INT-404

Artificial Intelligence

AUTOMATIC NUMBER PLATE RECOGNITION SYSTEM

End Term Report

Name	Reg. No	Roll No
G.Sridhar	11803722	48
P. Dhanushkumar Reddy	11804562	56
B.Sravan kumar	11804577	62
A.Hemanth Reddy	11803639	46

Section: K18HV

Department of Intelligent Systems

School of Computer Science Engineering

Lovely Professional University, Jalandhar

April-2020

STUDENT DECLARATION

This is to declare that this report has been written by us. No part of the report is copied from other sources. All information included from sources have been duly acknowledged. We aver that if any part of the report is found to be copied, we shall take full responsibility for it.

Name of student1:-

G.Sridhar

Reg. No: 11803722

Roll NO: 48

Name of Student3:-

P.DhanushKumar Reddy

Reg. No: 11804562

Roll No: 56

Name of Student2:-

B.Sravan Kumar

Reg. No: 11804577

Roll No: 62

Name of Student4:-

A.Hemanth Reddy

Reg. No: 11803639

Roll No: 46

TABLE OF CONTENTS

TITLE

BONAFIDE CERTIFICATE

Certified that this project report "Automatic Number Plate recognition" is the bonafide work of "G.Sridhar , P.DhanushKumar Reddy, B.Sravan Kumar, A.Hemanth Reddy" who carried out the project the project work under my supervision.

Mr. Dipen Saini Assistant Professor

Dept: Computer Science & Engineering

LPU

Phagwara PUNJAB

1. Background and Objective of the Project:

1.1 Introduction:

Automatic Number Plate Recognition (ANPR) is a mass surveillance system that captures the image of vehicles and recognizes their license number. ANPR can be assisted in the detection of stolen vehicles. The detection of stolen vehicles can be done in an efficient manner by using the ANPR systems located in the highways. This paper presents a recognition method in which the vehicle plate image is obtained by the digital cameras and the image is processed to get the number plate information. A rear image of a vehicle is captured and processed using various algorithms. In this context, the number plate area is localized using a novel „feature-based number plate localization" method which consists of many algorithms. But our study mainly focusing on the two fast algorithms i.e., Edge Finding Method and Window Filtering Method for the better development of the number plate detection system

1.2 Objective:

The main objective of this project is to recognise the number plates of vehicles from traffic camera to remove noise from that image,segmentation of characters and character recognition etc.

2. Description of the Project:

2.1 Data Acquisition and Wrangling

Most of the number plate localization algorithms merge several procedures, resulting in long computational (and accordingly considerable execution) time (this may be reduced by applying less and simpler algorithms). The results are highly dependent on the image quality, since the reliability of the procedures severely degrades in the

case of complex, noisy pictures that contain a lot of details. Unfortunately the various procedures barely offer remedy for this problem, precise camera adjustment is the only solution. This means that the car must be photographed in a way that the environment is excluded as possible and the size of the number plate is as big as possible. Adjustment of the size is especially difficult in the case of fast cars, since the optimum moment of exposure can hardly be guaranteed. Number Plate Localization on the Basis of Edge Finding: The algorithms rely on the observation that number plates usually appear as high contrast areas in the image (black-and-white or black-and-yellow). First, the original car image in color is converted to black and white image grayscale image as shown in figure 1.

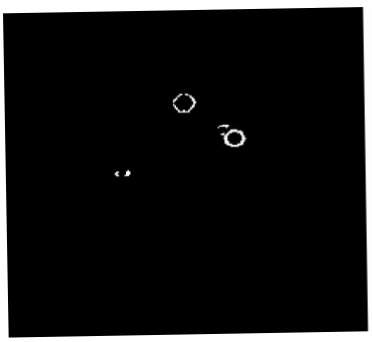
The original image is converted to grayscale image which is in high contrast as shown above. Now, we need to identify the location of the number plate horizontally in which row it‟s present. The letters and numbers are placed in the same row (i.e. at identical vertical levels) resulting in frequent changes in the horizontal intensity. This provides the reason for detecting the horizontal changes of the intensity, since the rows that contain the number plate are expected to exhibit many sharp variations. The Horizontal intensity graph is as follows, with the peaks indicating high contrast regions in the image:

The algorithm first determines the extent of intensity variation for each row, while in the second step it selects the adjacent rows which

exhibit the biggest changes. Number plates are highly probable to be in these rows. The horizontal position of the number plate must also be determined, which is done by using the previously determined values that characterize the changes. The variations are the highest at the letters (black letters on white background); therefore this is where the rate of change within a row is expected to be the highest. Sum of Filtered columns: The vertical position of the number plate must be found in the second step by using a picture obtained by band pass filtering. Having summed up the results of filtering for each row (sum of filtered rows) the vertical position of the number plate is determine on the basis of the statistical properties of the individual rows. To provide a fast algorithm simply the row featuring the highest amplitude is selected (the number plate is most likely to be located there)

2.2 Localisation On The basis Of Window Filtering:

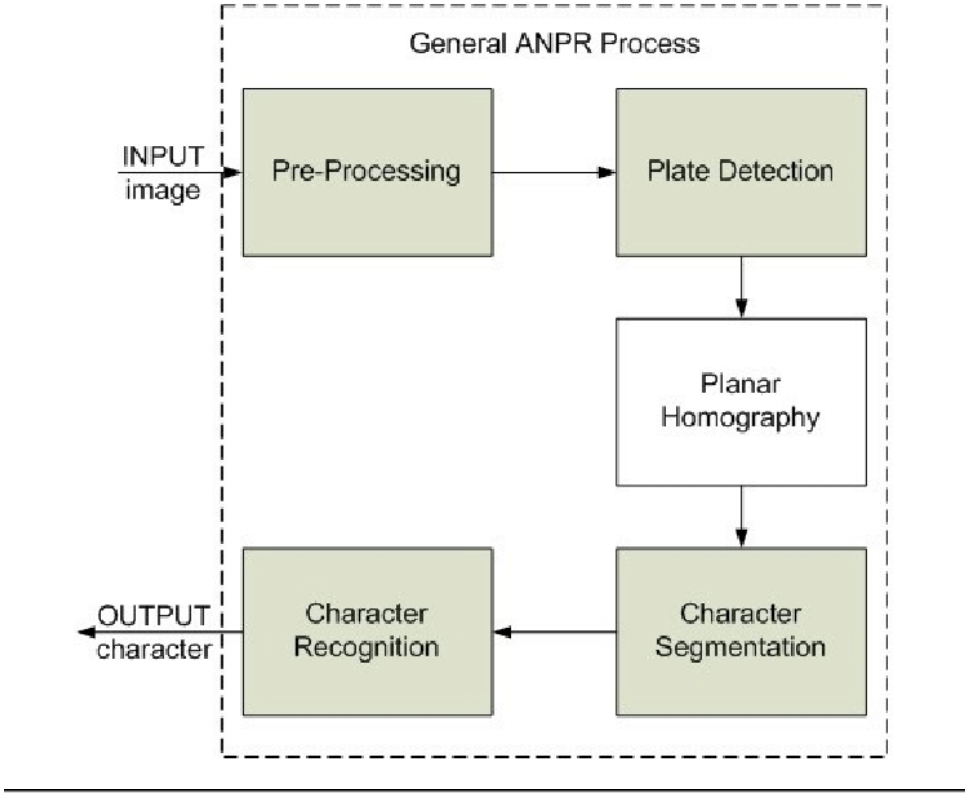
The drawback of the above solution (Edge Finding Methodology) is that after the filtering also additional areas of high intensity appear besides the number plate. If the image contains a lot of details and edges (example: complex background) the further areas. As a result, the SFR curve exhibits a smaller increment at the number plate and the edges in the surrounding areas may sometimes be more dominant.



Car image **Car image after removing noise**

The original image with complex Background is Filtered and the filtered image shows the High contrast regions apart from the number plate. The surroundings are unnecessarily included in the image which made the scene complex. We need to consider a window to exclude the surroundings from the image and concentrate on the actual image. For this we need to consider an appropriate window size. The window size is estimated on the basis of the expected size of the number plate. If the window is chosen to be as wide as the image, then the previously introduced algorithm is obtained, while too small window size leads to incorrect results. This latter algorithm reveals the number plate more effectively from its surroundings. The best result is obtained if the window size equals the width of the number plate, but smaller window dimensions provide fairly good values too. After determining the appropriate window size, we perform the sum of filtered rows and columns and the graph looks like this:

2.3 Proposed Study and Implementation:



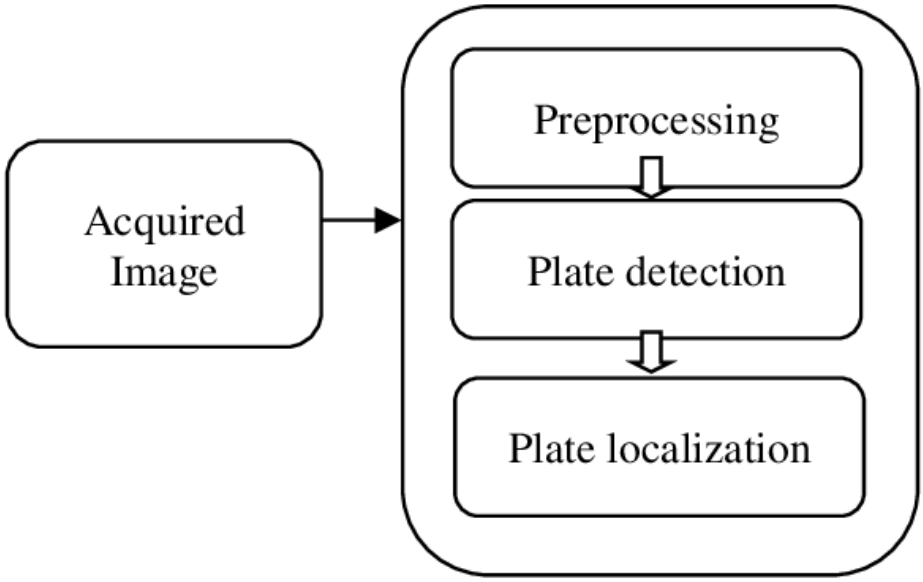


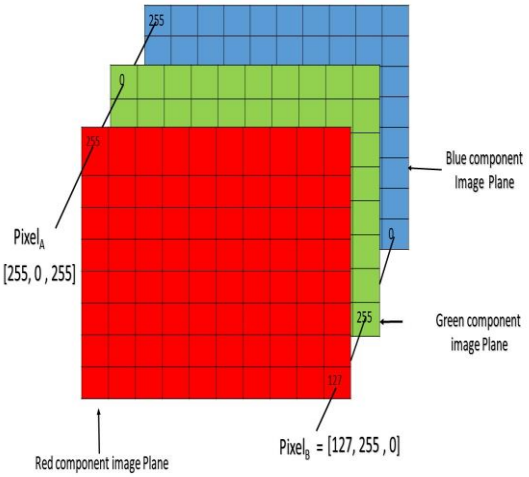
Fig 2: *Steps Involved in Proposed Method.*

2.4 Image Acquisition:

An image is a matrix with X rows and Y columns. It is represented as function say f(x, y) of intensity values for each color over a 2D plane. 2D points, pixel coordinates in an image, can be denoted using a pair of values. The image is stored as a small squared regions or number of picture elements called pixels as shown in the following figure:

TABLE IV: Character Recognition Confusion Matrix

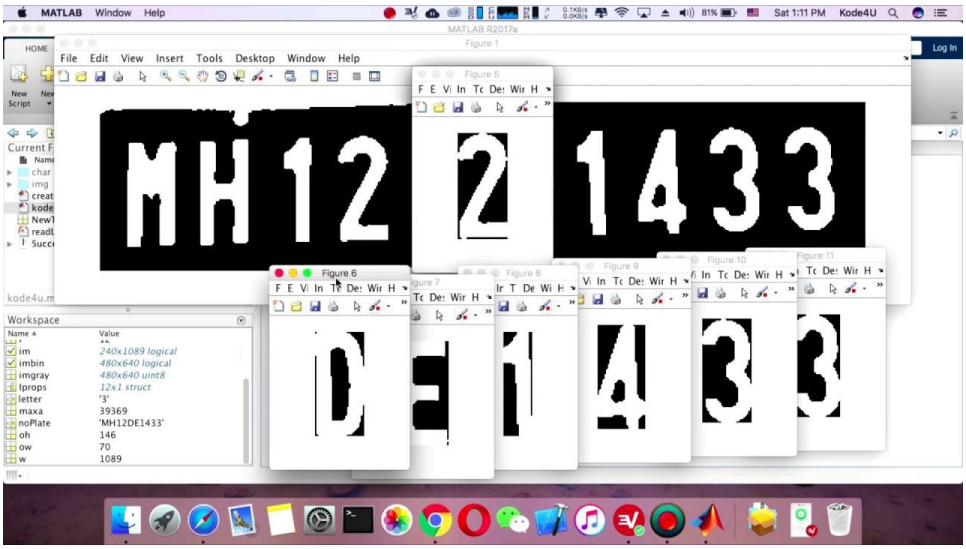
Class	0	1	2	3	4	5	6	7	8	9	BA	PA	Total
0	**11**	0	0	1	0	0	0	1	0	0	0	0	13
1	0	**25**	1	0	0	0	0	0	0	0	0	0	26
2	1	1	**34**	2	0	0	0	0	0	0	0	0	38
3	0	2	1	**30**	0	0	0	0	0	1	0	0	34
4	0	0	0	2	**32**	0	0	0	0	2	0	0	36
5	0	2	2	0	2	**25**	0	1	0	0	0	1	33
6	0	0	1	0	0	0	**32**	0	0	1	0	0	34
7	0	0	0	0	2	0	0	**29**	0	1	0	0	32
8	0	1	1	0	0	0	1	0	**18**	1	0	0	22
9	0	0	1	0	1	2	0	0	0	**13**	0	0	17
BA	0	2	0	1	0	2	0	0	0	2	**25**	2	34
PA	0	2	0	0	0	1	0	2	1	0	0	**35**	41
Total	12	35	41	36	37	30	33	33	19	21	25	38	



Pixel of an RGB image are formed from the corresponding pixel of the three component images

2.5 Image Processing:

The pre-processing is the first step in number plate recognition. It consists the following major stages: 1.Binarization, 2.Noise Removal Binarization: The input image is initially processed to improve its quality and prepare it to next stages of the system. First, the system will convert RGB images to gray-level images. Noise Removal: In this noise removal stage we are going to remove the noise of the image i.e., while preserving the sharpness of the image. After the successful Localization of the Number Plate, we go on with Optical Character Recognition which involves the Segmentation, Feature extraction and Number plate Recognition



2.6 Character Segmentation:

Segmentation is one of the most important processes in the automatic 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 can be improperly merged together. We can use a horizontal projection of a number plate for the segmentation, or one of the more sophisticated methods, such as segmentation using the neural networks. In this segmentation we use two types of segmentation: 1. Horizontal segmentation 2. Vertical segmentation. First we have performed vertical segmentation on the number plate then the characters are vertically segmented. After performing vertical segmentation we have to perform horizontal segmentation by doing this we get character from the plate.

2.7 Character Recognition:

We have to recognize the characters we should perform feature extraction which is the basic concept to recognize the character. The feature extraction is a process of transformation of data from a bitmap representation into a form of descriptors, which are more suitable for computers. The recognition of character should be invariant towards the

user font type, or deformations caused by a skew. In addition, all instances of the same character should have a similar description. A description of the character is a vector of numeral values, so called descriptors or patterns.

2.8 Code:

```
Import numpy as np
import cv2
from copy import deepcopy
from PIL import Image
import pytesseract as tess

def preprocess(img):
    cv2.imshow("Input",img)
    imgBlurred = cv2.GaussianBlur(img, (5,5), 0)
    gray = cv2.cvtColor(imgBlurred, cv2.COLOR_BGR2GRAY)

    sobelx = cv2.Sobel(gray,cv2.CV_8U,1,0,ksize=3)
    #cv2.imshow("Sobel",sobelx)
    #cv2.waitKey(0)
    ret2,threshold_img =
cv2.threshold(sobelx,0,255,cv2.THRESH_BINARY+cv2.THRESH_OT
SU)
    #cv2.imshow("Threshold",threshold_img)
    #cv2.waitKey(0)
    return threshold_img
```

```
def cleanPlate(plate):
    print("CLEANING PLATE. . .")
    gray = cv2.cvtColor(plate, cv2.COLOR_BGR2GRAY)
    #kernel = cv2.getStructuringElement(cv2.MORPH_CROSS, (3, 3))
    #thresh= cv2.dilate(gray, kernel, iterations=1)

    _, thresh = cv2.threshold(gray, 150, 255, cv2.THRESH_BINARY)
    im1,contours,hierarchy =
cv2.findContours(thresh.copy(),cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_NONE)

    if contours:
        areas = [cv2.contourArea(c) for c in contours]
        max_index = np.argmax(areas)

        max_cnt = contours[max_index]
        max_cntArea = areas[max_index]
        x,y,w,h = cv2.boundingRect(max_cnt)

        if not ratioCheck(max_cntArea,w,h):
            return plate,None

        cleaned_final = thresh[y:y+h, x:x+w]
        #cv2.imshow("Function Test",cleaned_final)
        return cleaned_final,[x,y,w,h]
```

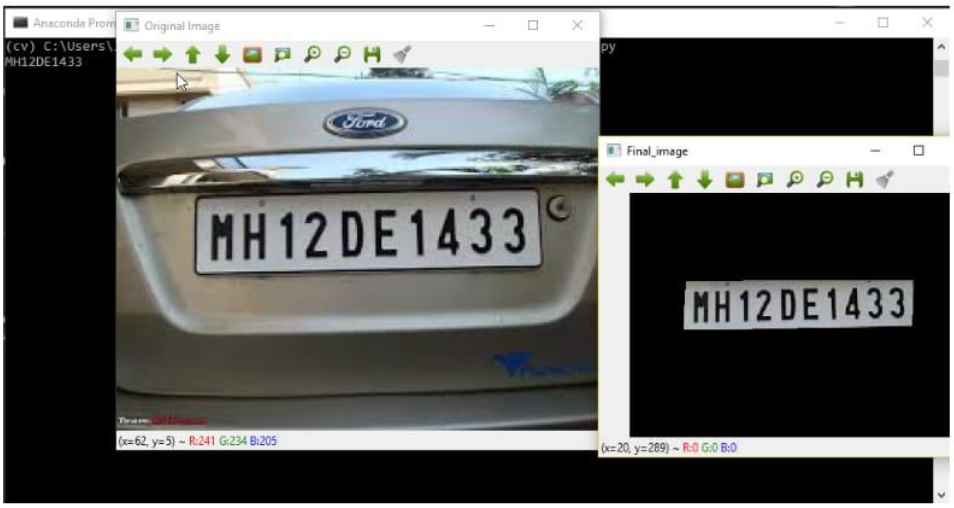
```
    else:
        return plate,None


def extract_contours(threshold_img):
    element = cv2.getStructuringElement(shape=cv2.MORPH_RECT, ksize=(17, 3))
    morph_img_threshold = threshold_img.copy()
    cv2.morphologyEx(src=threshold_img, op=cv2.MORPH_CLOSE, kernel=element, dst=morph_img_threshold)
    cv2.imshow("Morphed",morph_img_threshold)
    cv2.waitKey(0)


    im2,contours, hierarchy= cv2.findContours(morph_img_threshold,mode=cv2.RETR_EXTERNAL,method=cv2.CHAIN_APPROX_NONE)
    return contours


def ratioCheck(area, width, height):
    ratio = float(width) / float(height)
    if ratio < 1:
        ratio = 1 / ratio

    aspect = 4.7272
    min = 15*aspect*15  # minimum area
    max = 125*aspect*125  # maximum area
```

```
    rmin = 3
    rmax = 6

    if (area < min or area > max) or (ratio < rmin or ratio > rmax):
        return False
    return True

def isMaxWhite(plate):
    avg = np.mean(plate)
    if(avg>=115):
        return True
    else:
         return False

def validateRotationAndRatio(rect):
    (x, y), (width, height), rect_angle = rect

    if(width>height):
        angle = -rect_angle
    else:
        angle = 90 + rect_angle

    if angle>15:
         return False
```

```
    if height == 0 or width == 0:
        return False

    area = height*width
    if not ratioCheck(area,width,height):
        return False
    else:
        return True



def cleanAndRead(img,contours):
    #count=0
    for i,cnt in enumerate(contours):
        min_rect = cv2.minAreaRect(cnt)

        if validateRotationAndRatio(min_rect):

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


            if(isMaxWhite(plate_img)):
                #count+=1
                clean_plate, rect = cleanPlate(plate_img)
```

```
            if rect:
                x1,y1,w1,h1 = rect
                x,y,w,h = x+x1,y+y1,w1,h1
                cv2.imshow("Cleaned Plate",clean_plate)
                cv2.waitKey(0)
                plate_im = Image.fromarray(clean_plate)
                text = tess.image_to_string(plate_im,  lang='eng')
                print("Detected Text : ",text)
                img = cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),2)
                cv2.imshow("Detected Plate",img)
                cv2.waitKey(0)

    #print "No. of final cont : " , count



if __name__ == '__main__':
    print( "DETECTING PLATE . . .")

    #img = cv2.imread("testData/Final.JPG")
    img = cv2.imread("testData/test.jpeg")

    threshold_img = preprocess(img)
    contours= extract_contours(threshold_img)

    #if len(contours)!=0:
```

```
#print len(contours) #Test
# cv2.drawContours(img, contours, -1, (0,255,0), 1)
# cv2.imshow("Contours",img)
# cv2.waitKey(0)
```



cleanAndRead(img,contours)

3. Work Division:

This project can't be done by dividing the work of the project among us individually separately. So, everyone revised the every concept and methodology used/related in making this project successful.

4. Technologies and Framework used:

When it comes to the technologies and frameworks used in this project we choose to use python as programming language to program the project which is quite easy to use, portable, extensible and much efficient as of because it offers many libraries and

frameworks like Tesseract, matplotlib, pandas,numpy , open cvvetc which plays a crucial role in this project.

5. Conclusion:

This Report presents a recognition method in which the vehicle plate image is obtained by the digital cameras and the image is processed to get the number plate information. A rear image of a vehicle is captured and processed using various algorithms. Further we are planning to study about the characteristics involved with the automatic number plate system for better performance.