MINIPROJECT REPORT ON AUTOMATIC NUMBER PLATE RECONIGTION

In partial fulfillment of requirements for the degree of Bachelor of Technology Electronics and Communication Engineering

SUBMITTED BY: DEVESH SHARMA

(ROLL NO:1809131064)

Under the Guidance of

Mr. Anuranjan Kansal

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

JSS ACADEMY OF TECHNICAL EDUCATION
C-20/1 SECTOR-62, NOIDA
2020 - 2021

CERTIFICATE

Certified that mini project entitled “**Automatic Number Plate Recognition**” is a bonafide work carried out in the fifth semester by “Devesh Sharma” in partial fulfillment for the award of Bachelor of Technology in Electronics and Communication Engineering from JSS Academy of Technical Education, Noida during the academic year 2020- 2021.

SIGNATURE
Mr. Anuranjan Kansal

ACKNOWLEDGEMENT

With immense pleasure I, Devesh Sharma presenting Automatic Number Plate Recognition Seminar report as part of the curriculum of B. Tech, Third year (fifth semester).

I am extremely grateful to Prof. Gurulingappa M. Patil, Principal JSS Academy of Technical Education, Noida and Dr. Arun Kumar G., Head of Department, Department of Electronics & Communication Engineering, for providing all the required resources for the successful completion of my seminar. My heartfelt gratitude to my seminar guide Mr. Anuranjan Kansal, Assistant Professor, Department of Electronics & Communication Engineering, for his valuable suggestions and guidance in the preparation of the seminar report. I express my thanks to all staff members and friends for all the help and co-ordination extended in bringing out this seminar successfully.

TABLE OF CONTENTS

Chapter No.	Topic	PAGE No.
	Abstract	6
Chapter-1	Introduction	7
Chapter-2	Benefits	25
Chapter-3	Applications and Future prospects	26
	Conclusion	29
	References	30

LIST OF FIGURES

Figure **Page No.**

Fig 1.0: Factors to bring about a renaissance in computer vision………………………….………...7

Fig 1.1: Grayscale image is stored in numbers…………………………………………………8

Fig 1.2: How pixel stored in memory….…………………………………………………………..9

Fig 1.3: How to create color in rgb……………………………….…………………….....10

Fig 1.4: Steps involve in automatic number plate recognition……………………………..........12

Fig 1.5: Sample image….……………………………………………………………….....14

Fig 1.6: Sample image converted into grayscale….……………………………………………14

Fig 1.7: Applying the Gaussian blur to sample image.…………………………………………15

Fig 1.8: Applying canny edge detection on sample image.………………………………………..16

Fig 1.9:Number plate highlighted on sample image.……………………………………………18

Fig 1.10: Separated number plate image from the sample.……………….………….…….18

Fig 1.11: Number plate highlighted on sample image using trained model.……………….…....19

ABSTRACT

Computer is considered as one of the greatest invention of the mankind. Computer is responsible for changing the face of earth. Initially computer was designed to execute the complex mathematic the revolution in computer technology come after invention of transistors and microprocessors. Microprocessor enhance the computer speed and power to handle and execute the heavy task. Soon people realize that computer can be used for various task. now computer is every were we can’t imagine our world without computer. We are living in the era of Technology in this computer vison play important role. Computer vision is a field of artificial intelligence that trains computers to interpret and understand the visual world. Using digital images from cameras and videos and deep learning models, machines can accurately identify and classify objects and then react to what they “see”. 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. There are numerous ANPR systems available today. These systems are based on different methodologies but still it is really challenging task as some of the factors like high speed of vehicle, non-uniform vehicle number plate, language of vehicle number and different lighting conditions can affect a lot in the overall recognition rate.

Chapter 1

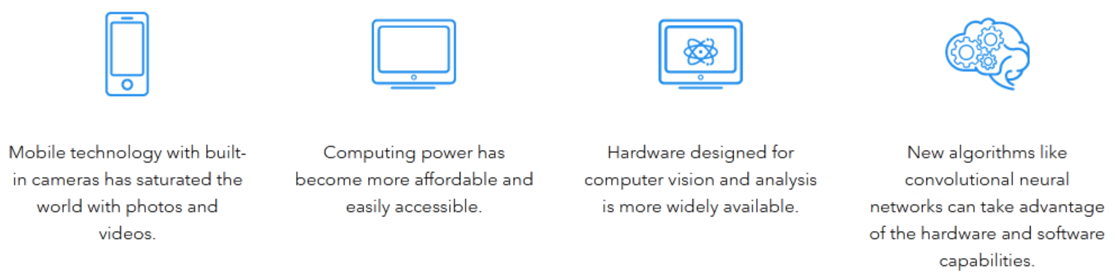
INTRODUCTION

1.1 BACKGROUND OF COMPUTER VISION

Early experiments in computer vision took place in the 1950s, using some of the first neural networks to detect the edges of an object and to sort simple objects into categories like circles and squares. In the 1970s, the first commercial use of computer vision interpreted typed or handwritten text using optical character recognition. This advancement was used to interpret written text for the blind.

As the internet matured in the 1990s, making large sets of images available online for analysis, facial recognition programs flourished. These growing data sets helped make it possible for machines to identify specific people in photos and videos.

Today, a number of factors have converged to bring about a renaissance in computer vision:



Mobile technology with built-in cameras has saturated the world with photos and videos.

Computing power has become more affordable and easily accessible.

Hardware designed for computer vision and analysis is more widely available.

New algorithms like convolutional neural networks can take advantage of the hardware and software capabilities.

Fig - 1.0

The effects of these advances on the computer vision field have been astounding. Accuracy rates for object identification and classification have gone from 50 percent to 99 percent in less than a decade and today's systems are more accurate than humans at quickly detecting and reacting to visual inputs.

1.2 WHAT IS COMPUTER VISION?

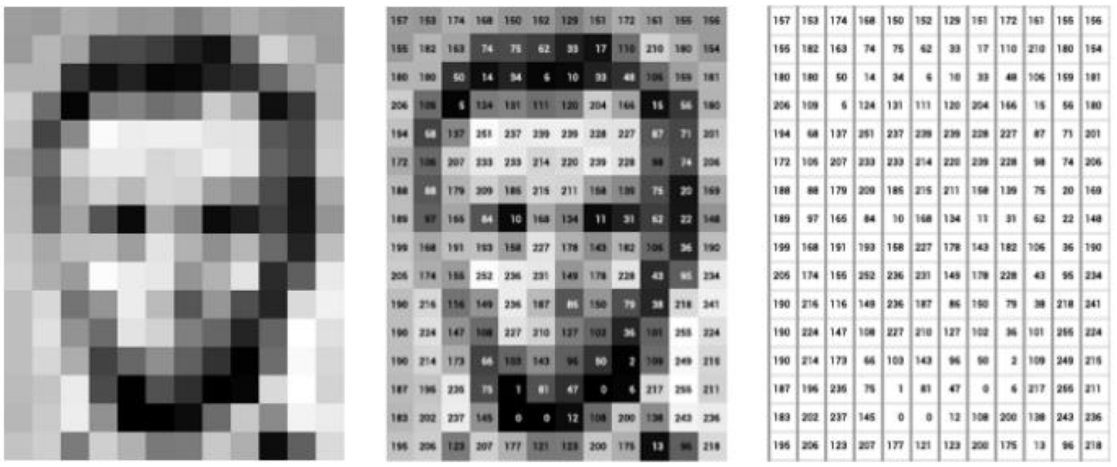
Computer vision is a field of artificial intelligence that trains computers to interpret and understand the visual world. Using digital images from cameras and videos and deep learning models, machines can accurately identify and classify objects — and then react to what they "see."

One of the major open questions in both Neuroscience and Machine Learning is: How exactly do our brains work, and how can we approximate that with our own algorithms? The reality is that there are very few working and comprehensive theories of brain computation; so despite the fact that Neural Nets are supposed to "mimic the way the brain works," nobody is quite sure if that's actually true.

The same paradox holds true for computer vision — since we're not decided on how the brain and eyes process images, it's difficult to say how well the algorithms used in production approximate our own internal mental processes.

On a certain level Computer vision is all about pattern recognition. So one way to train a computer how to understand visual data is to feed it images, lots of images thousands, millions if possible that have been labeled, and then subject those to various software techniques, or algorithms, that allow the computer to hunt down patterns in all the elements that relate to those labels.

Below is a simple illustration of the grayscale image buffer which stores our image of Abraham Lincoln. Each pixel's brightness is represented by a single 8-bit number, whose range is from 0 (black) to 255 (white):

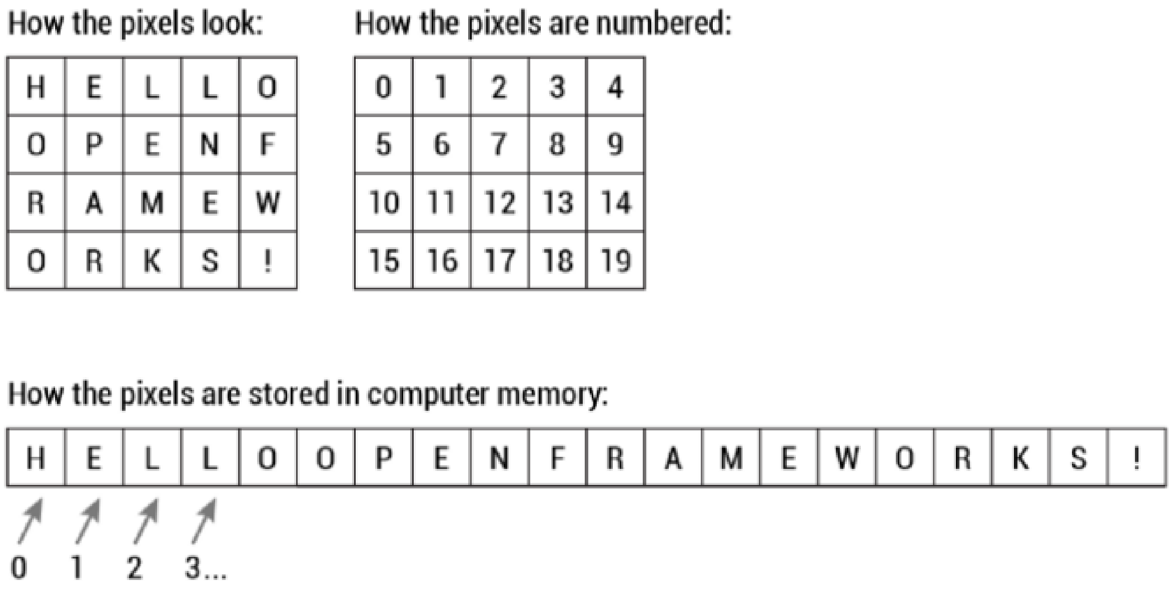


157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	236	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Pixel data diagram. At left, our image of Lincoln; at center, the pixels labeled with numbers from 0–255, representing their brightness; and at right, these numbers by themselves.

Fig - 1.1

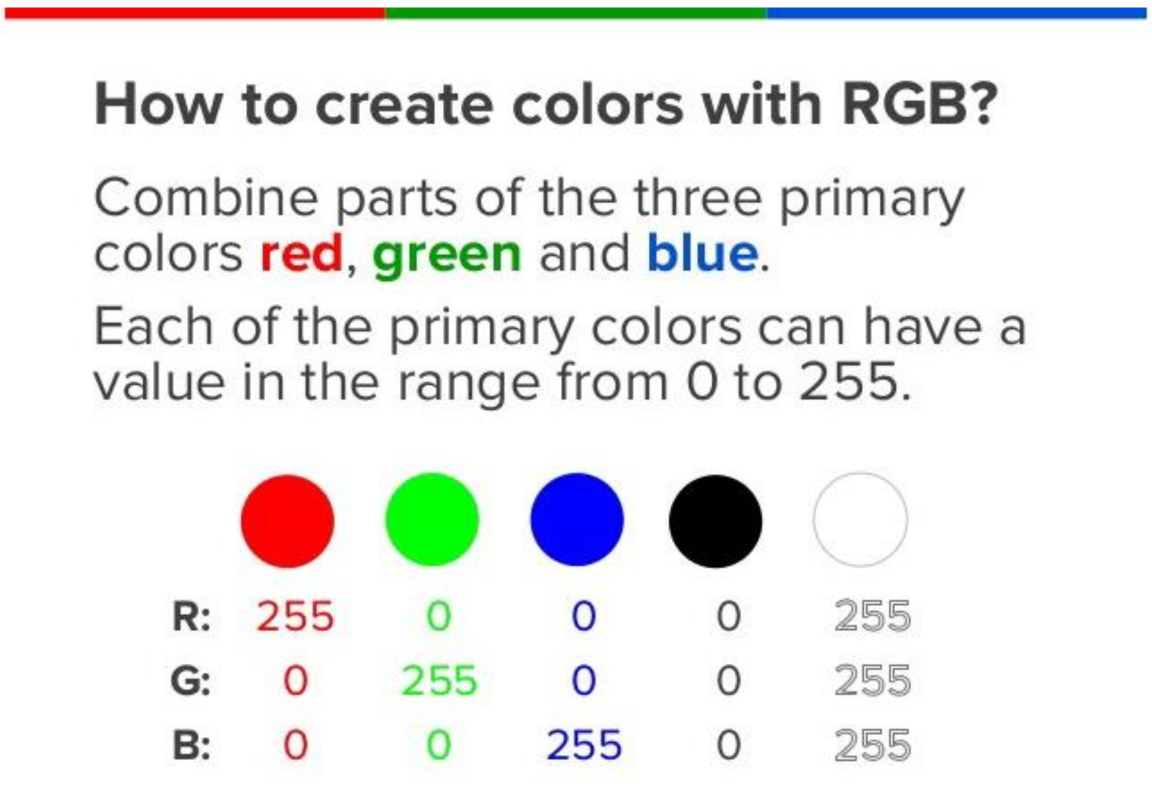
This way of storing image data may run counter to your expectations, since the data certainly *appears* to be two-dimensional when it is displayed. Yet, this is the case, since computer memory consists simply of an ever-increasing linear list of address spaces.


How the pixels look:
H E L L O
O P E N F
R A M E W
O R K S !
How the pixels are numbered:
0 1 2 3 4
5 6 7 8 9
10 11 12 13 14
15 16 17 18 19
How the pixels are stored in computer memory:
H E L L O O P E N F R A M E W O R K S !
0 1 2 3...

How pixels are stored in memory

Fig - 1.2

Imagine adding a colored image. Now things start to get more complicated. Computers usually read color as a series of 3 values — red, green, and blue (RGB) — on that same 0–255 scale. Now, each pixel actually has 3 values for the computer to store in addition to its position. If we were to colorize President Lincoln, that would lead to 12 x 16 x 3 values, or 576 numbers.



How to create colors with RGB?

Combine parts of the three primary colors **red**, **green** and **blue**.

Each of the primary colors can have a value in the range from 0 to 255.

	●	●	●	●	○
R:	255	0	0	0	255
G:	0	255	0	0	255
B:	0	0	255	0	255

Fig - 1.3

1.3 FIELD RELATED TO COMPUTER VISION.

1.Artificial intelligence

Areas of artificial intelligence deal with autonomous path planning or deliberation for robotic systems to navigate through an environment. A detailed understanding of these environments is required to navigate through them. Information about the environment could be provided by a computer vision system, acting as a vision sensor and providing high-level information about the environment and the robot.

Artificial intelligence and computer vision share other topics such as pattern recognition and learning techniques. Consequently, computer vision is sometimes seen as a part of the artificial intelligence field or the computer science field in general.

2.Solid-state physics

Solid-state physics is another field that is closely related to computer vision. Most computer vision systems rely on image sensors, which detect electromagnetic radiation, which is typically in the form of either visible or infra-red light. The sensors are designed using quantum physics. The process by which light interacts with surfaces is explained using physics. Physics explains the behavior of optics which are a core part of most imaging systems. Sophisticated image sensors even require quantum mechanics to provide a complete understanding of the image formation process. Also, various measurement problems in physics can be addressed using computer vision, for example motion in fluids.

3.Neurobiology

A third field which plays an important role is neurobiology, specifically the study of the biological vision system. Over the last century, there has been an extensive study of eyes, neurons, and the brain structures devoted to processing of visual stimuli in both humans and various animals. This has led to a coarse, yet complicated, description of how "real" vision systems operate in order to solve certain vision-related tasks. These results have led to a sub-field within computer vision where artificial systems are designed to mimic the processing and behavior of biological systems, at different levels of complexity. Also, some of the learning-based methods developed within computer vision (e.g. neural net and deep learning based image and feature analysis and classification) have their background in biology.

Some strands of computer vision research are closely related to the study of biological vision indeed, just as many strands of AI research are closely tied with research into human consciousness, and the use of stored knowledge to interpret, integrate and utilize visual information. The field of biological vision studies and models the physiological processes behind visual perception in humans and other animals. Computer vision, on the other hand, studies and describes the processes implemented in software and hardware behind artificial vision systems. Interdisciplinary exchange between biological and computer vision has proven fruitful for both fields.

4.Signal processing

Yet another field related to computer vision is signal processing. Many methods for processing of one-variable signals, typically temporal signals, can be extended in a natural way to processing of two-variable signals or multi-variable signals in computer vision.

However, because of the specific nature of images there are many methods developed within computer vision that have no counterpart in processing of one-variable signals. Together with the multi-dimensionality of the signal, this defines a subfield in signal processing as a part of computer vision.

5.Other fields

Beside the above-mentioned views on computer vision, many of the related research topics can also be studied from a purely mathematical point of view. For example, many methods in computer vision are based on statistics, optimization or geometry. Finally, a significant part of the field is devoted to the implementation aspect of computer vision; how existing methods can be realized in various combinations of software and hardware, or how these methods can be modified in order to gain processing speed without losing too much performance. Computer vision is also used in fashion ecommerce, inventory management, patent search, furniture, and the beauty industry.

1.4 APPLICATION AND TASK PERFORM BY COMPUTER VISION

Applications range from tasks such as industrial machine vision systems which, say, inspect bottles speeding by on a production line, to research into artificial intelligence and computers or robots that can comprehend the world around them. Some of the application are: -

1.Medicine

2.Machine Vision

3.Military

4.Autonomous vehicles

TASK PERFORM BY COMPUTER VISION: -

1.Recognition

2.Motion analysis

3.Scene reconstruction

4.Image restoration

1.5 OPENCV

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today’s systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as Numpy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. When OpenCV was designed the main focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.

1.6 AUTOMATIC NUMEBR PLATE RECOGNITION

In last few years, ANPR or number 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, car park system and Automatic vehicle parking system. ANPR algorithms are generally divided in four steps:

(1) Vehicle image capture

(2) Number plate detection

(3) Character segmentation

(4) Character recognition

.

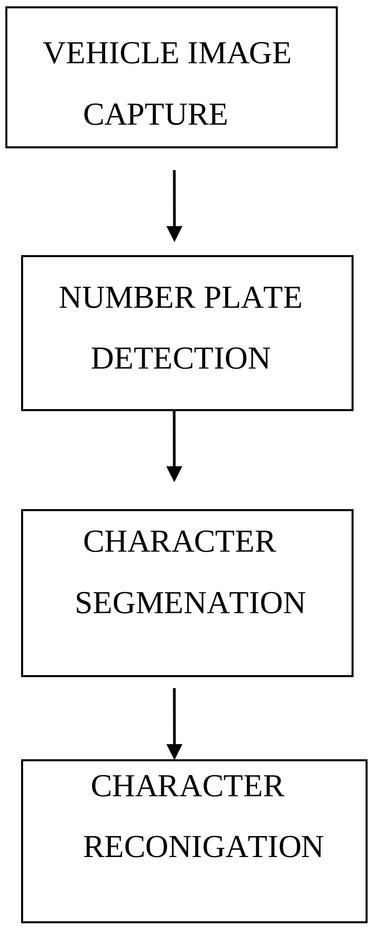


Fig - 1.4

As it is shown in Fig.1.4, the first step i.e. to capture image of vehicle looks very easy but it is quite exigent task as it is very difficult to capture image of moving vehicle in real time in such a manner that none of the component of vehicle especially the vehicle number plate should be miss. The success of fourth step depends on how second and third step are able to locate vehicle number plate and separate each character. Different methods follow different approaches to locate vehicle number plate from vehicle and then to extract vehicle number from that image.

STEP INVOLVE IN AUTOMATIC NUMBER PALTE RECONIZATION

1.Vehical image capture

Image can be capture using the camera or cctv.

2. Number Plate Detection

The first step is to detect the number plate from the car. We will use the contour option in OpenCV to detect for rectangular objects to find the number plate. The accuracy can be improved if we know the exact size, color and approximate location of the number plate. Normally the detection algorithm is trained based on the position of camera and type of number plate used in that particular country. This gets trickier if the image does not even have a car, in this case we will an additional step to detect the car and then the number plate.

3. Character Segmentation: Once we have detected the Number Plate we have to crop it out and save it as a new image. Again this can be done easily using OpenCV.

4. Character Recognition: Now, the new image that we obtained in the previous step is sure to have some characters (Numbers/Alphabets) written on it. So, we can perform OCR (Optical Character Recognition) on it to detect the number

PROCUDRE

The project is divide in two part:

PROJRCT PART 1

Use opencv to detect the number plate.

PROJRCT PART 2

Detect the plate using trained model. The model is trained to detect the number plate in given image.

The result of both part is compared and best result is shown to us.

Project part 1

1. **VECHICAL IMAGE CAPUTRE**

We are taking the photo of car given below

Fig - 1.5

2.NUMBER PALTE DETECTION

Step 1: Converting the color image into grayscale

##Coverting the image into the grayscale image

img_gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

Fig - 1.6

Gray scaling is common in all image processing steps. This speeds up other following process sine we no longer have to deal with the color details when processing an image. The image would be transformed something like this when this step is done

Step 2: Every image will have useful and useless information, in this case for us only the number plate is the useful information the rest are pretty much useless for our program. This useless information is called noise. Normally using a bilateral filter (Blurring) will remove the unwanted details from an image. The code for the same is blurred

img_gray=cv2.GaussianBlur(img_gray,(3,3),17,17)

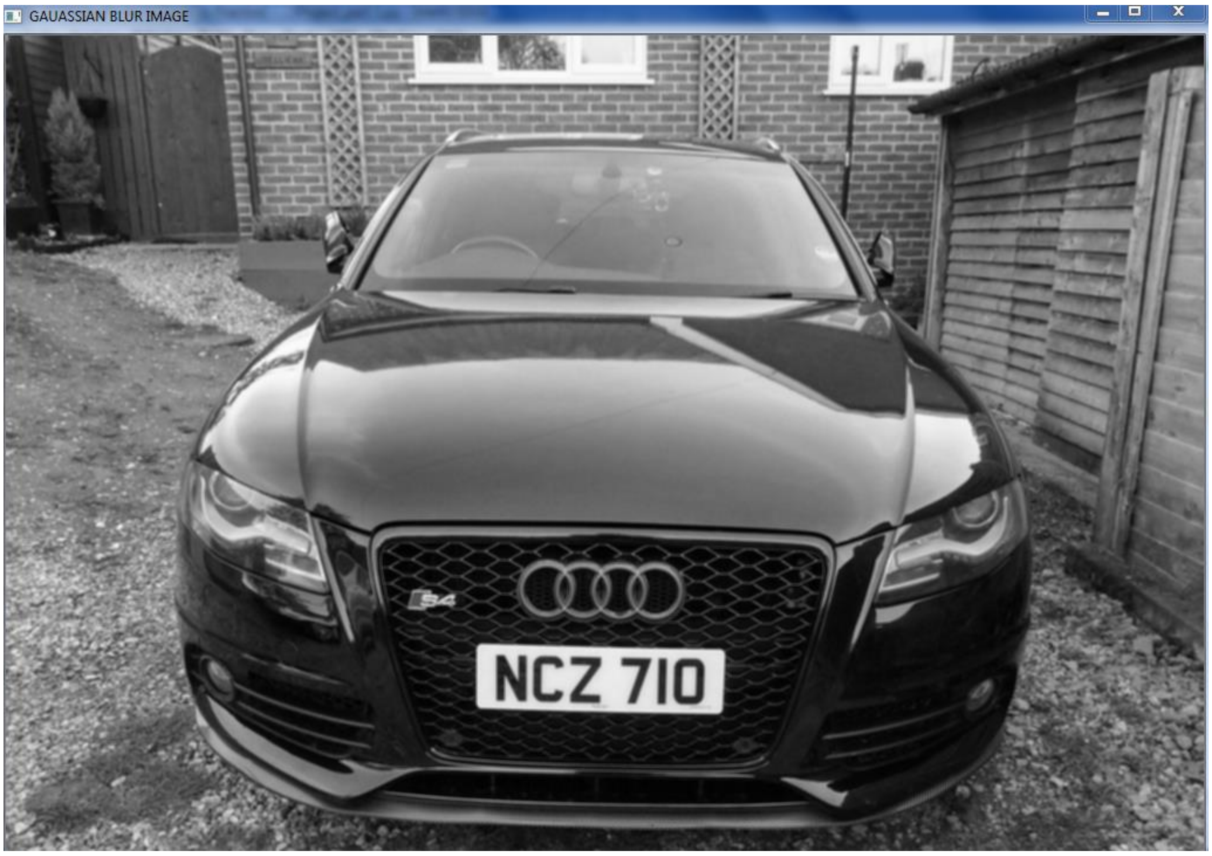


Fig - 1.7

Syntax is destination image = cv2.bilateralFilter(source image, diameter of pixel, sigma Color, sigma Space). You can increase the sigma color and sigma space from 17 to higher values to blur out more background information, but be careful that the useful part does not get blurred. The output image is shown below, as you can see the background details (tree and building) are blurred in this image. This way we can avoid the program from concentrating on these regions later.

Step 3: The next step is interesting where we perform edge detection. There are many ways to do it, the easiest and popular way is to use the canny edge method from OpenCV. The line to do the same is shown below

median=np.median(img_gray)

lower=int(max(0,0.7*median))

upper=int(min(255,1.3*median))

canny_edge=cv2.Canny(img_gray,lower,upper)

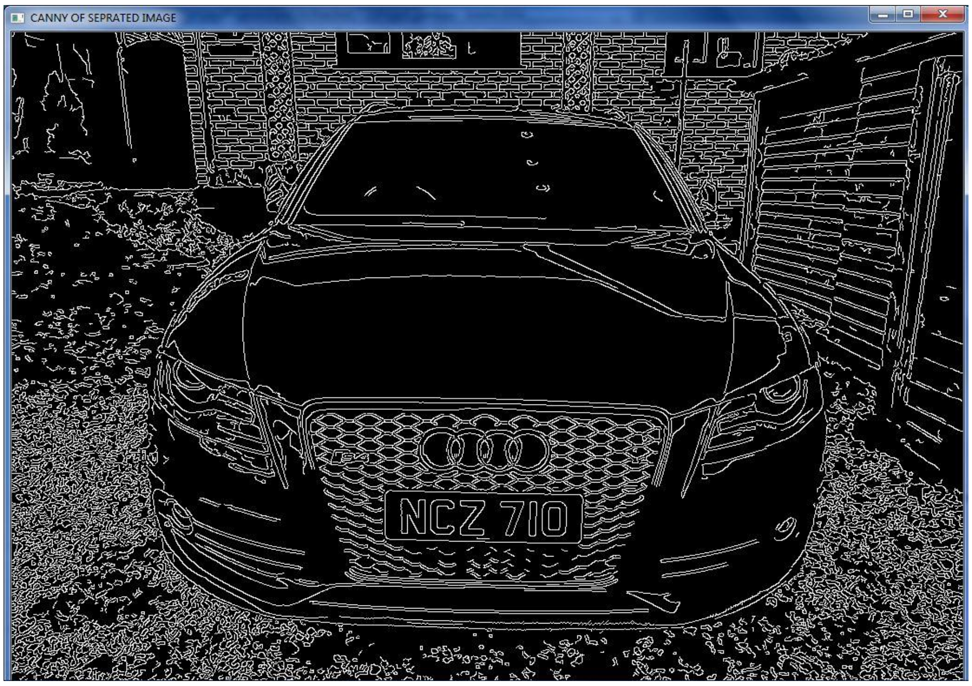


Fig - 1.8

The syntax will be destination image = cv2.Canny(source image, threshold Value 1, threshold Value 2). The Threshold Vale 1 and Threshold Value 2 are the minimum and maximum threshold values. Only the edges that have an intensity gradient more than the minimum threshold value and less than the maximum threshold value will be displayed. The resulting image is shown below

Step 4: Now we can start looking for contours on our image

counters,new=cv2.findContours(canny_edge,cv2.RETR_LIST,cv2.CHAIN_APPROX_NONE)

counters=sorted(counters,key=cv2.contourArea,reverse=True)

counters_with_image=None

license_plate=None

Once the counters have been detected we sort them from big to small and consider only the first 10 results ignoring the others. In our image the counter could be anything that has a closed surface but of all the obtained results the number plate number will also be there since it is also a closed surface.

To filter the number plate image among the obtained results, we will loop though all the results and check which has a rectangle shape contour with four sides and closed figure. Since a number plate would definitely be a rectangle four sided figure.

Once we have found the right counter we save it in a variable called screen Counture and then draw a rectangle box around it to make sure we have detected the number plate correctly.

```
x=None

y=None

w=None

h=None

for counter in counters:

    perimetrs=cv2.arcLength(counter,True)

    approx=cv2.approxPolyDP(counter,0.01*perimetrs,True)

    if len(approx)==4:

        x,y,w,h=cv2.boundingRect(approx)

        counters_with_image=approx

        x,y,w,h=cv2.boundingRect(counter)

        cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,23),2)

        license_plate=img[y:y+h,x:x+w]

        break
```

Step 5: Now that we know where the number plate is, the remaining information is pretty much useless for us. So we can proceed with masking the entire picture except for the place where the number plate is. The code to do the same is shown below The masked new image will appear something like below

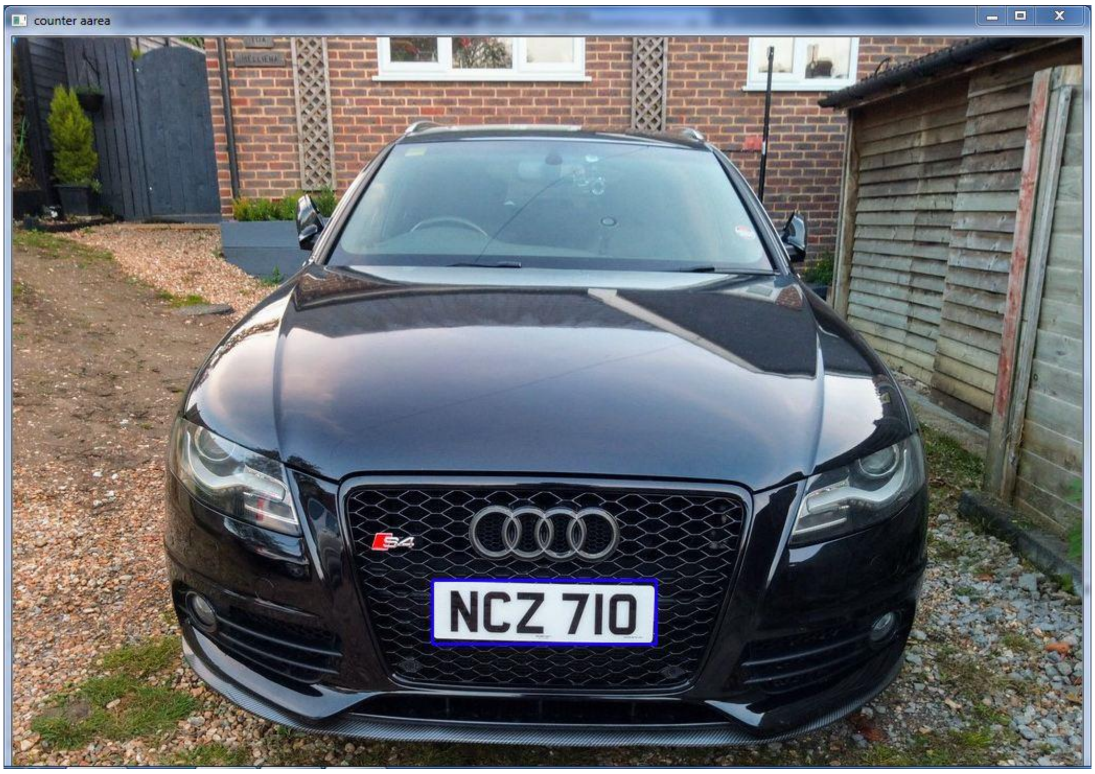


Fig - 1.9

2. Character Segmentation

The next step in Number Plate Recognition is to segment the license plate out of the image by cropping it and saving it as a new image**.** We can then use this image to detect the character in it. The code to crop the roi (Region of interest) image form the main image is shown below

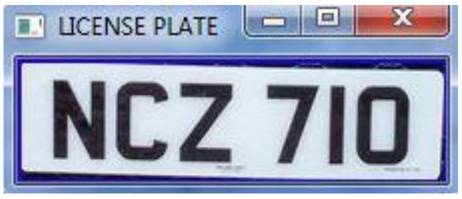


Fig - 1.10

3. Character Recognition

The Final step in this Number Plate Recognition is to actually read the number plate information from the segmented image. We will use the ***pytesseract*** package to read characters from image, just like we did in previous tutorial. The code for the same is given below

```
final_result=pytesseract.image_to_string(license_plate)
```

PROJECT PART TWO:-

2.NUMBER PALTE DETECTION

Step 1: Converting the color image into grayscale

##Coverting the image into the grayscale image

```
img_gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
```

Gray scaling is common in all image processing steps. This speeds up other following process sine we no longer have to deal with the color details when processing an image. The image would be transformed something like this when this step is done

Step 2: Every image will have useful and useless information, in this case for us only the number plate is the useful information the rest are pretty much useless for our program. This useless information is called noise. Normally using a bilateral filter (Blurring) will remove the unwanted details from an image. The code for the same is blurred

```
img_gray=cv2.GaussianBlur(img_gray,(3,3),17,17)
```

Syntax is destination_image = cv2.bilateralFilter(source_image, diameter of pixel, sigmaColor, sigmaSpace). You can increase the sigma color and sigma space from 17 to higher values to blur out more background information, but be careful that the useful part does not get blurred. The output image is shown below, as you can see the background details (tree and building) are blurred in this image. This way we can avoid the program from concentrating on these regions later.

Step 3: Decting plate using the haar cascade model for detecting the number plate

```
plate=cv2.CascadeClassifier(r"D:\opencv-master\data\haarcascades\haarcascade_russian_plate_number.xml"

plate_pos=plate.detectMultiScale(img_gray)
```

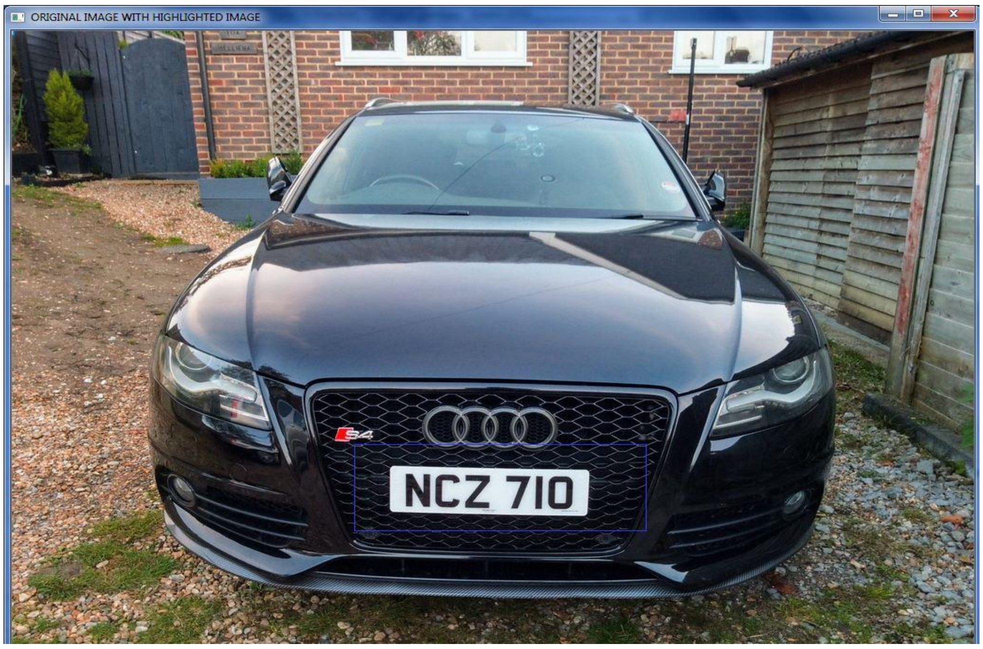


Fig - 1.11

##REST OF THE PORCESS IS SAME AS PROJECT 1

FULL CODE OF PORJECR PART 1

```
import tkinter.messagebox as tmsg

import string

import cv2

import pytesseract

##Reading the image from the system

img=cv2.imread(r"C:\Users\DEVESH\Downloads\CARS\FINAL
IMGE\44024595760_4afef7563e_b.jpg")

##Showing the original image

cv2.imshow("ORIGINAL IMAGE",img)

cv2.waitKey(0)

cv2.destroyWindow("ORIGINAL IMAGE")

##Coverting the image into the grayscale image

img_gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

#Applying the gaussianblur method for blurring the image

img_gray=cv2.GaussianBlur(img_gray,(3,3),17,17)

#Applying the pretrained model to detect the number plate

plate=cv2.CascadeClassifier(r"D:\opencv-
master\data\haarcascades\haarcascade_russian_plate_number.xml")

##storing the position ofthe number plate

plate_pos=plate.detectMultiScale(img_gray)

try:

   ##highlighting the number plate dected in the original image

  for (x,y,w,h) in plate_pos[0:1]:

    cv2.rectangle(img,(x,y),(x+w,y+h),(255,55,52),1)

    ##seperating the image of number plate from the original image

    sep=img[y:y+h,x:x+w]

  ##decting the text from the seprated image

  dected_number=pytesseract.image_to_string(sep)
```

```
except:
  plate_pos=plate.detectMultiScale(img_gray,1.2,5)
  for (x,y,w,h) in plate_pos[0:1]:
    cv2.rectangle(img,(x,y),(x+w,y+h),(255,55,52),1)
    sep=img[y:y+h,x:x+w]
    dected_number=pytesseract.image_to_string(sep)
##Showing the deted number plate image
cv2.imshow("DETECTED LICENSE PLATE AND SEPRATED FROM THE  ORIGINAL
IMAGE",sep)
cv2.waitKey(0)
cv2.destroyAllWindows()
##Showing the image
cv2.imshow("ORIGINAL IMAGE WITH HIGHLIGHTED IMAGE",img)
cv2.waitKey(0)
cv2.destroyAllWindows()


##Filtering the number plate number
valid_number=string.ascii_letters+string.digits
final_result_part1=""
for i in range(len(dected_number)):
   if dected_number[i] in valid_number:
       final_result_part1=final_result_part1+dected_number[i]
print("NUMBER PLATE IS =",final_result_part1)


##Calling the other method
from Project_part2 import *
number()
##Finaliszation of the number detected
if final_result_part1==final_result_part2:
      print("This is our final reslut",final_result_part1)
      TEMP=final_result_part1
```

```
elif len(final_result_part1)==0:
    print("THE RESLUT",final_result_part2)
    TEMP=final_result_part2
elif len(final_result_part2)==0:
    print("THE RESLUT",final_result_part1)
    TEMP=final_result_part1

import time
TIME=time.localtime()
value=tmsg.askquestion("Save","Do you want to save the details?")
if value=="yes":
  file=open("IMP","a")
  file.write("\nTHE PLATE NUMBER IS ={0:<10}".format(TEMP))
  file.write(f"\t\t\t\t  DATE= {TIME[2]}/{TIME[1]}/{TIME[0]}")
  file.write(f"\t\t\t\t  TIME={TIME[3]}:{TIME[4]}:{TIME[5]}")
  file.close()
  tmsg.showinfo("SAVED","Details are saved")
else:
     tmsg.showinfo("Exit","Details are not saved")
```

PROJECT PART 2

```
import string
import numpy as np
import cv2
import pytesseract
##Reading the image
img=cv2.imread(r"C:\Users\DEVESH\Downloads\CARS\FINAL IMGE\44024595760_4afef7563e_b.jpg")
img_gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
img_gray=cv2.GaussianBlur(img_gray,(3,3),17,17)
median=np.median(img_gray)
```

```
lower=int(max(0,0.7*median))
upper=int(min(255,1.3*median))
# canny_edge=cv2.Canny(img_gray,170,200)
canny_edge=cv2.Canny(img_gray,lower,upper)
cv2.imshow("CANNY OF SEPRATED IMAGE",canny_edge)
cv2.waitKey(0)
blank=np.zeros(img.shape,np.uint8)
##finding counters
counters,new=cv2.findContours(canny_edge,cv2.RETR_LIST,cv2.CHAIN_APPROX_NON
E)
counters=sorted(counters,key=cv2.contourArea,reverse=True)
counters_with_image=None
license_plate=None
x=None
y=None
w=None
h=None
for counter in counters:
    perimetrs=cv2.arcLength(counter,True)
    approx=cv2.approxPolyDP(counter,0.01*perimetrs,True)
    if len(approx)==4:
        x,y,w,h=cv2.boundingRect(approx)
        counters_with_image=approx
        x,y,w,h=cv2.boundingRect(counter)
        cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,23),2)
        license_plate=img[y:y+h,x:x+w]
        break
cv2.imshow("counter aarea",img)
cv2.waitKey(0)
cv2.imshow("LICENSE PLATE",license_plate)
cv2.waitKey(0)
```

```
final_result=pytesseract.image_to_string(license_plate)
alpha=string.ascii_letters
digit=string.digits
dict=digit+alpha
final_result_part2=""
for i in range(len(final_result)):
    if final_result[i] in dict:
        final_result_part2=final_result_part2+final_result[i]

print("NUMBER PLATE IS =",final_result_part2)
def number():
    return final_result_part2
```

CHAPTER 2

BENEFITS

- Automatic number plate recognition cameras are used to measure the average vehicle speed over longer distances
- Used to identify a motorist when he/she drives away without paying for their fuel
- Targeted advertisement
- Automatic number plate recognition cameras are used for Traffic management systems.
- Used to Analyze the behavior (route choice, origin-destination etc.) of a motorist for transport planning purposes
- ANPR camera solutions automatically recognize customers based on their number plate and provide them the complete information about the items that they ordered the last time they used the service.
- Automatic number plate recognition camera solutions are used to recognize the guest vehicles in order to assist visitor management systems.

CHAPTER - 3

Application areas of Automatic Number Plate Recognition: -

Parking

One of the main applications of ANPR is parking automation and parking security: ticketless parking fee management, parking access automation, vehicle location guidance, car theft prevention, "lost ticket" fraud, fraud by changing tickets, simplified, partially or fully automated payment process, among many others.

Access Control

Access control in general is a mechanism for limiting access to areas and resources based on users' identities and their membership in various predefined groups. Access to limited zones, however, may also be managed based on the accessing vehicles alone, or together with personal identity. Number plate recognition brings automation of vehicle access control management, providing increased security, car pool management for logistics, security guide assistance, event logging, event management, keeping access diary, possibilities for analysis and data mining.

Motorway Road Tolling

Road Tolling means, that motorists pay directly for the usage of particular segment of road infrastructures. Tolls are a common way of funding the improvements of highways, motorways, roads and bridges: tolls are fees for services. Efficient road tolling increases the level of related road services by reducing travel time overhead, congestion and improve roadways quality. Also, efficient road tolling reduces fraud related to non-payment, makes charging effective, reduces required manpower to process events of exceptions. License plate recognition is mostly used as a very efficient enforcement tool, while there are road tolling systems based solely on number plate recognition too.

Border Control

Border Control is an established state-coordinated effort to achieve operational control of the country's state border with the priority mission of supporting the homeland's security against terrorism, illegal cross border traffic, smuggling and criminal activities. Efficient border control significantly decreases the rate of violent crime and increases the society's security. Automatic number plate recognition adds significant value by event logging, establishing investigate-able databases of border crossings, alarming on suspicious passing, at many more.

Journey Time Measurement

Journey Time Measurement is a very efficient and widely usable method of understanding traffic, detecting conspicuous situations and events, etc. A computer vision based system has its well-known downfalls in Journey Time Measurement, while Automatic Number Plate Recognition has provided its viability: vehicle journey times can be measured reliably by automatic number plate recognition-based systems. Data collected by number plate recognition systems can be used in many ways after processing: feeding back information to road users to increase traffic security, helping efficient law enforcement, optimising traffic routes, reducing costs and time, etc.

Law Enforcement

Automatic number plate recognition is an ideal technology to be used for law enforcement purposes. It is able to automatically identify stolen cars based on the up-to date blacklist. Other very common law enforcement applications are red-light enforcement and over speed charging and bus lane control.

Toll Road Violations

Open road toll lanes appear to be the preferred way to keep traffic moving by eliminating the bottlenecks associated with gates and cash lanes. The reduction in air pollutants from the exhaust of queued vehicles and the lower operating costs also cannot be overlooked.

These benefits are offset by the requirement to provide accurate transaction accounting and an effective violation enforcement tool. Automatic Number Plate Reading has proven to be the right solution for those requirements. To be truly effective, the ALPR system needs to be able to read all plate types and at any speed anticipated in the lane.

Alert Notification

When you need to be notified that a vehicle of interest has been detected, you can add the Alert Notification and Management module.

- Each alert message contains plate and vehicle images along with vehicle information
- Alert messages can be sent to an individual or broadcast to a list of authorized recipients. of a match via email, SMS or MMS messaging
- Alert messages can be in e-mail, SMS or MMS format

Automatic Number Plate Recognition system to accurately read the number plates of passing vehicles. These reads are matched against a known list of vehicles of interest with all positive matches reported via email, SMS or MMS messaging. The system takes less than 500msec from the time a vehicle with a listed number plate is encountered to issuing the outbound alert messages. Each message contains an image of the number plate, an optional image of the scene with the vehicle present, and other pertinent data such as date, time, location and other user-supplied information in the blacklist.

Future prospect

Today advances technology took Automatic Number Plate Recognition (ANPR) systems from hard to set up, limited expensive, fixed based applications to simple mobile ones in which “point to shoot” method can be used. This is possible because of the creation of software which ran on cheaper PC based and also non specialist hardware in which their no need to give pre-defined direction, angels, speed and size in which the plate would be passing the camera field of view. Also Smaller cameras which can read number plates at high speed, along with smaller, more durable processors that can fit in police vehicles, allowed law enforcement officers to patrol daily with the benefit of number plate recognition in real time. We need to improve the accuracy of the automatic plate detection, if we able to increase thee accuracy we can also implement this system in drone that will enhance the speed and make a whole process a lot faster.

CONCLUSION

This system use image processing techniques using opencv for recognition of the vehicle from the database stored in the computer Along with recognizing the number plates also the timings of vehicle’s entrance are recorded and stored into database. The method works satisfactorily for wide variation of conditions and different types of number plates. This ANPR system works satisfactory however, it still room for improvement. This ANPR system speed can be increase 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 The method successful in recognizing the number plate of vehicles.

REFERENCE

- WIKIPEDIA

  https://www.wikipedia.org/

- YOUTUBE

  https://www.youtube.com/

- STACKOVERFLOW

  https://stackoverflow.com/

- GEEKSFORGEEKS

  https://www.geeksforgeeks.org/

- UDEMY

  https://www.udemy.com/