Mini Project

Shri Guru Gobind Singhji Institute of Engineering and Technology, Vishnupuri, Nanded.

Number Plate Detection Using MATLAB

Submitted by

(a) Gunjal Shahu Yogesh
(A09)

(b) Kulkarni Ajinkya Anantrao
(A17)

(c) Balki Saurabh Vilas
(A18)

Under the guidance of

Suraj Kulkarni Sir

Electronics & Telecommunication Engineering

2020 – 2021

ABSTRACT

In last couple of decades, the number of vehicles has increased drastically. With this increase, it is becoming difficult to keep track of each vehicle for purpose of law enforcement and traffic management. Vehicle number plate detection is used increasingly nowadays for automatic toll collection, maintaining traffic activities and law enforcement. Many techniques have been proposed for plate detection, each having its own advantages and disadvantages. The basic step in Vehicle number plate detection is localization of number plate. The approach mentioned in this project is a histogram-based approach. This approach has an advantage of being simple and thus faster. License plate localization is implemented using MATLAB and verified for its functionality.

This work proposes a method for the detection and identification of vehicle number plate that will help in the detection of number plates. This paper presents an approach based on simple but efficient morphological operation, Sobel edge detection and Image Processing method. This approach is simplified to segmented all the letters and numbers used in the number plate by using bounding box method and then to use Neural Networks to recognition of numbers and characters. The concentrate is given to locate the number plate region properly to segment all the number and letters to identify each number separately.

Keywords:

Vehicle Number Plate Detection (VNPD).

RED GREEN BLUE (RGB).

Gray Processing.

Image Acquisition.

Image Binarization, Template Matching.

ACKNOWLEDGMENTS

In the present world of competition there is a race of existence in which those are having will to come forward succeed. Project is like bridge between theoretical and practical working. With this willing we joined this particular project. First of all I would like to thank our supervisor for this mini project course Prof. Suraj Kulkarni who is obviously the one has guided us to work on this project. Without his enthusiasm this project would not become been reality. We are feeling oblige in taking the opportunity to sincerely thanks our supervisor and special thanks to International Journal of Computational Intelligence Research from where we collected the necessary information for the project. At last, but not least we are thankful for ourselves for doing all this work, giving best, dedicating oneself for this project in this crucial condition.

We are happy to work on this project which is very important for the conditions we are living in. Working on image processing topic has taught us lot of things and made us confident enough to work on any important projects in future. Thank you EXTC department for conducting such special course in the curriculum.

Thank you, Professor, all friends and fellow groupmates.

METHODOLOGY

The working of full VNPD system can be divided in to two broad sections. The hardware part and the software part. The working mechanism of all the parts is described in details below.

Software Model: The first and the most important part in this process is the software model. The software model uses the image processing technology. The programs are implemented in MATLAB. The algorithm is divided into following parts: Capture image, Pre-processing, Plate region extraction, Segmentation of character in the extracted number plate, Character recognition, Comparison with database and Indicate result. The flow chart of license plate recognition system implementation in this work is shown in the following figure. There are various steps in this approach and these are implementation in MATLAB.

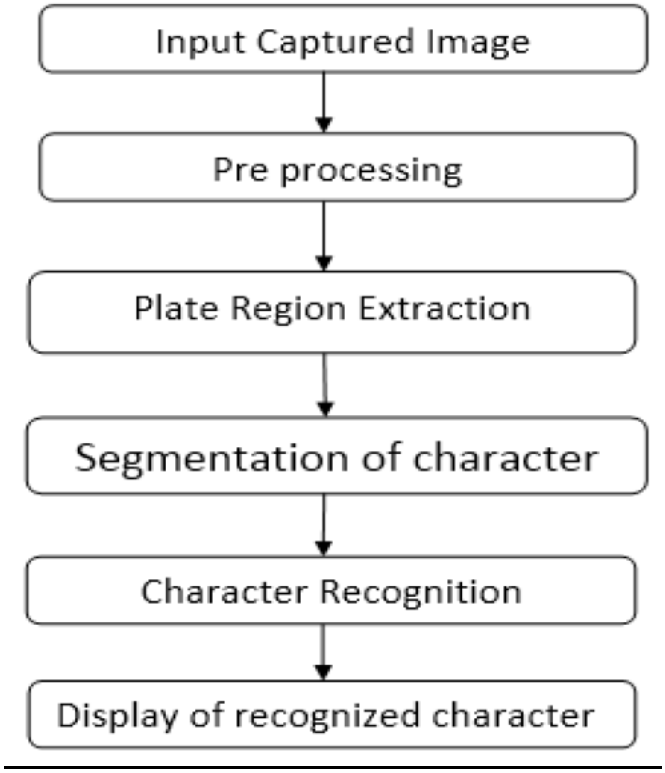

Fig. Flow Diagram of VNPD

INTRODUCTION

- **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 centre 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.

Various license plate detection algorithms have been developed in past few years. Each of these algorithms has their own advantages and disadvantages. But this method suggests a different approach of detection using binarization and elimination of unnecessary regions from an image. In this approach, initial image processing and

binarization of an image is carried out based on the contrast between characters and background in license plate. After binarizing the image, it is divided into different black and white regions. These regions are passed through elimination stage to get the final region having most probability of containing a number plate.

- **Purpose of this Project**

The main purpose of this project is to detect a license plate from an image of vehicles. An efficient algorithm is developed to detect a license plate in various luminance conditions. This algorithm extracts the license plate data from an image and provides it as an input to the stage of Vehicle number plate detection system. The image of a vehicle is given as an input from the camera. Extracted image of the number plate can be seen verification purpose.

We want to gain an insight on implementing image-processing algorithm. 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.

- **Significance of this Project**

Through this project, an algorithm for license plate detection from an image is implemented in MATLAB. On the hardware side, this algorithm is downloaded onto TI's Digital Video Development Platform that contains Texas Instrument's DaVinci Digital Signal Processors. Code Composer Studio is used to develop and implement the algorithm on actual hardware using C programming language. The EVM320DM6437 Development Kit is also studied for the purpose of available resources and their appropriate usage.

CONTENTS

1. 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 colour, 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 or YCbCr format. In case of an RGB image, all the three components, namely R, G and B combine together to convey information about the colour and brightness of a single pixel. Each component consumes certain memory space during image processing.

In case of a YCbCr image, each pixel in an image is represented as a combination of Y and Cb/Cr values. Here, Y stands for luminance, which describes light intensity, and Cb/Cr stands for chroma component, which describes colour information for an image. Over the time, it has been found that YCbCr components of an image convey sufficient amount of information compared to its counter parts RGB, with less amount of memory space. This is a major advantage nowadays, as most of the applications require sufficient information at very high speed and less storage.

- **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 up to 8 * 3 bits for its storage. An example of a representation of single pixel in RGB format is shown below.

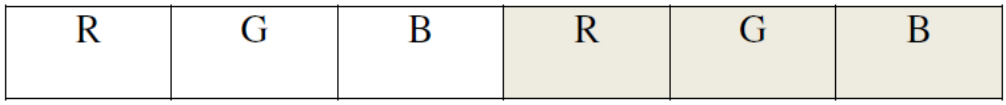

Fig. Representation of pixels in RGB format.

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.

- **YCbCr Format**

In contrast to RGB format, the YCbCr format is available with various kind of interleaving. For example, a 4:2:2 YCbCr format suggests that a single pixel is represented by two components, Y and C. Cb and Cr components are interleaved among the pixels. So if one pixel is represented by a combination of Y and Cb, the adjacent pixel will be represented by a combination of Y and Cr. Even if the Cb and Cr components are interleaved, its effect is not visible to human eye.

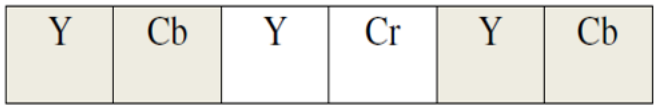

Fig. Representation of pixels in YCbCr format.

Values for Y, Cb and Cr vary from 0-255. Thus, to store a single pixel, the amount of storage required is 8 * 2 bits, which is less compared to that required by RGB format. For this project, Texas Instrument's EVM320DM6437 kit is to be used for license plate detection. The kit contains internal buffers to store the incoming frames of video. The format for the type of storage is shown below.

Cb	Y	Cr	Y
Cb	Y	Cr	Y
Cb	Y	Cr	Y

Fig. A part of frame buffer storage for input video frames.

From the above image, it is seen that the storage of frame starts with a C component and then a Y component. Therefore, at the 0th location, one can find the C component while at the 1st and alternate locations of Frame Buffer, one can find the Y component.

2. GENERAL VNPD SYSTEM

Block Diagram of VNPD is shown in Figure,

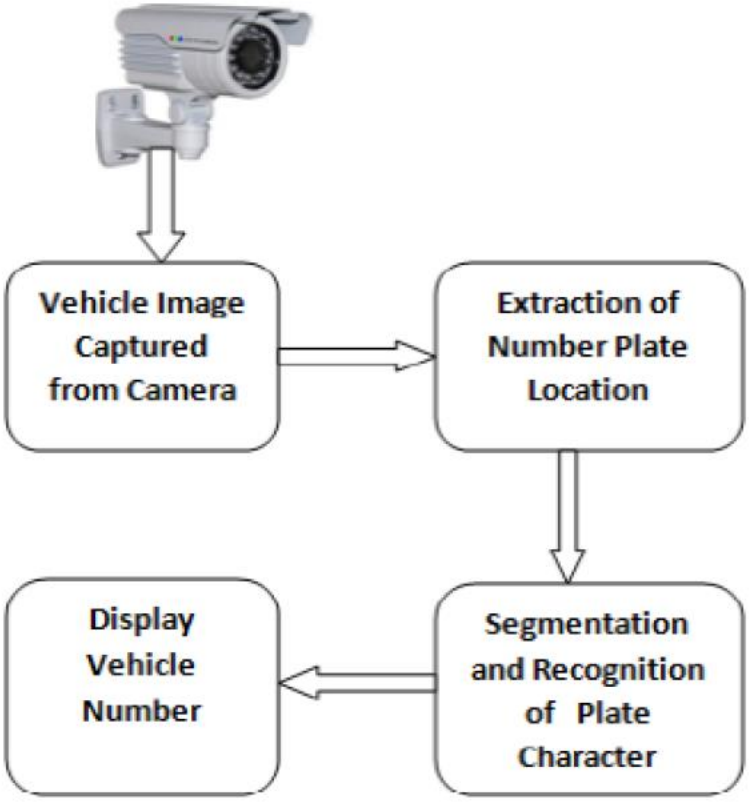

Fig. System Block Diagram

3. MATLAB IMPLEMENTATION

This part describes the implementation of Vehicle Number 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 user defined functions. A flow-chart showing the basic implementation of algorithm is shown below:

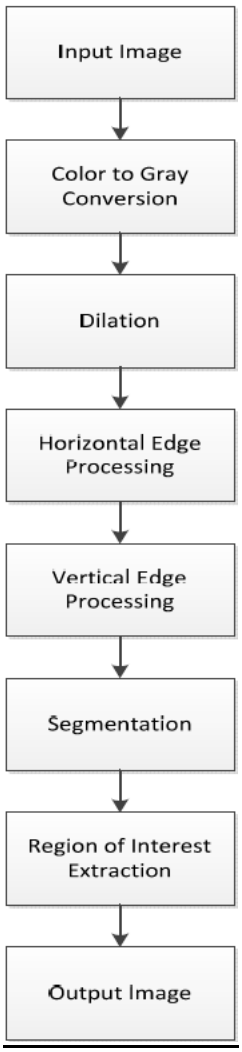

Fig. Flowchart showing license plate detection algorithm in MATLAB.

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

- **Convert a Coloured Image into Gray Image**

The algorithm described here is independent of the type of colours in image and relies mainly on the Gray level of an image for processing and extracting the required information. Colour components like Red, Green and Blue value are not used throughout this algorithm. So, if the input image is a coloured 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. Original colour image.

Fig. Gray image.

- **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 neighbouring 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 colour, lighter edges of object, etc. may get lost. The process of dilation will help to nullify such losses.

Fig. Dilated Image

- **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 neighbouring 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 neighbouring 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.

- **Image 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. Output of Segmentation

- **Character Segmentation**

Segmentation is one of the most important processes in the number plate recognition, because all further steps rely on it. If the segmentation fails, a character can be improperly divided into two pieces, or two characters. The ultimate solution on this problem is to use bounding box technique. Once a bounding box created over each character and numbers presented on number plate, each character & number is separate out for recognition of number plate.

In the bounding box technique we used some instructions, bwlabel to detect the number of connected elements in the image and the matrix of the image <480*640>.

Regionprops.bounding box instruction detects matrices, which contain the coordinates of upper left corner of the bounding box and specifies the width of the bounding box along each dimension.

And then for each connected element we draw a rectangle as we can see in the code.

Fig. Character Segmentation

- **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. 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 colours 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 can be coded in C for implementation on actual hardware for real time applications.

RESULTS AND PERFORMANCE ANALYSIS

This part contains the output results of Car License Plate Detection algorithm implementation on MATLAB, advantages and disadvantages of the applied algorithm and future scope of improvements. The Vehicle Number Plate Detection algorithm can be successfully implemented on EVM320DM6437 hardware using Code Composer Studio and C language for real time application. Performance analysis is done by implementing various optimization techniques described later.

- **RESULTS**

The algorithm was tested using different license plates having various background conditions, light condition and image quality. Some of the output results are shown below:

(a) (b)

(c)

(d)

(e)

(f)

(g) (h)

(i) (j)

Fig. (a)(c)(e)(g)(i) Images of actual cars (b)(d)(f)(h)(j) Images of detected license plates.

The table below shows the type and number of plates and success ratio for detection of plates.

License Plate Conditions	Success using MATLAB Implementation (%)	Success using Hardware Implementation(%) (THEROTICAL)
Sunlight	100	94
Cloudy weather	100	92
Shade	100	94
Different Backgrounds	100	95

Table. Performance of License Plate Detection under Various Conditions

The MATLAB implementation helped to verify the algorithm very efficiently. Due to certain limitations of hardware like input image from camera and processing overhead on kit, the success ratio for the algorithm on hardware is slightly less compared to the implementation in MATLAB.

APPENDIX

- **MATLAB Simulation Code**

Number Plate Detection code:

```
close all;
clear all;

im = imread('Number Plate Images/image5.png');
imgray = rgb2gray(im);
imbin = imbinarize(imgray);
im = edge(imgray, 'prewitt');

%Below steps are to find location of number plate
Iprops=regionprops(im,'BoundingBox','Area', 'Image');
area = Iprops.Area;
count = numel(Iprops);
maxa= area;
boundingBox = Iprops.BoundingBox;
for i=1:count
   if maxa<Iprops(i).Area
       maxa=Iprops(i).Area;
       boundingBox=Iprops(i).BoundingBox;
   end
end

im = imcrop(imbin, boundingBox);%crop the number plate area
im = bwareaopen(~im, 500); %remove some object if it width is too long or too
small than 500

 [h, w] = size(im);%get width

imshow(im);

Iprops=regionprops(im,'BoundingBox','Area', 'Image'); %read letter
count = numel(Iprops);
noPlate=[]; % Initializing the variable of number plate string.

for i=1:count
   ow = length(Iprops(i).Image(1,:));
   oh = length(Iprops(i).Image(:,1));
   if ow<(h/2) & oh>(h/3)
       letter=Letter_detection(Iprops(i).Image); % Reading the letter
corresponding the binary image 'N'.
       noPlate=[noPlate letter] % Appending every subsequent character in
noPlate variable.
   end
end
```

Letter Detection code:

```
function letter=readLetter(snap)

load NewTemplates
snap=imresize(snap,[42 24]);
rec=[ ];

for n=1:length(NewTemplates)
    cor=corr2(NewTemplates{1,n},snap);
    rec=[rec cor];
end

ind=find(rec==max(rec));
display(ind);

% Alphabets listings.
if ind==1 || ind==2
    letter='A';
elseif ind==3 || ind==4
    letter='B';
elseif ind==5
    letter='C';
elseif ind==6 || ind==7
    letter='D';
elseif ind==8
    letter='E';
elseif ind==9
    letter='F';
elseif ind==10
    letter='G';
elseif ind==11
    letter='H';
elseif ind==12
    letter='I';
elseif ind==13
    letter='J';
elseif ind==14
    letter='K';
elseif ind==15
    letter='L';
elseif ind==16
    letter='M';
elseif ind==17
    letter='N';
elseif ind==18 || ind==19
    letter='O';
elseif ind==20 || ind==21
    letter='P';
elseif ind==22 || ind==23
    letter='Q';
elseif ind==24 || ind==25
    letter='R';
elseif ind==26
    letter='S';
```

```
elseif ind==27
    letter='T';
elseif ind==28
    letter='U';
elseif ind==29
    letter='V';
elseif ind==30
    letter='W';
elseif ind==31
    letter='X';
elseif ind==32
    letter='Y';
elseif ind==33
    letter='Z';
    %*-*-*-*-*
% Numerals listings.
elseif ind==34
    letter='1';
elseif ind==35
    letter='2';
elseif ind==36
    letter='3';
elseif ind==37 || ind==38
    letter='4';
elseif ind==39
    letter='5';
elseif ind==40 || ind==41 || ind==42
    letter='6';
elseif ind==43
    letter='7';
elseif ind==44 || ind==45
    letter='8';
elseif ind==46 || ind==47 || ind==48
    letter='9';
else
    letter='0';
end
end
```

Template Creation code:

```
%CREATE TEMPLATES
%Alphabets
A=imread('alpha/A.bmp');B=imread('alpha/B.bmp');C=imread('alpha/C.bmp');
D=imread('alpha/D.bmp');E=imread('alpha/E.bmp');F=imread('alpha/F.bmp');
G=imread('alpha/G.bmp');H=imread('alpha/H.bmp');I=imread('alpha/I.bmp');
J=imread('alpha/J.bmp');K=imread('alpha/K.bmp');L=imread('alpha/L.bmp');
M=imread('alpha/M.bmp');N=imread('alpha/N.bmp');O=imread('alpha/O.bmp');
P=imread('alpha/P.bmp');Q=imread('alpha/Q.bmp');R=imread('alpha/R.bmp');
S=imread('alpha/S.bmp');T=imread('alpha/T.bmp');U=imread('alpha/U.bmp');
V=imread('alpha/V.bmp');W=imread('alpha/W.bmp');X=imread('alpha/X.bmp');
Y=imread('alpha/Y.bmp');Z=imread('alpha/Z.bmp');

%Natural Numbers
one=imread('alpha/1.bmp');two=imread('alpha/2.bmp');
three=imread('alpha/3.bmp');four=imread('alpha/4.bmp');
five=imread('alpha/5.bmp'); six=imread('alpha/6.bmp');
seven=imread('alpha/7.bmp');eight=imread('alpha/8.bmp');
nine=imread('alpha/9.bmp'); zero=imread('alpha/0.bmp');

%Creating Array for Alphabets
letter=[A B C D E F G H I J K L M N O P Q R S T U V W X Y Z];
%Creating Array for Numbers
number=[one two three four five six seven eight nine zero];

NewTemplates=[letter number];
save ('NewTemplates','NewTemplates')
clear all
```

CONCLUSION

- **DISCUSSION**

In this paper, the license plate location is the first step in this system of license plate recognition. Therefore, this step play an extremely role and it can affect the accuracy of following steps directly. In this step, the quality of the edge detection of plate affects the process of plate location in the morphological algorithm. Then, the binarization is of great importance in cutting the character out from the license plate. If the above operations are well done, the features of license plate will be clear in order to improve accuracy of the character matching. On the other hand, if the plates are in the different angle to the observer, it is necessary to adjust the angle in order to facilitate the characters matching. The Rando algorithm is used in this step. The advantage of Rando is too fast and accurate on the field seeking for angle compared with the horizontal edge. Matching template is the last step in this programming. We prepared a lot of relevant templates including the test characters and test numbers. In this step, characters are matched with templates by the black points‟ calculation algorithm. The advantage is that is can improve the accuracy of matching. Oppositely, the disadvantage of this method is that it may lengthen the time of matching.

- **Conclusion and Further Scope**

To sum up, this paper describes a technique which recognizes the license plates by use of image processing. The application of license plate system not only can distinguish Europe Union license plates from Swedish license plates, but also can recognize the characters on license plates. Therefore, the aim of this system is fully accomplished and its application may inspire a series of research questions (see Chapter 1.4). Typically, this system can also be applied to collect the information of vehicles on road, since it can recognize the character and output the license plate number automatically.

Obviously, the accuracy of the recognition is the most important in this system. Therefore, this application should be optimized and modified for overcoming the accuracy limitations. In order to make the recognition more precise, we should add some pre-processes to remove the interferences. Moreover, we would continue the further study for license plate recognition in some complicated environments, such as vehicles at dark night or in heavy rains and so on. If we could accomplish all of the objectives, this application would have a very promising future.

- The Automatic Vehicle Number Plate Detection system plays an important role in detecting security threats.
- The system uses series of image processing techniques for identifying the vehicle from database stored in PC.
- The system is implemented in MATLAB and its performance is tested on real images.
- The system robustness and speed can be increased if high resolution camera is used.

REFERENCES

[1] R. Radha and C.P.Sumathi, “A Novel approach to extract text from license plate of vehicle”, Signal & Image Processing.

[2] Shen Zheng Wang & His-Jian Lee “Detection and Recognition of License Plate Characters with Different Appearances

[3] T. Pratheeba, “Morphology Based Text Detection and Extraction from Complex Video Scene.

[4] **Ms. Shilpi Chauhan** and **Vishal Srivastava**

Research Scholar, Dept. of CSE, Arya Institute of Engineering and IT, Jaipur, India. Associate Professor, Dept. of CSE, Arya College of Eng. and IT, Jaipur, India.


THANK
YOU