International Journal on Applications in Information and Communication Engineering
Volume 8 : Issue 3 : July 2022, pp 15 – 19 www.aetsjournal.com ISSN (Online) : 2394 - 6237

AUTOMATIC NUMBER PLATE RECOGNITION

PRADEESH P, Dr. A. SIVAKUMAR

***Abstract*—**Number Plate recognition, also called License processing methods is a potential research area in smart cities and the Internet of Things. An exponential in-crease in the number of vehicles necessitates the use of automated systems to maintain vehicle infor-mation for various purposes. In the proposed algo-rithm an efficient method for recognition of Indian vehicle number plates has been devised. We are able to deal with noisy, low illuminated, cross angled, non-standard font number plates. This work employs sev-eral image processing techniques such as, morpho-logical transformation, Gaussian smoothing, Gaussi-an thresholding and Sobel edge detection method in the pre-processing stage, afterwhich number plate segmentation, contours are applied by border follow-ing and contours are filtered based on character di-mensions and spatial localization. Finally we apply Optical Character Recognition (OCR) to recognize the extracted characters. The detected texts are stored in the database, further which they are sorted and made available for searching. The project has its own drawbacks and limitations as we are not using higher machine learning or deep learning algorithms but it works efficiently for an average use case.

***Keywords*—** Gaussian smoothing, Gaussi-an thresholding, Sobel edge detection method, Data Preprocessing, Optical Character Recognition.

I. INTRODUCTION

Large number of vehicles around us in daily life creates disturbances such as heavy traffic, stealing of vehicles at the places like toll plazas, parking areas, heavy traffic roads. Management of vehicles transpor-tation is tedious and time-consuming task if it is com-pletely done manually and which results in huge errors and faults. Therefore, it is necessary to develop automatic license plate recognition system to solve the problems discussed above which will automatically recognize number from front side image of vehicle. The detection number plate goes through following steps: finding plate location in image, segmenting and recognizing characters. Number of license plate is displayed on graphical user interface and stored in database with time and date for further use and alarm will ring of sto-len vehicle is detected. The system can be used for pur-pose of security as well as ahhutomatic highway speed detection, traffic violation cases, toll plazas, parking area.

Vehicles are increasing in enormously as they are necessary to travel from one place to another place in little time. We see number of vehicles around us in our daily life and everyone needs it but with population increase, vehicles increased last decades in large quanti-ty. But it created disturbances to human life such as huge traffic, large sound, crime cases such as stealing of vehicles, accidents, etc. and therefore management of vehicle is very necessary. As a result, there is a lot of work going on to improve the transportation of vehi-cles. Out of these, vehicle Plate Recognition System is the most attractive research issue and this manuscripts discusses some practical aspect of recognizing number written on vehicle number plate. A Vehicle Plate Recognition System is a tracking system that identifies the vehicle so that the car is tracked down through the existing database.

II. APPLICATIONS

Automatic Number Plate Recognition (ANPR) has a wide range of applications since the license number is the primary, most widely accepted, human readable, mandatory identifier of motor vehicles. Some of the applications are:

1. Housing societies / Apartments: ANPR can be used in housing societies to let in resident's vehicles inside by storing the number plate details in the database.

This can hence reduce the man-force near the security gate. Our project works efficiently in this case provided the society or apartment is not very large.

2. Parking: Ticketless parking fee management, parking access automation, vehicle location guidance, parking fee charging, car theft prevention,

Pradeesh P, Student, B.Sc Information Technology, Rathinam College of Arts and Science, Coimbatore, Tamil Nadu, India – 641021, (e-mail: piccasopradeesh@gmail.com).

Dr. A. Sivakumar, Assistant Professor, Department of Computer Science, Rathinam College of Arts and Science, Coimbatore, Tamil Nadu, India – 641021, (e-mail: sivamgp@gmail.com).

"lost ticket" fraud, fraud by changing tickets are some areas where ANPR is helpful .

3. Access Control: License plate recognition brings au-tomation 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. It is very effective in Border and Law controls too.

4. Motorway Road Tolling: Tolls are a common way of funding the improvements of highways, motorways, roads and bridges. Efficient road tolling reduces fraud related to non-payment, makes charging effective, re-duces required manpower to process events of excep-tions and these can be implemented using ANPR.

5. Journey Time Measurement: Data collected by li-cense 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 and reducing costs and time.

III. PROPOSED METHODOLOGY

1. To reduce the noise we need to blur the input Image with Gaussian Blur then convert the it to grayscale
2. Find vertical edges in the image.
3. To reveal the plate we have to binarize the image. For this apply Otsu's Thresholding on the vertical edge image. In other thresholding methods we have to choose a threshold value to binarize the image but Otsu's Thresholding determines the value automatically.
4. Apply Closing Morphological Transformation on thresholded image. Closing is useful to fill small black regions between white regions in a thresholded image. It reveals the rectangular white box of license plate
5. To detect the plate we need to find contours in the image. It is important to binarize and morph the image before finding contours so that it can find more relevant and less number of contours in the image.
6. Now find the minimum area rectangle enclosed by each of the contour and validate their side ratios and area. We have defined the minimum and maximum area of the plate as 4500 and 30000 respectively.
7. Now find the contours in the validated region and validate the side ratios and area of the bounding rectangle of the largest contour in that region. After validating you will get a perfect contour of a license plate. Now extract that contour from the original image. You will get the image of plate.

This step is performed by clean_plate and ratioCheck method of class PlateFinder

```
defclean_plate(self, plate):
gray = cv2.cvtColor(plate, cv2.COLOR_BGR2GRAY)
thresh = cv2.adaptiveThreshold(gray, 255,

  cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
        cv2.THRESH_BINARY, 11, 2)
  _, contours, _ = cv2.findContours(thresh.copy(),
            cv2.RETR_EXTERNAL,
            cv2.CHAIN_APPROX_NONE)
  if contours:
    areas = [cv2.contourArea(c) for c in contours]
    # index of the largest contour in the
    # areas array
    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 self.ratioCheck(max_cntArea,
              plate.shape[1],
              plate.shape[0]):
      return plate, False, None
    return plate, True, [x, y, w, h]
    else:
    return plate, False, None
def ratioCheck(self, area, width, height):
  min = self.min_area
  max = self.max_area
  ratioMin = 3
  ratioMax = 6
```

```
ratio = float(width) / float(height)
    if ratio < 1:
    ratio = 1 / ratio
if (area < min or area > max) or (ratio <ratioMin or ratio >ratioMax):
    return False
    return True
```

8. To recognize the characters on license plate precisely, we have to apply image segmentation. For that first step is to extract the value channel from the HSV format of the plate's image. It would look like.
9. Now apply adaptive thresholding on the plate's value channel image to binarize it and reveal the characters. The image of plate can have different lightning conditions in different areas, in that case adaptive thresholding can be more suitable to binarize because it uses different threshold values for different regions based on the brightness of the pixels in the region around it.
10. After binarizing apply bitwise not operation on the image to find the connected components in the image so that we can extract character candidates.
11. Construct a mask to display all the character components and then find contours in mask. After extracting the contours take the largest one, find its bounding rectangle and validate side ratios.
12. After validating the side ratios find the convex hull of the contour ad draw it on the character candidate mask. The mask would look like-
13. Now find all the contours in the character candidate mask and extract those contour areas from the plate's value thresholded image, you will get all the characters separately.

Steps **8** to **13** are performed by **segment_chars** function that you can find below in the full source code. The driver code for the functions used in steps 6 to 13 is written in the method **check_plate** of class **PlateFinder**.

The proposed methodology consists of four major phases: pre-processing, detection, recognition and searching as shown in figure below.

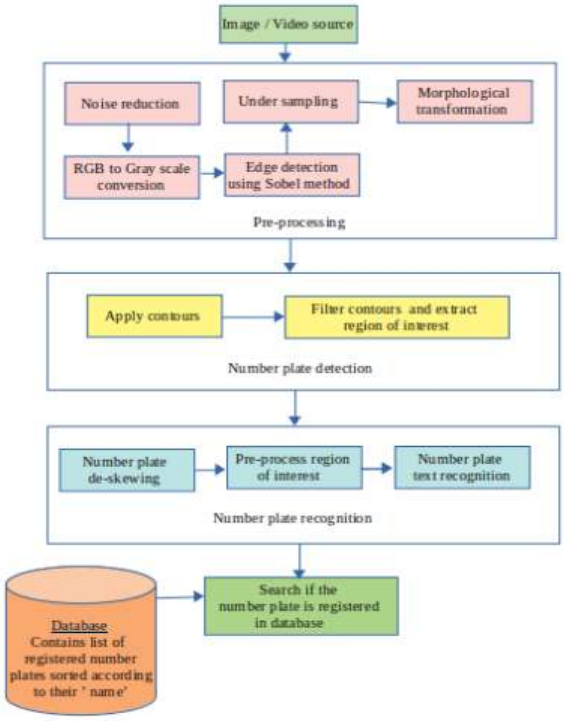

Figure 1: Proposed Architecture

IV. BRIEF DESCRIPTION OF STEPS

Step 1: Image pre-processing

Step 1.1: Noise reduction: The objective of Gaussian filtering/ Gaussian smoothing is to reduce noise and detail. This will serve well for further image processing steps. For an Image, mathematically, gaussian filter can be expressed as:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

The input image is made to convolve with this 2-D 'G' matrix to obtain a smoothened image. In OpenCV, Gaussian smoothing can be applied using the following function:cv2. GavssianBlur (image, (5,5), 0) , where (5,5) refers to the filter size and '0' indicates the model to find the value of standard deviation (sigma) itself.

Step 1.2: RGB to Grayscale conversion: Converting RGB image to grayscale saves a lot of time since we have to perform convolution of the image with sobel filter over only one 2D matrix rather than RGB image having 3 channels and making it complicated. Another reason is, in case of image edge detection we are focussed on observing

the intensity change and it is easier to analyse it in a gray-scaled image.

$$\frac{\partial f}{\partial x} = S_x \otimes f \qquad \frac{\partial f}{\partial y} = S_y \otimes f$$

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

Step 1.3: Edge detection using sobel method: Sobel edge detection works by calculating the gradient of image intensity at each pixel within the image. It finds the direction of the largest increase from light to dark and the rate of change in that direction. The following terms are used to perform edge detection using Sobel method:

In OpenCV, cv2.Sobel (img2 , cv2.CV_8U, 1,0, ksize = 3) is used to perform edge detection using kernel size of 3.

Step 1.4: Under-sampling: The Number plate detection and recognition algorithm are supposed to work at a steady and consistent frame rate. Unsurprisingly, for high-resolution images, image processing algorithms tend to work slow. It is in fact unnecessary to consider images with such a high resolution. This stage reduces the resolution if it crosses a predefined threshold.

Step 1.5: Morphological transformation: Top-hat and Black-hat filters are part of Morphological transformations . The Top-hat operation is used to enhance bright objects of interest in a relatively dark background, while the black-hat operation (also known as bottom-hat) is used to enhance dark objects of interest in a relatively bright background. In this work, top-hat results are added to the original image and black-hat results are subtracted from it.

Step 2: Number plate detection

Step 2.1: Apply Counters: Contour Tracing, also called as Border following is the algorithm used for generating Contours. A contour is a link of equal intensity points along the boundary. In OpenCV, finding contours is like finding a white object from the black background, therefore during the Adaptive Gaussian Thresholding stage, Inversion operation has to be applied.

Step 2.2: Filter Contours and extract region of interest: For small regions, especially sharp edges and noise outliers, contours are applied. A human eye can easily figure out that such contours are unnecessary, but this must be incorporated into a program. Initially, Bounding boxes were applied to each contour. Then, for each contour, the following factors were considered such as minimum contour area, minimum contour width and height, minimum and maximum possible aspect ratios. This resulted in the filtering of most of the unnecessary contours, propelling us near to our objective, ie, Detect number plate.

Step 3: Number plate recognition

Step 3.1: Number plate de-skewing: Skew is the amount of rotation necessary to return an image to horizontal and vertical alignment. Skew is measured in degrees. Deskewing is a process whereby skew is removed by rotating an image by the same amount as its skew but in the opposite direction. This results in a horizontally and vertically aligned image where the text runs across the page rather than at an angle. In our project, this step is done using ratio_and_rotation()

Step 3.2: Pre-process region of interest: It is possible that two or more contours may completely overlap with each other, as in the case with the number ‘zero’. The inner contour, if detected in the contour process, may lie completely inside its outer contour. Due to this phenomenon, both contours may get recognized as separate characters during the recognition process. If needed, we also resize the image before doing the recognition step.

Step 3.3:Number plate text recognition: Python-tesseract is an optical character recognition (OCR) tool for python. That is, it will recognize and “read” the text embedded in images. We have used this tool finally to obtain the text present in the filtered, de-skewed contour.

Step 4: Searching unknown image Step 4.1: Create database: Using Step-1,2 and 3, register all the vehicles in the dataset and store them in a database after removing other special characters. Step 4.2: Sorting: To make the final stage of searching more efficient, we are performing sorting operations on the detected texts. This is done using a quick sort algorithm. Quick sort is a divide and conquer algorithm. It is not stable and does in-place sorting.

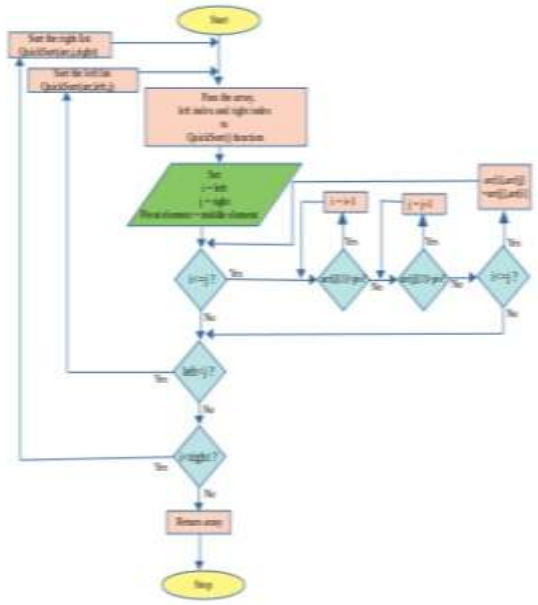

Figure 2: Flowchart

Step 4.3: Searching : Pass a new image and follow steps 1,2,3. Obtain the new vehicle's registration number and check if it is present in the database using Binary search method. Binary search is another simple divide and conquer algorithm that is performed on a sorted array/list. It works better than linear search in case of more images in the dataset.

Binary search: Flowchart:

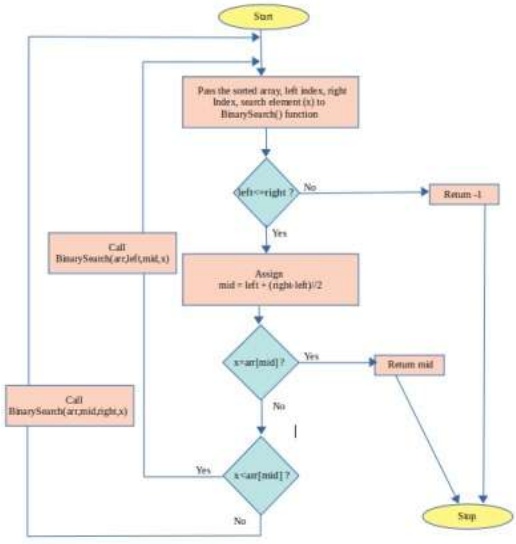

Figure 3: Binary search Flowchart

V. EXPERIMENTAL RESULTS

VI. CONCLUSION

Since we didn't use complex machine learning and deep learning algorithms there are some drawbacks of this project but it will work efficiently if implemented in housing society/ apartment/ institution to allow resi-dent's vehicles inside and almost all the challenges we faced while solving the problem are resolved to a good extent.

REFERENCES

[1] M M Shidore, and S P Narote. (2011) "Number Plate Recognition for Indian Vehicles" International Journal of Computer Science and Network Security 11 (2): 143-146.
[2] Sang Kyoon Kim, D. W. Kim and Hang Joon Kim. (1996) "A recognition of vehicle license plate using a genetic algorithm based segmentation," Proceedings of 3rd IEEE International Conference on Image Processing, Lausanne.
[3] https://docs.opencv.org/master/.
[4] https://github.com/anuj-badhwar/Indian-Number-Plate-Recognition-System.