International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 07 | July 2021 www.irjet.net p-ISSN: 2395-0072

Vehicle Number Plate Recognition Using MATLAB

Utsav Dedhia[1]

[1]UG Student, NMIMS University, Maharashtra, India

Abstract – *The project is about recognizing the number plate of a vehicle when it is stationary or from the vehicle's image. The aim is to use MATLAB software to recognize individual characters from the number plate which can further be compared from the license database to identify the owner of the vehicle. However, the project is limited to image processing and character recognition of the number plate.*

Key Words: MATLAB, Number Plate, Recognition, Vehicle

1.INTRODUCTION

License plates are used for the identification of vehicles and their owners in every country. The system to identify the number plate is either manual (operator-controlled) or automatic. Systems for recognizing and then identifying the license plates are utilized in traffic control and for security purposes such as control of access to restricted areas and vehicle tracking. Each country has a different system for recognizing number plates as per their need. For example, Indian license plates are difficult to recognize as there is no standard law followed for the aspect ratio and characters of the license plate. The identification task, in general, is challenging because of various reasons such as broken number plate, color, poor maintenance, written in a different language, etcetera. Number plate detection method scans the input image to identify and locate the license plate in the image. Since a plate can exist anywhere in an image with various sizes, it is not possible to check every pixel of the image to locate it. The number plate of the vehicle is automatically recognized and identified by comparing it with a database when it enters a gate, or a toll stop. It is difficult to detect the boundary of the Number plate from the image of the car in an outdoor scene due to the color of characters and background on the license plate. Some algorithms are based on a combination of morphological operation, segmentation, and Canny edge detector. The process of locating the number plate consists of steps like Edge Detection, Morphological operations, noise filtration, character segmentation, and template matching.

2. OBJECTIVE AND APPLICATION

2.1 Objective

At the end of the project, the characters from the number plate should be recognized and displayed. The recognized characters should be correct, and the goal is to keep the process as simple as possible.

2.2 Application

- *Parking*: To permit parking whose license plate is recognize
- *Community Safety*: Only registered number plates vehicles are allowed inside
- *Toll Stops:* Prevents theft and fraud
- *Border Security*: The vehicle gets registered when leaving or entering a country
- *Traffic Control*: Traffic is controlled more efficiently

3. METHODOLOGY

3.1 Introduction

MATLAB software is used for image processing and character recognition. A step-by-step process is followed to get the characters of the number plate recognized and displayed. Processes for image enhancement will be done before character segmentation and recognition. Fig. 1 shows how the process steps will be executed.

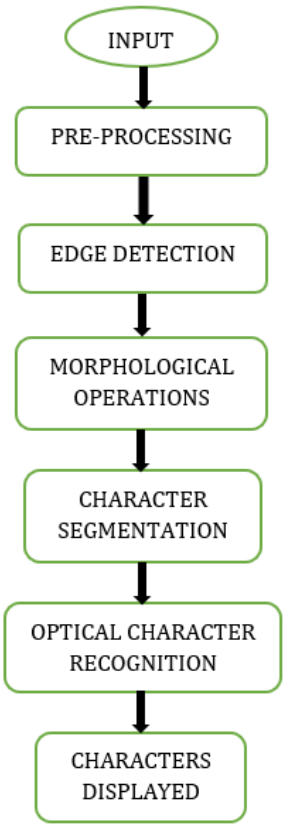

Fig.1 Flowchart of the process

3.2 Image Input

The image of the number plate used as an example for this project is taken from the internet. First, we must write (input) the image in MATLAB and then display it. The fig.2 shows the code and the result.

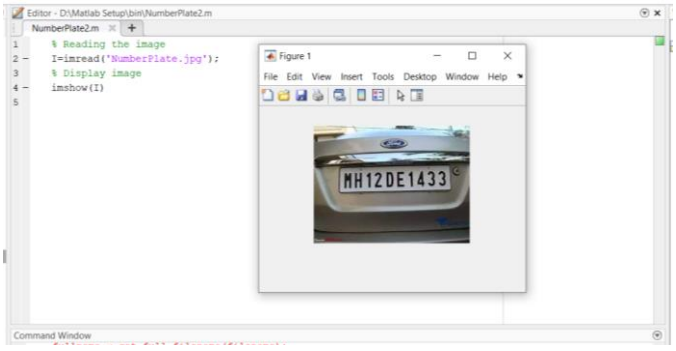

Fig.2 Code for reading and displaying the image

3.3 Pre-Processing

In Pre-processing, we use various methods to improve the image. Filter out noise, improve contrast, convert the image to a more computable form.

3.2.1 *Converting from RGB to Grayscale:* The grayscale conversion helps us to reduce the luminance effect and enhance the features of the image segregating the darker values of the image more distinct. Fig.3 shows the code and the converted grayscale image.

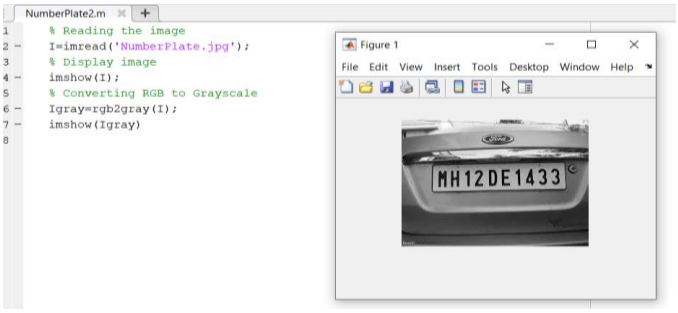

Fig.3 Grayscale image and code

3.2.2 *Converting Grayscale to Binary:* The image, which was converted into black and white, will now be converted to a binary image. Converting to Binary is often used for distinguishing objects in the image from the background as well as from the other objects in it. I have used *the imbinarize* method which uses thresholding to convert images into binary form by replacing all values above a globally determined threshold with 1s and setting all other values to 0s. Fig.4 shows the code and the binary image.

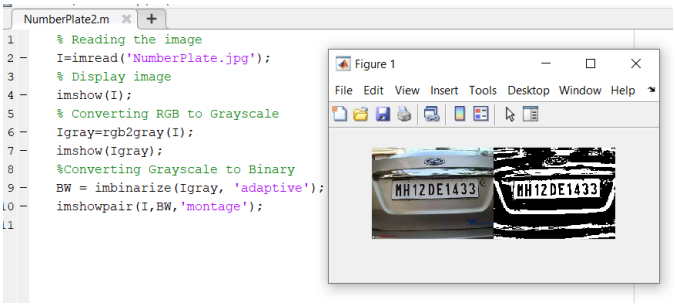

Fig.4 Image converted to binary image

3.2.3 *Edge Detection:* The Edge Detection method reduces considerable amount of data from the image and conserves the properties of the image for further processing steps. To detect and locate sharp discontinuities, I have used Edge Detection method. The *Canny* operator was decided to be an optimal edge detector. The code and resulted image after applying the *Canny* operator is shown in fig.5.

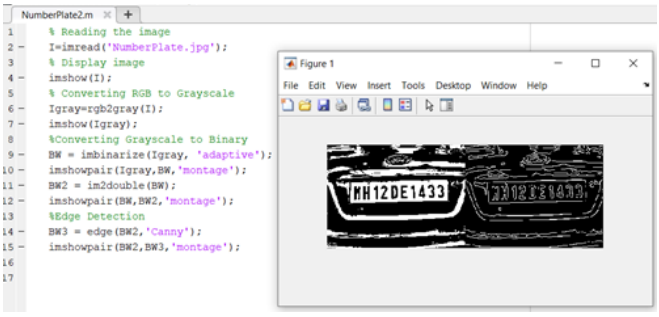

Fig.5 The detected edges

3.2.4 *Morphological Operations:* Morphology operations are used to process an image based on the shapes. Erosion and Dilation are two of the basic morphological operations. Dilation is performed by adding pixels to the boundaries of objects for all the pixels in the input pixel's neighborhood. Dilation is used to increase the thickness of the number plate edges. So, we can find the characters easily. Fig.6 shows the thickened border.

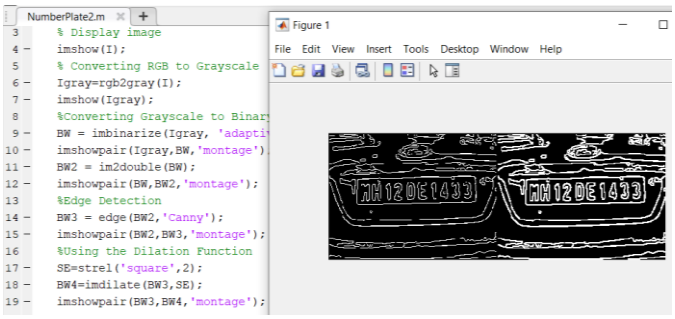

Fig.6 Dilated image on the left

3.2.5 *Character Segmentation:* After the dilation step, the next step is to segment all the characters, without losing the features of the characters. One of the most important process to recognize the characters is the Segmentation process. There is function in the MATLAB toolbox called *regionprops* (). The set of properties of each labeled region are measured in the label matrix by this function and the properties of the image region are measured by the bounding box. This technique is used for checking the numbers with a template used by the template matching algorithm in Optical Character Recognition (OCR).

3.2.6 *Letter Detection:* I have created a function to get the alphanumeric output from the input image, then load the template we have saved. It compares the input image with every image in the template to get the match. The code is shown in fig.7.

```
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
```

Fig. 7 Letter Detection Code

3.2.7 *Template Matching:* The processed image of the number plate is now used to compare each character against the complete alphanumeric database using template matching. The matching process moves the template image to all possible positions in a larger source image and computes a numerical index that indicates how well the template matches the image in that position. Matching is done on a pixel-by-pixel basis. The size of the template is 42 x 42. The recognition is accurate as the template size is fixed. Fig.8 shows the code for the template file.

```
%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
```

Fig.8 Template Code

3.2.8 *Code for Plate Detection*: A comprehensive code was written to locate and crop the number plate region for faster and better image processing. Fig.9 below shows the code for number plate detection.

```
im = imread('Number Plate Images/image4.jpg');
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 th

 [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 th
        noPlate=[noPlate letter] % Appending every subsequent character in noPlate vari
    end
end
```

Fig.9 Number Plate Detection Code

4. EXPERIMENTATION AND RESULTS

Different number plate images were used, and results were analyzed. The results from different images helped me to understand the limitations of the program. In fig.10, it shows two different results from two different images. The program was not able to detect the middle image number plate. This is due to the automatic cropping of the image; this problem can be resolved simply by keeping the cropping step manual.

Fig.10 Three different images with different results

5. LIMITATIONS

- A broken number plate cannot be detected
- Number plates in different languages or contain different symbols cannot be detected if not in the database
- Poor maintenance of number plates can lead to similarities between characters. For example, 5 & 8, 0 & D
- Low resolution or blurry number plate

6. CONCLUSION

A vehicle number plate recognition system can be very helpful for tracking or identifying a vehicle by recognizing its number plate. The system works fine for a fixed image style. It has a huge potential for improvement and making it precise.

REFERENCES

[1] Recognition Of Vehicle Number Plate Using MATLAB https://www.irjet.net/archives/V3/i5/IRJET-V3I5177.pdf

[2] MATLAB Based Vehicle Number Plate Recognition https://www.ripublication.com/ijcir17/ijcirv13n9_10.pdf

[3] Ragini Bhat, Bijender Mehandia, "Recognition of Vehicle Number Plate using MATLAB", IJIREEICE, Vol. 2, Issue 8, August 2014

[4] Automatic Vehicle Number Plate Recognition System using MATLAB, IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) e-ISSN: 2278-2834, p- ISSN: 2278-8735.Volume 11, Issue 4, Ver. II (Jul.-Aug .2016)

[5] Number plate recognition using template comparison for various fonts in MATLAB, DOI: 10.1109/ICCIC.2016.7919542

[6] MATLAB Based Vehicle Number Plate Recognition, International Journal of Computational Intelligence Research ISSN 0973-1873 Volume 13, Number 9 (2017)