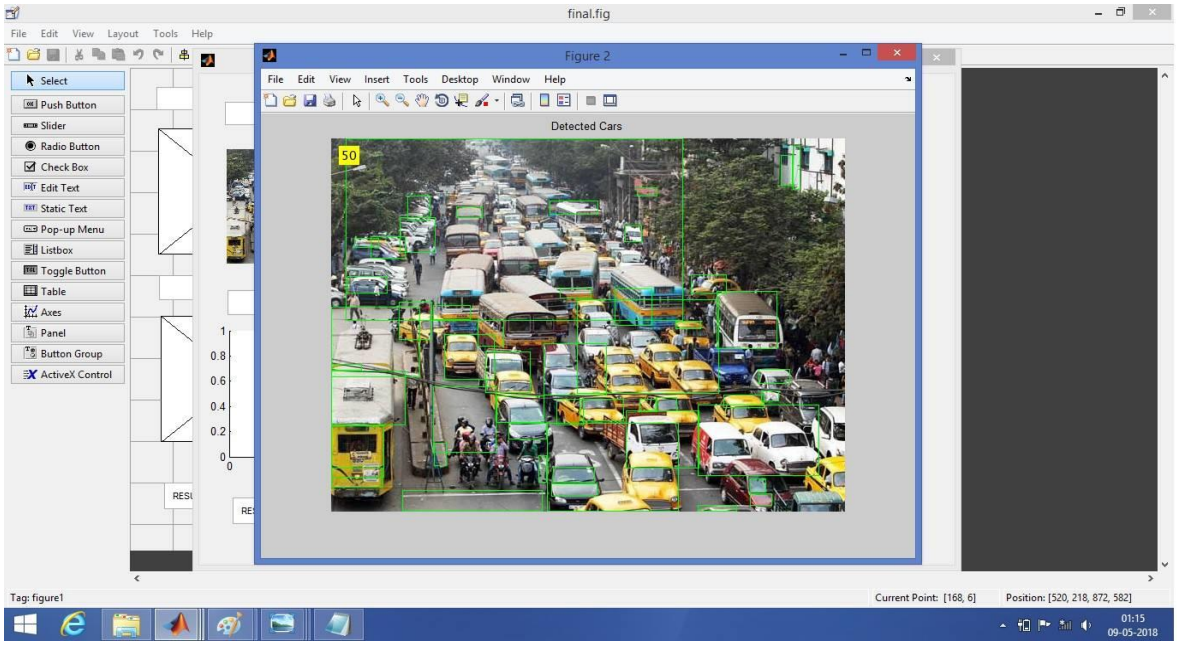


Fig -8.3 the system is ready to take another image input.

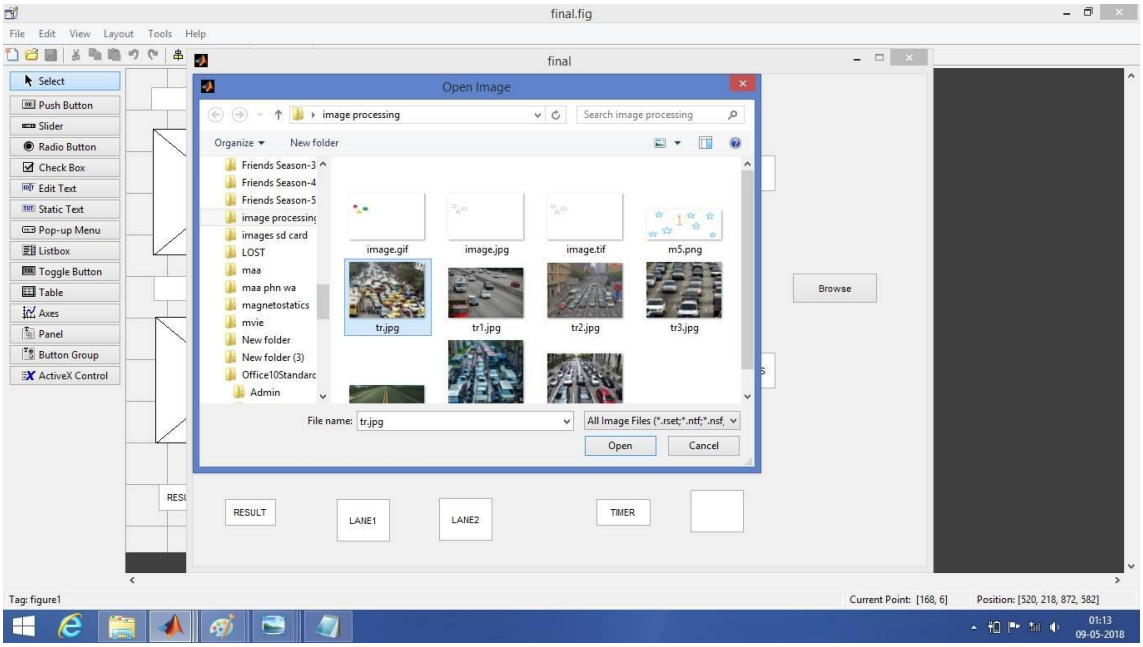



Fig-8.4 the image is converted into binary image

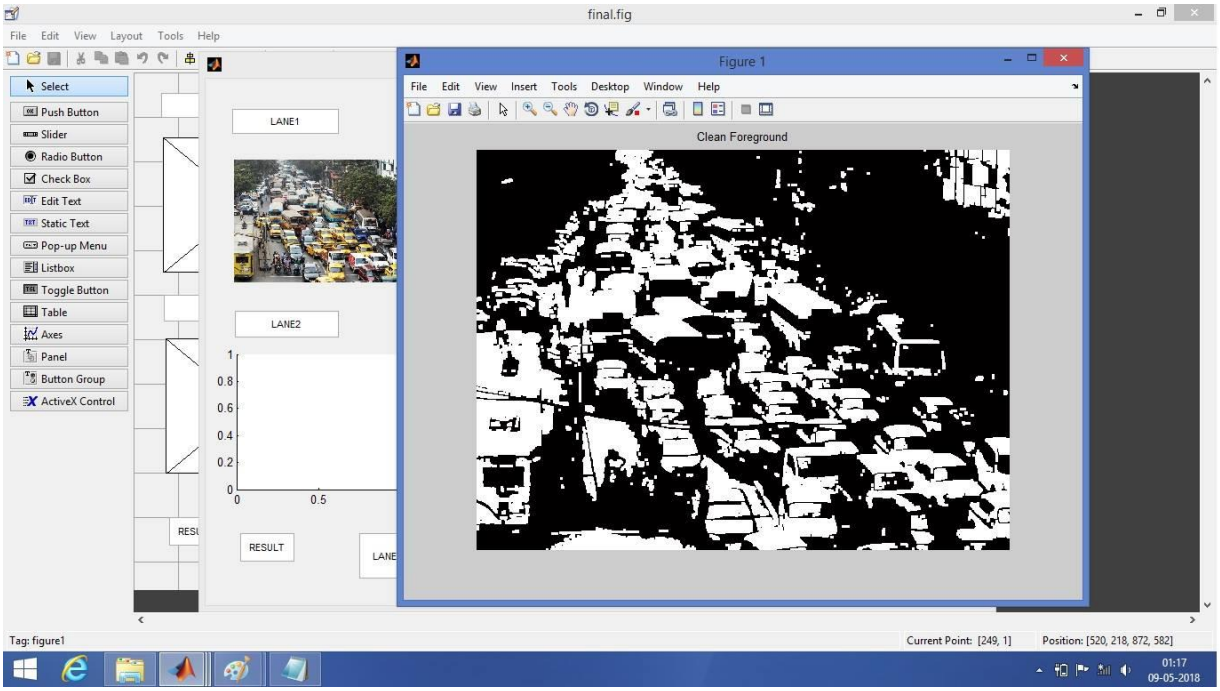


Fig-8.5 Comparison of car density between two lanes and controlling the signal accordingly with the timer for case 1.

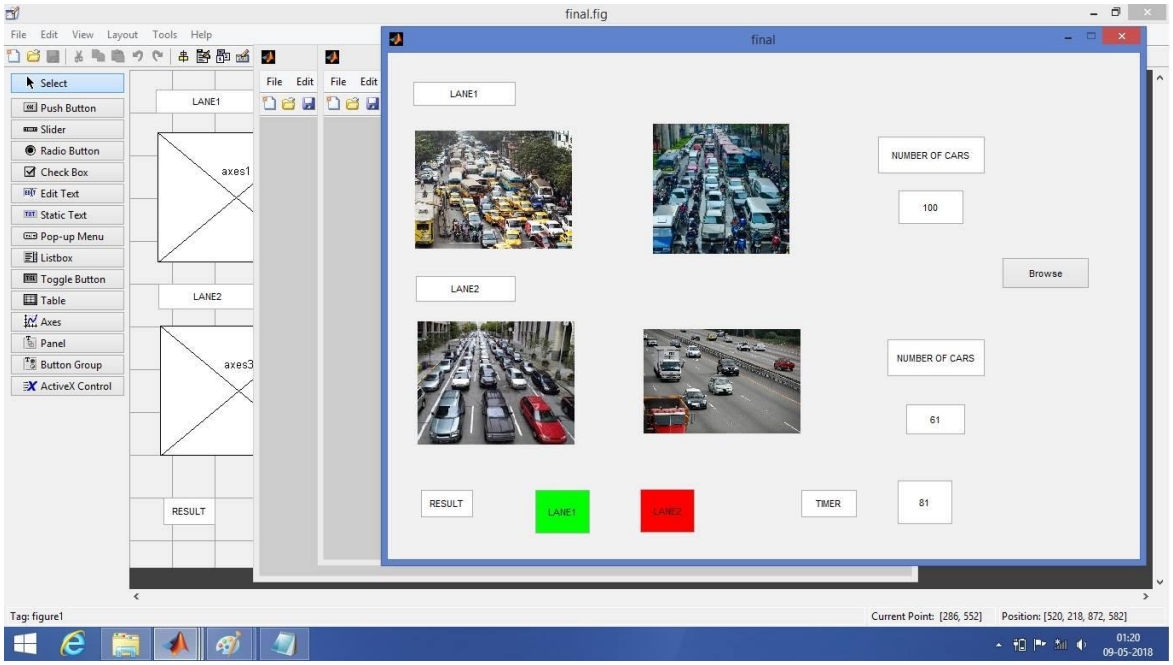



Fig-8.6 Comparison of car density between two lanes and controlling the signal accordingly with the timer for case 2.

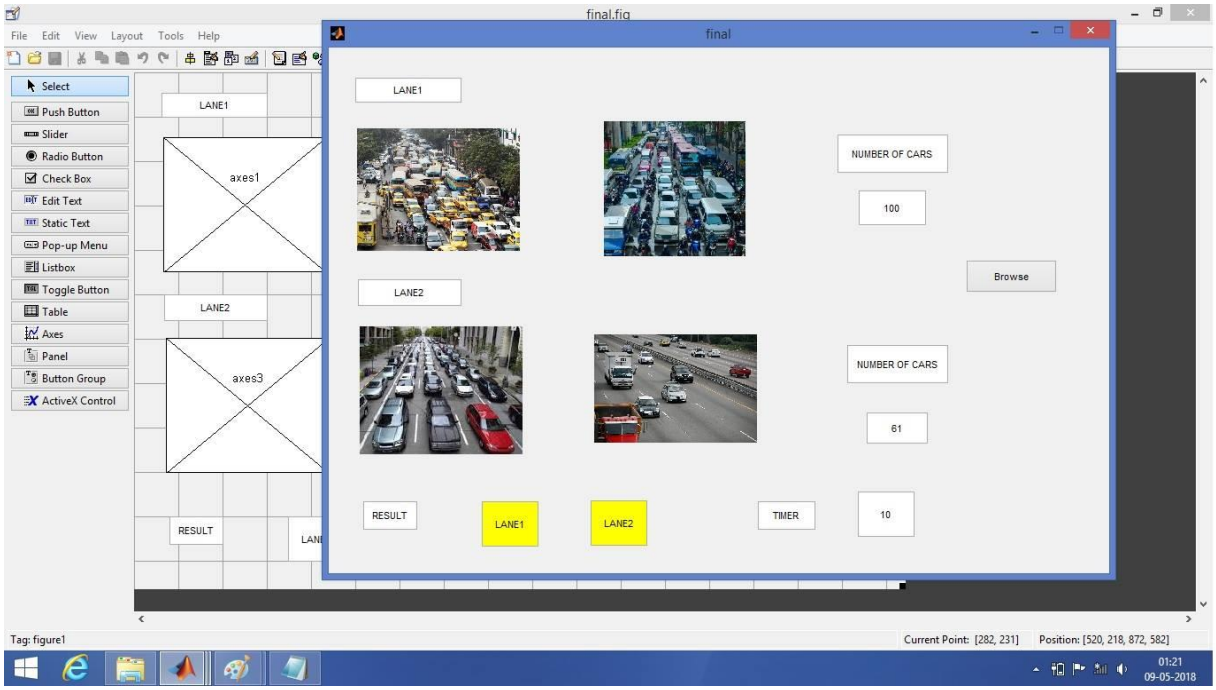
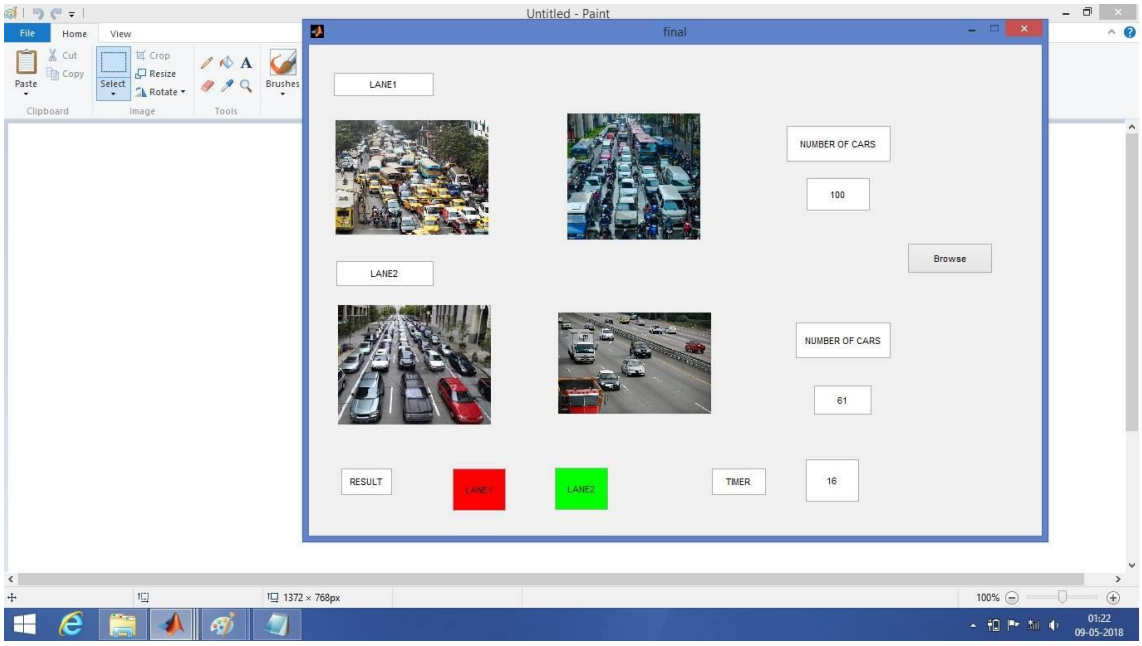


Fig-8.7 Comparison of car density between two lanes and controlling the signal accordingly with the timer for case 3.



# 9.Advantages and Dis-advantages:

## 9.1. Advantages-

The advantages of this new method include such benefits as use of image processing over sensors, low cost, easy setup and relatively good accuracy and speed. Because this method has been implemented using Image Processing and Mat lab software, production costs are low while achieving high speed and accuracy.

The advantages of this proposed technique are that there is no need to use aerial imagery or complex sensor-based systems. The proposed system is very

cost effective as it does not require installation of any additional devices, such as RFIDs.

It reduces the traffic congestion by reducing unwanted long-time delay and it keeps a track of the vehicles.

The advantages of this new method include such benefits as: 1) Non-use of sensors 2) Low cost and easy setup and relatively good accuracy and speed. Because this method has been implemented using Image Processing and MATLAB software, production costs are low while achieving high speed and accuracy.

## 9.2. Disadvantages-

Some of the researchers have focused in their work on traffic flow estimation. It is measured as the rate at which vehicles pass a fixed point (e.g. vehicles per minute). They used spot sensors such as loop detectors and pneumatic sensors to quantify the traffic flow.

However, the sensors are very expensive and need a lot of maintenance especially in developing countries because of the road ground de-formations.

It is also found that traffic congestion also occurred while using the electronic sensors for controlling the traffic.

most of the previous vision-based monitoring systems suffered from lack of robustness on dealing with continuously changing environment such as lighting conditions, weather conditions and unattended vehicles. All these factors considerably affect the traffic density estimation

The problem with the traffic density measurement is that the traffic density of a road with stationary or unattended vehicles is the same as the traffic density of a road with no stationary vehicles. Traffic flow counts the number of vehicles that passes through the frame during a certain time interval. However, it may give an empty road a higher priority than a congested road, because fewer vehicles are passing though the given point in that empty road. Therefore, we will

concentrate on the detection of the delayed and unattended vehicles in the proposed approach for computing more informative metric about the traffic congestion in order to have more effective way of traffic. This metric is very similar to the traffic density, but with taking the traffic flow into consideration. So, it can be considered as a combination of both traffic density and traffic flow.

# 10.Drawbacks:

1. This system is not adaptable for night vision images.
2. If the congestion has an ambulance struck inside it, then the system will consider it as a normal car count, and hence cannot understand the emergency.
3. The stand-by cars are also counted in the account, and hence even if no car is causing the traffic congestion.
4. Real life camera is very costly; hence it would provide some hindrance to hardware module of the system.

# 11.Conclusion:

In this paper, a method for estimating the traffic using Image Processing is presented. This is done by using the camera images captured from the highway and images taken are converted to the image sequences. Each image is processed separately and the number of cars has been counted. If the number of cars exceeds a specific threshold, warning of heavy traffic will be shown automatically. In this respect, the method is superior to previously published designs. The method presented in this paper is simple and there is no need to use sensors that have been commonly used to detect traffic in the past. With

some improvements, this method can be used to detect road accidents and identify violations of the spiral movements of cars.

# 12.Reference:

1. Digital image processing -Richard E Woods.

2. S Tridhar- digital image processing

3. https://en.wikipedia.org/wiki/Digital_image_processing