VEHICLE LICENCE PLATE RECOGNITION

Issac Koshy Panicker - 50249131
Pretty Mary Philip - 50247311
Vineesh Ravindran - 50249458

CSE 573 Computer Vision and Image Processing
Fall 2017 Final Project

Abstract

Number plate recognition system retrieves the registration number of a vehicle when provide with an image of the vehicle's rear view. In the system the number plate is extracted out from the image which is then provided to a character recognition system to predict the characters present in the image. The project is implemented using MATLAB.

1. Introduction

License plate recognition (LPR) plays an important role in numerous real-life applications, such as automatic toll collection, traffic law enforcement, parking lot access control, and road traffic monitoring. LPR recognizes a vehicles license plate number from an image or images taken by either a color, black and white camera. It is fulfilled by the combination of a lot of techniques, such as object detection, image processing, and pattern recognition. LPR is also known as automatic vehicle identification, car plate recognition, automatic number plate recognition, and optical character recognition (OCR) for cars. The variations of the plate types or environments cause challenges in the detection and recognition of license plates. The LPR system that extracts a license plate number from a given image can be composed of three stages.
1. License Plate Extraction
2. Character Segmentation
3. Optical Character Recognition

2. License Plate Extraction

The License Plate Extraction module of the algorithm is used to find one or more probable sub section of the original input image, one of which is supposed to contain the number plate region. All of these regions are then fed into the Optical Character Recognition Module for character analysis and letter extraction. Therefore, it is necessary that the LPE module finds the number plate region successfully in the first place. Over the years, a lot of algorithms are developed to detect the license plate area of an image each with their own advantages and disadvantages. In this algorithm, we have used an approach that is both easy to implement and has a high success rate. The algorithm calculates the sum of differences of gray values between neighboring pixels of an image, column wise and row-wise. After that, only the regions with larger than average sum-of-difference values are kept intact, while all the other areas are converted into black pixels thus segmenting the input image into probable candidate region.

The steps of implementing License Plate Detection algorithm in MATLAB are described below.

2.1. Converting a Colored Image into Gray Image

The algorithm described here is independent of the type of colors in image and relies mainly on the gray level of an image for processing and extracting the required information. Color components like Red, Green and Blue value are not used throughout this algorithm. So, if the input image is a colored image represented by 3-dimensional array in MATLAB, it is converted to a 2- dimensional gray image before further processing. Also, the contrast of the image is further increased, as the algorithm depends on the difference of pixel values of light and dark area. The sample of original input image is shown below:

2.2. Dilating the Image

Dilation is a process of improvising given image by filling holes in an image, sharpen the edges of objects in an image, and join the broken lines and increase the brightness of an image. Using dilation, the noise with-in an image can also be removed. By making the edges sharper, the difference of gray value between neighboring pixels at the edge of an object can be increased. This enhances the edge detection. In Number Plate Detection, the image of a car plate may not always contain the same brightness and

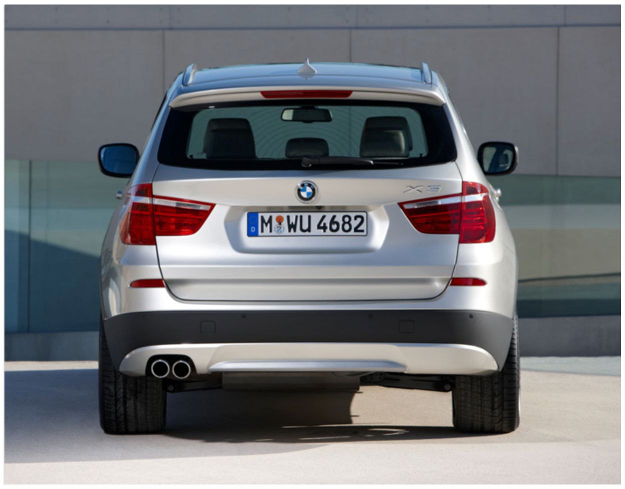


Figure 1. Original Image.

shades. Therefore, the given image has to be converted from RGB to gray form. However, during this conversion, certain important parameters like difference in color, lighter edges of object, etc. may get lost. The process of dilation will help to nullify such losses.

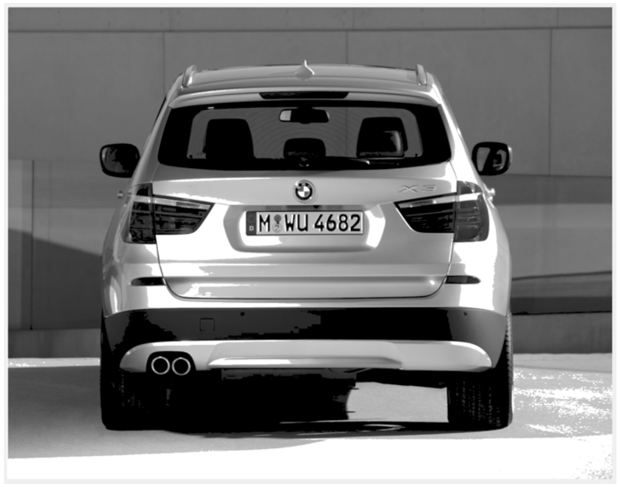


Figure 2. Dilated Image.

2.3. Horizontal and Vertical Edge Processing of an Image

Histogram is a graph representing the values of a variable quantity over a given range. In this Number Plate Detection algorithm, the writer has used horizontal and vertical histogram, which represents the column-wise and row wise histogram respectively. These histograms represent the sum of differences of gray values between neighboring pixels of an image, column-wise and row wise. In the above step, first the horizontal histogram is calculated. To find a horizontal histogram, the algorithm traverses through each column of an image. In each column, the algorithm starts with the second pixel from the top. The difference between second and first pixel is calculated. If the difference exceeds certain threshold, it is added to total sum of differences.

Then, algorithm will move downwards to calculate the difference between the third and second pixels. So on, it moves until the end of a column and calculate the total sum of differences between neighboring pixels. At the end, an array containing the column-wise sum is created. The same process is carried out to find the vertical histogram. In this case, rows are processed instead of columns.

2.4. Passing Histograms through a Low Pass Digital Filter

Referring to the figures shown below, one can see that the histogram values changes drastically between consecutive columns and rows. Therefore, to prevent loss of important information in upcoming steps, it is advisable to smooth out such drastic changes in values of histogram. For the same, the histogram is passed through a low-pass digital filter. While performing this step, each histogram value is averaged out considering the values on it right-hand side and left-hand side. This step is performed on both the horizontal histogram as well as the vertical histogram. Below are the figures showing the histogram before passing through a low-pass digital filter and after passing through a low pass digital filter.

2.5. Filtering out Unwanted Regions in an Image

Once the histograms are passed through a low-pass digital filter, a filter is applied to remove unwanted areas from an image. In this case, the unwanted areas are the rows and columns with low histogram values. A low histogram value indicates that the part of image contains very little variations among neighboring pixels. Since a region with a license plate contains a plain background with alphanumeric characters in it, the difference in the neighboring pixels, especially at the edges of characters and number plate, will be very high. This results in a high histogram value for such part of an image. Therefore, a region with probable license plate has a high horizontal and vertical histogram values. Areas with less value are thus not required anymore. Such areas are removed from an image by applying a dynamic threshold. In this algorithm, the dynamic threshold is equal to the average value of a histogram. Both horizontal and vertical histograms are passed through a filter with this dynamic threshold. The output of this process is histogram showing regions having high probability of containing a number plate. The filtered histograms are shown below:

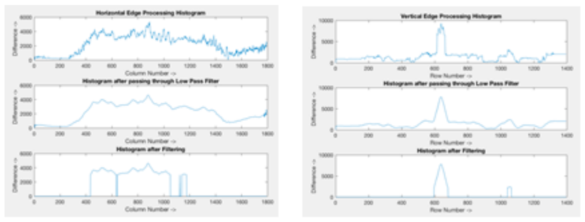


Figure 3. Generated histograms.

2.6. Segmentation

The next step is to find all the regions in an image that has high probability of containing a license plate. Co-ordinates of all such probable regions are stored in an array. The output image displaying the probable license plate regions is shown below. The images contained vehicles of different colors and varying intensity of light. With all such images, the algorithm correctly recognized the number plate.

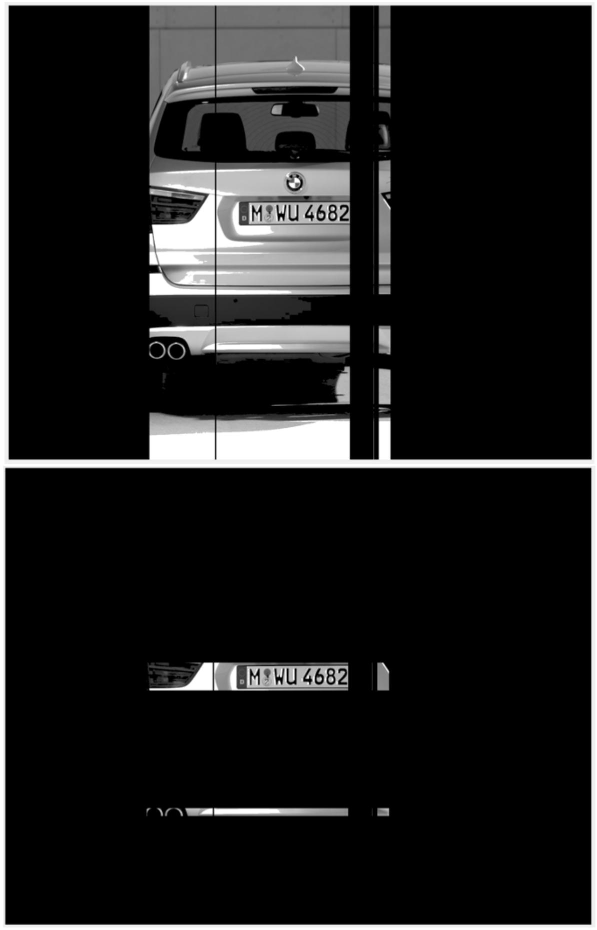


Figure 4. Segmenting the car image.

2.7. Cropping

From the segmented image, we crop the image having the number plate section and is saved as plate.jpg into the system.

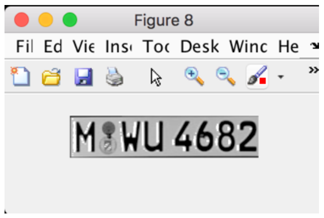


Figure 5. Extracted number plate

2.8. Limitations

However, this output depends on the properties of the camera and background subject to the car. For a speed detection camera when both of these conditions are fixed, this algorithm will provide maximum output. We need to change some parameters based on the camera and background.

3. Character Segmentation

Once we receive the extracted number plate, we will have to extract out the characters involved in the plate. As the number plates usually contain patterns other than the registration number characters we have to ensure that only relevant characters are extracted out before moving to character recognition.

3.1. Detecting connected components

We utilize the pixel connectivity of the number plate characters to separate out the registration number characters from other patterns. Using the image that we receive from the number plate extraction part we first use the MATLAB method bwareaopen to remove all unnecessary patterns. This ensures that only the components relevant to the character recognition part remain in the image.

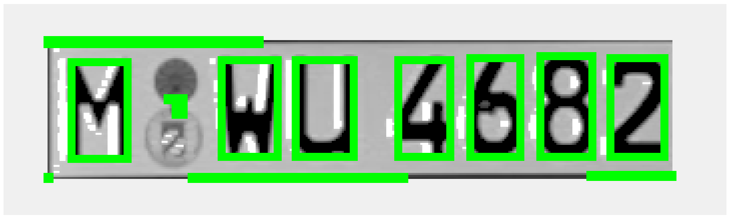


Figure 6. Segmented characters

3.2. Component Labeling

The next step would be to label the generated components in the image. The labeling helps us in separating out each character from the image. The cropped out characters are stored in a cell array and provided to the character recognition part. Each cropped portion is resized so that its dimensions match the templates that we would use in the character recognition part.

3.3. Limitations

The major limitation that we came across in this approach was the fact that components other than the registration number characters get detected during segmentation even after changing the intensity threshold to higher values.

4. Character Recognition

Optical Character Recognition is the conversion of images which are types, handwritten or printed text. It is a common method of digitizing printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as cognitive computing, machine translation, (extracted) text-to-speech, key data and text mining. We utilize optical character recognition to identify the characters involved in the number plate.

4.1. Template matching

Template matching involves comparing an image to a stored glyph on a pixel-by-pixel basis. It is the process of finding the location of a sub image called a template inside an image. This relies on the input glyph being correctly isolated from the rest of the image, and on the stored glyph being in a similar font and at the same scale. This technique works best with typewritten text and does not work well when new fonts are encountered. Since License plates are mostly typed-text we have implemented the OCR using template matching approach. We first read the provided templates and their corresponding labels into different arrays. In our case the label names are same as the file names of the given templates. These templates are then compared with each segmented character.

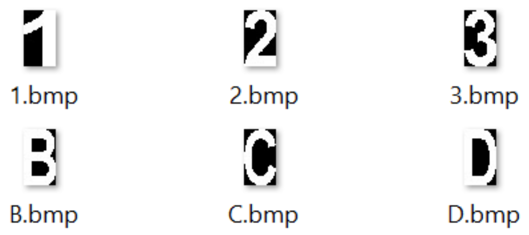


Figure 7. Template Samples

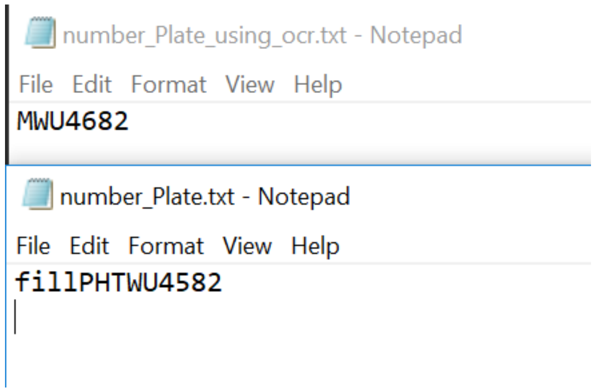
The template that is the most similar to the character is recognized as the target. Each template scans the character column by column to calculate the normalized cross correlation. The label corresponding to the template is set as the value of the recognized character. If a character is different from the template due to any font change, rotation, or noise, the template matching produces incorrect recognition. The problem of recognizing tilted characters is solved by storing several templates of the same character with different inclination angles.

4.2. Limitations

Template matching method depends highly on the templates that we use for comparison and as a result the extent of character recognition could get limited if the font used in number plates varies highly from that used in the templates.

4.3. Using OCR function

Computer Vision System Toolbox provides an OCR function which provided excellent results when fed with the same extracted image. The function was able to distinguish between the characters and other patterns involved in the image and generated retrieved the words involved in the image.


number_Plate_using_ocr.txt - Notepad
File Edit Format View Help
number_Plate.txt - Notepad
File Edit Format View Help

```
MWU4682
```

```
fillPHTWU4582
```

Figure 8. Comparison between results generated using the toolbox OCR and template matching

5. Conclusion

While the number plate extraction and character segmentation worked almost perfectly, the character recognition using template matching method had limitations. The system recognized most of the characters well but had limitations identifying characters such as 'M' an 'W', often mistaking them to 'H'. The recognition of the numbers involved was better than that of the alphabetic characters. On the other hand, the results generated by the OCR function of the image processing toolbox were excellent. The function recognized most of the extracted number plates that were provided to it with good accuracy.

6. References

[1] Sheetal Mithun Kawade , M.M.Mukhedkar."A Real Time Vehicles License Plate Recognition System". International Journal of Science and Engineering. Volume 1, Number 2 - 2013, PP-41-48 IJSE.

[2] R. T. Lee, K. C. Hung, and H. S. Wang,Real Time Vehicle License Plate Recognition Based on 2D Haar Discrete Wavelet Transform, International Journal of Scientific & Engineering Research, Volume 3, Issue 4, ISSN 2229-5518, April-2012.

[3] Hamed Saghaei, Proposal for Automatic License and Number Plate Recognition System for Vehicle Identification, 2016 1st International Conference on New Research Achievements in Electrical and Computer Engineering

[4] Muhammad H Dashtban, Zahra Dashtban, Hassan Bevrani, A Novel Approach for Vehicle License Plate Localization and Recognition, International Journal of Computer Applications (0975 8887), Volume 26 No.11, July 2011.

[5] M.-S. Pan, J.-B. Yan, and Z.-H. Xiao, Vehicle license plate character segmentation, Int. J. Automat. Comput., vol. 5, no. 4, pp. 425432,2008