CS 410/510: Introduction to Computer Vision
Final Project Report (due 4:30 pm, 03/17)
Haomin He & Boxuan Zhang

License Plate Blurring, Deblurring and Recognition

Abstract

This paper demonstrates a variety of license plate operations, includes license plate detection, blur, deblur/sharpness and recognition. Utilize algorithms from opencv, papers and so on. Build object detection of Haar Cascades with Russian number plate, blurring plate by using median value replacement, sharpening plate number with Laplacian-based method and recognize plate number by implementing text binarization and Tesseract optical character recognition engine.

Introduction

Google map is a popular application to show a place, people can look at images or videos of a place. It also provides privacy protection. For instance, license plates and faces are always blurred. Thus, we want to achieve implementation of blurring license plate on an image. At the same time, the deblurring process of license plate number also sounds interesting to us. The deblurring process assists us to recognize unreadable plate number in images. After getting readable numbers, we can achieve our the next goal, getting precise plate number from the image.
For this project, Haar Cascades use the opencv functions and follow the rule from official website of opencv. By learning algorithms from opencv, we build median blur function by ourselves. Picture sharpening uses the Laplacian method and then use the filter2d function from opencv. In the end, we achieve plate number recognition through image foreground and background text binarization. We then use Tesseract, an optical character recognition engine, to convert image of text into a text file. This paper presents all the methods we have implemented, and experiments we have tested on our programs.

Graph of Project Process

Detect license plate from images	→ blur the detected license plate		
	→ deblur the detected license plate	→ plate number recognition	→ output license plate number

Method

Haar Cascades Method

Object Detection can become effective by using Haar feature-based cascade classifiers. The method proposed by Paul Viola and Michael Jones in this paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach, the haar cascade function is trained from a lot of correct and fault image, after training, it can be used to detect object in the other images.
For license plate detection, Haar Cascades use the same method like face detection, the algorithm needs lots of correct image (image of plate) and fault picture (image without plate) to train the classifier. Then, it needs the extract features from it.
Haar-like features are the digit feature to work on object detection. Hal features use window detection and place the circumscribed rectangle places at the feature area, and them make computation. At begin of test, the detection uses a same size detect window and move on the input graph, calculate the feature rate on the sub-areas on the input picture. The difference of computation will make comparison with the threshold, and then split the goal areas. And then decide the correct area of object.
Hal feature's main advantage is fast computations. Project decide to use the Russian license plate .xml file to determine the location of plate in the input graph, and then using functions from haar cascades, it can have the plate to be marked. It will mark the plate with red color and it will show a rectangle that include the plate.

Median Blur Method

For privacy protection, blur method can effectively suppress the noise point, making the picture become smooth. Median filtering is one typical nonlinear filtering, it is a nonlinear signal processing technology that can effectively suppress noise based on the theory of sorting statistics. The basic idea is to replace the center pixel within the neighborhood with the median value. The value is close enough to the true value to eliminate isolated noise points. The method preserves the edge details of the image while taking out impulse noise and salt-and-pepper noise points. These excellent characteristics are not available in linear filtering. The median filter first needs to create a filter template to sort the pixel values. Usually in the size of 3*3 or 5*5 or in different shapes such as lines, circles, crosses, rings and so on.

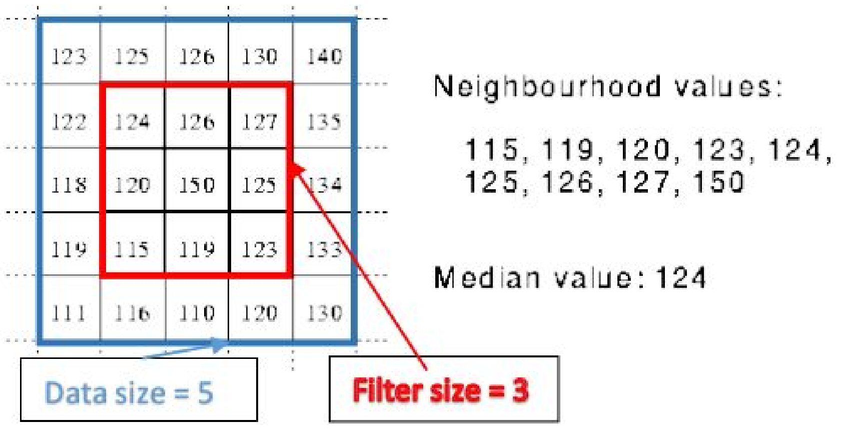


Figure 1. 3 * 3 filter, current value, neighborhood values and median value.

Picture Sharpness Method

In the image operation, blur can eliminate noise in the image or reduce contrast. Conversely, in order to emphasize the edge and detail of images, it is necessary to sharpen the image and improve the contrast. The Laplacian sharpening image is related to the degree of abrupt change of the surrounding pixels, it is based on the degree of change of the image pixel. When the value of the neighborhood center pixel is lower than the average value of other pixels in the field in which it is located, the value of center pixel should be further reduced. When the average value in the neighborhood is located, the value of center pixel should be further increase to achieve sharpening of the image.

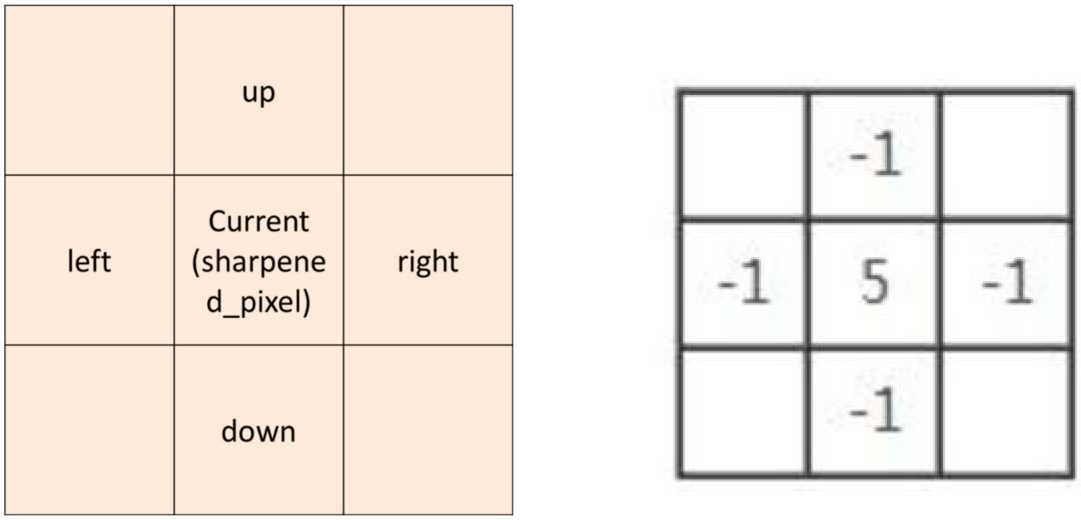


Figure 2. Pixels calculate and parameter to use.

sharpened_pixel = 5 * current – left – right – up – down ;

Figure 3. Function of calculation.

Plate Number Recognition Method

For plate number recognition we have consulted paper “Font and Background Color Independent Text Binarization” to get enhanced images, and used Tesseract (an optical character recognition engine abbreviated as OCR) to recognize text on a license plate. OCR is the electronic conversion of images of typed, handwritten, or printed text into machine-encoded text.

Next, we present the binarization method that the paper talks about. As the paper discuss, images taken by camera may suffer from uneven lighting, low resolution, blur, and perspective distortion. We need to overcome these challenges to acquire text information from these images. A binarization process precedes the analysis and recognition procedures. Simply speaking, it uses threshold to classify image pixels into foreground or background classes. Our goal is to have images be properly binarized without the loss of textual information. This binarization process reduces the computational load and the complexity of the analysis algorithms later on.

In the paper, authors proposed method uses an edge-based connected component approach to automatically obtain a threshold for each component. Canny edge detection is performed on three color channels. E_R, E_G, E_B are the corresponded edge images and they compose the edge map E ($E = E_R \vee E_G \vee E_B$, $\vee$ denotes the logical OR operation). The associated bounding box information is computed during edge detection step, we call it edge-box (EB). The size and the aspect ratios of the EBs can help filter out the non-text regions. We only consider and process EBs that are greater than 15 pixels but smaller than 1/5th of the image dimension. To eliminate highly elongated regions, we constrain the aspect ratio lie between 0.1 to 10.

The edge detection captures both the inner(EBint) and outer(EBout) boundaries of the characters. The following figure shows edge-boxes for the English alphabet and numerals. As you can see there is no character that completely encloses more than two edge components. Therefore, we can filter out the non-text regions by following constraints:
If a EB has 1 or 2 EBs that lie completely inside it, we conclude this EB is a text character and we retain this edge component.
If a EB has 3 or more EBs that lie completely inside it, we conclude this EB is a non-text character and we remove this EB.

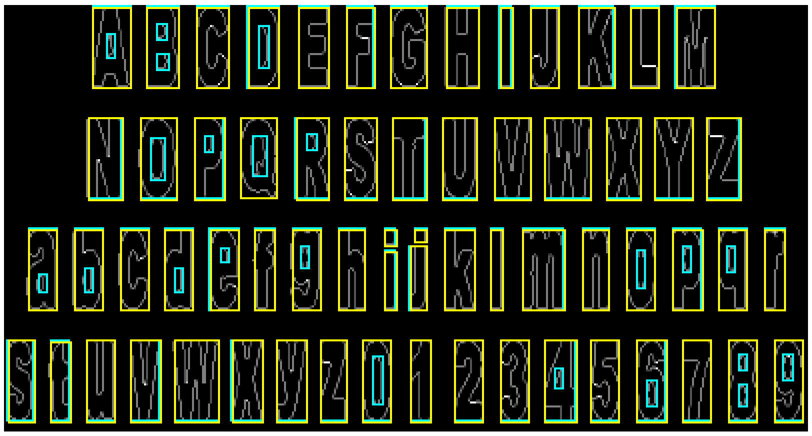


Figure 4.

Next, we do binarization on the filtered set of EBs. For each EB, we estimate the foreground and background intensities. In figure 5, there are foreground and background pixels that are used for calculating the threshold and inversion of the binary output.

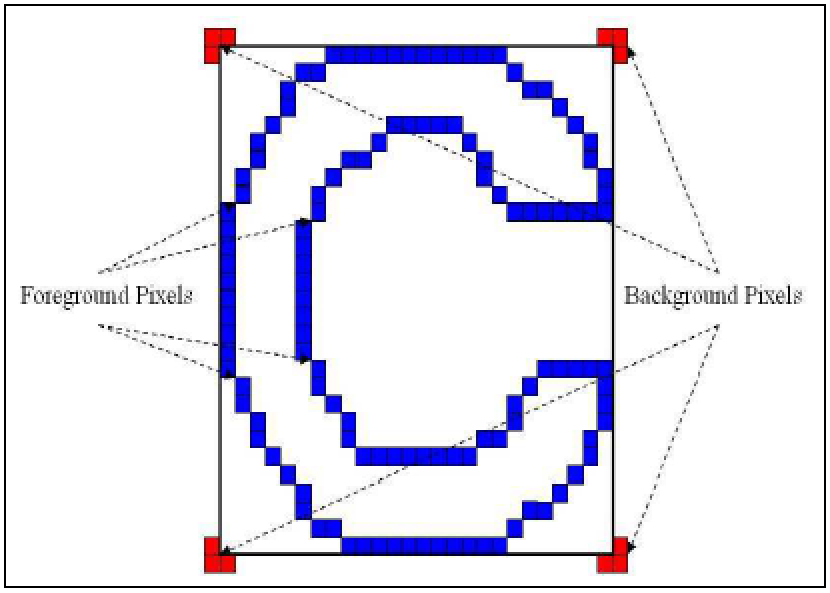


Figure 5. The foreground and the background pixels of each edge component

The foreground intensity (F_{EB}) is computed as the mean gray level intensity of the pixels that correspond to the edge pixels. $F_{EB} = \frac{1}{N_E} \sum_{(x,y) \in E} I(x, y)$ where E is the edge pixels, I(x,y) represent the intensity value at the pixel (x,y) and N_E is the number of edge pixels in an edge component.

The background intensity (B_{EB}) is computed based on intensity of three pixels each at the periphery of the corners of the bounding box:
$B = \{I(x-1, y-1), I(x-1, y), I(x, y-1),$
$I(x+w+1, y-1), I(x+w, y-1), I(x+w+1, y),$
$I(x-1, y+h+1), I(x-1, y+h), I(x, y+h+1),$
$I(x+w+1, y+h+1), I(x+w, y+h+1), I(x+w+1, y+h)\}$
where (x, y) is the coordinates of the top-left corner of the bounding-box of each edge component and w and h are its width and height. Authors indicate that when the text is aligned diagonally, the mean intensity of the background pixels is affected due to overlapping of the adjacent bounding boxes. You can see this in figure 6. Hence, the local background intensity can be estimated more reliably by considering the median intensity of the 12 background pixels.
$B_{EB} = \text{median}(B)$

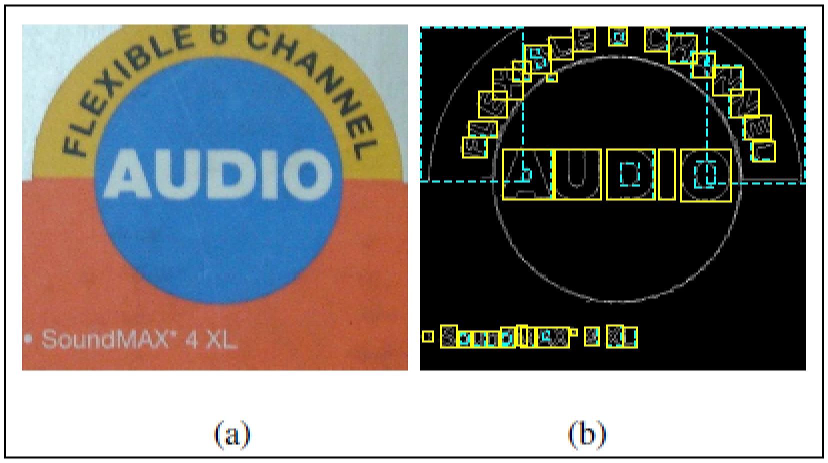


Figure 6. (a) Input Image (b) Output of Edge-Box filtering.

Next, we binarize each edge component using the estimated foreground intensity (F_{EB}) as the threshold. While binarization process, foreground text is output as black and the background as white.

If the foreground intensity is lower than the background:

$$BW_{EB}(x, y) = \begin{cases} 1, & I(x, y) \geq F_{EB} \quad (white) \\ 0, & I(x, y) < F_{EB} \quad (black) \end{cases}$$

If the foreground intensity is higher than the background

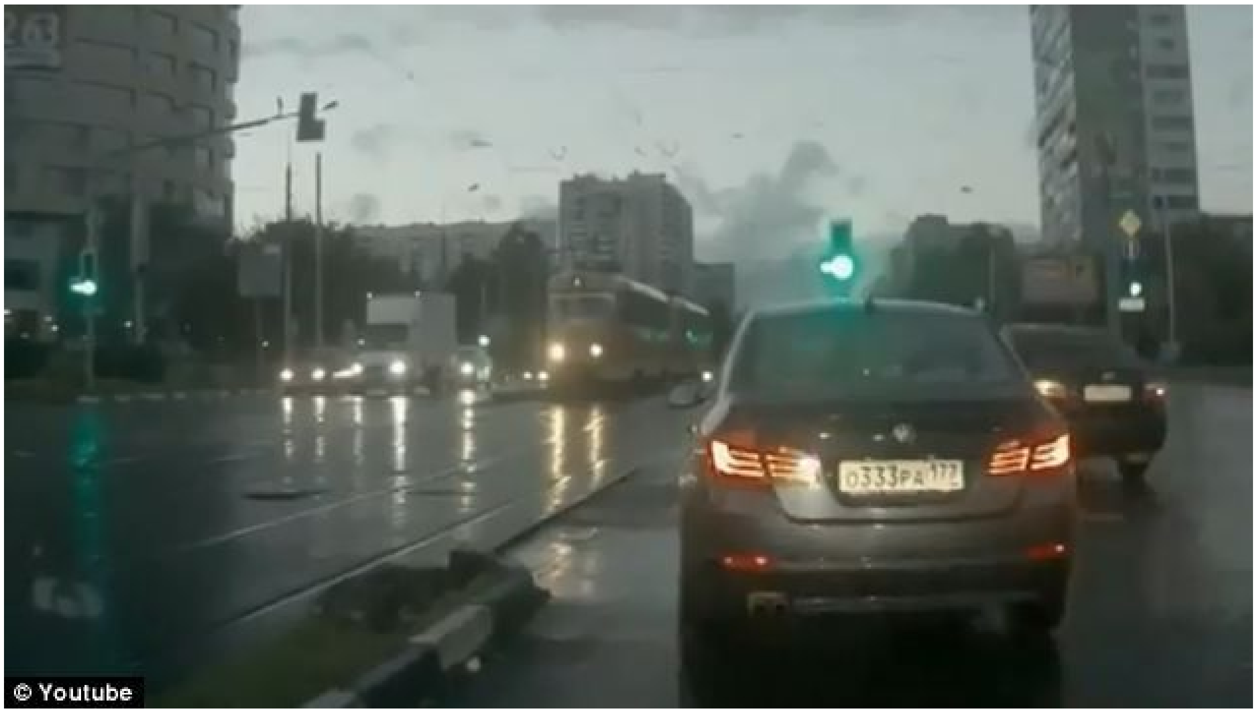
$$BW_{EB}(x, y) = \begin{cases} 0, & I(x, y) \geq F_{EB} \quad (black) \\ 1, & I(x, y) < F_{EB} \quad (white) \end{cases}$$

where BW_{EB} stands for binarized output.

Final step, we run Tesseract, optical character recognition engine, on the binarized output image in order to convert the image into machine-encoded text. In the end, we obtain the license plate number in a text file.

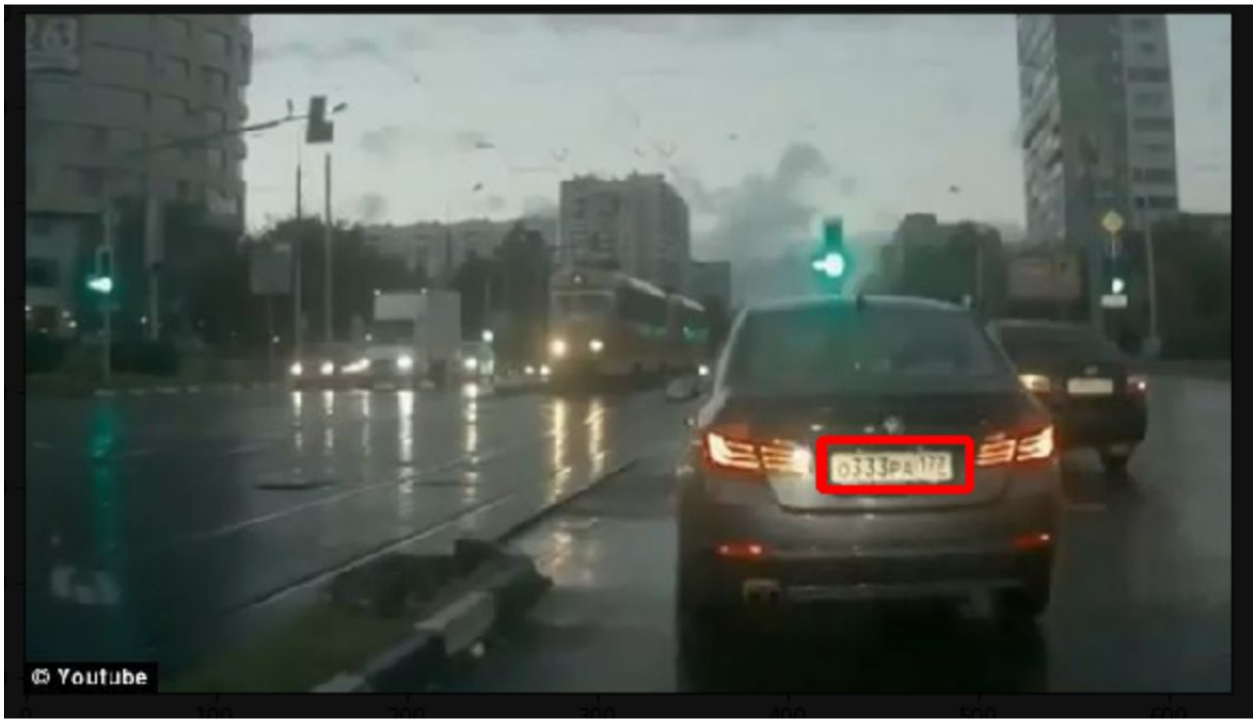
Experiments

Haar Cascades, Median Blur and Picture Sharpening use the same input test graph.
Original image:



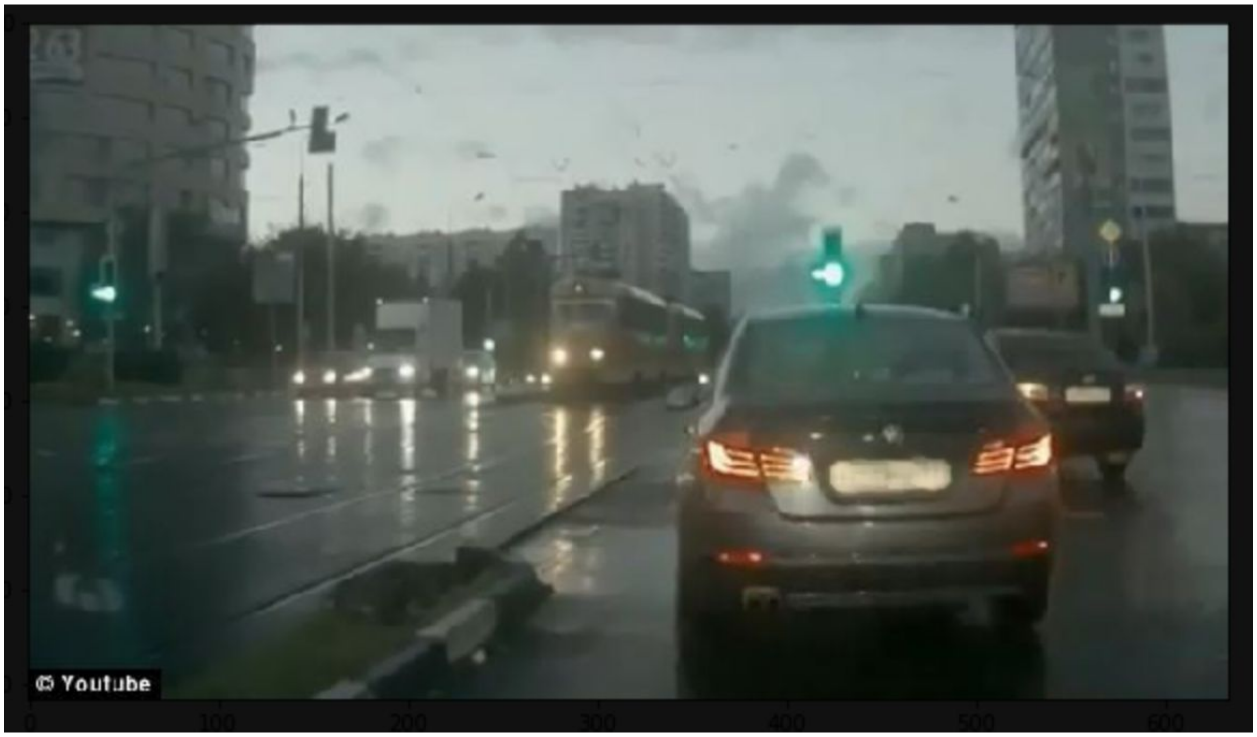
Haar Cascades Method

Use a red rectangle to mark the location of license plate find by Haar Cascades.



Median Blur Method

The Haar Cascades method already figure out the location of license plate, and then operate the plate with median blur, make it become smooth.



Picture Sharpening Method

The Haar Cascades method already figure out the location of license plate, and then operate the plate with picture sharpen, make the number plate become more sharp.

Plate Number Recognition Method

We have found some license plate images online to test out our method. Following is a testing example.

Original image:

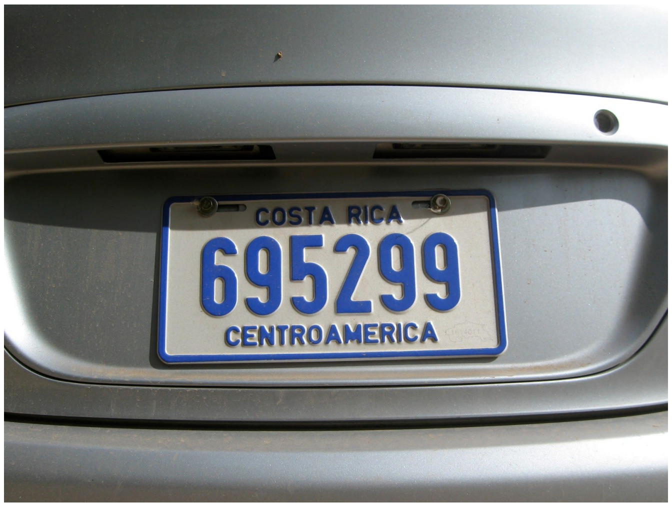
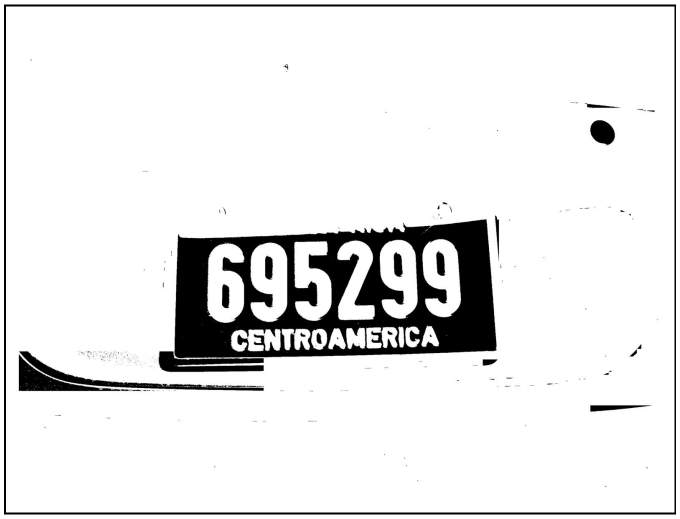
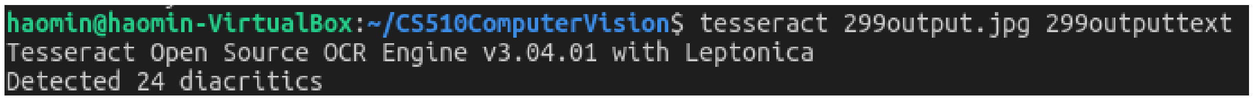


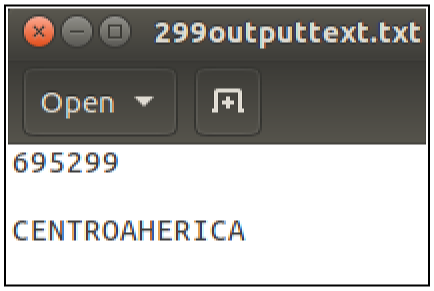
Image after Text Binarization:



Output text file after running Tesseract:



```
haomin@haomin-VirtualBox:~/CS510ComputerVision$ tesseract 299output.jpg 299outputtext
Tesseract Open Source OCR Engine v3.04.01 with Leptonica
Detected 24 diacritics
```



```
695299

CENTROAHERICA
```

References

Cascade Classification - OpenCV 2.4.13.7 Documentation, docs.opencv.org/3.1.0/d4/d13/tutorial_py_filtering.html.

“Image Filtering.” Cascade Classification - OpenCV 2.4.13.7 Documentation, docs.opencv.org/2.4/modules/imgproc/doc/filtering.html.

Kasar, Thotreingam & Kumar, Jayant & Ramakrishnan, A.G.. (2007). “Font and Background Color Independent Text Binarization.” 10.13140/RG.2.1.1349.4880.

Jasonlfunk. “Jasonlfunk/Ocr-Text-Extraction.” GitHub, 15 Oct. 2014, github.com/jasonlfunk/ocr-text-extraction.

“Optical Character Recognition.” Wikipedia, Wikimedia Foundation, 6 Mar. 2019, en.wikipedia.org/wiki/Optical_character_recognition.

Piscani, Francesco. “License Plate Recognition with OpenCV 3 : OCR License Plate Recognition.” YouTube, YouTube, 18 Feb. 2015, www.youtube.com/watch?v=nmDiZGx5mqU.