A Project Report on

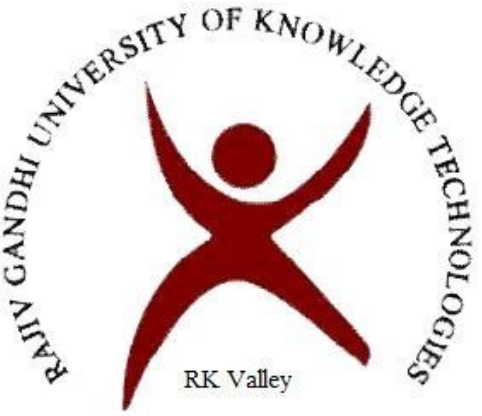
AUTOMATIC NUMBER PLATE RECOGNITION

Submitted by

Galla Varalakshmi R161609

Bolisetty Navaneetha R161610

Submitted to



Under the supervision of

Mr. G.Siva Rama Sastry

Assistant Professor

Department of Computer Science and Engineering

As a part of

Partial fulfilment of the degree of Bachelor of Technology in Computer Science and Engineering.

Date: 24-05-2021

CERTIFICATE

This is to certify that the report submitted by **Galla Varalakshmi and Bolisetty Navaneetha** in partial fulfilment of the requirements for the award of Technology in Computer Science and Engineering is a bonafide work carried out by her under my supervision and guidance.

Mr. G.Siva Rama Sastry,
Project Guide,
Assistant Professor,
Computer Science and Engineering,
RGUKT, R.K Valley.

Ms. Shabana,
Project Coordinator,
Assistant Professor,
Computer Science and Engineering,
RGUKT, R.K Valley.

Mr. T.Sandeep Kumar Reddy
Head of the Department,
Assistant Professor,
ComputerScienceandEngineering,
RGUKT, R.K Valley.

Declaration

I Galla Varalakshmi(R161609) and Bolisetty Navaneetha(R161610) hereby declare that this report submitted by me under the guidance and supervision of Mr. G.Siva Rama Sastry is a bonafide work.

I also declare that it has not been submitted previously in part or in full to this university or other university or institution for the award of any degree or diploma.

Date:24-05-2021,
Place: RK Valley.

Galla Varalakshmi(R161609),
Bolisetty Navaneetha(R161610).

TABLE OF CONTENT

`

ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistancefrom many people and I am extremely privileged to have got this all along the completion of my project. All that I have done is only due to such supervision and assistance and we would not forget to thank them.

We would like to express my sincere gratitude to Mr.G.Siva Rama Sastry for valuable suggestions and keen interest throughout the progress of my course of research.

We are grateful to Ms. Shabana for encouragement and more over timely support and gudiance.

We are very much thankful to our senior Mr.Tejdeep for encouragement and guidance till the completion of our project work.

At the outset, I would like to thank Rajiv Gandhi University of Knowledge and Technologies, R.K Valley for providing all the necessary resources forthe successful completion of my course work.

With Sincere Regards,
Varalakshmi Galla,
Navaneetha Bolisetty.

AUTOMATIC NUMBER PLATE RECOGNITION SYSTEM (ANPR)

Abstract

Traffic control and vehicle owner identification has become major problem in every country. Sometimes it becomes difficult to identify vehicle owner who violates traffic rules and drives too fast. Therefore, it is not possible to catch and punish those kinds of people because the traffic personal might not be able to retrieve vehicle number from the moving vehicle because of the speed of the vehicle. Therefore, there is a need to develop Automatic Number Plate Recognition (ANPR) system as a one of the solutions to this problem.
Automatic Number Plate Recognition system i.e.ANPR system is an image processing technology. In this technology we use license plate of vehicle to recognize the vehicle. Automatic Number Plate Recognition system is used in various areas such as automatic toll collection, Border crossings, parking system, Traffic control, stolen cars tracking, maintaining traffic activities and law enforcement etc.

Keywords

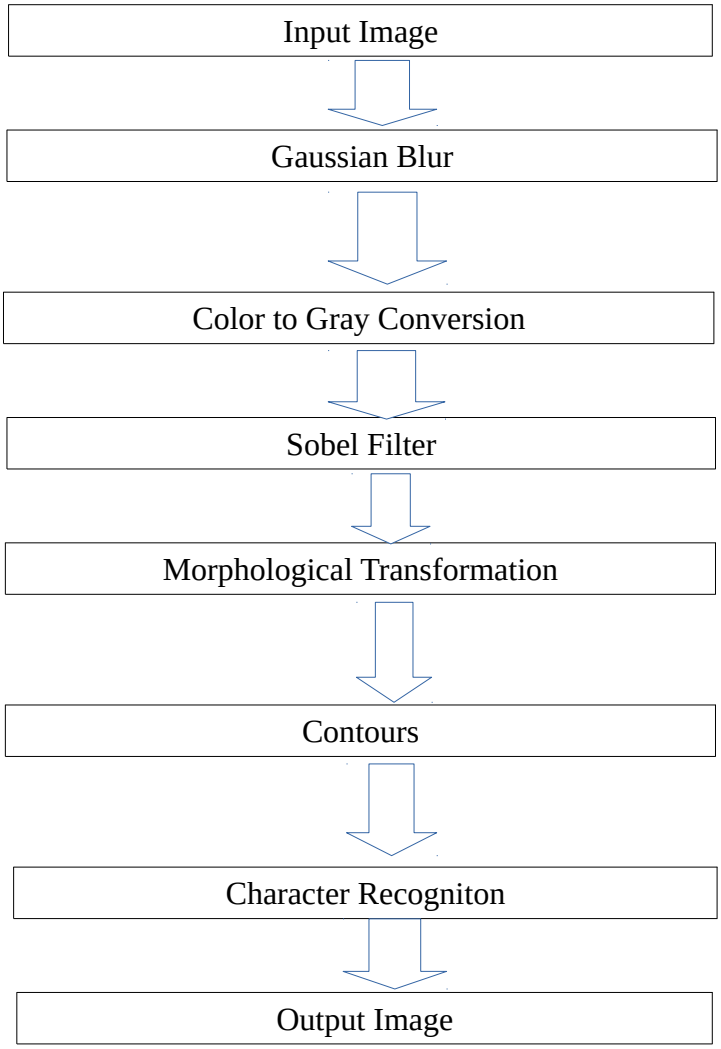
Automatic Number Plate Recognition(ANPR), Optical character Recogniton(OCR),

INTRODUCTION

With rising traffic on roads and number of vehicles on roads, it is getting very difficult to manually handle laws, traffic rules and regulations for smooth traffic moment. Toll-booths are installed on freeways and parking complex, where the vehicle has to stop to pay the toll or parking charge fees. Also, Traffic maintenance systems are constructed on freeways to analyze for vehicles moving at speeds not allowed by law. All these operations have a scope of improve development. In the center of all these processes lies a vehicle. The vital question here is how to find a particular vehicle? The obvious response to this question is by using the vehicle's number plate. This number differentiate one vehicle from the other, which is effective especially when both are of same type of make and model. An automated system can be developed to find the license plate of a vehicle and recognize the characters from the region having a license plate. The vehicle license plate number which can be utilized to fetch farther information data about the vehicle and its owner, which can be used for further processing.
In last few years, ANPR or license plate recognition (LPR) has been one of the useful approaches for vehicle surveillance. It is can be applied at number of public places for fulfilling some of the purposes like traffic safety enforcement, automatic toll text collection [1], car park system [2] and Automatic vehicle parking system [3].

We have implemented algorithm for Automatic Number Plate Recognition System using python. This algorithm initially used different inbuilt functions and implemented some user defined techniques related to image processing. Once the algorithm was implemented, it was checked with multiple input images having vehicle number plates. The input vehicle images consist of number plates. A flowchart shown below is the basic implementation algorithm which is shown in figure number 1.An OCR is used to recognize each character.

IMPLEMENTATION

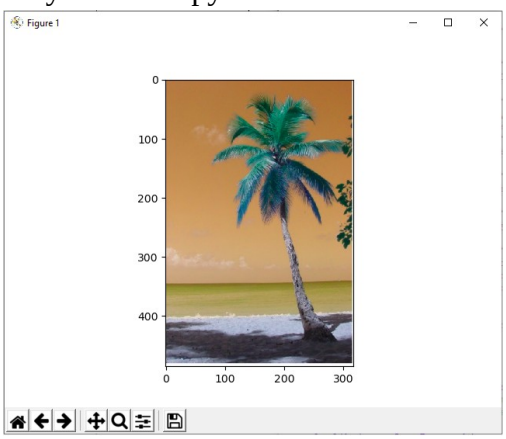


Ch.1 Implementation process

a.Matplot

matplotlib is a User friendly, but powerful, plotting library for python. I is commonly used with OpenCv images. ... **pyplot** is a module in matplotlib. matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats.

```
import cv2
import matplotlib.pyplot as plt
image= cv2.imread('Tropical-tree.jpg')
plt.imshow(image)
plt.show()
```



b.Input Images

Here, we are providing the image dataset to the algorithm for giving those images to algorithm, we are using **glob l**library of python.

```
path=glob.glob("/content/drive/MyDrive/data/data/*.jpg")
```

1.Gaussian Blur

Image smoothing is a technique which helps in reducing the noise in the images. Image may contain various type of noise because of camera sensor. It basically eliminates the high frequency (noise, edge) content from the image so edges are slightly blurred in this operation. OpenCV provide gaussianblur() function to apply smoothing on the images.

It is a widely used effect in graphics software, typically to reduce image noise and reduce detail. The visual effect of this blurring technique is a smooth blur resembling that of viewing the image through a translucent screen, distinctly different from the bokeh effect produced by an out-of-focus lens or the shadow of an object under usual illumination.

```
img2 = cv2.GaussianBlur(img, (3,3), 0)
```

Ch2. Original image

Ch3. Blurred Image

2.Conversion of color image to Gray scale Image

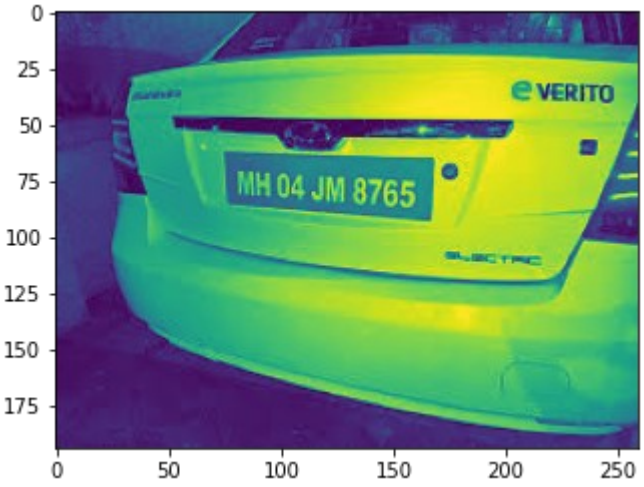
The above algorithm shown is independent of the type of colors(RGB) in image and depends mainly on the gray scale of an image for processing and extracting the essential data. Color components like Red, Green and Blue value are not used in this algorithm. Thus, if the input image is a colored i.e. RGB image represented by three-dimensional array , it is converted to a two-dimensional gray scale image before further processing. The example of original color i.e. RGB input image and converted gray scale image is shown below. So first step is conversion of color to gray is implemented.

An RGB image can be viewed as three images(a red scale image, a green scale image and a blue scale image) stacked on top of each other. RGB image is basically a M*N*3 array of colour pixel, where each colour pixel is a triplet which corresponds to red, blue and green colour component of RGB image at a specified spatial location.

Similarly, A Grayscale image can be viewed as a single layered image. In MATLAB, a grayscale image is basically M*N array whose values have been scaled to represent intensities.

```
img2 = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)
```

Ch .4 Colored Image



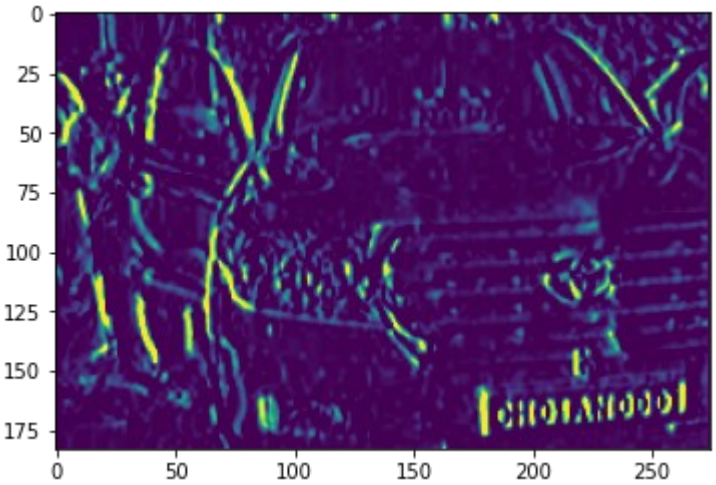
Ch.5 Gray Scale Image

3.Sobel Filter

It is a derivate mask and is used for edge detection. Like Prewitt operator sobel operator is also used to detect two kinds of edges in an image.
The **Sobel filter** is used for **edge detection**. It works by calculating the gradient of image intensity at each pixel within the image. ... The result shows how abruptly or smoothly the image changes at each pixel, and therefore how likely it is that that pixel represents an edge.

```
img2 = cv2.Sobel(img2,cv2.CV_8U,1,0,ksize=3)
```

Ch.6 Original image



Ch.7 Edged Image

4.Morphological Transformation

Morphological transformations are some simple operations based on the image shape. It is normally performed on binary images. It needs two inputs, one is our original image, second one is called structuring element or kernel which decides the nature of operation. Two basic morphological operators are Erosion and Dilation. Then its variant forms like Opening, Closing, Gradient etc also comes into play. We will see them one-by-one with help of following image:

```
element = cv2.getStructuringElement(shape=cv2.MORPH_RECT, ksize=(17, 3))
```

4.1 Erosion

The basic idea of erosion is just like soil erosion only, it erodes away the boundaries of foreground object (Always try to keep foreground in white).

4.2 Dilation

It is just opposite of erosion. Here, a pixel element is '1' if atleast one pixel under the kernel is '1'. So it increases the white region in the image or size of foreground object increases. Normally, in cases like noise removal, erosion is followed by dilation. Because, erosion removes white noises, but it also shrinks our object. So we dilate it. Since noise is gone, they won't come back, but our object area increases. It is also useful in joining broken parts of an object.

4.3 Opening

Opening is just another name of **erosion followed by dilation**. It is useful in removing noise, as we explained above. Here we use the function, **cv.morphologyEx()**

4.4 Closing

Closing is reverse of Opening, **Dilation followed by Erosion**. It is useful in closing small holes inside the foreground objects, or small black points on the object.

```
cv2.morphologyEx(src=img2, op=cv2.MORPH_CLOSE, kernel=element,
dst=morph_img_threshold)
```

4.5 Thresholding

Here, the matter is straight-forward. For every pixel, the same threshold value is applied. If the pixel value is smaller than the threshold, it is set to 0, otherwise it is set to a maximum value. The function **cv.threshold** is used to apply the thresholding. The first argument is the source image, which **should be a grayscale image**. The second argument is the threshold value which is used to classify the pixel values. The third argument is the maximum value which is assigned to pixel values exceeding the threshold. OpenCV provides different types of thresholding which is given by the fourth parameter of the function. Basic thresholding as described above is done by using the type cv.THRESH_BINARY.

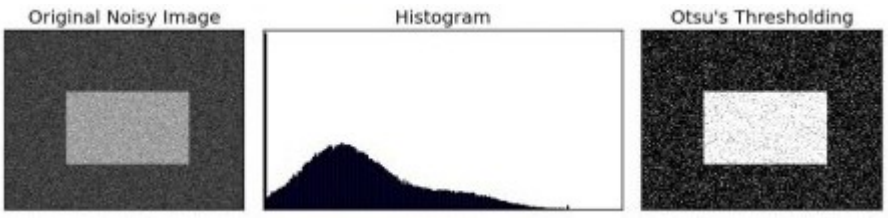
4.5.1 Otsu's Binarization

In global thresholding, we used an arbitrary chosen value as a threshold. In contrast, Otsu's method avoids having to choose a value and determines it automatically.

Consider an image with only two distinct image values (*bimodal image*), where the histogram would only consist of two peaks. A good threshold would be in the middle of those two values. Similarly, Otsu's method determines an optimal global threshold value from the image histogram.

In order to do so, the cv.threshold() function is used, where cv.THRESH_OTSU is passed as an extra flag. The threshold value can be chosen arbitrary. The algorithm then finds the optimal threshold value which is returned as the first output.

```
_,img2 = cv2.threshold(img2,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
```



5.Contours

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition.

5.1 Draw contours

To draw the contours, cv.drawContours function is used. It can also be used to draw any shape provided you have its boundary points. Its first argument is source image, second argument is the contours which should be passed as a Python list, third argument is index of contours (useful when

drawing individual contour. To draw all contours, pass -1) and remaining arguments are color, thickness etc.

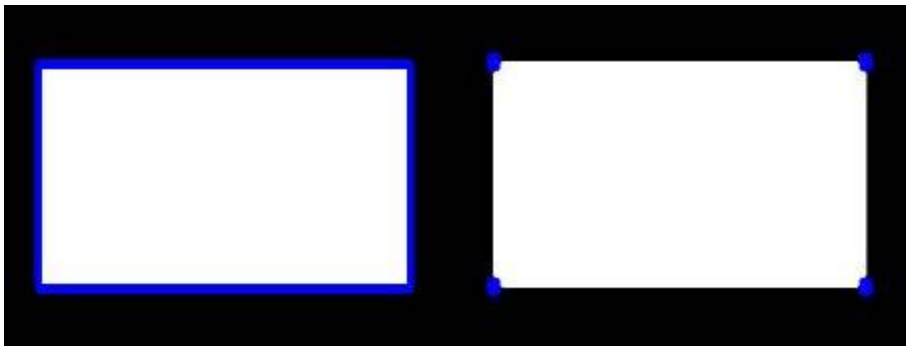
5.2 Contour Approximation Method

Above, we told that contours are the boundaries of a shape with same intensity. It stores the (x,y) coordinates of the boundary of a shape. But does it store all the coordinates ? That is specified by this contour approximation method.

If you pass cv.CHAIN_APPROX_NONE, all the boundary points are stored. But actually do we need all the points? For eg, you found the contour of a straight line. Do you need all the points on the line to represent that line? No, we need just two end points of that line. This is what cv.CHAIN_APPROX_SIMPLE does. It removes all redundant points and compresses the contour, thereby saving memory.

Below image of a rectangle demonstrate this technique. Just draw a circle on all the coordinates in the contour array (drawn in blue color). First image shows points I got with cv.CHAIN_APPROX_NONE (734 points) and second image shows the one with cv.CHAIN_APPROX_SIMPLE (only 4 points). See, how much memory it saves!!!

```
num_contours, hierarchy=
cv2.findContours(morph_img_threshold,mode=cv2.RETR_EXTERNAL,method=cv2.CHAIN
_APPROX_NONE)
cv2.drawContours(img2, num_contours, -1, (0,255,0), 1)
```

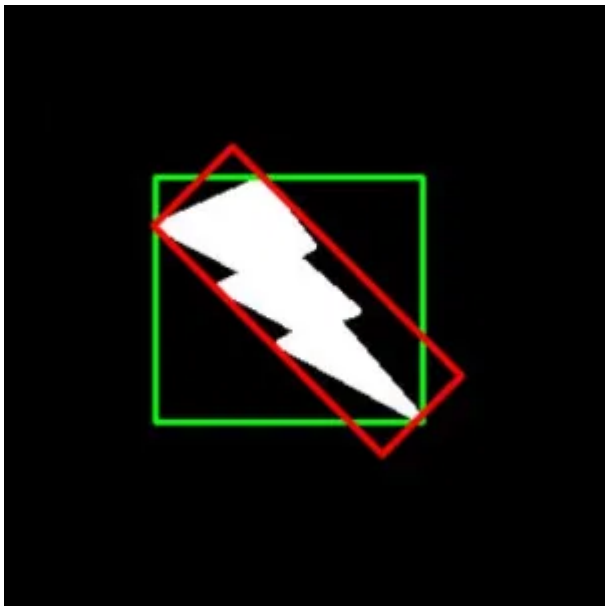


5.3 OpenCV Minimum Area Rectangle

Draw a minimum area rectangle around the region of interest. This is one of the most common tasks performed when working with contour.

The bounding rectangle is drawn with a minimum area. Because of this, rotation is also considered. The below image shows 2 rectangles, the green one is the normal bounding rectangle while the red one is the minimum area rectangle. See how the red rectangle is rotated.

```
min_rect = cv2.minAreaRect(cnt)
```



For each contour, we draw a rectangle with minimum area around it by using cv.minAreaRect(cnt) function in opencv.it returns a origin point,width,height and angle of the rectangle. By observing those parameters,we will calculate the area compare those things with some standard values with for implementing we have write functions ratio_and_rotation(min_rect) if this function returns true then we can conclude that the selected contour is almost the licence plate.

5.4 Cropping of licence plate

Now,we need to crop the number plate by using contour,which satisfies the function ratio_and_rotation().

Here we use cv.boundingRect(cnt) for getting the position of licence plate,width and height

```
x,y,w,h = cv2.boundingRect(cnt)
plate_img = img[y:y+h,x:x+w]
```

here, number plate is present in plate_img.

6.Character Recognition

EasyOCR, as the name suggests, is a Python package that allows computer vision developers to effortlessly perform Optical Character Recognition.

EasyOCR is implemented using Python and the PyTorch library. If you have a CUDA-capable GPU, the underlying PyTorch deep learning library can speed up your text detection and OCR speed *tremendously.*

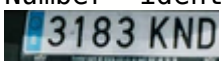
Next, we need to tell EasyOCR which language we want to read. EasyOCR can read multiple languages at the same time but they have to be compatible with each other. English is compatible with all languages. Languages that share most of character (e.g. latin script) with each other are compatible.

EasyOCR will then check if you have necessary model files and download them automatically. It will then load model into memory which can take a few seconds depending on your hardware. After it is done, you can read as many images as you want without running this line again.

```
reader=easyocr.Reader(['en'])
result=reader.readtext(plate_im)
```

output:

```
Number  identified number plate...
```




```
CUDA not available - defaulting to CPU. Note: This module is much faster with a
GPU.

Number  Detected Plate Text :  [([[2, 0], [98, 0], [98, 24], [2, 24]], '3183
KND', 0.4561689326064402)]
```

7.Output Images

Number input image...

Number identified number plate...

CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.
Downloading detection model, please wait. This may take several minutes depending upon your network connection.

Downloading recognition model, please wait. This may take several minutes depending upon your network connection.

Number Detected Plate Text : [([[46, 4], [168, 4], [168, 30], [46, 30]], 'LA 03 MG . 2784', 0.1524223205104773)]
Number input image...

Number identified number plate...

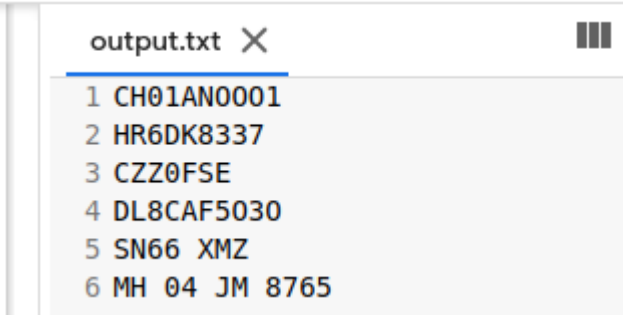
CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.

```
Number  Detected Plate Text :  [([[0.36153615896191904, 7.853075311273344],
[86.91759622541677, -0.568176659382748], [88.63846384103807,
17.146924688726656], [2.0824037745832253, 24.568176659382747]], 'CHO1ANOOOO1',
0.244699481425359)]
```

8.Licence numbers in file

here ,we are saving the numbers of number plate in the file.

```
with open('/content/drive/MyDrive/data/output.txt', 'a') as writefile:
writefile.write(result[0][1])
writefile.write("\n")
```



Conclusion

In this paper, the Automatic Number Plate Recognition system using vehicle license plate is presented. This system use image processing techniques for recognition of the vehicle from the database stored in the computer. The system works satisfactorily for wide variation of conditions and different types of number plates. The system is implemented and executed in Matlab and performance is tested on genuine images. This ANPR system works quite well however, there is still room for improvement. This ANPR system speed can beincrease with high resolution camera. Which can be able to capture clear images of the vehicle. The OCR method is sensitive to misalignment and to different sizes, so we have to create different kind of templets for different RTO specifications. The statistical analysis can also be used to define the probability of detection and recognition of the vehicle number plate. At present there are certain limits on parameters like speed of the vehicle, script on the vehicle number plate, skew in the image which can be removed by enhancing the algorithms further.

Reference

1.https://techvidvan.com/tutorials/python-project-license-number-plate-recognition/

2.https://medium.com/programming-fever/license-plate-recognition-using-opencv-python-7611f85cdd6c

3.https://www.youtube.com/watch?v=WQeoO7MI0Bs

4.https://docs.opencv.org/master/d4/d73/tutorial_py_contours_begin.html

5.https://www.ijert.org/research/automatic-number-plate-recognition-system-anpr-system-IJERTV3IS071132.pdf