Automatic Detection and Segmentation of License Plates

Muhammad Saad Malik and Reinhard Klette

Computer Science Department, Tamaki Campus,
The University of Auckland, New Zealand
Email: mmal000@aucklanduni.ac.nz

Abstract. This paper presents algorithms for license-plate detection and segmentation. The algorithms achieve respectable success rates and run times. Experiments and results are also discussed in this paper.

1 Introduction

Automatic license-plate-recognition (LPR) systems are useful in the automotive and transport industry. For example, see [2–8]. License-plate detection and segmentation are essentials of an LPR system.

In this paper we present algorithms for both detection and segmentation. Our algorithms focus on daylight scenarios with cars in a stationary position (i.e. parked cars). See Fig. 1 for an example of a typical input image as processed in this study. The figure also shows seven detected regions of interest (four true-positives and three false-positives).

Fig. 1. A typical scene as considered in this study: Parked cars, recorded with a mobile camera. The figure also illustrates results of license-plate detection (four true-positives, and three true-negatives).

2 License-Plate Detection

The first step in a license-plate-recognition system is license-plate detection. In this section we discuss our method of license plate detection and how it works.

License Plate Localisation. Our approach to license plate localisation is somewhat similar to the one proposed in [6]. In the following we go over the steps taken in our approach.

Step 1. The algorithm starts off with an RGB input image which is converted into grayscale. Then, to reduce noise, we apply a 5×5 Gaussian blur filter to the image. This is necessary as it improves the results of further processing on the image.

Step 2. In this step we detect edges. We use the simple Sobel operator. License plate characters have more dominant and meaningful vertical edges in comparison to horizontal edges. Hence, we apply only the vertical component of the Sobel operator.

Step 3. After edge detection we apply the morphological operation of closing to the image. The closing operation helps to dilate white regions as well as to erode away small unwanted details.

The structuring element used for closing is given a similar aspect ratio as that of a license plate. This is to ensure that the resulting image produces rectilinear shapes (i.e. like the shape of a license plate). Next, to simplify the image, we apply Otsu's binarization (e.g., see description in [1]).

When the binary image is obtained we further apply two more morphological operations, an erosion followed by a dilation. One could simply apply an opening operation but the structuring element will then have to be the same for both operations. For our approach we require two different structuring elements so the opening operation on its own would not work. For erosion we use a structuring element of a size slightly larger than the one used for the closing operation before. Using erosion again further helps in removing shapes that are too small to be a license plate. It also helps break apart large shapes which may contain the license plate (i.e. due to the application of these morphological operations the license plate may sometimes merge into other objects). For dilation we use a larger and more square-shaped structuring element. Dilation is used to simply bring back important already-removed information. For example, a license plate may already be in perfect shape but by applying erosion it may have lost its shape. Dilation using a larger structuring element also allows remaining shapes to gain more area around them. This helps because, if some shape is eventually a license plate, the shape will cover as much of the license plate as possible.

Step 4. By the end of Step 3, an image of several regions is produced which may also contain a license plate. The focus in this step is to extract regions which are most likely to be license plates. To do this we used connected-component analysis (CCA).

First, we calculate the minimum rectangular area for each detected region; we obtain rectangles that represent each one region. Then we compare the properties of the detected rectangles with those of a license plate. Based on the comparison we eliminate false-positives, i.e. regions which are unlikely to be a license plate.

The properties we use to determine whether a region is a license plate or not include the area, the angle (of the main axis), and the aspect ratio of an average license plate. If the area of a rectangle is either too big or too small, the corresponding region is ignored. If the angle of a rectangle suggests that the corresponding region is tilted too much (e.g. by 75°), then this region is also ignored. Lastly, if the aspect ratio of a rectangle is quite different from that of a license plate's aspect ratio then that region is also ignored. In the end, regions that correctly meet the properties of a license plate are extracted from the original grayscale image. See Fig. 1. Note that we still have several false-positives in this example at this stage.

License Plate Validation. License plate validation is a process of validating rectangular windows as being a license plate. The considered windows are the candidate regions extracted in the localisation process described before. Our approach to validate a region as a license plate involves running a simple test. The candidate region must pass this test in order to be classified as a license plate. This test follows [8].

Validation Test. A candidate region must contain at least two identical or similarly sized rectangles each circumscribing one character. We use this condition for validation because all characters on a license plate have about the same width and height.

If two similarly sized rectangles are found then they identify most likely two license plate characters. On average, license plates have at least five characters which means two characters should be detected if the candidate region is a license plate. Once such rectangles are detected, we calculate the center points of the detected rectangles. We then check whether the center points lie in reasonable positions to be considered as license plate characters.

At the end the test checks whether the candidate region contains a character-sized rectangle somewhere close to the center of the candidate region. To check this we simply make use of center points calculated earlier.

On-going Validation. Apart from the validation test, the candidate regions are tested throughout the segmentation process to ensure that they are in fact license plates. The segmentation algorithm is discussed in the next subsection.

License Plate Segmentation. The process starts with *License Plate Tilt Correction.* In this step, following validation, we correct any undesired tilt that the license plates may have. This is the start of the segmentation algorithm. Once again we were inspired by [8].

For correcting the tilt we simply gather all the calculated center points from the validation process and sort them in ascending order according to their x coordinates. After sorting the center points we calculate the tilt angle θ of the license plate using a function available in OpenCV. See Equ. (1), where *cvFastArctan* is a function that takes in a x and y value and calculates the associated angle:

$$\theta = cvFastArctan(|y_{diff}|, |x_{diff}|); \tag{1}$$

The provided x_{diff} value is the absolute difference between the x coordinates of the lowest and the highest center points (with respect to x). Similarly, the

Fig. 2. Detected (*left column*) and segmented (*right column*) license plates.

provided y_{diff} value is the absolute difference between the y coordinates of the lowest and the highest center points (with respect to y). Once the angle is found, we use a 2D rotation matrix to rotate the license plate accordingly.

License Plate Pre-processing. Pre-processing of license plates is required to support better results during character segmentation. License plates which are not pre-processed will have noise and other undesired conditions that may produce incorrectly segmented characters. The main focus of this pre-processing step is to cleanly convert the license plates into binary images. For reducing noise, we apply again a 3×3 Gaussian filter to the license plate window. Then we use a simple contrast enhancement method to give more dominance to the darker pixels in the region. We do this for making the characters more dominant (assuming black characters on white background). This approach is similar to the object enhancement algorithm described in [3]. The contrast enhancement we apply is specified by the following linear transform:

$$g(x, y) = a \cdot f(x, y) + b \tag{2}$$

where g is the output and f is the input window, and a and b are the gain and bias which are assumed to be known for controlling contrast and brightness. We preselected a as being 2, and b as being -100; we obtained these values through experimentation with different license plates. Finally, to convert the candidate regions into a binary image, we use Otsu's binarization again as we already did in the localisation process.

Character Extraction. This is the final stage of the segmentation algorithm. This step simply involves using connected-component analysis to extract the characters from the license-plate region. After extraction they can be processed individually during a subsequent character recognition process. Figure 2 illustrates results of the segmentation process.

3 Experiments and Results

In this section we discuss results of our experiments with the proposed license plate detection and segmentation algorithms. All images used had a resolution of $3,264 \times 2,448$ pixels.

Discussion of Detection Experiments. The outcome from our detection experiments suggested that the detection success rate is high. The images used were taken in different scenarios to test the detection algorithm under various scene-geometry or lighting situations. The detections were almost always successful in every case. Problems occurred when the distance of license plates from the camera changed. License plates being very close to the camera may not be detected correctly. The experiments suggested that the detection algorithm has a limitation of being only optimally working in a certain range of distances. Detection in such cases was only successful if the algorithm was adjusted to support the new distances. In some cases, detections were successful but the detected regions consisted of extra information that was not associated with the plates (i.e. noise). This commonly occurred in tilted view images of parked cars.

Discussion of Segmentation Experiments. Segmentation experiments also produced good results. We tested the segmentation algorithm on several license plates that were detected using the detection algorithm. Plates with various tilt angles (in a defined range) were used in the experiments. Plates with no, or very minimal tilt were also considered to ensure the algorithm only fixes tilt on plates that actually require it to do so.

The sample of plates also included plates recorded under different lighting conditions., varying from ideal lighting to really poor lighting conditions (e.g. very bright or dark). Different plate sizes were also used, some of which were detections closer to the camera. From the experiments we understood that the segmentation algorithm would only fail if the detected license plate was dirty or there was some other form of noise around the license plate characters. However, a limitation was identified: Even though the tilt of the plates was corrected, the perspective of the characters was not. This limitation could increase the chance of poor character extraction and character recognition.

Results and Overall Performance. Overall, the detection and segmentation algorithms performed well, especially on images with frontal views of parked cars. The major advantage of the detection algorithm was that it was able to detect several plates in one execution. This makes the proposed algorithm very efficient in comparison to other methods. The detection algorithm was also robust to variations in lighting conditions. The segmentation algorithm was robust to different lighting conditions as well as to various tilt angles and plate sizes.

To evaluate the performance of the algorithms we only considered images with frontal views of parked cars. The detection algorithm was tested on 13 images, which consisted of in total 45 license plates (from 45 parked cars). Some of the license plates were repeated in some images. For the purpose of testing it was not important for the license plates to be different in every sample. What was important however was that we had different conditions and situations when the images were taken. The sample of 13 images covered different conditions (i.e. in lighting) and situations (i.e. such as the number of cars being parked, and how they are parked). The segmentation algorithm was tested on 11 license-plate images. In both tests the sample size was quite low. This was due to a lack in more test data. Table 1 summarizes a few results of the experiments.

Algorithm	Sample size	Success rate
Detection	45	100%
Segmentation	11	100%
Overall	56	100%

Table 1. Summary on experiments.

The experimental results in Table 1 show an overall success rate of 100% for the considered cases. This is obviously very optimistic and unreal for general expectations in the real world. The clear reason for this is the low sample size. Apart from that, by looking at the results we can conclude that the algorithms actually produced high success rates in real life situations.

Run-times. The average run-time for the detection algorithm was 2.6 seconds per input image. This was regardless of how many plates were detected. The average run-time for the segmentation algorithm was 0.31 seconds. This run-time was for segmenting a single license plate. We did not aims at any run-time optimization.

4 Conclusions

In this paper we presented algorithms for license plate detection and segmentation. Both algorithms were tested in different scenarios and they produced promising results with respectable run-times. The experiments suggested that the algorithms work optimally on images with frontal views of cars. A few limitations were also identified, and addressing them might be the first task in future work. There may also be further unidentified limitations because the algorithms were only tested on a relatively small sample size of test data.

References

1. Klette, R., *Concise Computer Vision*, Springer, London (2014)
2. Kodwani, L., Meher, S., Automatic license plate recognition in real time videos using visual surveillance techniques, Proc. *Int. Joint Conf. Advance Engineering Technology* (2013) 6 pages
3. Kranthi, S., Pranathi, K., Srisaila, A., Automatic number plate recognition. *Int. J. Advancements Technology* (2011) 2:408–422
4. Kumary, S., Angel, B., Laya, T., License plate detection and character recognition using contour analysis, *Int. J. Advanced Trends Computer Science Engineering* (2014) 3:15–18
5. Romić, K., Galić, I., Baumgartner, A., Character recognition based on region pixel concentration for license plate identification, *Tehnicki vjesnik* (2012) 19:321–325
6. Thome, N., Vacavant, A., Robinault, L., Miguet, S., A cognitive and video-based approach for multinational license plate recognition, *Machine Vision Applications* (2011) 22:389–407
7. Wang, Y.-R., Lin, W.-H., Horng, S.-J., A sliding window technique for efficient license plate localization based on discrete wavelet transform, *Expert Systems Applications* (2011) 38:3142–3146
8. Wen, Y., Lu, Y., Yan, J., Zhou, Z., Deneen, K., Shi, P., An algorithm for license plate recognition applied to intelligent transportation system, *IEEE Trans. Intelligent Transportation Systems* (2011) 12:830–845