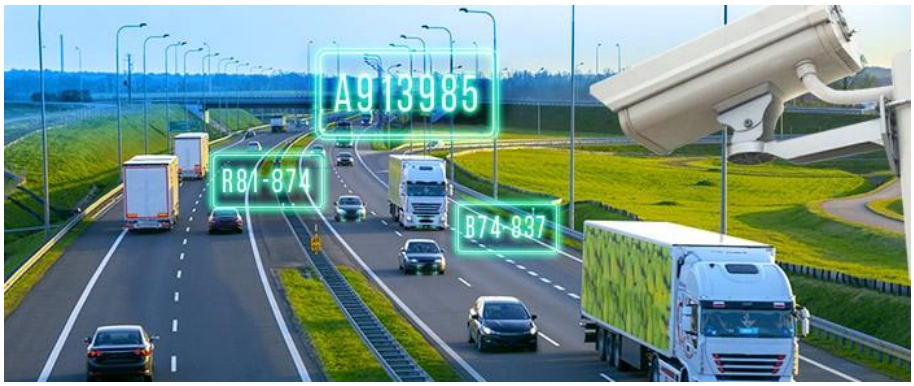
PROJECT REPORT

APRIL 18, 2020



LICENSE PLATE DETECTION

COMP 4102

PRESENTED BY: Sahil Kapal - 101092630

Nicolas Mak-Fan – 100964074

Rayhane Bournas – 101042914

Umar Farooq – 101094104

CONTENTS

Tables

Figures

1. ABSTRACT

Taking pictures of moving vehicles can cause the entire image to be blurry. Obtaining the license plate information from these pictures can be challenging because the entire image needs to be unblurred. This report proposes a method of implementing computer vision components to detect where in the blurry image the license plate is located and only focusing on unblurring this portion of the image. To successfully unblur the image, the method will first detect edges that correspond to the license plate of the vehicle. Then, the image of the license plate will be unblurred. Lastly, Optical Character Recognition (OCR) algorithm is used to detect the characters of the license plate. As a result, license plate detection achieved a success rate of 80%.

2. INTRODUCTION

The technological advancement in the past decade has led to the increase in research and study in the field of Computer vision. Computer vision studies ways computers can gain perfection in vision and interpreting visual images the same way as a human or animal. Many real-world applications of computer vision are being used in today's time, for example, from medical imaging to machine learning and fingerprint recognition. Among these applications, advancements are also being made in license plate detection.

License plate detection is a technology that can read and store data of the license plate of a car using optical character recognition (OCR). This application is necessary to detect and classify plate numbers of vehicles involved in accidents, law breaking, and theft. However, this technology faces many problematic challenges such as extreme weather conditions causing blurriness, vehicles driving too fast, other conditions like too much sun light or darkness during the nighttime, as well as camera color issues which make it difficult to develop this technology into a robust system.

There have been many studies proposing automated license plate detection methods such as 1) dynamic programming-based method to isolate letter from an image, 2) Hough transform to detect lines in binary images, 3) AdaBoost which involves machine learning and many more. However, all these methods add a degree of complexity to developing a universal detection system.

This report proposes a robust system to extract license plate number through three stage process as described in **Figure 1**. At first, the system will detect where in blurry image the license plate is located and then the system will deal with unblurring this portion of the image. At last, Optical Character Recognition (OCR) algorithm is performed to detect the characters of the license plate.

FIGURE 1: LICENSE PLATE DETECTION ALGORITHM STEPS

3. BACKGROUND

Previously, many studies have been done on license plate detection which focuses mainly on providing robust techniques to solve certain problems but not all. The previous studies are as follow:

1) Dynamic programming-based method to isolate letters directly. This method applies a range of threshold to identify image regions with similar properties. However, this method is only robust to detecting license plates that have poor lighting (J., 2009).
2) Identification of license plate using Hough transform. This method detects lines in binary images by searching sets of parallel lines that matches the rectangular shape. However, the algorithm is computationally intensive for high resolution images (T., 2005).
3) Use of Gabor transform to analyze textures in all directions and different scales and orientations of the license plate or text based on the convolution of the images. This method is robust to orientation and scaling, but the algorithm is computationally intensive and difficult to implement (Kahramen F., 2003).
4) AdaBoost, which is an innovative approach involving elements of machine learning. This method used a training dataset of different license plate images. However, this algorithm is difficult to implement in real-time (L., 2004).

4. APPROACH

This section provides the algorithmic steps that were taken in order to successfully detect, extract, filter and recognize license plate numbers.

4.1 LICENSE PLATE DETECTION

This approach detects where in the blurry image the license plate is located. Following are the algorithm steps:

1. Retrieve the image and convert into a grayscale.
2. Perform a bilateral Filter on the grayscaled image which reduces the noise while preserving edges. This will help in future steps for edge detection.
3. Apply a gaussian filter to the image by convolving a gaussian kernel to the product of step 2.
4. Once the blur has been applied, pass the image into a threshold function. The product of this function will return an image with all the edges, ultimately prepping for the location of the license plate by contouring the right shape.
5. Dilate step. 4 which fills the unnecessary areas of the image with white, allowing the contouring to be an easier task.
6. Erode step 5. Which keeps the foreground of the image, white, which makes the plate easier to spot out.
7. Locate the contours of the image by passing in step 6. In the findContours built-in openCV function. The parameters will define how the contour is found which pays no attention to hierarchy and draws the contour with full lines.
8. A loop is performed to scan the image and if 4 points are found, the application assumes the contour is found.
9. Once the license plate contour is found, then the next step is to perform another loop over the image with the spotted contour and mark circles on each vertex of the license plate.
10. Extract the coordinates of each vertex (x,y) to be passed onto the cropping portion of the algorithm.

4.2 LICENSE PLATE EXTRACTION

After the license plate has been detected, the next step is to extract it out of the full image. In no order, coordinates of the top left, top right, bottom left and bottom right of the license plate are

passed from the previous step. These coordinates contain the x and y values of each location. In order to extract the license plate from the image, we need to determine which coordinate corresponds to which corner of the license plate. We can then save these values in a dictionary with “top_left”, “bottom_left”, “top_right”, and “bottom_right” as the keys. The algorithm to detect these corners is as follows:

1. Create minX and minY variables to save the minimum values. The minX value will correspond to the left side of the plate, and the minY values will correspond to the top of the plate.
2. Loop over the given coordinates
    a. If the current coordinates x value is lower than minX, replace minX with current x value
    b. If the current coordinates y value is lower than minY, replace minY with current y value
3. Create maxX and maxY variables to save the minimum values. The maxX value will correspond to the left side of the plate, and the maxY values will correspond to the top of the plate.
4. Loop over the given coordinates
    a. If the current coordinates x value is higher than maxX, replace maxX with the current x value
    b. If the current coordinates y value is higher than maxY, replace maxY with the current y value
5. We can save these values as follows:

```
coords = {
    "top_left": [minX, minY],
    "bottom_left": [maxX, minY],
    "top_right": [minX, maxY],
    "bottom_right": [maxX, maxY]
}
```

6. Now that we have the correct positions, we can extract the license plate.

4.3 LICENSE PLATE FILTERING AND CHARACTER RECOGNISTION

Once the license plate has been cropped, the first step will be to resize the image, so that we can have a bigger picture of the plate to make it easier to be read by the OCR. After the image is resized, we need to filter the image to get a new refined license plate. In the process, we call on multiple different open cv functions to blur the image. After the image is blurred, we then filter the image so that we can receive a new sharpened image of the license plate. Finally, we pass in this image to the OCR so that the image can be read by the system. After receiving the text back from the system, some testing will be done. This testing is done so that we can remove all the unnecessary characters that the OCR could have picked up while trying to process the image. Once all of that is done, we now have the clean text of the plate, which gets returned to the user.

5. RESULTS

5.1 LICENSE PLATE DETECTION RESULTS

Original image is presented on the left side of **Figure 2**, whereas the right side of the image shows the location of the contours. The result was achieved by performing a bilateral filter on the grayscale to reduce noise, then convolving the image with Gaussian Kernel to produce a blurred image. After this, the image was dilated and eroded, and then the contours on the license plate were located. Lastly, circles were marked on each vertex of the license plate, which can be seen on the right side of **Figure 2**, highlighted in green.

FIGURE 2: LICENSE PLATE CONTOUR LOCATION

5.2 LICENSE PLATE EXTRACTION RESULTS

Figure 3 provides the list of license plates that were successfully spotted on the contoured image from **Section 5.1.** The result in **Figure 3** was achieved through finding the minimum area enclosing the input 2D point set with the located contours in **Figure 2**. Algorithm was then performed, as mentioned in **Section 4.2**, to detect the corners of the license plate, which is shown with blue circles in the figure below. Lastly, corners were extracted from the image.

FIGURE 3: LIST OF SUCCESSFULLY EXTRACTED LICENSE PLATE NUMBERS

5.3 LICENSE PLATE FILTERING AND CHARACTER RECOGNISTION RESULTS

Cropped image from **Section 5.2** was used and python character reading library was applied to successfully read the characters of the license plate. The result of the procedure mention in **Section 4.3** resulted in the following figure:

FIGURE 4: OPTICAL CHARACTER RECOGNITION ALGORITHM RESULT

5.4 SUCCESS RATE

The algorithm that was developed has an average success rate of about 80%. This rate was achieved with the vigorous testing that was done and with certain challenges that we had faced. Since there was no actual function that can be done that can truly unblur an image, the only way to get it unblurred is by neural enhancement, which is out of scope of the course, hence were restricted to slightly blurry plates. There were also other challenges where, if the image was out of focus, the odds of finding the contour points were very low due to the quality of the picture. With the pictures that we were able to find the contours, we were very successful in getting all the data needed to display the license plate to the user.

6. APPENDIX

6.1 LIST OF WORK

TABLE 1: PROJECT WORK SCHEDULE

Dates	Rayhane Bournas	Sahil Kapal	Nick Mak-Fan	Umar Farooq
February 1	Research methods to accomplish project	Research methods to accomplish project	Research methods to accomplish project	Research methods to accomplish project
February 8	Determine Non-Functional Requirements	Determine Non-Functional Requirements	Determine Non-Functional Requirements	Determine Non-Functional Requirements
February 15	Overview of what different classes will contain	Overview of what different classes will contain	Overview of what different classes will contain	Overview of what different classes will contain
February 22	Work on the skeleton code	Work on the skeleton code	Work on the skeleton code	Work on the skeleton code
February 29	Work on implementation concepts	Work on implementation concepts	Work on implementation concepts	Work on implementation concepts
March 7	Work on implementation concepts	Work on implementation concepts	Work on implementation concepts	Work on implementation concepts
March 14	Peer code review	Peer code review	Peer code review	Peer code review
March 21	Project testing	Project testing	Project testing	Project testing
March 28	Documentation	Documentation	Documentation	Documentation
April 4	Rehearse for in-class presentations	Rehearse for in-class presentations	Rehearse for in-class presentations	Rehearse for in-class presentations
April 18	Submission	Submission	Submission	Submission

6.2 LINKS

Presentation video: https://youtu.be/RMcZEHyrdTo

Code Repository: https://github.com/makfni/LicenseDetector

7. REFERENCES

J., K. D. (2009). Dynamic Programming Based Method for Extraction of License Plate Numbers of Speeding Vehicles on the Highway. *Internation Journal of Automative Technology*, 205-210.

Kahramen F., K. B. (2003). *License Plate Character Segmentation based on the Gabor Transform and Vector Quantinization.* ICSIC.

L., D. (2004). *License Plate Detection using Adaboost.* San Diego: University of California San Diego.

T., D. H. (2005). Building an Automative Vehicle License Plate Recognition System. *Computer Science*.