Computer Vision

Project Report

Automatic Number Plate Recognition for Indian vehicles

Submitted by:

Aditya Pawar (23PGAI0081)

Punit Drolia (23PGAI0113)

Harshada Jhadav (23PGAI0101)

Nanda Kishore (23PGAI0111)

Shubham Gupta (23PGAI0123)

Table of contents:

Abstract:

Number Plate recognition, also called License Plate realization or recognition using image processing methods is a potential research area in smart cities and the Internet of Things. An exponential increase in the number of vehicles necessitates the use of automated systems to maintain vehicle information for various purposes. In the proposed algorithm 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 several image processing techniques such as, morphological transformation, Gaussian smoothing, Gaussian thresholding and Sobel edge detection method in the pre-processing stage, afterwhich number plate segmentation, contours are applied by border following and contours are filtered based on character dimensions 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.

Requirements:

Dataset: Images of Indian vehicles where the number plate is clearly visible.

Operating System: Windows 10

Programming language: Python3

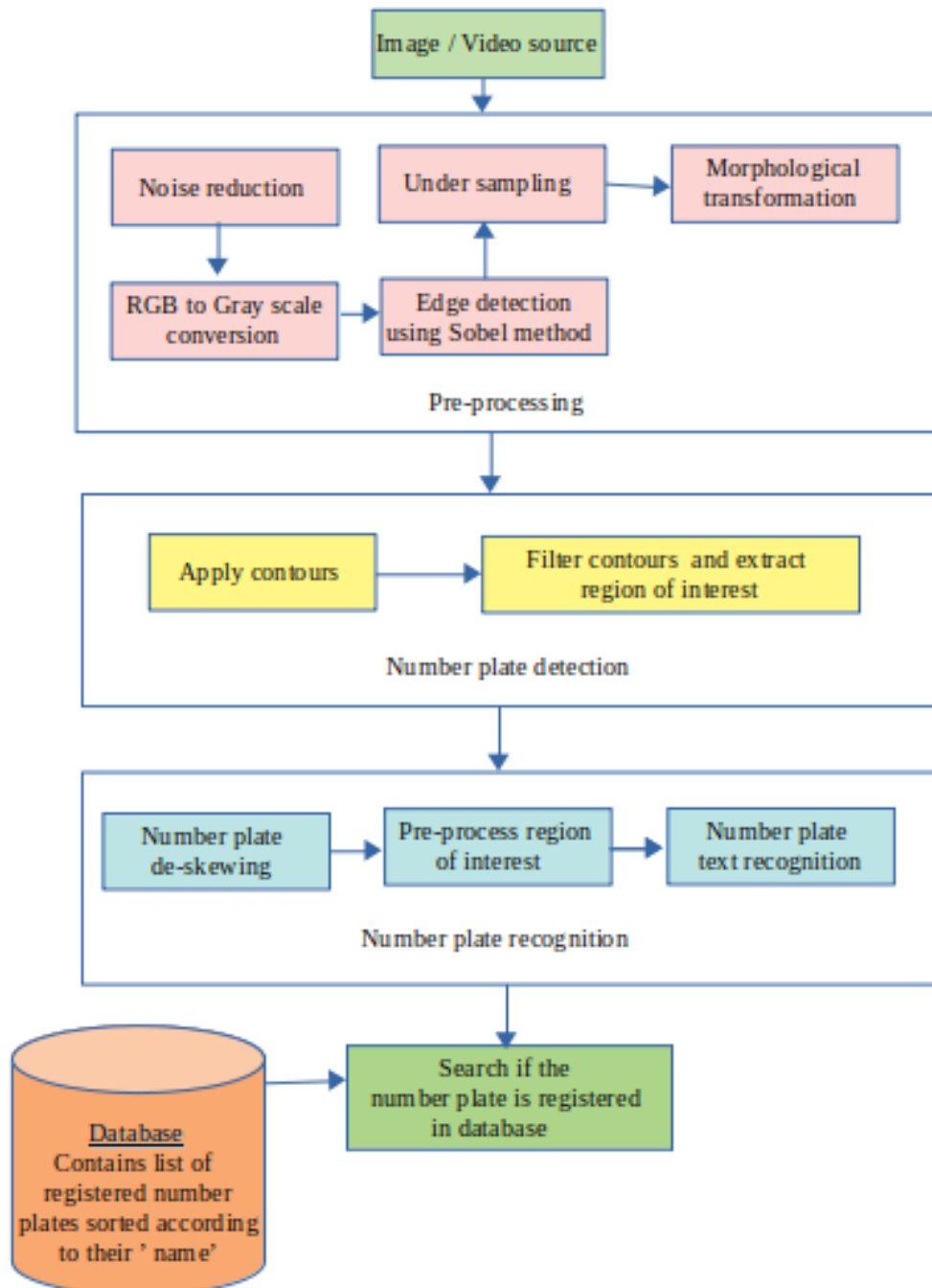**Libraries used:** sys, glob, os, glob, numpy, cv2, PIL, pytesseract, re

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, "lost ticket" fraud, fraud by changing tickets are some areas where ANPR is helpful .
3. **Access Control:** License 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. 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, reduces required manpower to process events of exceptions and these can be implemented using ANPR.
5. **Journey Time Measurement:** Data collected by license 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.

Methodology:

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

```
                        ┌──────────────────────┐
                        │  Image / Video source│
                        └──────────┬───────────┘
                                   │
    ┌──────────────────────────────┼─────────────────────────────────┐
    │                              ▼                                  │
    │  ┌─────────────────┐   ┌──────────────┐    ┌──────────────────┐ │
    │  │ Noise reduction │   │ Under sampling│──▶│  Morphological    │ │
    │  └────────┬────────┘   └──────▲───────┘    │  transformation   │ │
    │           │                   │            └──────────────────┘ │
    │           ▼                   │                                 │
    │  ┌─────────────────┐   ┌──────────────────┐                     │
    │  │ RGB to Gray scale│──▶│ Edge detection   │                    │
    │  │ conversion       │   │ using Sobel method│                   │
    │  └─────────────────┘   └──────────────────┘                     │
    │                                                                 │
    │                      Pre-processing                             │
    └──────────────────────────────┬─────────────────────────────────┘
                                   ▼
    ┌──────────────────────────────────────────────────────────────┐
    │  ┌─────────────────┐        ┌───────────────────────────┐     │
    │  │ Apply contours  │──────▶│ Filter contours  and extract│    │
    │  └─────────────────┘        │ region of interest         │    │
    │                             └───────────────────────────┘     │
    │                  Number plate detection                       │
    └──────────────────────────────┬───────────────────────────────┘
                                   ▼
    ┌──────────────────────────────────────────────────────────────┐
    │ ┌──────────────┐   ┌──────────────┐   ┌──────────────┐         │
    │ │ Number plate │──▶│ Pre-process  │──▶│ Number plate │        │
    │ │ de-skewing   │   │ region of    │   │ text         │        │
    │ │              │   │ interest     │   │ recognition  │        │
    │ └──────────────┘   └──────────────┘   └──────────────┘         │
    │                 Number plate recognition                      │
    └──────────────────────────────┬───────────────────────────────┘
                                   ▼
   ┌──────────┐            ┌──────────────────────────┐
   │ Database │───────────▶│  Search if the           │
   │ Contains │            │  number plate is registered│
   │ list of  │            │  in database             │
   │ ...      │            └──────────────────────────┘
   └──────────┘
```

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: $cv2.\,GaussianBlur(image, (5,5), 0)$, 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.

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:

Run filter over image

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

Image gradient

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

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.

(New image)   (Step 1.1)   (Step 1.2)

(Step 1.3)   (Step 1.4)   (Step 1.5)

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 2)

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.

```
Number plate...
KL 26 H 5009
Text :   KL 26 H5009
```

(Step 3)

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.

Quick sort:

Flowchart:
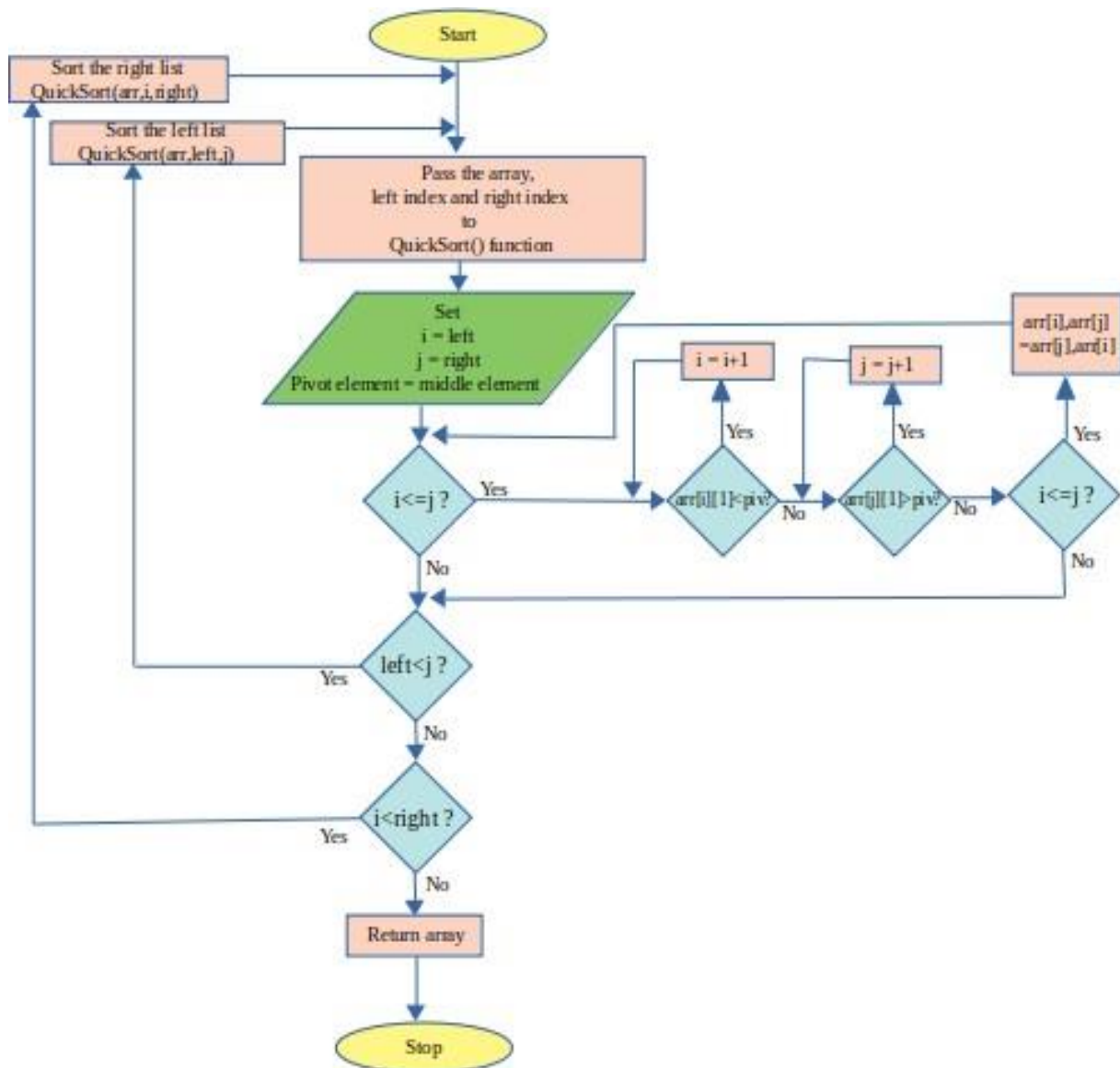
Time complexity:
Best case: O(n log(n))
Average case: O(n log(n))
**Worst case:** O(n^2)

Space complexity:
Best case: O(log(n))
Worst case: O(n)

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:
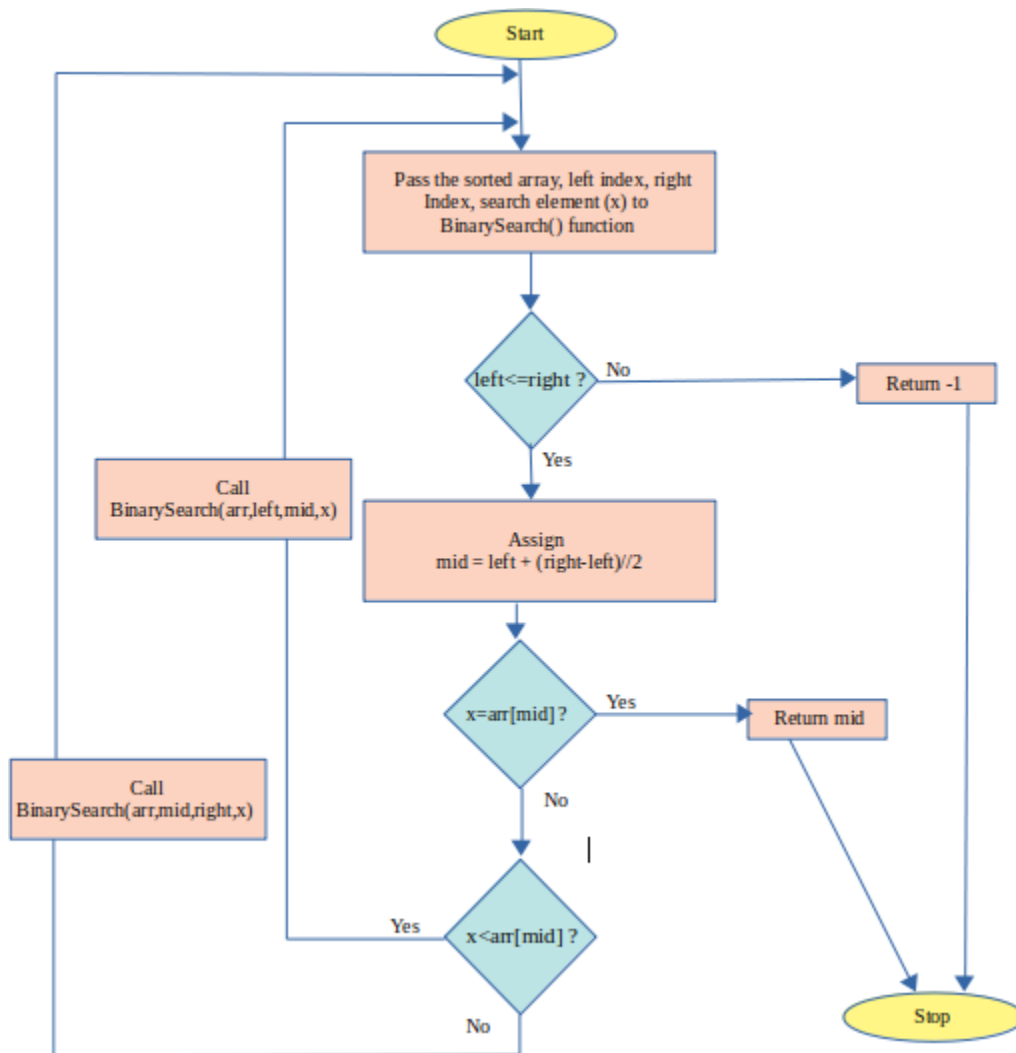
Time complexity:

It takes O(log(n)) time complexity, which when compared with linear search is much better, especially when the array size gets larger.

Space complexity:
O(1)

(Detected number plate)

(Sorted number plates)

(New image)

(Searched - output)

Challenges and solution:

1. **Dealing with bright and dark objects.**

   Solution: Many details present in an RGB image become less salient when we convert it into grayscale. To visualize the change in intensity and thus to deal with bright and dark objects in the image, converting it to grayscale works best.

2. **Dealing with noisy images.**

   Solution: We have convolved the image with a Gaussian filter in the pre-processing stage to reduce the noise present in the image.

3. **Dealing with cross-angled or skewed number plates.**

   Solution: De-skewing, i.e. rotating the image to the required position can be done to detect the text present on the number plate.

4. **Dealing with non-standard number plates.**

   Solution:The code we implemented won't give any results in case of non-standard/ partially torn number plates. Hence, they won't be stored in the database and searching is not possible on it.

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 resident's vehicles inside and almost all the challenges we faced while solving the problem are resolved to a good extent.