Number Plate Detection System

CT/DT Number: CT20172340445
Contestant Name: ARUN SINGH
College Name: Krishna Institute of Engineering & Technology, Ghaziabad

1. Background

Vehicle Number Plate detection –

Identify the license place in the image and do an OCR to extract the characters from the detected license plate.

A Vehicle Number Plate detection uses optical character recognition technology to automatically read vehicle license plate as an Image. It is also a versatile and surveillance technology that used in various traffic applications. It is the cost-effective way to increase the securities premises by giving the capability to control, monitor and log access of any vehicles. This technology adopted and can be used for civil enforcement, car park management, toll booth, Smart city, police and security. Detection technology embedded with security cameras to providing the astounding response.

License Plate Recognition systems have been implemented in many countries like United States of America, Australia, Korea and few others. Strict implementation of license plate standards in these countries has helped the early development of License Plate Recognition systems. These systems use standard features of the license plates such as: dimensions of plate, border for the plate, color and font of characters, etc. help to localize the number plate easily and identify the license number of the vehicle. In India, number plate standards are rarely followed. Wide variations are found in terms of font types, script, size, placement and color of the number plates. In few cases, other unwanted decorations are present on the number plate. Also, unlike other countries, no special features are available on Indian number plates to ease their recognition process.

Even though automatic number plate detection and recognition has already been widely tested and adopted by a lot of countries for surveillance purposes, but in India where the size of number plates on Indian vehicles are not fixed and the CCTV cameras which are used for surveillance purpose are not of high resolution — ANPR(Automatic Number Plate Recognition) remains a challenge to be solved.

The efficiency of most of the plate recognition systems is poor particularly for Indian License Plates which are generally highly complex in localizing the plate area, segmenting the characters within the license plate and the character recognition of segmented characters as a valid license plate number (a valid sequence of ASCII characters as plate number to be stored in a database for later use).

2. Your Understanding

In this project, our model has to identify the license place in the image and do an OCR to extract the characters from the detected license plate. It is a two-step process. In step 1, localisation of a number plate is to be done i.e. the position where the number plate is located within the image has to be identified. In step 2, Optical character recognition is to be done on the identified image and final output should be number plate number as a text. This automatic process can be used in various traffic application. It can also be used in car park management, smart city etc. In and out time from the official record can be maintained and the method is more reliable.

Researchers have to face various problems while automatic detection and recognition of license plate. Here we have explored few major problems. The first problem is the non uniformity of the license number plate models for different cities. It may also varies from one state to another and thus from vehicle to vehicle. Length of the number plates may also vary. Second prime difficulty is the low resolution of the number plates for vehicles in video frames under typical surveillance systems. Number Plate Detection System is a challenging application due to variations in vehicle scale, perspective and brightness, together with motion blur, occlusions and clutter typically encountered in practical application.

Following is the two-step approach that has been applied in developing our model.

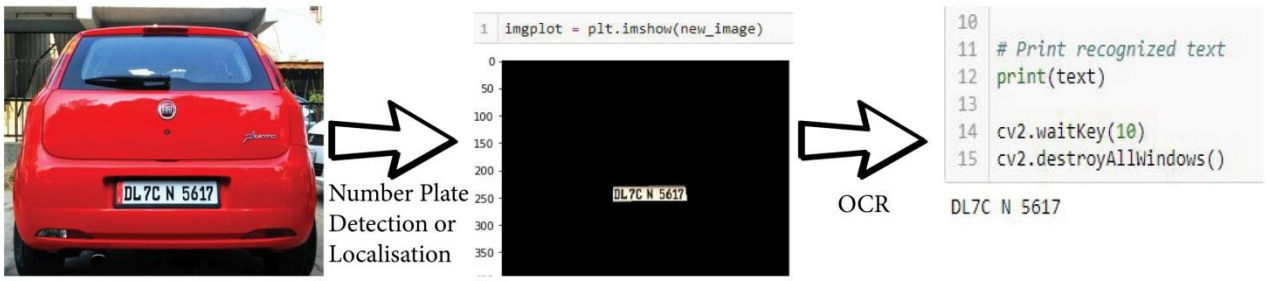


Fig1. Step 1 : Number plate Detection
Step 2 : Optical character recognition

3. Scope

- Number plate and text is written within, can be of any colour or font.
- The number plate can be at any degree of rotation within the image.
- It detects the license plate out of images with a single car. Multiple images of the car in the same picture are not handled.
- It can detect a number plate of any vehicle.

Fig.3.1

Fig.3.2

Fig.3.3

4. Out of Scope

- Number Plate number depicting some other character.
- Number Plated is randomly strongly illuminated –(difficulty in OCR).
- Multiple images of the car in the same picture are not handled.
- Images taken from too far and too blur images are not taken into consideration.

Fig.4.1

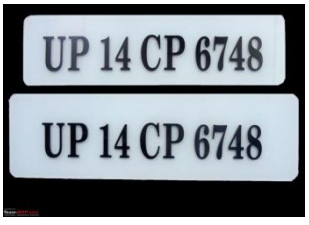


Fig.4.2

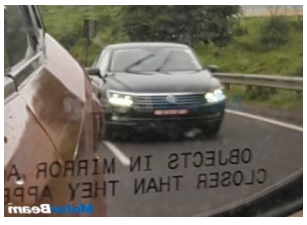


Fig.4.3

5. Assumptions

General Assumptions:

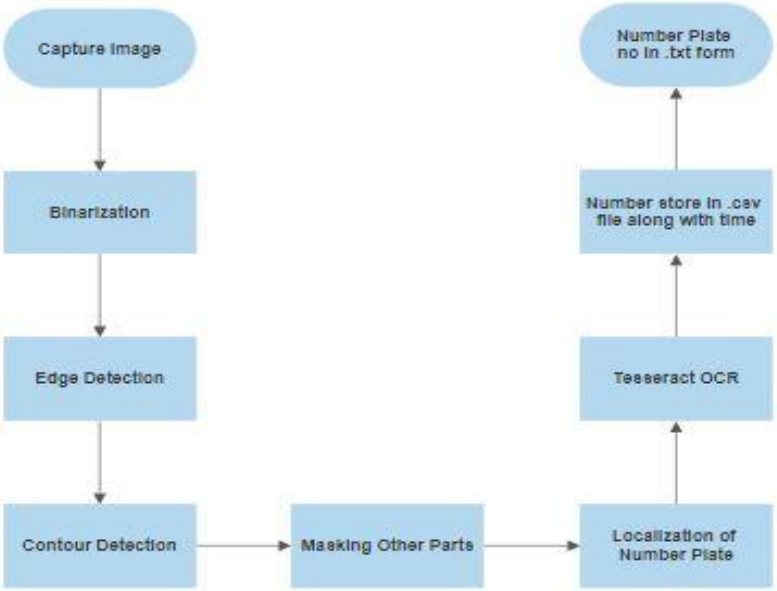
- Number Plate is of Rectanglur shape.
- Input provided is in *.jpg, *.jpeg, *.png format.
- Images are not blurr and doesnot contain any filters.

Technical Assumptions

- Python is installed in a System.
- Tesseract ocr is cloned from github for ocr.
- System on which model is running having minimum requirement of 2 gb ram and Intel core-2 processor.
- All required libraries are already installed before running the required code.

6. Solution Approach

High level Solution Approach



Models/ Algorithms proposed

Image binarization: Image binarization is a process to convert an image to black and white. In this method, certain threshold is chosen to classify certain pixels as black and certain pixels as white. But the main problem is how to choose correct threshold value for particular image. Sometimes it becomes very difficult or impossible to select optimal threshold value. Adaptive Thresholding can be used to overcome this problem. A threshold can be selected by user manually or it can be selected by an algorithm automatically which is known as automatic thresholding

Edge Detection: Edge detection includes a variety of mathematical methods that aim at identifying points in a digital image at which the image brightness changes sharply or, more formally, has discontinuities. The points at which image brightness changes sharply are typically organized into a set of curved line segments termed *edges*. The same problem of finding discontinuities in one-dimensional signals is known as step detection and the problem of finding signal discontinuities over time is known as change detection. Edge detection is a fundamental tool in image processing, machine vision and computer vision, particularly in the areas of feature detection and feature extraction.

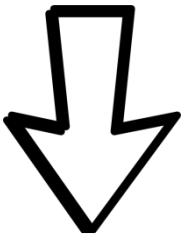
Contour Detection: Contours detection is a process can be explained simply as a curve joining all the continuous points (along with the boundary), having same colour or intensity. The contours are a useful tool for shape analysis and object detection and recognition. Contours can do a bit more than "just" detect edges. The algorithm does indeed find edges of images but also puts them in a hierarchy. This means that you can request outer borders of objects detected in your images. To do contours detection, OpenCV provides a function called *FindContours*which intent to find contours in the image. Of course to some treatment should be applied to the picture to get a good contours detection.

Tesseract 4 OCR –Used to detect text on number plate. Tesseract 4 adds a new neural net (LSTM) based OCR engine which is focused on line recognition, but also still supports the legacy Tesseract OCR engine of Tesseract 3 which works by recognizing character patterns. Compatibility with Tesseract 3 is enabled by using the Legacy OCR Engine mode (--oem 0). It also needs trained data files which support the legacy engine.

7. Implementation Framework

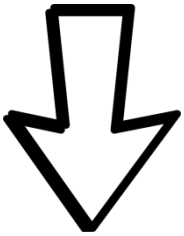
```
In [ ]:
1 import numpy as np
2 import cv2
3 import  imutils
4 import matplotlib.image as mpimg
5 import matplotlib.pyplot as plt
6 import sys
7 import pytesseract
8 import pandas as pd
9 import time
```

Install all the required packages before proceeding to run a code.



```
pytesseract.pytesseract.tesseract_cmd = r"C:\Program Files\Tesseract-OCR\tesseract.exe"
```

Give path of tesseract model.



```
11 image = cv2.imread('7.jpeg')
12
13 image = imutils.resize(image, width=500)
14
15 cv2.imshow("Original Image", image)
16
```

Taking input by cv2.imread in which name of the input image has to be passed considering image is in same directory

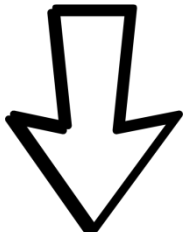
```
cv2.imshow("Original Image", image)

gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
cv2.imshow("1 - Grayscale Conversion", gray)#1

gray = cv2.bilateralFilter(gray, 11, 17, 17)
cv2.imshow("2 - Bilateral Filter", gray)#2

edged = cv2.Canny(gray, 170, 200)
cv2.imshow("4 - Canny Edges", edged)#3
#print("1")
cv2.waitKey(100000)
cv2.destroyAllWindows()#4
```

Edge detection of the original image.

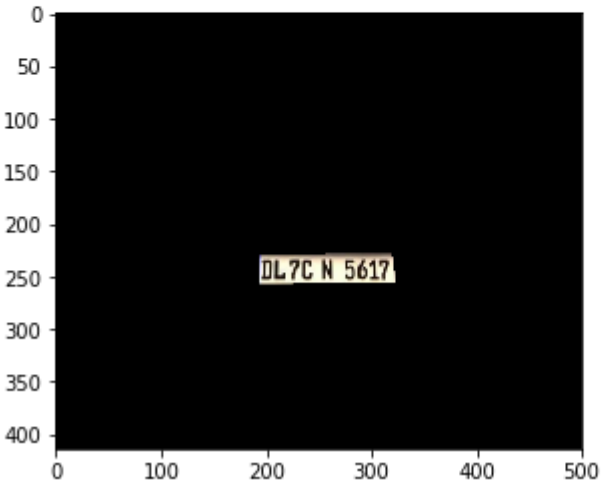
```
(_,cnts, _) = cv2.findContours(edged.copy(), cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)#removed new
#print("3")
#cv2.imshow("check",edged)#5
#print("4")
```

```
cnts=sorted(cnts, key = cv2.contourArea, reverse = True)[:30]
NumberPlateCnt = None

count = 0
for c in cnts:
        peri = cv2.arcLength(c, True)
        approx = cv2.approxPolyDP(c, 0.02 * peri, True)
        if len(approx) == 4:
            NumberPlateCnt = approx
            break
```

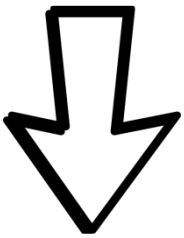
Finding contours from the image for locatisation.



```
In [9]:  1  mask = np.zeros(gray.shape,np.uint8)

In [10]: 1  new_image = cv2.drawContours(mask,[NumberPlateCnt],0,255,-1)

In [11]: 1  new_image = cv2.bitwise_and(image,image,mask=mask)

In [12]: 1  imgplot = plt.imshow(new_image)
```

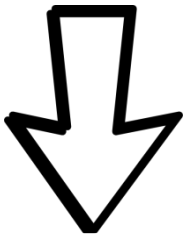


Masking of other parts of the image and localisation of number plate from the image.



```
In [1497]: 1  config = ('-l eng --oem 1 --psm 3')

In [1498]: 1  text = pytesseract.image_to_string(new_image, config=config)
           2  #print(text)#5
           3
           4  #Data is stored in CSV file
           5  raw_data = {'date':[time.asctime( time.localtime(time.time()))],'':[text]}
           6  #raw_data = [time.asctime( time.localtime(time.time()))],[text]
```

Calling tesseract model for character recognition of the number plate and noting time at which engine gets executed i.e. when did car entered.



```
7
8  df = pd.DataFrame(raw_data)
9  df.to_csv('data.csv',mode='a')
10
11 # Print recognized text
12 print(text)
13
14 cv2.waitKey(10)
15 cv2.destroyAllWindows()
```

```
DL7C N 5617
```

Final output is in the text form. And time and number plate number is been stored in *.csv file as output.

Software Details:

Anaconda 3
Python 3.6
Required libraries and packages mentioned above in the document
Tesseract 4

8. Solution Submission

https://github.com/aspapa/tcs_humAin_Number_plate_detection.git

9. Appendix

[1] https://github.com/tesseract-ocr/tesseract

[2] https://opencv-python-tutroals.readthedocs.io/

[3]https://www.researchgate.net/publication/236888959_Automatic_Number_Plate_Recognition_System_ANPR_A_Survey

10. References

https://shodhganga.inflibnet.ac.in/bitstream/10603/71675/7/07_chapter%201.pdf

https://medium.com/datadriveninvestor/license-plate-detector-code-build-and-deploy-790579a18402

https://arxiv.org/ftp/arxiv/papers/1710/1710.10418.pdf

https://medium.com/@deepakkumar1984/contour-detection-in-an-image-c-c65ed2b47c25

https://en.wikipedia.org/wiki/Edge_detection

https://github.com/tesseract-ocr/tesseract