3^{rd} year 1^{st} semester B. Sc (hons) Final Examination 2018

IT 3100: Semester Project & Viva

Project on

Vehicle Number Plate Recognition

(By using MATLAB)

Project Supervisor

Dr Rashed Mazumder

Associate Professor

Prepared By

Name	Exam Roll
Md. Nahiduzzaman	160533
Saria Jahin	160561
Humayra Binte Arfan	160515
Zinia Anzum Tonni	160531

Institute of Information Technology
Jahangirnagar University

DECLARATION

This project report is submitted to the Institute of Information Technology, Jahangirnagar University, Savar, Dhaka in partial fulfillment of the requirements for having the B.Sc (Hons.) degree in IT. This is also needed to certify that the project work is under the 3rd Year 1st Semester course of the IIT "IT-3100: Semester Project & Viva". So, we, here declaring that this project report has not been submitted elsewhere for the requirement of any kind of degree, diploma or publication.

Name	Exam Roll
Md. Nahiduzzaman	160533
Saria Jahin	160561
Humayra Binte Arfan	160515
Zinia Anzum Tonni	160531

ACCEPTANCE

This project report is submitted to the Institute of Information Technology, Jahangirnagar University, Savar, Dhaka in partial fulfillment of the requirements for having the B.Sc (Hons.) degree in IT.

Dr Rashed Mazumder

Associate Professor

Institute of Information Technology (IIT)

Jahangirnagar University

Savar, Dhaka-1342, Bangladesh

ACKNOWLEDGEMENTS

First of all we would like to thank the Almighty for giving us the opportunity to complete this work successfully. Our acknowledgement is meant to express our sincere gratitude to all those people who have been associated with this project and have helped us with it and by sharing their experiences and valuable opinions through which we received the required information crucial for our project. We are thankful to our parents for their relentless support. Most importantly we are grateful to our honorable supervisor who took time out to guide us and provide us with all the necessary materials and sufficient knowledge that was the major requirement.

Finally, we express our thanks to our honorable teacher **Dr Rashed Mazumder** for giving us the opportunity to learn the subject in a practical.

ABSTRACT

Vehicle Number Plate Detection

In our daily life the use of vehicles is rising exponentially and as increasing vehicles are violating the rules of traffic, theft of vehicles, ingoing to restricted areas, abnormal number of accidents lead to upturn in the crime rates linearly. For any vehicle to be acknowledged, vehicle license plate detection will play a main significant job in this active world. The commonly used in field of safety and security system, we need to identify vehicles registration number .The methodology which we have used is modest but appropriate. First the segmente of all characters in the image (License Plate). Ultimately, the recognition of each character is done. The pattern matching method is used for recognition each character in the vehicle license plate.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

With increasing number of vehicles on roads, it is getting difficult to manually enforce laws and traffic rules for smooth traffic flow. Toll-booths are constructed on freeways and parking structures, where the car has to stop to pay the toll or parking fees. Also, Traffic Management systems are installed on freeways to check for vehicles moving at speeds not permitted by law. All these processes have a scope of improvement. In the center of all these systems lies a vehicle. In order to automate these processes and make them more effective, a system is required to easily identify a vehicle. The important question here is how to identify a particular vehicle? The obvious answer to this question is by using the vehicle's number plate.

Vehicles in each country have a unique license number, which is written on its license plate. This number distinguishes one vehicle from the other, which is useful especially when both are of same make and model. An automated system can be implemented to identify the license plate of a vehicle and extract the characters from the region containing a license plate. The license plate number can be used to retrieve more information about the vehicle and its owner, which can be used for further processing. Such an automated system should be small in size, portable and be able to process data at sufficient rate.

1.2 Purpose of this Project

The main purpose of this project is to detect and recognize a license plate from an image provided by a camera. An efficient algorithm is developed to detect a license plate in various conditions. This algorithm extracts the license plate data from an image and provides it as an input to the stage of Car License Plate Recognition. Extracted image of the number plate can be seen on monitor for verification purpose. From image the character also be extracted which can be further use to verify the details of the vehicle.

The scope of this project is to detect the license plate from the given image and observe the output on television. This project can work as a base for future improvements in the field of image processing, especially in license plate extraction and plate number recognition.

1.3 Organization of Report

Chapter two of the report explains the fundamentals of image processing, information about pixel in an image, RGB .

Chapter three explains the implementation of the above fundamental topic and algorithm for this project. It discusses about various steps of algorithm in detail. Suitable images are also provided to show output at different stages in applied algorithm.

The information about the results obtained after implementing this project and its performance analysis is concluded in Chapter four of this report.

Chapter 2

FUNDAMENTALS OF IMAGE PROCESSING

An image is used to convey useful information in a visible format. An image is nothing but an arrangement of tiny elements in a two-dimensional plane. These tiny elements are called Pixels. A large number of pixels combine together to form an image, whether small or large. Each pixel represents certain information about the image, like color, light intensity and luminance. A large number of such pixels combine together to form an image. Pixel is the basic element used to describe an image. Mostly, each pixel in an image is represented in either RGB (Red Green Blue) format. In case of an RGB image, all the three components, namely R, G and B combine together to convey information about the color and brightness of a single pixel. Each component consumes certain memory space during image processing.

2.1 RGB Format

In case of an RGB image, each pixel is represented by three different components R, G and B. Each of these components requires at least 8 bits for their storage. In general, a single pixel may require upto 8 * 3 bits for its storage. An example of a representation of single pixel in RGB format is shown below.

R	G	B	R	G	B

Fig. 2.1 Representation of pixels in RGB format.

The value of R, G and B each ranges from 0-255. A value of (0, 0, 0) represents a black pixel, (255, 0, 0) represents a red pixel and (0, 255, 0) represents a green pixel. So, 8 bits are required to store value for a single component.

2.2 Morphological Image Processing

Binary images may contain numerous imperfections. In particular, the binary regions produced by simple thresholding are distorted by noise and texture. Morphological image processing pursues the goals of removing these imperfections by accounting for the form and structure of the image. These techniques can be extended to greyscale images.

Morphological image processing is a collection of non-linear operations related to the shape or morphology of features in an image. Morphological operations rely only on the relative ordering of pixel values, not on their numerical values, and therefore are especially suited to the processing of binary images. Morphological operations can also be applied to greyscale images such that their light transfer functions are unknown and therefore their absolute pixel values are of no or minor interest.

2.2.1 Morphological Dilation and Erosion

The most basic morphological operations are dilation and erosion. Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries. The number of pixels added or removed from the objects in an image depends on the size and shape of the structuring element used to process the image. In the morphological dilation and erosion operations, the state of any given pixel in the output image is determined by applying a rule to the corresponding pixel and its neighbors in the input image. The rule used to process the pixels defines the operation as a dilation or an erosion. This table lists the rules for both dilation and erosion.

2.3 Convolution

Convolution is a simple mathematical operation which is fundamental to many common image processing operators. Convolution provides a way of `multiplying together' two arrays of numbers, generally of different sizes, but of the same dimensionality, to produce a third array of numbers of the same dimensionality. This can be used in image processing to implement operators whose output pixel values are simple linear combinations of certain input pixel values.

In an image processing context, one of the input arrays is normally just a graylevel image. The second array is usually much smaller, and is also two-dimensional (although it may be just a single pixel thick), and is known as the kernel.

2.4 Thinning

Thinning is a morphological operation that is used to remove selected foreground pixels from binary images, somewhat like erosion or opening. It can be used for several applications, but is particularly useful for skeletonization. In this mode it is commonly used to tidy up the output of edge detectors by reducing all lines to single pixel thickness. Thinning is normally only applied to binary images, and produces another binary image as output.

The thinning operation is related to the hit-and-miss transform, and so it is helpful to have an understanding of that operator before reading on.

2.5 Character Recognition using template matching

Template matching or matrix matching, is one of the most common classification methods. Here individual image pixels are used as features. Classification is performed by comparing an input character with a set of templates (or prototypes) from each character class. Each comparison results in a similarity measure between the input characters with a set of templates. If the pixels differ the measure of similarity may be decreased. After all templates have been compared with the observed character image, the character's identity is assigned the identity of the most similar template.

Chapter 3

MATLAB IMPLEMENTATION

3.1 Introduction

This chapter describes the implementation of License Plate Detection algorithm using MATLAB. MATLAB is a very powerful software tool used to implement the tasks that require extensive computation. It provides easy and quicker implementation of algorithms compared to C and C++. The key feature in MATLAB is that it contains a rich library functions for image processing and data analysis. This makes MATLAB an ideal tool for faster implementation and verification of any algorithm before actually implementing it on a real hardware. Sometimes, debugging of errors on actual hardware turns out to be a very painful task. MATLAB provides an easy approach for debugging and correction of errors in any algorithm. Other than this, MATLAB contains many features including workspace, plot, imread, imhist, imshow, etc. for data analysis and image processing, which makes it a better choice over other software languages like C and C++.

Considering the above advantages, the writer of this project initially implemented an algorithm for License Plate Detection using MATLAB. The algorithm initially used various inbuilt functions and implemented few user defined routines related to image processing. Once the algorithm was developed, it was verified with multiple input images containing car number plates. The input images contained number plates that were aligned horizontally as well as at some angle from horizontal axis. Once the algorithm was completely verified, the in-built functions of MATLAB were replaced by userdefined functions. A flow-chart showing the basic implementation of algorithm is shown below:

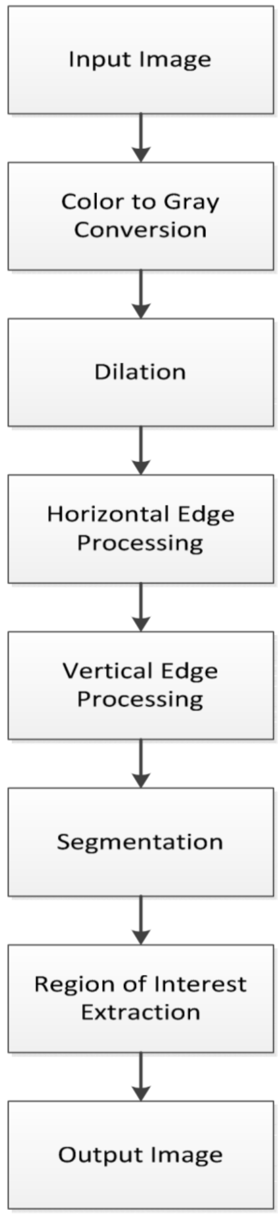


Fig. 3.1 Flowchart showing license plate detection algorithm in MATLAB.

The steps of implementing License Plate Detection algorithm in MATLAB are described below.

3.2 Convert a Colored Image into Gray Image

The algorithm described here is independent of the type of colors in image and relies mainly on the gray level of an image for processing and extracting the required information. Color components like Red, Green and Blue value are not used throughout this algorithm. So, if the input image is a colored image represented by 3-dimensional array in MATLAB, it is converted to a 2-dimensional gray image before further processing. The sample of original input image and a gray image is shown below:

Fig 3.2 Original color image.

Fig. 3.3 Gray image.

3.3 Dilate an Image

Dilation is a process of improvising given image by filling holes in an image, sharpen the edges of objects in an image, and join the broken lines and increase the brightness of an image. Using dilation, the noise with-in an image can also be removed. By making the edges sharper, the difference of gray value between neighboring pixels at the edge of an object can be increased. This enhances the edge detection.

In Number Plate Detection, the image of a car plate may not always contain the same brightness and shades. Therefore, the given image has to be converted from RGB to gray form. However, during this conversion, certain important parameters like difference in

Fig. 4.4 Dilated image.

color, lighter edges of object, etc. may get lost. The process of dilation will help to nullify such losses.

3.4 Horizontal and Vertical Edge Processing of an Image

Histogram is a graph representing the values of a variable quantity over a given range. In this Number Plate Detection algorithm, the writer has used horizontal and vertical histogram, which represents the column-wise and row-wise histogram respectively. These histograms represent the sum of differences of gray values between neighboring pixels of an image, column-wise and row-wise.

In the above step, first the horizontal histogram is calculated. To find a horizontal histogram, the algorithm traverses through each column of an image. In each column, the algorithm starts with the second pixel from the top. The difference between second and first pixel is calculated. If the difference exceeds certain threshold, it is added to total sum of differences. Then, algorithm will move downwards to calculate the difference between the third and second pixels. So on, it moves until the end of a column and calculate the total sum of differences between neighboring pixels. At the end, an array containing the column-wise sum is created. The same process is carried out to find the vertical histogram. In this case, rows are processed instead of columns.

3.5 Passing Histograms through a Low Pass Digital Filter

Referring to the figures shown below, one can see that the histogram values changes drastically between consecutive columns and rows. Therefore, to prevent loss of important information in upcoming steps, it is advisable to smooth out such drastic changes in values of histogram. For the same, the histogram is passed through a low-pass digital filter. While performing this step, each histogram value is averaged out considering the values on it right-hand side and left-hand side. This step is performed on both the horizontal histogram as well as the vertical histogram. Below are the figures showing the histogram before passing through a low-pass digital filter and after passing through a low-pass digital filter.

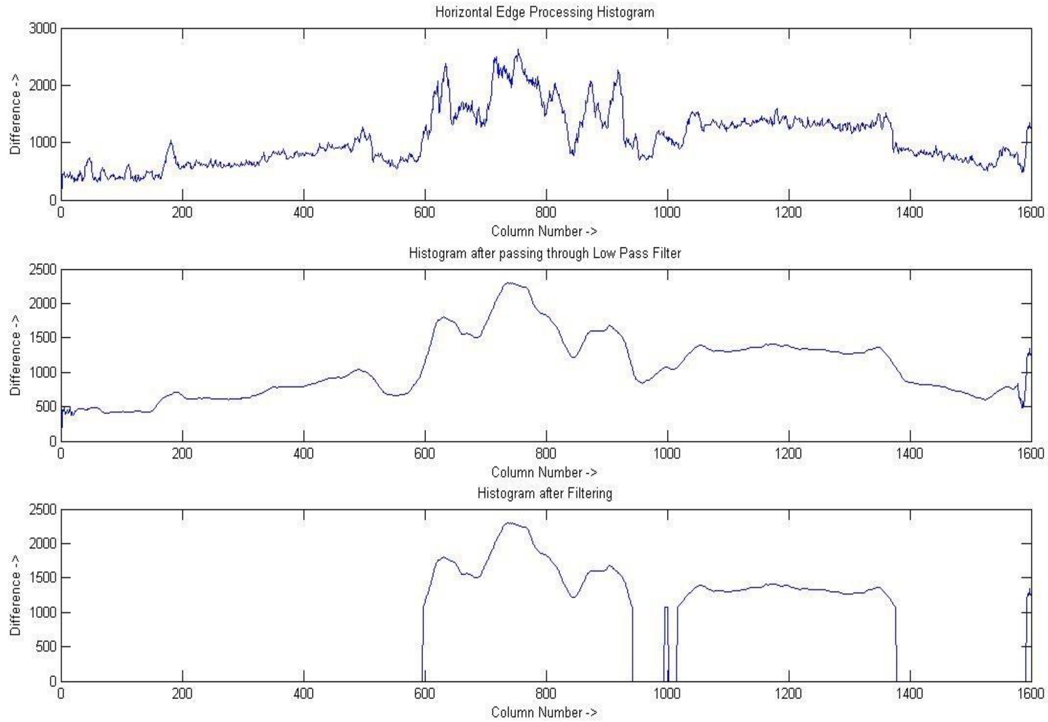


Fig. 3.5 Horizontal edge processing histogram.

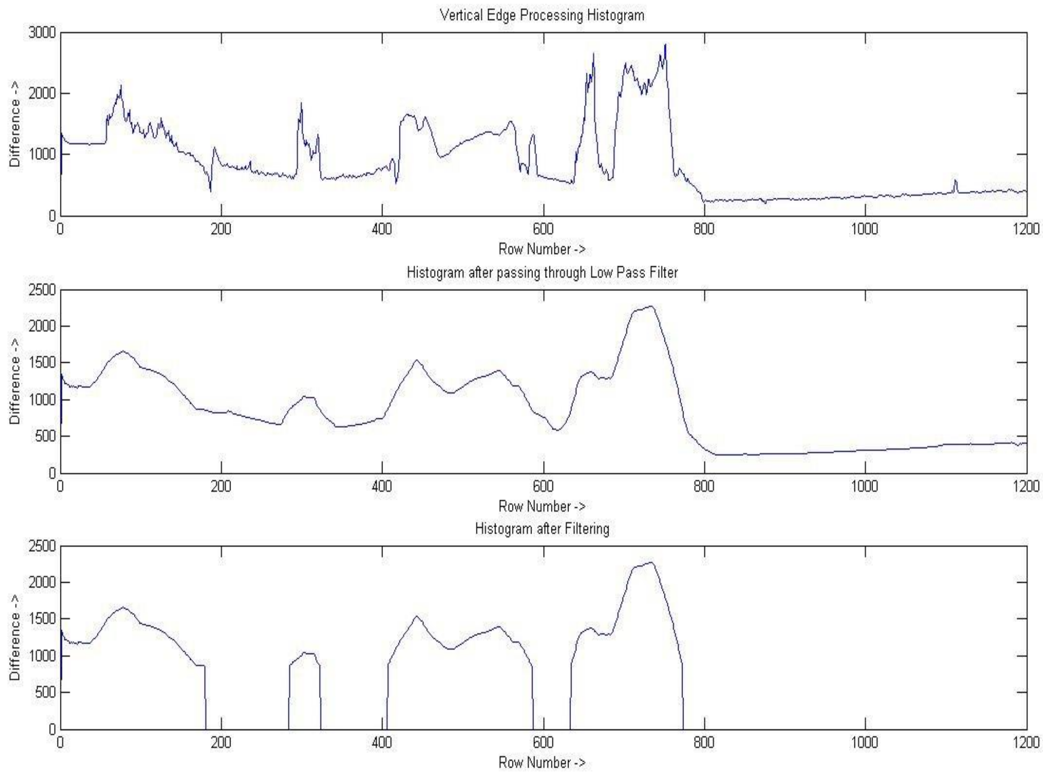


Fig. 3.6 Vertical edge processing histogram.

3.6 Filtering out Unwanted Regions in an Image

Once the histograms are passed through a low-pass digital filter, a filter is applied to remove unwanted areas from an image. In this case, the unwanted areas are the rows and columns with low histogram values. A low histogram value indicates that the part of image contains very little variations among neighboring pixels. Since a region with a license plate contains a plain background with alphanumeric characters in it, the difference in the neighboring pixels, especially at the edges of characters and number plate, will be very high. This results in a high histogram value for such part of an image.

Therefore, a region with probable license plate has a high horizontal and vertical histogram values. Areas with less value are thus not required anymore. Such areas are removed from an image by applying a dynamic threshold.

In this algorithm, the dynamic threshold is equal to the average value of a histogram. Both horizontal and vertical histograms are passed through a filter with this dynamic threshold. The output of this process is histogram showing regions having high probability of containing a number plate.

3.7 Segmentation

The next step is to find all the regions in an image that has high probability of containing a license plate. Co-ordinates of all such probable regions are stored in an array. The output image displaying the probable license plate regions is shown below.

Fig. 4.7 Output of segmentation.

3.8 Region of Interest Extraction

The output of segmentation process is all the regions that have maximum probability of containing a license plate. Out of these regions, the one with the maximum histogram value is considered as the most probable candidate for number plate. All the regions are processed row-wise and column-wise to find a common region having maximum horizontal and vertical histogram value. This is the region having highest probability of containing a license plate. The image detected license plate is shown below:

Fig. 3.8 Detected license plate.

This algorithm was verified using several input images having resolution varying from 680 * 480 to 1600 * 1200. The images contained vehicles of different colors and varying intensity of light. With all such images, the algorithm correctly recognized the number plate. This algorithm was also tried on images having number plate aligned at certain angle (approximately 8-10 degree) to horizontal axis. Even with such images, the number plates were detected successfully. After successfully implementing and verifying the algorithm in MATLAB, it was coded in C for implementation on actual hardware.

The details about the same are covered in next chapter.

3.9 Template Matching

After getting gray scale image we separate the word using blob analysis. Each character is called as blob. Each character is match with the training dataset. Template matching algorithm is used for matching blob with the training database. To match blob with training dataset feature extraction algorithm is used. If character is matched accurately then character is given as input to template generator.

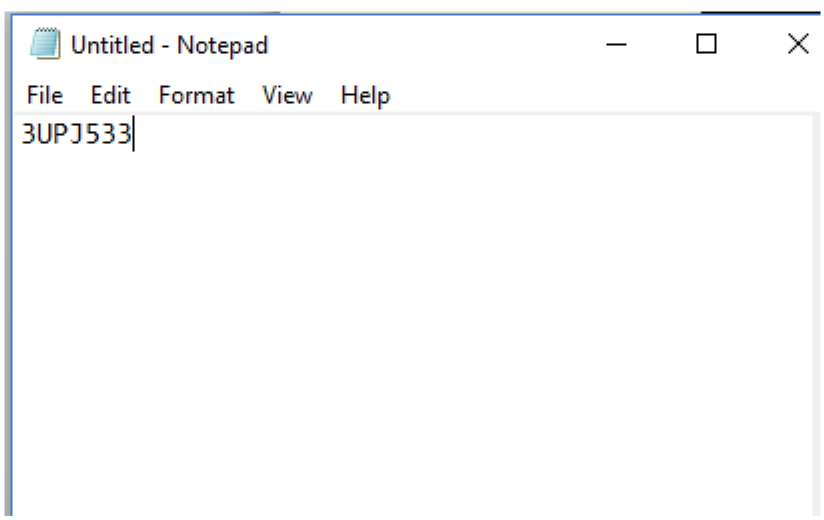


Fig. 3.9 Template Matching

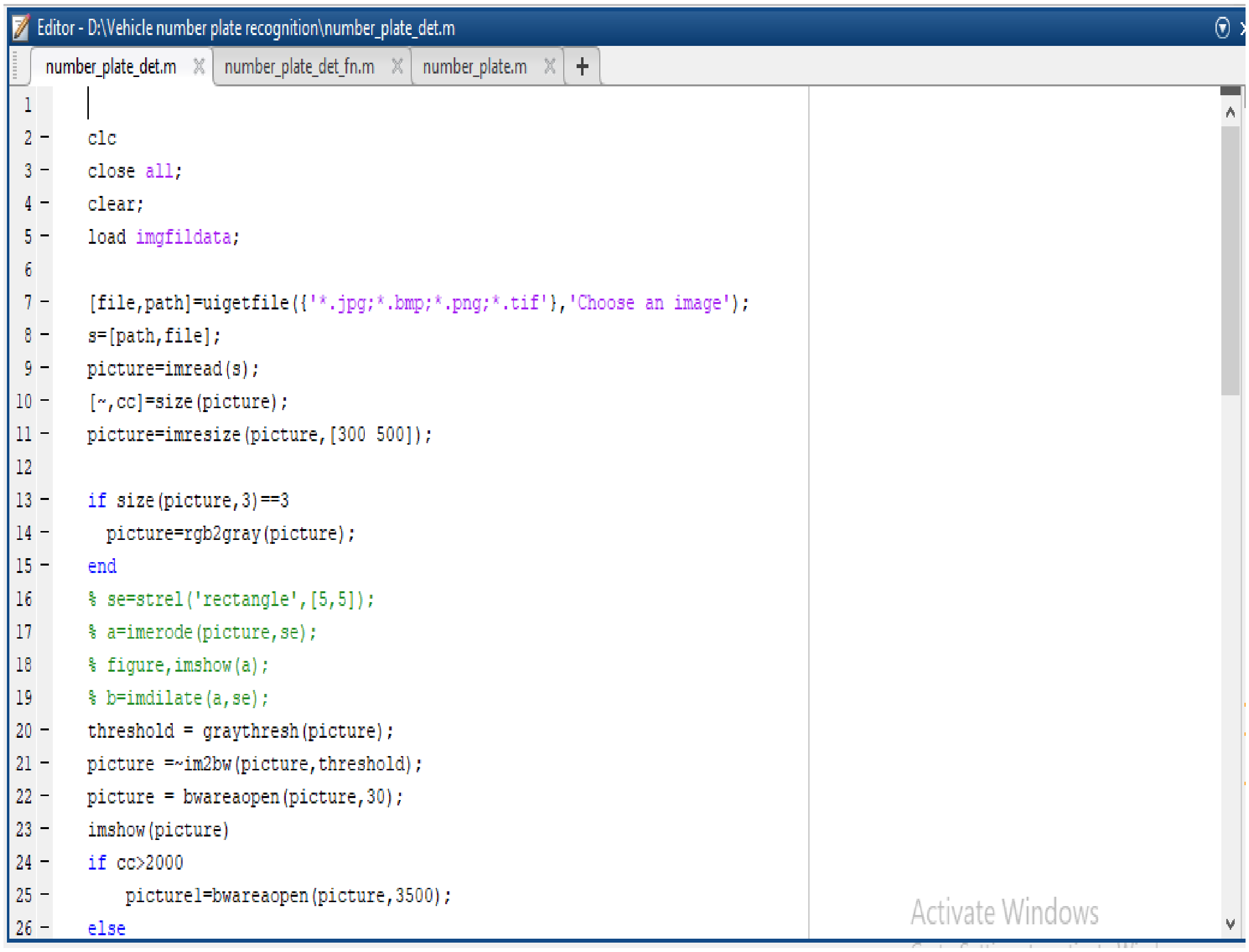
Chapter 4

RESULTS AND PERFORMANCE ANALYSIS

4.1 Results

Some screen shot of the code and Result is given bellow.

Fig4.1 This is the given image




```
clc
close all;
clear;
load imgfildata;

[file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
s=[path,file];
picture=imread(s);
[~,cc]=size(picture);
picture=imresize(picture,[300 500]);

if size(picture,3)==3
  picture=rgb2gray(picture);
end
% se=strel('rectangle',[5,5]);
% a=imerode(picture,se);
% figure,imshow(a);
% b=imdilate(a,se);
threshold = graythresh(picture);
picture =~im2bw(picture,threshold);
picture = bwareaopen(picture,30);
imshow(picture)
if cc>2000
    picture1=bwareaopen(picture,3500);
else
```

Fig4.2 Matlab Code

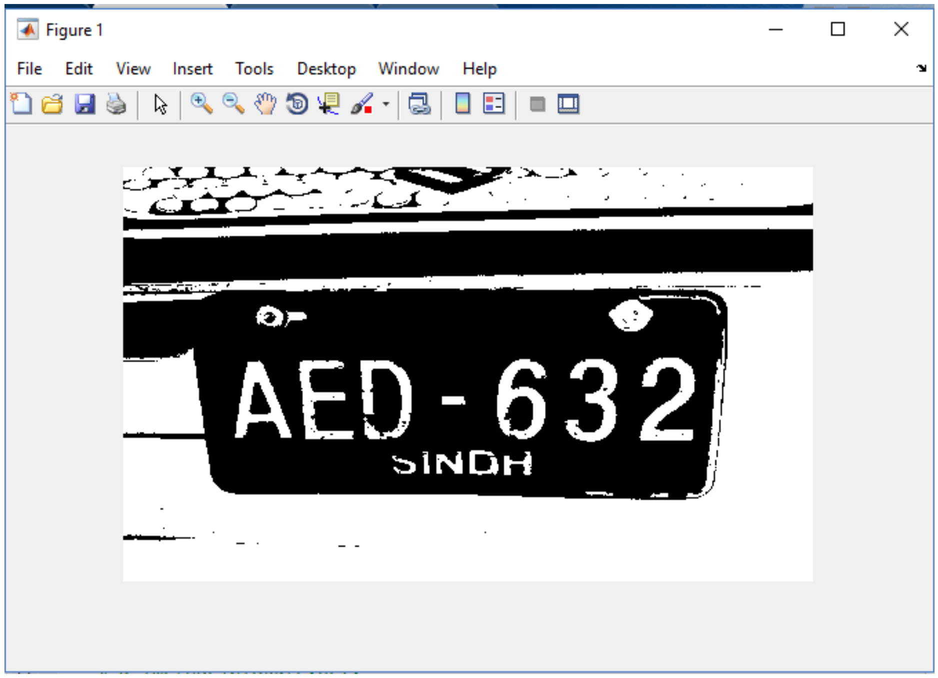


Fig4.3 Output of the code(figure 1)

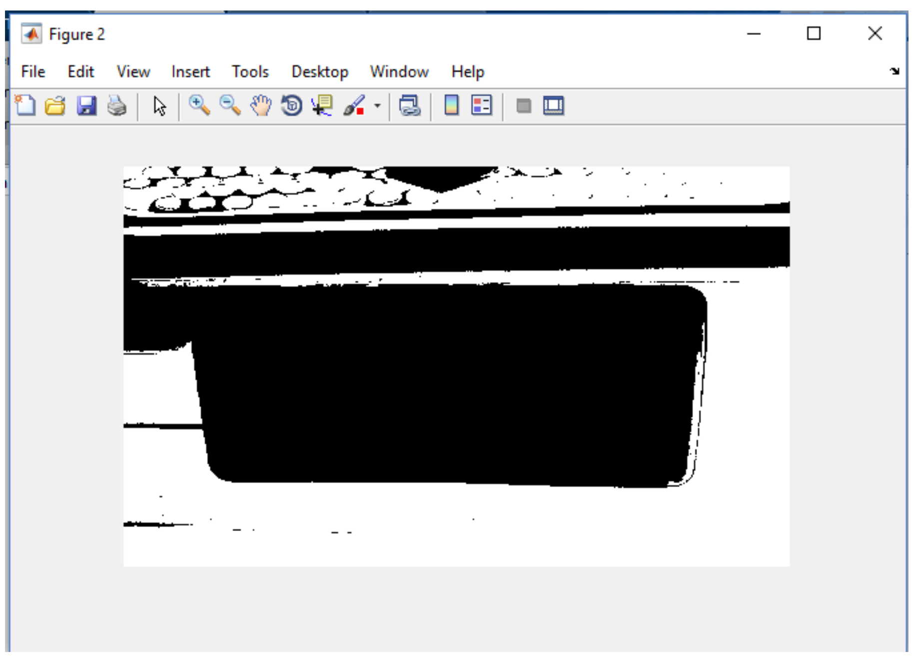


Fig 4.4 Output of the code(figure 2)

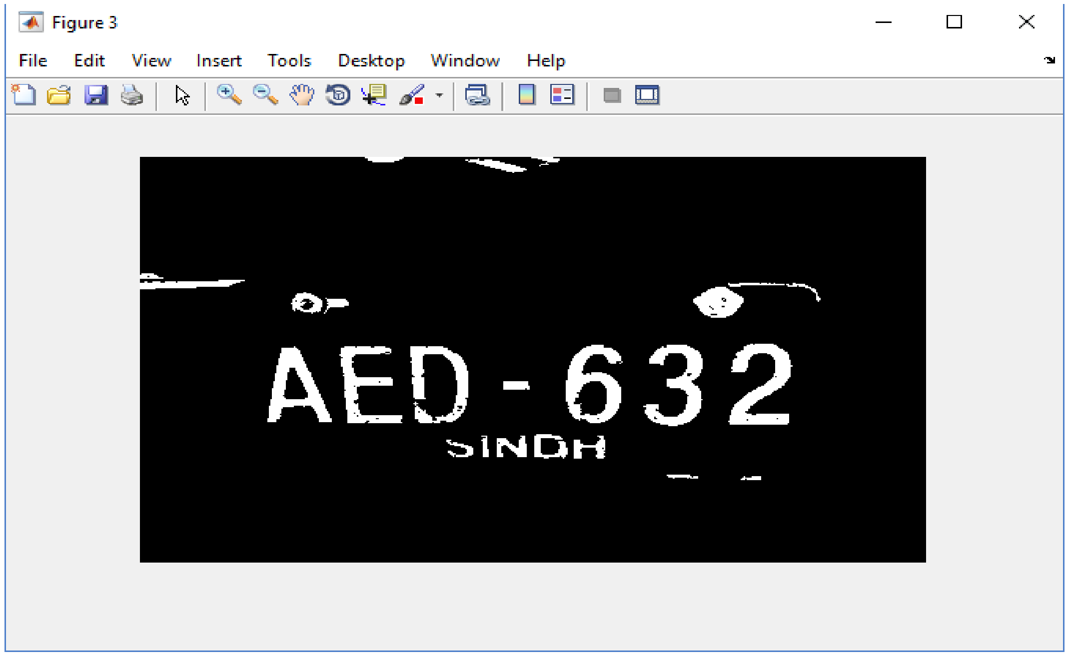


Fig 4.5 Output of the code (figure 3)

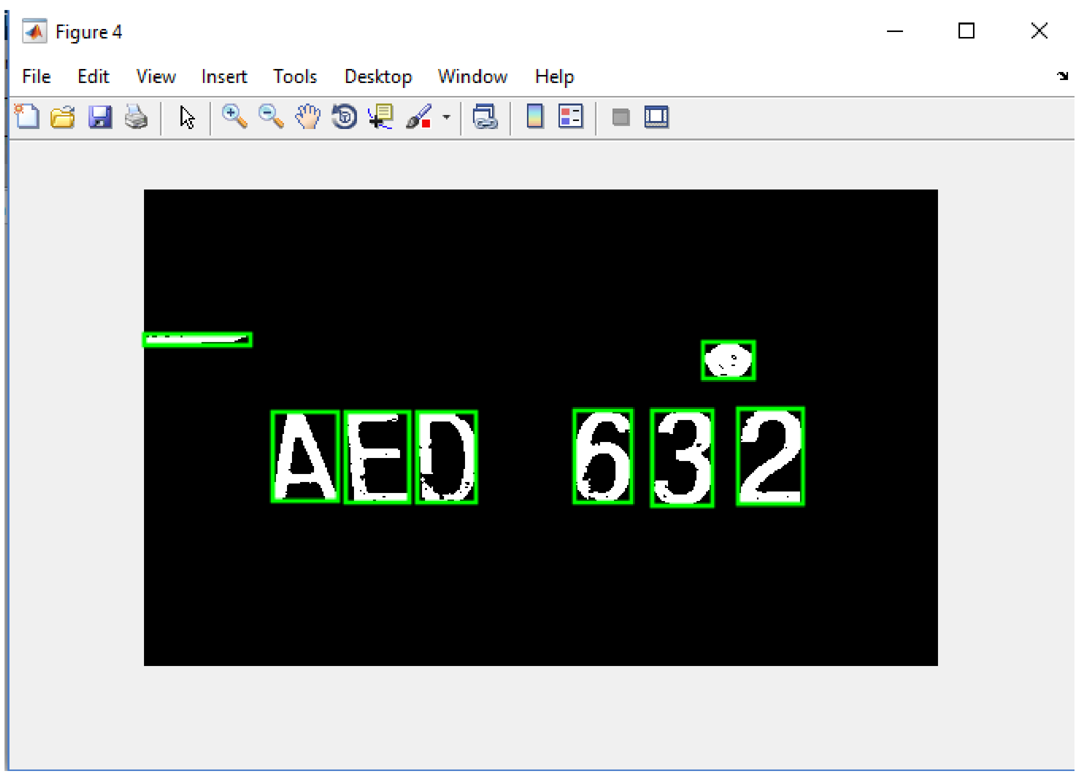


Fig 4.6 Output of the code (figure 4)

Fig 4.7 Output of the code (figure 5)

4.2 Conclusion

The purpose of this study has been to create an automatic number plate recognition system. It describes complete intelligent digital image processing system to recognize the vehicle number plate using a sequential multistage approach performed over grayscale images, consists of two parts extraction of vehicle number plate from the image, and recognition of number plate. For extracting the vehicle number plate edge detection algorithm used. The experimental results show that this method is of higher recognition accuracy . This new approach provides a good direction for automatic number plate recognition. The system was designed using MATLAB.

BIBLIOGRAPHY

[1] Clemens Arth, Florian Limberger and Horst Bischof, "Real-Time License Plate Recognition on an Embedded DSP-Platform", Proceedings of IEEE conference on Computer Vision and Pattern Recognition, pp 1-8, June 2007.

[2] Halina Kwasnicka and Bartosz Wawrzyniak, "License plate localization and recognition in camera pictures", AI-METH 2002, November 13-15, 2002.

[3] Texas Instruments Inc., "TMS320DM6437 DVDP Getting Started Guide", Texas, July 2007.

[4] Texas Instruments Inc., "TMS320DM6437 Digital Media Processor", Texas, Page 5, November 2006 – Revised June 2008.

[5] Pramod Kapadia, "Car License Plate Recognition Using Template Matching Algorithm", Master Project Report, California State University, Sacramento, Fall 2010.