An Approach To Licence Plate Recognition

J.R. Parker
Laboratory for Computer Vision
University of Calgary

Pavol Federl
Computer Graphics Laboratory
University of Calgary

Abstract

Here we describe a project that deals with license plate location in raster images. The algorithm takes a raster image as input, and yields the position of a plate in the image. After the position is determined, the algorithm can determine the locations of the license plate characters, which could be easily combined with an OCR algorithm to convert the license plate number into an ASCII string.

1. Introduction

Recent interest in photo-radar systems has sparked interest in the old topic of license plate location and recognition. In spite of general commercial interest in the topic there has been relatively little published information. The essential problem is to locate a motor vehicle license plate in an otherwise untouched video frame or photograph. Clearly, such a scene will be cluttered by such items as shadows, bumper stickers, dents, and model identification plates. What will be proposed here is a detailed and novel method for location of vehicle plates.

The basic method for license plate location can be described by the following pseudo-code:

[1] input: original image

[2] preprocessing (RGB conversion/smoothing)

[3] edge detection

[4] location of characters

[5] genetic algorithm to find the license plate

[6] output: bounding box with the best score (from GA)

Each of these steps will be discussed in appropriate detail.

The authors would like to acknowledge the support provided by the Natural Sciences and Engineering Research Council of Canada.

2. Pre-processing and edge detection

The first step converts an RGB image to a gray-level image, using the method standard for NTSC images:

gray=(red*299+green*587+blue*114)/1000

Figure 1 shows an example of a picture that has been converted to gray scale. We will show the various steps of the algorithm based on this reference image.

In the second step a median filter (5x5) is applied to the gray-level image in order to remove the noise, while preserving the sharpness of the image. The median filter is a non-linear filter, which replaces each pixel by a value obtained by computing the median of values of pixels in an, in this case, 5x5 neighborhood of the original pixel. Figure 2a shows the result of applying the median filter to the original image.

After preprocessing the image, it is necessary to find the edges in the image. To this end the Shen-Castan edge detector [7] is employed, partly due to its excellent localization properties and because it can deal with significant amounts of noise. The algorithm amounts to a convolution with a smoothing kernel (the infinite symmetric exponential filter) followed by a search for the edge pixels (zero crossings of the Laplacian) and hysteresis thresholding.

The edge image obtained will help us to look for rectangular shapes in the image. For example, the objective function for the genetic algorithm will be based on how close the outline of a bounding box is to the edges in this image. Since there are cases when the edges around

Figure 1. A sample image converted to grey levels

the license plate are not identified correctly (i.e. white license plate on a white car), the score function for GA does not rely only on this edge image. Figure 2b shows the result of applying Shen-Castan edge detector to the smoothed image.

3. Character Location

Next step involves finding regions in the original preprocessed image that are likely to be characters, using a method similar to one described by Fletcher and Kasturi in [4]. We start by thresholding the preprocessed image to obtain a binary picture. Then we apply binary erosion and dilation [5] to the result, to separate the foreground regions from each other. After that all of the 8-connected regions in the new binary image are identified, and bounding boxes are calculated for them. The bounding boxes with attributes exceeding certain thresholds (described below) are deleted. From this set of bounding boxes we also erase all boxes that do not belong to strings of at least 3 characters; the result is a set of bounding boxes that are very good candidates for characters in a license plate. In the following section we describe these steps in greater detail.

We have tried experimenting with global thresholding on various images of cars, but have concluded that it is very difficult to automatically determine a threshold value that would separate the letters from a license plate for any image. There are various problems with various pictures of cars: license plate can be dirty, the car can be of a bright color, poor lighting, etc. Therefore, it was decided to adopt a region based approach. On all observed images of cars, the gray level difference between the characters and the background of the license plate is very noticeable; this is a design characteristic. Therefore, an algorithm was designed to take advantage of this.

The local neighborhood of a pixel is found, as are the maximum and minimum pixel in this neighborhood. If the difference between max and min is less than the threshold t, the whole neighborhood is considered to have approximately the same shade of gray; therefore we assign the value of a new pixel based on whether the old pixel is bright or dark. If the difference between max and min is greater than or equal to the threshold, we assign the value of a new pixel to be foreground when the old pixel is closer to the maximum, otherwise the old pixel is closer to minimum, in which case new pixel will be as-

(a)

(b)

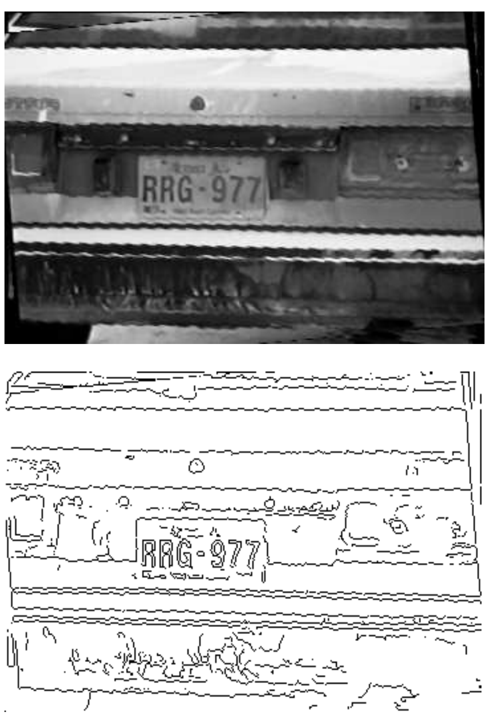

Figure 2. Pre-processing

(a) Figure 1 image after smoothing (median filter). (b) Edges found by the Shen-Castan edge detector.

(a)

(b)

(c)

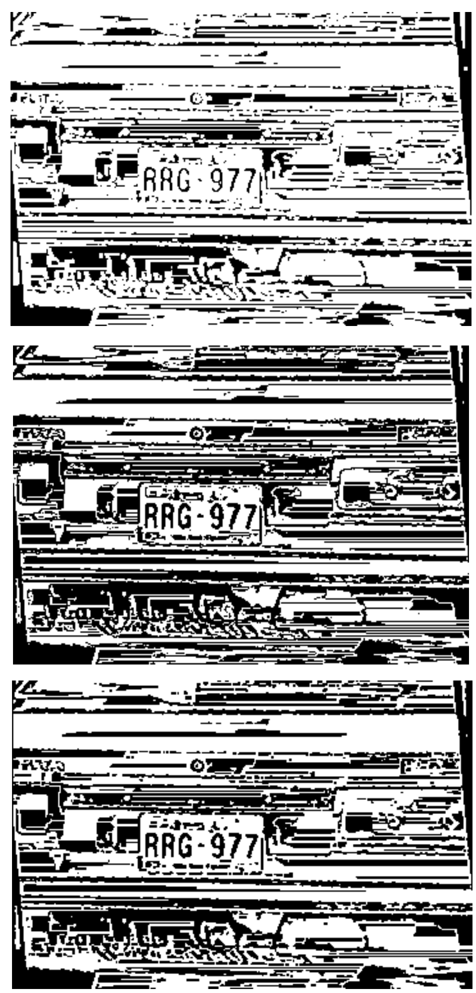

Figure 3. Use of morphology.

(a) The thresholded and smoothed image. (b) The thresholded image after dilation. (c) Dilated image after erosion. No digits are touching here, so no improvement is expected or observed.

signed background. Figure 3a shows the result of the thresholding the image in figure 2a using this algorithm.

In the original image, the thresholded characters are quite often connected to each other due to noise or illumination. This could produce bounding boxes that would be rejected later on due to bad aspect ratios, or they might be considered too big to be portions of strings. Therefore, a morphological opening step follows thresholding (Figures 3b, 3c)

Next, each 8-connected region is located and filled using a non-recursive flood-fill algorithm and a FIFO linked list. Bounding boxes for each region are computed in the obvious fashion, followed by the deletion of all boxes that have 'bad dimensions', remembering that we are looking for characters. The box is deleted if one of the following conditions is satisfied:

- width is less than a threshold (default = 4)
- height is less than a threshold (default = 10)
- area of the bounding box is smaller than a threshold (default =100)
- aspect ratio (width/height) is less than a threshold (default = 0.1)
- aspect ratio is greater than a threshold (default = 1.3)
- ratio between the area of the 8-connected region and the area of the bounding box is less than a threshold (default = 0.2)

The first three conditions guarantee that all boxes that are too small are deleted from the list; the next two conditions are responsible for keeping only bounding boxes that have reasonable aspect ratios. For example there are no characters found on license plates that are more than twice as wide as they are tall. The last condition removes all boxes are relatively too large compared to the area of the 8-connected region. This effectively removes most of 'graphics' in the picture, such as long diagonal lines, circles, etc. All of the constants used above were determined empirically, and can be easily changed in the program.

Finally, all bounding boxes that do not belong to strings of characters of length 3 or more are removed. If there are no strings found of length at least 3, the condition of a membership is relaxed to strings of length 2 or more. While the detailed description of the method would be long, it is clear that a string will have three bounding boxes that form a straight line, and that are relatively close to each other (distance between them is less than their own width). Figure 4 contains a snapshot of a result with the bounding boxes after this last step of box elimination.

4. Genetic Algorithm

To find a license plate in the image the problem is expressed as an optimization, and a genetic algorithm is used to solve it [1,2,3,8]. The input to our genetic algorithm routine is the edge image, and the set of bounding

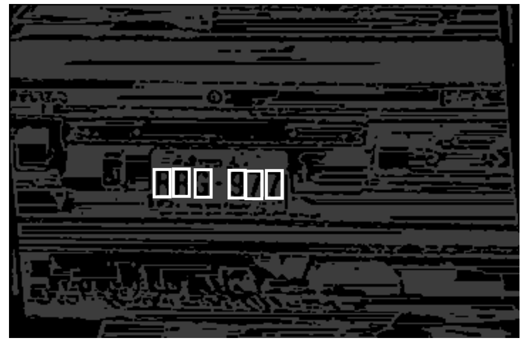

Figure 4. The characters identified in the test image.

boxes of candidate characters. The score (objective) function is based on the content of this edge image and the properties of the boxes.

In the implementation, each chromosome represents one possible location of a license plate. Each chromosome has 4 genes, coordinates (x1,y1) for left upper, and (x2,y2) for lower bottom corner of the plate [9].

The function that scores a chromosome returns a value between 0 and 1, 1 denoting a perfect license plate, 0 denoting a bad plate. The scoring function is based on rating each plate in four categories. The rating for each category is also a number between 0 and 1, and each category has a different weight in the total rating. The ratings are:

rating1 = rating based on the dimension of the license plate

rating2 = rating based on how closely the rectangle of the license plate is represented in the edge image

rating3 = rating based on what area ratio of the license plate is covered by the bounding boxes of potential characters

rating4 = rating based on how close is the license plate centered around the bounding boxes it encompasses

total_rating = (weight1 * rating1 + weight2 * rating2 +weight3 * rating3 + weight4 * rating4) / (weight1 + weight2 + weight3 + weight4)

The weights for the 4 ratings have been chosen as 1.0, 10.0, 10.0, 3.0, which means that the dimension of the license plate has the smallest effect on the total score. The largest effect comes from rewards based on how closely the rectangle of the license plate fits the edge image, and the area of the license plate covered by the potential character boxes.

When calculating rating 1, based on the dimensions of the rectangle, we first check whether the rectangle intersects the boundary of the image. If it does, we return -1 as the result for the total rating. We also return -1 when the license plate is too small (i.e. width or height are below some thresholds). If the license plate has an

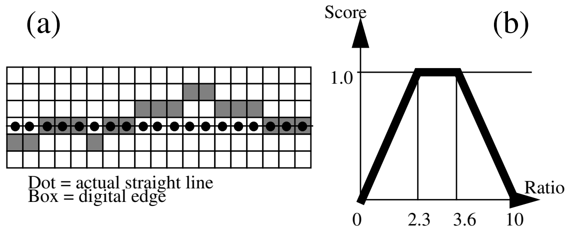

Figure 5. How to calculate the similarity score for horizontal lines, as in thecomputation of rating 2. (b) Computing the scoring function for the area ratio (rating 3)

unreasonable dimension ratio, we penalize such choice also by returning -1 as the total cost; he respective min and max thresholds for the dimension ratios are 1.0 and 5.0. When the dimensions and location of the license plates represented by a chromosome are acceptable, we return a reward based on how close is the dimension ratio of the license plate close to a desired dimension ratio, which is by default 2.0.

When computing rating 2, the closeness of the rectangle to edges in the image is measured by determining how closely each of the 4 lines of the boundary approximates the edges in the edge image. Each line is rated between 0 and 1, where 1 is the best approximation, which means that all pixels on the line are set in the edge image.To find the score for a line based on the information in the edge image, we calculate the score of each pixel on that line, which are also values between 0 and 1.

For horizontal lines we look for vertical pixel neighbors, and for vertical lines, we look for horizontal pixel neighbors. The score of each pixel is then calculated based on the determined distance to the closest pixel in the edge image. If the distance is 0, the score is 1.0. If the distance is 1, the score is 0.9. These values are taken from the empirically calculated array of scores 100,90,70,52,50,48,20,10,5,0,0,0....

This means that if a pixel has its closest neighbor more than 8 pixels apart, that pixel is assigned score of 0. For example, the score of the line in figure 4 is calculated as average of values:

0.9 0.9 1.0 1.0 1.0 0.9 1.0 1.0 0.9 0.9 0.9 0.7 0.7 0.9 0.9 0.9 1.0 1.0 1.0

which is 0.921. Since it is computationally expensive to determine the scores of each pixel, we use a dynamic programming approach. When a score is calculated for a given pixel for the first time, this score is saved; w it is necessary to calculate the score of the same pixel again, it can be had by table lookup.

Rating 3 is calculated using the ratio of the area of all of the bounding boxes found in the rectangle to the total area. If this ratio is between 0.23 and 0.36 (empirical), the score is 1.0. Otherwise the score linearly decreases to 0 based on how much the ratio is close to 0, or to 1. This scoring function is shown in figure 4.

Rating 4 is a measure of how well the characters are centred in the plate. First find all the bounding boxes that intersect the rectangle defined by the chromosome. Then calculate the average center coordinate of all these rectangles, and compute a distance between this center and the center of the rectangle defined by the chromosome. Then we assign the reward a value based on this distance, which is linearly decreasing as the distance increases.

Using the weighted sum of the above ratings for the objective function, the genetic algorithm filters through possible license plates and evaluates them. If the entire population contains the same chromosome and the score of this chromosome has not increased in the last generation of the new population, we say that the population has converged. In such case we simply mutate all the chromosome (except the first one), to obtain some randomness in our population. Parents are selected by roulette wheel, and the usual mutation and crossover operators are applied.

In figure 5 the output of the GA can be seen - a located license plate. The score determined for this plate was 0.9033.

5. Results and Future Work

Figure 5 shows the results of some of the experiments that were performed. A failure in locating a license plate can be attributed to many factors; most commonly the character location is unsuccessful, usually because the characters touch. However, the algorithm was successful in finding a plate on a white car, where the edge image contains almost no edges around the license plate.

Future work could focus on determining the positions of characters more accurately. Determining the correct positions of all characters in a license plate would dramatically improve the results of the license plate recognition.

6. References

[1] D. Beasley, D.R. Bull, R.R. Martin, An Overview of Genetic Algorithms: Part 1, Fundamentals. University Computing, 1993 15(2) 58-69.

[2] D. Beasley, D.R. Bull, R.R. Martin, An Overview of Genetic Algorithms: Part 2, Research Topics. University Computing, 1993 15(4) 170-181.

[3] J.L.R. Filho, P.C. Treleaven, C. Alippi, Genetic-Algorithm Programming Environments. Computer, June 1994, pages 28-43.

[4] L.A. Fletcher, R. Kasturi, A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images. IEEE transactions on pattern analysis and machine intelligence, Vol. 10, No. 6, November 1988, pages 910-917.

[5] J.R. Parker, Practical Computer Vision Using C, Wiley 1994, New York.

[6] Prakash, M. and Murty, M.N., A Genetic Approach For Selection Of Near Optimal Subsets Of Principal Components For Discrimination, Pattern Recognition Letters, Vol. 16, 1995. Pp. 781-787.

[7] Shen, J. and Castan, S., An Optimal Linear Operator for Step Edge Detection, Computer Vision, Graphics, and Image Processing: Graphical Models and Understanding, Vol. 54, No. 2, March, 1992. Pp. 112-133.

[8] M. Srinivas, L.M. Patnaik, Genetic Algorithms: A Survey. Computer, June 1994, pages 17-26.

[9] Toet, A., and Hajema, W.P., Genetic Contour Matching, Pattern Recognition Letters, Vol. 16, 1995. Pp. 849-856.

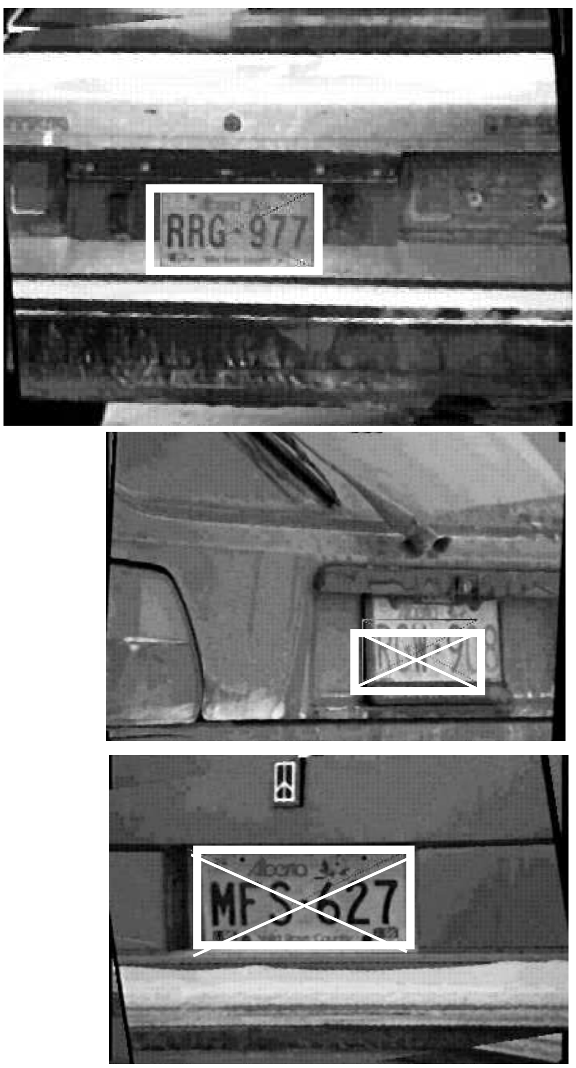

Figure 6. Various results from the license plate location system.