Automatic Recognition of Car License Plate Numbers

Minya Chen and Edward K. Wong

Department of Computer and Information Science
Polytechnic University
5 Metrotech Center
Brooklyn, NY 11201

Abstract

In this paper, we present a novel method to extract and recognize car license plate numbers embedded in 3-D scenes. The method is divided into two stages: extraction and recognition. Two-D rotational angles of the car license plates are estimated in the extraction stage, and subsequently used to compute 3-D invariant features for character recognition in the recognition stage.

Keywords: license plate recognition; character extraction; character recognition.

1. Introduction

Automatic recognition of car license plates plays an important role in traffic surveillance systems. Such systems, which could be installed in places like parking areas and highway tollgates, can greatly reduce the workload of human operators. Any situation requiring the identification of a vehicle provided with a license plate number represents a potential application. Recently, we have seen considerable amount of research efforts on the automatic detection and recognition of license plate numbers [1-5]. There are also many commercial license plate recognition products available today. PIPS Technology [6] offers a wide range of license plate recognition systems, including a number of mobile/portable products, for applications in parking lots, toll stations, access control, law enforcement and security. Zamir Recognition Systems Ltd. [7] develops and markets InSignia, a license plate recognition system for the automatic identification of vehicle in real time. Adaptive Recognition Hungary Ltd [8] develops, manufactures and sells intelligent software and devices that process images within the context of security and traffic control. Geovision LPR system [9] uses Neural Networks technology to identify vehicles by their license plates. The SeeCar [10] product line is a set of vision-based license plate recognition systems that detect and read vehicle license plates for vehicle access control, parking lot billing, and security applications.

In this paper, we propose a novel method to extract and recognize car license plate numbers embedded in 3-D scenes. The method is divided into two stages: extraction and recognition. In the extraction stage, license plate characters are extracted based on computing the *maximum gradient difference* (MGD) at each pixel. The 2-D angles of the characters in the image plane are also estimated in the extraction stage. Details will be described in Section 2. In the recognition stage, 3-D perspective invariant features are extracted and used for recognition. Details will be described in Section 3. In Section 4, we present simulation and experimental results. Finally, in Section 5, we give our concluding remarks.

2. License Plate Number Extraction

In most application scenarios, license plates are captured with small amount of rotational and perspective distortions. The license plates' 2-D projections have shapes that can be approximated by parallelograms. License plate number extraction can be divided into four steps as described below:

Step 1: Identify license plate region

This step is to detect the license plate region from the input image. We use the fact that in a license plate region the variation in color or grayscale values is high due to the presence of characters. We compute a measure called *maximum gradient difference* (MGD) for each pixel in the gray scale image obtained from the original color image that contains the license plate. The *MGD* is computed as follows: first the horizontal luminance gradient *dx* of each pixel is computed by using the mask [-1, 1], then the difference between the maximum and minimum gradient values is computed within a local window of size $1 \times n$. The window size n is chosen based on the resolution of the image and the approximate size of the license plate in the image.

For our test image set we choose $n = 21$, which results in reasonable good performance. Size n can be adjusted according to different application scenarios. We then threshold the MGD to isolate regions with high MGD values, which correspond to potential text regions. A license plate typically has one or more high MGD value regions. We apply morphological dilation to combine spatially close regions. We then remove false text regions based on the geometrical property of license plates. Details about the use of *MGD* measure for character extraction in video and color images can be found in [11]

Step 2: Identify region of interest

The current algorithm extracts only the license plate number and ignore the name of the state. The data set we experimented contains license plate images of

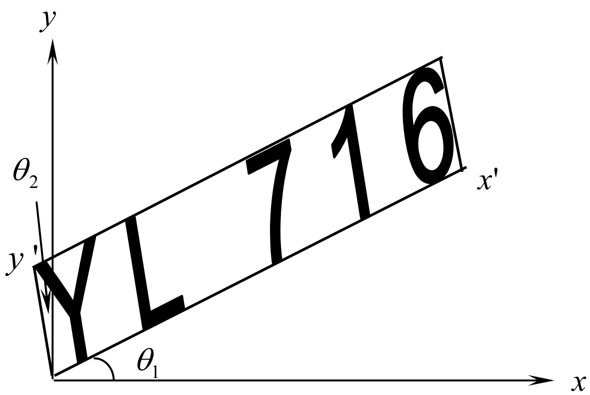


Figure 1. Projected license plate forms a parallelogram

New Jersey State, with the license plate number located in the center part of the license plate. The background of the middle region is of light color and the license number is of dark color. In Step 2, we binarize the license plate region identified in Step 1, and apply connected component analysis on the light pixels. The background of the license plate region will then form the largest connected component. Using this property, we can narrow the region of interest to the center part of the license plate, which will be used for character extraction in Step 3.

Step 3: Character extraction

After we locate the center part of the license plate, we extract the license plate characters based on their geometric properties. License plate characters have some uniform characteristics that could be utilized in the extraction process. First the characters are of the same font and size. Second, the number of characters on the license plate is within a certain range.

Step 4: Estimate 2-D angles of license plates

We estimate the 2-D angles of license plates in Step 4. Here we assume that the camera is far enough from the license plate so that the perspective effect introduced by the camera lens system is negligible. The 2-D projection of a rectangular license plate on the image plane can therefore be approximated by a parallelogram. The horizontal side of the license plate will make an angle of θ_1 with the horizontal *x* axis, and the vertical side of the license plate will make an angle of θ_2 with the vertical *y* axis (see Figure 1 for illustration.). The values for θ_1 and θ_2 depend on the

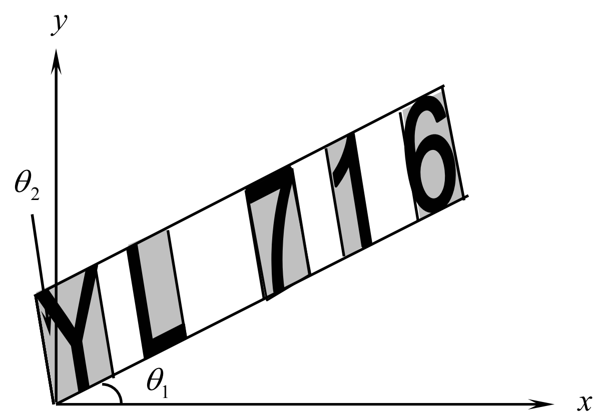


Figure 2. θ_1 and θ_2

3-D angle of the license plate relative to the camera.

To estimate the angle θ_1, we find the angle of the line passing through the highest point of every license plate character, and the angle of the line passing through the lowest point of every character. Angle θ_1 is computed as the average of these two angles. From the geometry of the parallelogram as shown in Figure 1, we see that θ_2 is close to θ_1 in value. After θ_1 is estimated, we can estimate θ_2 by a constrained search process with iterative refinement. For every value of θ_2 (with a certain step size) within the range $[\theta_1 - \alpha, \theta_1 + \alpha]$, where α is a small positive angle, we compute the area of the shaded regions as shown in Figure 2. Every shaded region is tightly bounded by a parallelogram with angle values θ_1 and θ_2. The θ_2 value which leads to the minimum areas for the shaded regions is chosen as the value for θ_2. After the first iteration, we refine our search with a smaller step size by searching in the neighborhood of the estimated θ_2 value, and so on in subsequent iterations.

Using the line passing through the highest points and the line passing through the lowest points of license plate characters, we can use them to further trim the connected components of the characters. Pixels above the top line and pixels below the bottom line are removed in the trimming process.

3. License Plate Number Recognition

The segmented connected components of the license plate characters and the estimated θ_1 and θ_2 values are input to the recognition phase. The 3-D invariant features we choose for recognition are *slanted* x-*projection, slanted* y-*projection*, and *slanted foreground-to-background (FB)* ratio. Normally, x- and y-projections for binary images are obtained by computing the number of foreground pixels projected onto the *x*- and *y*-axes for each bin respectively. To compensate for the rotational distortion caused by the imaging angle of the camera, we compute the projections onto the slanted horizontal axis *x'* and the slanted vertical axis *y'* (See Figure 1.) We illustrate this in Figure 3. Figure 3 (a) shows the original image of the letter 'G'. Figure 3 (b) shows the projected letter 'G' with rotational distortion. Figures 3 (c) and (e) show the *x*- and *y*-projections of the image in Figure 3 (a), respectively. Figures 3 (d) and (f) show the slanted *x*- and slanted *y*- projections of the image in Figure 3 (b), respectively. We see that the shape for the *x*-projection in Figure 3 (c) is very close to that of the slanted *x* projection in Figure 3 (d). We see the same similarity in the shapes of Figures 3 (e) and 3 (f).

We define the feature *foreground-to-background (FB) ratio* as the ratio of the number of foreground to background pixels in a local rectangular block. For a given input image, we divide the image into $n \times m$ rectangular blocks and compute the FB ratio for each block. The *slanted FB ratio* is computed in a similar manner except the local block is a parallelogram. For a given image, parallel lines at angles of θ_1 and θ_2 as shown in Figure 4 are used to divide the image.

Using the 3-D invariant features, a simple nearest neighbor classifier is used to recognize license plate characters. An unknown character is compared to a set of prototype characters representing different character classes.

4. Simulation and Experimental Results

We conducted simulations and experiments to validate our extraction and recognition algorithms. The prototype characters we used in the nearest neighbor classifier are manually extracted from a set of digital images of New Jersey license plates. The prototypes include 10 digits and 26 capital letters, with an image resolution of 180 x 324 per character.

To test the robustness of the invariant features against 3-D rotational effects, we graphically generate a set of test images with different rotational degrees using an OpenGL graphics package. As these images are graphically generated, they represent characters of perfect segmentation results. Simulation results showed that for the features slanted-*x* and slanted-*y* projections, if the rotating angle about the horizontal *x*-axis is within $[-10^\circ, 10^\circ]$, and the rotating angle about the vertical *y*-axis is within $[-40^\circ, 40^\circ]$, the recognition rate is above 90%. For the feature slanted FB ratio, if the rotating angle about the x-axis is within $[-10^\circ, 10^\circ]$ and the rotating angle about the y-axis is within $[-30^\circ, 30^\circ]$, the recognition rate is above 90%. From the results, we can say that the features we used is robust against small amount of rotation about the *x*-axis, and can tolerate a larger amount of rotation about the *y*-axis.

We tested our algorithm on 35 real images of resolution 640 x 480, containing New Jersey State license plates. These images contain license plates of different sizes (on image), illuminations and quality. They were taken either form the front or the rear a vehicle at different camera angles. Figure 5 shows a sample test image from the test data set. The algorithm correctly detected 86% of the license plate regions. A total of 182 characters were correctly extracted, which represents 85% of the total number of license plate characters. It took about one second per image for license plate character extraction on a 1.8GHz Pentium 4 PC. Out of the 182 characters, the algorithm correctly recognized 83% of the characters when the slanted *x*- and slanted *y*-projections are used as features. The recognition rate was 74% when the FB ratio was used as feature.

5. Concluding Remarks

We have presented an effective method for extracting and recognizing license plate numbers embedded in 3-D scenes. License plate detection and character extraction was shown to be robust and accurate in color images with complex background. Character recognition was shown to have good accuracy. A unique characteristic of our approach is the use of simple 3-D invariant features, which were shown to give good performance under a moderate amount of 3-D rotations of the license plate.

6. References

[1] W. He, Z. Wang, and W. Su, “Active electronic license plate-an intelligent identification technology,” *Proceedings of the IEEE International Vehicle Electronics Conference,* Vol. 1, pp 459-461, 1999.

[2] T. Naito, T. Tsukada, K. Yamada, K. Kozuka, and S. Yamamoto, “Robust License-plate recognition method for passing vehicles under outside environment,” *IEEE Transactions on Vehicular Technology,* Vol. 49 Issue 6, pp 2309-2319, November 2000.

[3] T. Naito, T. Tsukada, K. Yamada, K. Kozuka, and S. Yamamoto, “License plate recognition method for inclined plates outdoors,” *Proceedings of 1999 International Conference on Information Intelligence and Systems*, pp 304-312, 1999.

[4] D.S. Gao, and J. Zhou, “Car license plates detection from complex scene,” *Proceedings of 5th International conference on Signal Processing,* Vol. 2, pp 1409-1414, 2000.

[5] K.K. Kim, K.I. Kim, J.B. Kim and H.J. Kim, “Learning-based approach for license plate recognition,” *Proceedings of the 2000 IEEE Signal Processing Society Workshop on Neural Networks,* Vol. 2, pp 614-623, 2000.

[6] http://www.pipstechnology.com/

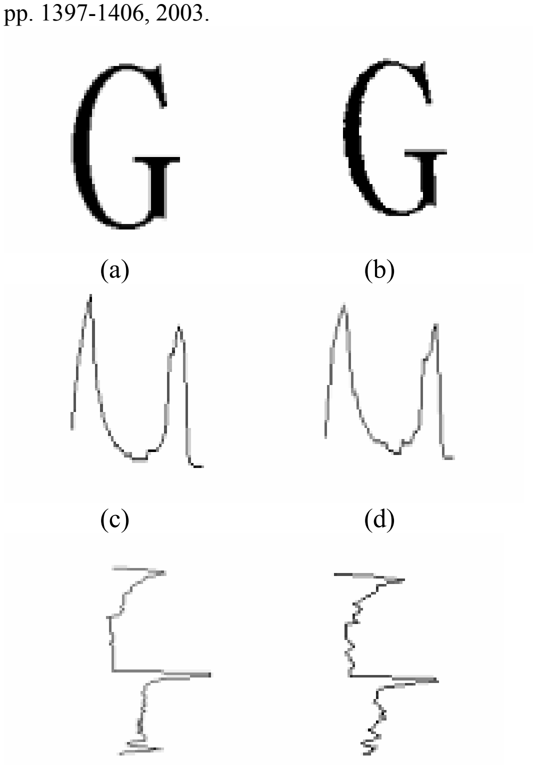
[7] http://www.zamir.co.il/

[8] http://www.arhungary.hu/

[9] http://www.geovision.com.tw/

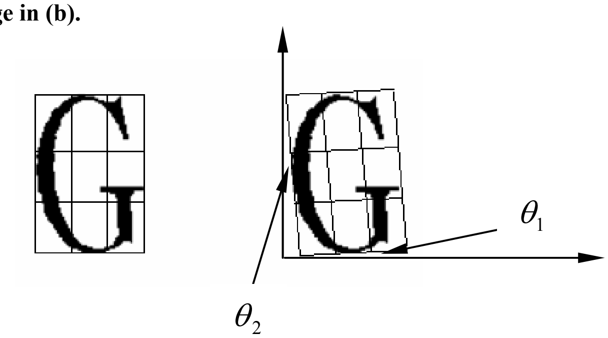
[10] http://www.htsol.com/

[11] E.K. Wong, and M. Chen, “A new robust algorithm for video text extraction,” *Pattern Recognition,* Vol. 36, Issue 6, pp. 1397-1406, 2003.



(a) (b) (c) (d) (e) (f)

Figure 3. (a) Original image with character G. (b) Projected character ‘G’ with rotational distortions ($\theta_1 = 2^\circ$ and $\theta_2 = 4^\circ$). (c) x-projection of image in (a). (d) Slanted x-projection of image in (b) (e) y-projection of image in (a.) (f) Slanted y-projection of image in (b).



(a) (b)

0.37	0.09	0.29
0.49	0.01	0.21
0.44	0.04	0.49

(c)

0.38	0.11	0.28
0.54	0	0.2
0.49	0.05	0.45

(d)

Figure 4. (a) 3×3 blocks for computation of FB ratio. (b) 3×3 blocks for computation of slanted FB ratio ($\theta_1 = 2^\circ, \theta_2 = 4^\circ$). (c) 3×3 FB ratios of image in (a). (d) 3×3 slanted FB ratios of image in (b)

Figure 5. A sample test image.