Journal of Computer and System Sciences 79 (2013) 245–255

Contents lists available at SciVerse ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

An algorithm for accuracy enhancement of license plate recognition

Lihong Zheng [a,*], Xiangjian He [b], Bijan Samali [b], Laurence T. Yang [c]

[a] School of Computing and Mathematics, Charles Sturt University, Australia
[b] Faculty of Engineering and IT, University of Technology, Sydney, Australia
[c] Department of Computer Science, St. Francis Xavier University, Canada

ARTICLE INFO

Article history:
Received 5 January 2011
Received in revised form 14 November 2011
Accepted 1 May 2012
Available online 9 May 2012

Keywords:
License plate detection
Statistical features
Haar-like features
Image segmentation
Blob detection algorithm
OCR

ABSTRACT

This paper presents an algorithm for extraction (detection) and recognition of license plates in traffic video datasets. For license plate detection, we introduce a method that applies both global edge features and local Haar-like features to construct a cascaded classifier consisting of 6 layers with 160 features. The characters on a license plate image are extracted by a method based on an improved blob detection algorithm for removal of unwanted areas. For license plate recognition (i.e., character recognition), an open source OCR is modified and used. Our proposed system is robust under poor illumination conditions and for moving vehicles. Our overall system is efficient and can be applied in real-time applications. Experimental results are demonstrated using a traffic video.

1. Introduction

License Plate Recognition (LPR) has found numerous applications in various areas. It can be used for automatically identifying vehicles in a car park, for vehicle access control in a restricted area and for detecting and verifying stolen vehicles. A LPR system consists of two major components: license plate detection and character recognition.

License plate detection is a crucial step in a LPR system. The quality of a license plate detection algorithm influences the accuracy of license plate recognition. In addition, many factors can affect the accuracy and efficiency of license place detection. For example, the quality of license plate detection may be degraded due to ambient lighting conditions, image perspective distortion, texts on images and so forth. Most of the existing license plate detection algorithms are restricted by various controlled conditions such as fixed backgrounds [1], known color [2], or designated ranges of the distance between cameras and vehicles [3]. Therefore, it remains to be a challenging problem regarding to detecting license plates under complex environments.

There are two types of approaches for character recognition. One type of approaches uses commercial Optical Character Recognition (OCR) software to recognize the characters [4]. The other type of approaches uses learning based methods to identify the characters [5]. Both approaches have reached to as high as 99% of accuracy under controlled conditions and when the cameras are mounted in fixed locations with no mobility. However, when illumination conditions are poor or when the vehicles are moving, the accuracy is impaired.

In this paper, we present a LPR system containing the following contributions. Firstly, we introduce a cascaded classifier [7] that uses the AdaBoost algorithm [6] to detect a license plate accurately. Then, we propose a method that uses

* Corresponding author.
E-mail addresses: lzheng@csu.edu.au (L. Zheng), xiangjian.he@uts.edu.au (X. He), bijan@eng.uts.edu.au (B. Samali), ltyang@stfx.ca (L.T. Yang).

0022-0000/$ – see front matter
http://dx.doi.org/10.1016/j.jcss.2012.05.006

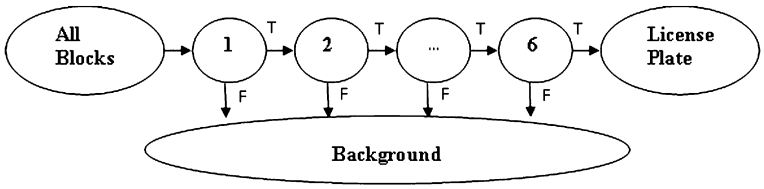


Fig. 1. Process of constructing a cascaded classifier [7].

a blob detection algorithm [8] and identification algorithm to efficiently improve the accuracy of character segmentation. To validate our LPR system, we apply an OCR for character recognition.

License plate detection algorithm as shown in [7] constructs a cascaded classifier of six layers for license plate detection. The classifiers of the first two layers are based on two global statistical features respectively and are trained through simple learning procedures. Then, the layers according to the remaining four classifiers are trained by AdaBoost learning algorithm based on local Haar-like features. The well-trained classifiers are ready to identify the location of license plate in the whole input image. The classifiers based on global features decrease the complexity of detection algorithm. The classifiers based on local Haar-like features further improve the detection rate and reduce the false positive rate. It shows better performance than [9] which applied discrete wavelet transform (DWT) to locate the license plate.

OCR has shown its advantage in recognizing printed document or text where the background has no or very little noise. However, license plate images captured in real-time usually contain heavy noise and are with complex backgrounds. Therefore, it is challenging to correctly remove the boundary of a license plate and segment the characters on the license plate. No matter which OCRs are used, the recognition accuracy will be significantly reduced if the characters are not properly segmented. In recent years, there have been many algorithms developed for character segmentation on license plates. These algorithms include the works shown in [10] based on Hough transform, in [11] based on horizontal and vertical projections, and in [12] using the Operator Context Scanning algorithm to increase the processing speed and accuracy. However, it is still a problem for accurate and real-time character segmentation under the situations when license plate boundaries are connected to inside characters, characters are connected to each other, and characters are broken [13,14].

In this paper, we adopt our recent approach that can correctly and efficiently segment the characters on license plates. This approach is based on some well-known techniques including image binarization, edge detection, image projections, and connected component analysis. We also integrate a blob detection algorithm into the method for character segmentation.

The rest of the paper is organized as follows. In Section 2, we give an overview of license plate detection and localization. Two global statistical features and local Haar-like features are reviewed in Section 3. The techniques for character segmentation, including color reverse, edge detection, and blob extraction, are presented in Section 4. Experimental results are demonstrated in Section 5. Comparison and discussion are described in Section 6. Finally, we conclude this paper in Section 7.

2. License plate detection

As shown in [7], the basic idea of the detection algorithm is to use a variable scanning window moving around on the input vehicle image. At each position, the image area covered by the scanning window is classified using a pre-trained classifier as either a license-plate area (a positive decision) or a non-license-plate area (a negative decision). The classifier used in this algorithm [7] is a significant extension of Viola and Jones' work shown in [15] to license plate detection.

In this algorithm [7], a six-layer cascaded classifier is constructed to increase the detection speed, in which the first two layers are based on global features and the last four layers are based on local Haar-like features. The classification process can be taken as a degenerate decision tree containing multi-layer classifiers as shown in Fig. 1. A positive result from the first classifier triggers the evaluation of a second classifier. A positive result from the second classifier triggers a third classifier, and so on. A negative outcome at any layer leads to the immediate rejection of the image region (block). In other words, those image regions that are not rejected by the initial classifier will be processed by a sequence of classifiers. If any classifier rejects a selected image region, no further processing will be performed for this region. It is commonly seen that, for a given vehicle image, the majority of evaluated image regions are negative. Therefore, the cascaded classifier shown in Fig. 1 attempts to reject as many negatives as possible at the earlier stages. As its consequence, this cascaded classifier leads to fast license plate detection. It is also worth to note that the detection algorithm (classifier) acts on the vertical edge maps of input vehicle images rather than on the raw image intensity values. This further enhances the efficiency of the cascaded classifier.

In the following, the algorithm [7] is further described in two aspects: training and testing. Training is a process that the classifier is learning to make correct decisions using pre-classified samples. Testing is a process to use the pre-trained classifier to classify individual image blocks.

2.1. Training

To obtain the cascaded classifier which can make correct decisions, pre-classified positive samples (images containing license plates) and negative samples (images containing non-number-plates) are selected for training. In our experiments, all positive samples are obtained through manually labeling license plate areas in vehicle images captured under various conditions. All negative samples are extracted from vehicle images from which license plates have already been removed manually.

The individual classifiers that together construct the cascaded classifier are trained independently. Recall that the classifiers on the first two layers are based on global features, and the classifiers on the other four layers are based on local Haar-like features.

To train the classifier on the first layer, the value of the first global feature, called Edge Density (to be defined in Section 3), is computed for each input sample. Statistical methods are used to select a threshold that can correctly classify all positive samples (i.e., using the selected threshold, the edge densities of all positive samples are on the "positive" side). Note that the threshold is not unique. Also note that, for a given threshold, some negative samples may be wrongly classified as "positive", i.e., some non-number-plate blocks may be classified as a "license plate". These are referred as false positives. Hence, a threshold which can correctly classify all positive samples and produce the least number of false positives is selected.

For the second layer classifier, another global feature is employed. It is called the Edge Density Variance that will also be defined in Section 3. All input samples used to train the classifier on the second layer are from "positive" classification outcomes using the first classifier after training. Since the false positive rate of the first classifier is usually non-zero, samples which are classified as "positive" by the first classifier contain both real positive samples and some negative samples. By properly selecting another threshold based on the second global feature, we can classify all positive samples as "positive" and produce minimum false positives. The classifier on the second layer based on the second global feature is thus obtained. Again, the training of the second classifier is implemented using statistical methods similar to those used for the first classifier.

Similarly, the samples used to train the classifier on the third layer are those samples which are filtered as "positive" by the classifiers on the first two layers. Unlike the first two layers, which are both based on global features, the classifiers on the third through sixth layers are all based on local Haar-like features. More details about the features will be given in Section 3. We will find that, within any image area (region), the total number of Haar-like features is very large and much larger than the total number of pixels within the area. To ensure fast classification, the AdaBoost learning algorithm is used to select best-performing classifiers (called weak classifiers), each based on a Haar-like feature, and to combine these multiple weak classifiers to construct one classifier (referred as strong classifier). This procedure is completed in multiple rounds. In each round, an optimal weak classifier is selected. The AdaBoost algorithm introduces a weight for each sample. Through continuously increasing the weights of "hard" samples in each round and selecting the corresponding best-performing weak classifiers until the constructed strong classifier meets the predefined accuracy requirement, a strong classifier is then constructed and the training on this layer is finished.

Similarly, the samples classified as positive ones by the third layer are input to the fourth layer, and so forth. Finally, a six-layer cascade classifier is constructed.

Since both global and local features in this algorithm are generated from vertical edge maps of input images, the vertical edge maps of images are computed first before any feature is extracted. Section 3 defines a new vertical edge map on which our detection algorithm acts. Furthermore, for the convenience of training, all sample image regions are scaled to 48×16 pixels before they are used for training. This size is selected as roughly the size of the smallest license plate which is still recognizable in a vehicle image.

2.2. Testing

After the cascaded classifier has been trained and satisfies pre-defined classification accuracy, it can be used for license plate detection.

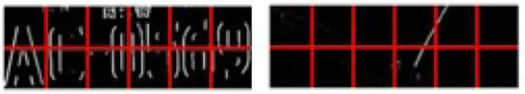
In the testing procedure, an input image is selected and a scanning window moves around the whole image space with a horizontal step of half the window width and a vertical step of half the window height. At each position, global and local features of the sub-region (that is covered by the scanning window) are extracted as needed and then the trained cascaded classifier acting on those features classifies the image region covered by the scanning window as either positive, i.e. a "license plate", or negative, i.e., a "non-license-plate". This procedure proceeds through the whole image until all image regions have been scanned.

Furthermore, since the detection algorithm acts on vertical edge maps of input images, vertical edge map detection is first performed before the detection begins.

To detect license plates of multiple sizes, the detection is carried out using multiple scales. The size of basic scanning window is set to be 48×16. The window size is then scaled up to 300×100 with a scaling factor of 1.2. This design makes it possible for our algorithm to detect license plates with various sizes. Fig. 2 gives an example that shows detectable license plates using our algorithm that have approximately the minimum size of 48×16 pixels (see Fig. 2(a)) and maximum size of 300×100 pixels (see Fig. 2(b)). In both cases, the vehicle images have 512×384 pixels.

(a) (b)

Fig. 2. Examples of small and large license plates.



(a) License plate (b) Background

Fig. 3. Distribution of vertical edges in two different regions.

In our detection algorithm, image sub-regions covered by the scanning window at each position are classified by a pre-trained classifier based on feature values obtained from each region. After the preprocessing for vertical edge detection, all vehicle images with various appearances are converted to vertical edge maps which are black and white images represented in grey-level intensities indicating the strength of the vertical edge at each image pixel. We can hence employ Haar-like features similar to those that have been successful in the systems presented in [15] for license plate detection. Since only local features are used in [15], classifiers based on such simple features need many features. This, in turn, makes the system computation expensive and unstable. In our algorithm, in addition to local Haar-like features, we also bring in global statistical features. We will describe these two kinds of features in detail in the following section.

3. Global and local features

In Sections 3.1 and 3.2, two global features as defined in [7] are reviewed and they are Edge Density and Edge Density Variance. In Section 3.3, Haar-like local features [6,7] are discussed. The regions of license plates have some common, obvious and simple characteristics. Firstly, a license plate region usually contains rich edge information due to the presence of text. Secondly, most edges in the region of a license plate are vertical edges. Thirdly, the vertical edges are distributed relatively uniformly in the region of a license plate. Therefore, we include the two global edge features for license plate detection to address the above three characters. These two features describe the global distribution of edges in the region containing a license plate, and are described in more detail in the following.

3.1. Edge density [7]

The edge density describes an image region as a whole. It is defined as

$$D_E = \frac{1}{N}\sum_i \sum_j E_V(i,j), \tag{1}$$

where $E_V(i,j)$ is the magnitude of the vertical edge at location (i,j), and N is the number of non-zero vertical edge pixels in the image region. The vertical edges can be computed using

$$E_V(i,j) = \begin{cases} G(i,j), & \text{if } 45^\circ \leqslant \alpha(i,j) \leqslant 135^\circ, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

where $G(i,j)$ and $\alpha(i,j)$ represent the gradient magnitude and edge angle at location respectively.

Recall that, during training, the size of the block is fixed to be the size of sample images, which is always 48×16. During the testing procedure, the size of image region varies according to the change of the scale of the scanning window.

The Sobel gradient operator is employed to produce gradient map, where the resulted gradient magnitudes are normalized by the maximum gradient strength in the image.

3.2. Edge density variance [7]

Besides the abundant edge information, note that the foreground characters in a license plate are usually distributed with relatively even interval. As its consequence, the gradients and hence the corresponding vertical edges in the block of

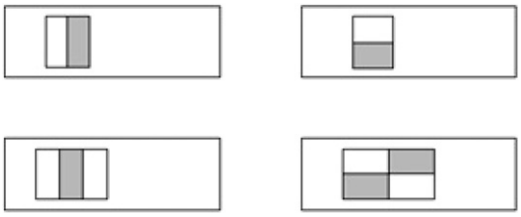


Fig. 4. Four types of Haar-like features [7].

a license plate are distributed more evenly with similar strength, compared with other regions. Fig. 3 gives examples of foreground and background images.

To obtain the density variance feature, a block is divided into 12 equal-sized sub-blocks as shown in Fig. 3. Let g_i denote the mean value of the vertical edge strength at i-th sub-block, and g denote the mean value of the vertical edge strength of the whole block. Then, the density variance of the block, denoted as V_G, is defined as

$$V_G = \frac{\sum_{i=1}^{n} |g_i - g|}{n \cdot g} \tag{3}$$

where n is the number of the sub-blocks, e.g., $n = 12$ in above example.

The density variance defined above, which takes value from 0 to 1, is a ratio to the mean vertical edge strength of the block. In this way, no matter whether the vertical edge is strong or weak, the density variance keeps low as long as there are similarly strong or weak vertical edges distributed evenly through the block.

3.3. Local Haar-like features [6,7]

The Haar-like features are originated from Haar basis functions. They consist of a number of rectangles covering adjacent image regions (see Fig. 4). The value of a Haar-like feature is the difference between the average of the pixel values (in our algorithm in this paper, they are the vertical edge strengths) in white rectangles and grey rectangles. The Haar-like features represent the local properties of the image block because every white rectangle is adjacent to the grey ones.

A Haar-like feature is determined by its type and by the size and position of the rectangles in the image region. These rectangles have the same size and shape for computing a single Haar-like feature and are horizontally or vertically adjacent. Since the number, size and position of the rectangles in the image region are all variable, there will be many Haar-like features in an image region. We use four types of Haar-like features [7] in this paper as shown in Fig. 4. The value of a two-rectangle feature is the difference between the sums of the pixel values within two rectangular regions which are vertically or horizontally adjacent (see the upper two figures in Fig. 4). The value of a three-rectangle feature is the sum of pixel values within two outside rectangles subtracted from the sum of pixel values in a center rectangle (see the bottom-left figure in Fig. 4). A four-rectangle feature value is computed as the difference between sums of pixel values in the diagonal pairs of rectangles (see the bottom-right figure in Fig. 4). These rectangles can have any size and position, as long as they are located within the image region.

Haar-like features obtained in the above way form a set of over-complete features [15], which can capture the interior structure of an object and are invariant to certain transformations. However, the dictionary consisting of all the Haar-like feature values is too large. There are hundreds of thousands of such Haar-like features in a 48×16 image region. Clearly, it is too time-consuming to compute all the features for classification. A challenge is then to find the best features.

The AdaBoost algorithm [7] is a good choice for selecting a small number of features from a very large number of potential features. After being trained, the selected best-performing classifiers (called weak classifiers) are combined to construct a strong classifier. The basic idea of the AdaBoost algorithm [7] is as follows. After constructing a weak classifier, the samples are re-weighted with higher weights assigned those which are incorrectly classified. Then, the next weak classifier is trained with the re-weighted samples. A number of weak classifiers are trained in this way till the given false positive rate is reached. The final classifier (called a strong classifier) is constructed by combining these weak classifiers using a set of weights. These weights are determined by classification error of each weak classifier.

4. Character segmentation of license plates

Once the license plate area is found, the following step is to find the characters contained in the image. The procedures of character segmentation include three different tasks. The first one is character height estimation. The upper and lower boundaries of character are located and used to obtain the character height. Then, we move to estimate character width and segment license plate. The final step is to construct the labeled character segments based on a block extraction algorithm for verification of character segments.

Fig. 5. Images samples of located license plates.

4.1. Character height estimation of license plates

This step contains three parts: color reverse, vertical edge detection and horizontal projection histogram.

4.1.1. Color reverse

The license plates in the New South Wales (NSW) State, Australia have many different formats, colors, and alignments. For instance, white in black, black in white, black in yellow, etc., are commonly used color combinations. Color reserve step is necessary before we assign right color (i.e., black) to the characters of a license plate and hence obtain a correct binary image of the license plate. It makes the color of the characters on a license plate be black. It is done based on a statistical analysis of edges. Given the located image, we pick l horizontal symmetrical lines on the license plate image. The color index C_I is calculated as the average amount of the cross points (where pixel value changes between black and white) along each line in horizontal direction. Cross point number is increased by one if there is a foreground point.

$$C_I = \frac{1}{l} \sum_j \sum_i F(i, j), \tag{4}$$

where j is from 1 to l and i is from 1 to N, and $F(i, j)$ is 1 if it is foreground point at location (i, j). If Color Index value is over a statistical selected threshold, the candidate image is labeled as an image to be converted. Otherwise keep the original value for next step.

Therefore, all candidate license plates are assumed as black-on-white, and the consistency of interested area is kept.

4.1.2. Vertical edge detection

Once a license plate area is detected and located, a rectangular area of the license plate obtained by using the algorithm described in Section 2 may not be big enough to cover the whole license plate. We hence perform a simple area enlargement computation so that the enlarged area will fully contain the license plate. Fig. 5 shows some examples of final rectangular areas of the license plate images.

Given that the located area shows stronger connectivity in vertical direction than horizontal direction, we perform vertical edge detection on the license plate images as shown in Fig. 5 through the computation of horizontal gradient. At each pixel, use the Sobel mask of [−3 0 3; −10 0 10; −3 0 3] to compute the horizontal gradient value [16]. Then, use the Otsu [17] method for binarization to obtain the vertical maps. More about the Otsu method will be addressed in Section 4.2.1 below. The edge pixels are represented using white pixels and other pixels using black pixels. Fig. 6(a) shows the vertical edge pixels on the images in Fig. 5.

4.1.3. Horizontal projection histogram

Although projection histogram is not a new concept, it is used here to find the upper and lower bounds of a license plate after the vertical edge are obtained. We perform a horizontal projection to find the top and bottom position of characters. The value of a histogram bin is the sum of the white pixels along a particular line in horizontal direction. When all values along all lines in the horizontal direction are computed, the horizontal projection histogram is obtained. The mean value of the histogram is then used as a threshold to decide the upper and lower bounds. The middle area of which the histogram segment is bigger than the threshold is recorded as the area bounded by the upper and lower bounds. The horizontal projection histograms of the six images shown in Fig. 6(a) are displayed in Fig. 6(b).

Finally, the distance between upper and lower boundaries is recorded as the height value of characters.

4.2. Character width estimation of license plates

After the area bounded by the upper and lower bounds of a license plate is found, the areas above the upper bound and below the lower bound are removed. The remaining area without the upper and lower boundaries of the license plate is considered for character segmentation on the license plate. Image binarization and vertical projection are two steps for segmentation here. Each segment here may contain one or two characters. Note that we use segmentation results to estimate the width of the characters on the license plate for the following processes. Each segment does not need to be accurate to contain exactly one character.

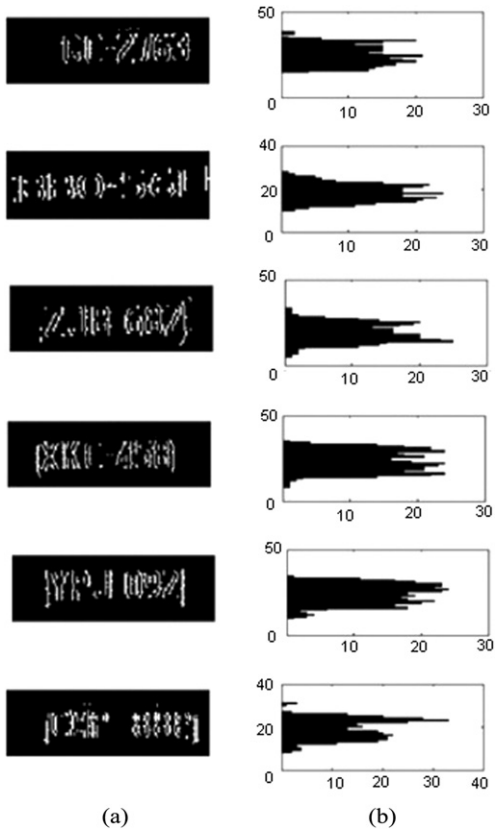


Fig. 6. (a) Vertical edge maps of images in Fig. 5. (b) Horizontal projection histogram.

Fig. 7. Binarized images of the images in Fig. 5 with upper and lower bounds removed.

4.2.1. Image binarization

As well known, image binarization is to change grey values of an image into binary values and re-represent the image as a binary image accordingly. Image binarization highlights the pixels of interest and suppresses the background pixels. The simplest way for image binarization is to choose a threshold value, and classify all pixels with values above this threshold as white (255 grey value), and all other pixels as black (0 grey value). Otsu [17] gave an idea to select a good threshold globally. Otsu's method is based on an analysis of the gray scale level histogram of the whole image and selects an optimal threshold for a given image by maximizing a discriminant criterion, i.e., the separability of the resultant classes in gray levels. Fig. 7 shows the results of binarization on the images in Fig. 5 after cutting the upper and lower boundaries of the license plates.

4.2.2. Vertical projection histogram

We perform a vertical projection to find the gaps between characters on a license plate. The value of a histogram bin is the sum of the white pixels along a line in vertical direction. When all values along all lines in the vertical direction are computed, the vertical projection histogram is obtained. Fig. 8 show the vertical projection of images in Fig. 7.

Based on the results of vertical projection, each license plate is separated into blocks horizontally by the zero points in the projection histogram. Fig. 9 shows the segments (or blocks) of the images in Fig. 5 obtained from the results of vertical projection after upper and lower bounds removed.

To estimate the width of a character after the segmentation process above, we take into account only widths of all blocks except the two smallest blocks and the two biggest ones. The averaged widths of these blocks are used as the estimated width of characters.

The estimated height and width of characters will be used in the following step for blob checking (extraction).

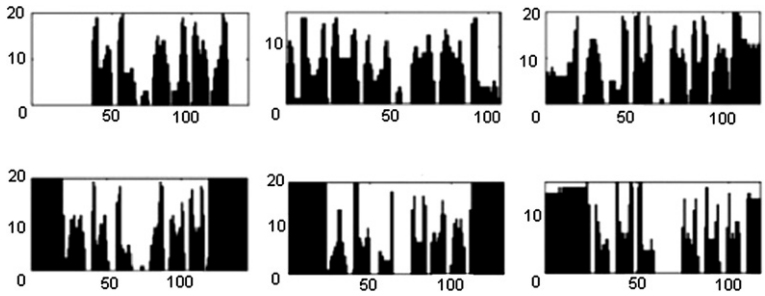


Fig. 8. Vertical projection of the images in Fig. 7.

Fig. 9. Character segments of license plates.

4.3. Distinguishing character blocks by blob extraction

As shown in Fig. 9, not all the segments obtained in previous procedure are blocks containing characters. Some blocks actually contain nothing but vehicle background and/or left and right bounds of the license plates. The statistical features of such background are very similar to those of the characters. So these features are not good enough to distinguish background blocks from character blocks. Roughly, only 70% character blocks can be identified successfully. Therefore, another method is sought to increase the successful rate. This method aims to keep only the blocks containing characters and removes those blocks that do not contain characters. By doing this, a blob extraction algorithm is applied for tracking character blocks on the left and the right side of a license plate. The left and right boundaries of the license plate will also be located and removed. This step consists of two parts: blob detection and blob checking algorithms.

4.3.1. Blob detection algorithm

Blob detection algorithm [8] is another kind of image segmentation techniques that labels the pixels in an image as belonging to one of many different groups. It is a sort of extension method based on Connected Component Analysis (CCA) algorithm [8]. Blob detection algorithm is limited to the binary image only in our system. It includes region labeling, connected-component labeling, blob discovery, and region extraction. Blobs can be counted, filtered and tracked.

Well-known algorithms for blob detection can be sorted into two types: the pixel-based algorithms [8] and the run based algorithms [14]. Given an image, they both assign labels to a pixel such that adjacent pixels of the same features are assigned the same label. The image is copied into a small or large array. The pixel-based algorithms do multiple scans. On the other hand, the run based method only search connected pixels in one run. In this paper, for the fast processing speed the first approach is applied in blob finding process. The brief idea is described as follows.

First, a label counter is created and initialized to one. Then, the binary image is scanned from left to right and from top to bottom.

- For every pixel, check the northeast, north, northwest, and west pixel for 8-connectivity for a given region criterion (i.e., intensity value of 1 in binary image).
  - (a) If none of the neighbors fits the criterion then assign to region the value of the region counter. Increase the region counter by one.
  - (b) If only one neighbor fits the criterion assign the pixel to that region.
  - (c) If multiple neighbors match and are all members of the same region, assign the pixel to their region.
  - (d) If multiple neighbors match and are members of different regions, assign the pixel to one of the regions (it doesn't matter which one). Record all these regions as 'equivalent regions'.
- Scan image again, and assign all equivalent regions the same region value.
- Scan stops until there are no unlabeled pixels in the image.

Fig. 10 shows the extracted blocks in red boxes from the located license plates.

4.3.2. Blob checking algorithm

Note that there are still some missing blocks where characters are connected with the boundary. Next, we move to identify the blocks based on character block checking criteria, so that the non-character blobs can be removed after the process. There are many different conditions that can be selected as the checking criteria. The examples of the conditions are the upper and lower boundary positions, the blob area, the blob perimeter, the mean, the density, the estimated character

Fig. 10. Extracted blocks from images in Fig. 5.

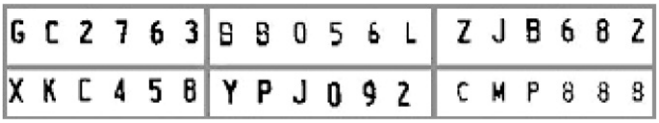


Fig. 11. The binary and enlarged license plate images.

width and height as shown in Sections 4.2.1 and 4.2.2, the height to width ratio, and maximum characters amount in one license plate. Most of these can be included in the character block checking criteria.

Following are the criteria used in our algorithm.

Rule 1. *If the upper boundary of one blob is smaller than the overall UPBound, or if its upper boundary is bigger than the overall LOWBound, the candidate blob is a non-character one.*

Rule 2. *If the height to width ratio of a blob is bigger than ThHigh* (*i.e.* 0.8) *or smaller than ThLow* (*i.e.* 0.3)*, the candidate blob is assumed to be a non-character one.*

Rule 3. *If the area of blob is bigger than one sixth of the whole image size, it needs further segmentation.*

Rule 4. *If any two neighbor segments obtained of which the widths are* 2/3 *smaller than the character average width on the same license plate, they are linked* (*combined*) *into one segment.*

Rule 5. *If any segment obtained in the previous step having width* 1/3 *bigger than the character average width, it is separated into two segments.*

The blob recorded as a possible character block must match all the conditions to be recorded as a character block and retained. Otherwise, the blob is regarded as a non-character block and hence is removed. After all blocks on the given license plate are checked, the process completes. Then, the selected blobs (characters) are enlarged and grouped together after the unwanted non-character areas (NCRs) of the license plate are removed.

5. Experimental results

To test the performance of our proposed detection algorithm, a total of 460 vehicle images are used for the experiments. 300 vehicle images are taken as training images that contain 305 visible number plates. The other 160 images are used as test images that contain 169 visible number plates. Vehicle images used in the experiments have been taken in various circumstances under various illuminations conditions and view angles. There is a great variety of colors, view angles and styles of number plates.

The negative samples used to train the classifiers based on global features are collected by randomly selecting 28,000 regions from 50 images which do not contain any license plate. The negative samples used in AdaBoost learning procedure are randomly extracted from a total of 220 vehicle images that do not contain any number plates.

In the experiments, a six-layer cascaded classifier is obtained. Each of the first two layers uses one of the global features defined in Section 3. On the last four layers, the numbers of the features in the strong classifiers are 19, 34, 47 and 58 respectively. So our final cascaded classifier has 6 layers and uses 160 features. This classifier is much simpler than Viola & Jones' classifier which has 38 layers and uses 6060 features.

In the experiments, of the 169 visible number plates in 160 testing images (in some images there is more than one number plate), 163 number plates are detected, giving a detection rate of 96.4% [7]. Meanwhile, only 8 false positive regions are detected. Since tens of thousands of background regions have been automatically generated during the testing procedure, this false positive rate is actually very low (less than 0.0001) [7]. Fig. 12 shows some of the detection results, where the detected license plates are marked by red boxes. From the examples, we may see that our algorithm can work under various complex environments, various illuminations, and various view angles. The algorithm can detect the license plates with various sizes, positions and colors. Fig. 12(a) shows examples of license plates with strong illumination. Fig. 12(b) shows the result of detecting multiple license plates in one image.

To test the segmentation algorithm, we apply the proposed algorithm on 587 license plate images with 3502 characters located by the license plate detection step. All characters are correctly segmented. Fig. 11 shows some of the characters segmented after binarization. The false positives (i.e., non-character areas that are segmented) are mainly due to the false detection of license plates. Only 7 out of total 594 license plate detected regions are non-license plates. This rate is $(594 - 587)/594 = 1.18\%$. Therefore, our correct segmentation rate is still very high and is about 98.82% even taking into account the rate of wrongly detected license plate regions. All candidate images are processed as one format as black characters on white plate. The binary enlarged license plate images are obtained by our proposed method and are sent to

(a) Single license plate in an image (b) Multiple license plates in an image

Fig. 12. Examples of detected license plates.

Table 1
Performance comparison.

References	Detection accuracy	Segmentation/Recognition accuracy
[18]	93.2%	95%
[19]	97.1%	96.4%
Proposed	96.4%	98.82%

Fig. 13. Comparison of clarity of original license plate with images developed using three alternative approaches. Notes: (a) Original image. (b) Traditional CCA method. (c) Active contour based method. (d) Our approach.

the OCR software (Tesseract) for recognition. With slight modification of the Open Source Tesseract OCR code, all correctly segmented characters on the license plates are recognized correctly.

A performance comparison with various methods is shown in Table 1. [18] uses projection based method and [19] applies hybrid binarization techniques to segment the license plate. Our license plate detection rate is 96.4% and the accuracy rate for license plate segmentation is 98.82% as shown in Table 1. Furthermore, our system can detect and recognize a 648×486 license plate image in real-time and within 0.1 seconds on a PC with Pentium 2.8 GHz CPU with a lot of rooms for optimization of our algorithms to be carried out in the future.

6. Comparison and discussion

In this section, an empirical evaluation of the proposed segmentation method(s) and other segmentation methods such as traditional CCA (connected component analysis) and active contour based method [20] is presented in details.

Three different approaches are tested on the same sample license plates. Fig. 13 shows one of the examples. It has been found experimentally that blob detection algorithm based approach has the best performance as shown in Figs. 13(b), 13(c), and 13(d).

Firstly, traditional CCA uses the connectivity between a pixel and its neighbors to label the pixel and merge into same group. Given an image, CCA assigns labels to a pixel such that adjacent pixels of the same features are assigned the same label. The image is copied into a small or large array. It does multiple scans to label the pixels as belonging to one of many different groups. However, if characters are connected each other, it is hard to separate them individually. Secondly, active contour based approach starts with some initial boundaries and iteratively modify them by applying some shrinking or expansion operations. It gets stuck in local minima states due to overlook minute features in the process of minimizing the energy over the entire path of their contours. It is a time consuming method. It takes more than eleven times of the time that CCA takes and ten times of the time that our approach uses. Lastly, blob detection algorithm based approach cut the character through the stroke connectivity and histogram projection. It detects the local peaks more efficiently. As shown in Table 2, our approach is developed to find the best interested area of clear characters finally.

7. Conclusions

In this paper, we have constructed a cascaded classifier consisting of 6 layers for license plate detection using both global edge features and local Haar-like features. The classifiers on the first two layers are based on the two global edge features. These two layers exclude more than 80% non-plate regions from further training or testing and hence greatly increase the detection speed in the next four layers [7]. The classifiers on the next four layers, trained by AdaBoost learning procedure,

Table 2
Comparison of experimental results with images developed using three alternative approaches.

Accuracy rate of LPR	Traditional CCA	Active contour	Our approach
Detection rate of license plates	96.4%	96.4%	96.4%
Accuracy of character segmentation	78.6%	76.1%	98.82%
Average time of segmentation (seconds)	0.196	2.21	0.204
Character recognition rate	98.7%	98.7%	98.7%
Overall recognition rate	74.5%	71.9%	94.03%

are based on local Haar-like features. In our algorithm, a real-time detection speed is achieved. With a small number of features, we can obtain very high detection rate with very low false positive rate even when the license plate detection algorithm is used under various complex environments.

Our algorithm for detection of license plates in a video does not use motion information and any other pre-determined knowledge, and is not restricted by any controlled conditions. Therefore, it is more flexible than any existing systems for license plate recognition, and can hence be applied to any areas where the cameras are mobile and the backgrounds are changeable.

Moreover, we have proposed a method to segment the characters of car license plates. This is a crucial work after the detection of license plates and before the use of OCRs for character recognition. The process is successful through the steps of character vertical height estimation, character width estimation, and segmentation of a license plate into blocks and the identifying of character blocks. The techniques used include image binarization, vertical edge detection, horizontal and vertical image projections, and blobs extraction and identification. Various well-known techniques are applied to come out with the innovative algorithm in this paper.

The Tesseract OCR software is used to test our results. The experimental results show a high accuracy of non-character area removal and significantly higher recognition rate after character is segmented. The overall recognition accuracy is about 94.03% after we remove all unwanted background areas. Therefore, the recognition accuracy has been enhanced.

References

[1] Hongliang Bai, Changping Liu, A hybrid license plate extraction method based on edge statistics and morphology, in: Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004), vol. 2, 2004, pp. 831–834.
[2] Sang Kyoon Kim, Dae Wook Kim, Hang Joon Kim, A recognition of vehicle license plate using a genetic algorithm based segmentation, in: Proceedings, International Conference on Image Processing, vol. 2, September 1996, pp. 661–664.
[3] Sunghoon Kim, Daechul Kim, Younbok Ryu, Gyeonghwan Kim, A robust license-plate extraction method under complex image conditions, in: Proceedings of the 16th International Conference on Pattern Recognition, vol. 3, 2002, pp. 216–219.
[4] Yefeng Zheng, Huiping Li, D. Doermann, Machine printed text and handwriting identification in noisy document images, IEEE Trans. Pattern Anal. Mach. Intell. 26 (3) (2004) 337–353.
[5] Bin Zhao, Yong Liu, Shao-Wei Xia, Support vector machine and its application in handwritten numeral recognition, in: Proceedings of the 15th International Conference on Pattern Recognition, vol. 2, 2000, pp. 720–723.
[6] Yoav Freund, Robert Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, in: Paul Vitnyi (Ed.), Computational Learning Theory, in: Lecture Notes in Comput. Sci., vol. 904, Springer, Berlin, Heidelberg, 1995, pp. 23–37, http://dx.doi.org/10.1007/3-540-59119-2_166.
[7] Huaifeng Zhang, Wenjing Jia, Xiangjian He, Qiang Wu, Learning-based license plate detection using global and local features, in: Proceedings of the 18th International Conference on Pattern Recognition (ICPR 2006), vol. 2, 2006, pp. 1102–1105.
[8] Berthold K. Horn, Robot Vision, 1st ed., McGraw–Hill Higher Education, 1986.
[9] Yuh-Rau Wang, Wei-Hung Lin, Shi-Jinn Horng, A sliding window technique for efficient license plate localization based on discrete wavelet transform, Expert Syst. Appl. 38 (4) (2011) 3142–3146.
[10] Yungang Zhang, Changshui Zhang, An new algorithm for character segmentation of license plate, in: Proceedings of the IEEE Intelligent Vehicles Symposium, 2003, pp. 106–109.
[11] Jian Zhang, Xiaoping Fan, Cailun Huang, Research on characters segmentation and characters recognition in intelligent license plate recognition system, in: Proceedings of the 25th Chinese Control Conference (CCC 2006), 2006, pp. 1753–1755,.
[12] Ioannis Giannoukos, Christos-Nikolaos Anagnostopoulos, Vassili Loumos, Eleftherios Kayafas, Operator context scanning to support high segmentation rates for real time license plate recognition, Pattern Recogn. 43 (11) (2010) 3866–3878.
[13] Xiaodan Jia, Xinnian Wang, Wenju Li, Haijiao Wang, A novel algorithm for character segmentation of degraded license plate based on prior knowledge, in: Proceedings of the IEEE International Conference on Automation and Logistics, 2007, pp. 249–253.
[14] Lifeng He, Yuyan Chao, Kenji Suzuki, Hidenori Itoh, A run-based one-scan labeling algorithm, in: Mohamed Kamel, Aurlio Campilho (Eds.), Image Analysis and Recognition, in: Lecture Notes in Comput. Sci., vol. 5627, Springer, Berlin, Heidelberg, 2009, pp. 93–102, http://dx.doi.org/10.1007/978-3-642-02611-9_10.
[15] Paul Viola, Michael J. Jones, Robust real-time face detection, Int. J. Comput. Vis. 57 (2004) 137–154, http://dx.doi.org/10.1023/B:VISI.0000013087.49260.fb.
[16] Rafael C. Gonzalez, Richard E. Woods, Digital Image Processing, 3rd ed., Pearson Prentice Hall, 2008.
[17] N. Otsu, A threshold selection method from gray-level histograms, IEEE Trans. Syst. Man Cybern. 9 (1) (1979) 62–66.
[18] Cheokman Wu, Lei Chan On, Chan Hon Weng, Tong Sio Kuan, Kengchung Ng, A Macao license plate recognition system, in: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, vol. 7, 2005, pp. 4506–4510.
[19] Jing-Ming Guo, Yun-Fu Liu, Jiann-Der Lee, License plate localization and character segmentation with feedback self-learning and hybridbinarization techniques, in: TENCON 2007 – 2007 IEEE Region 10 Conference, 2007, pp. 1–4.
[20] T.F. Chan, L.A. Vese, Active contours without edges, IEEE Trans. Image Process. 10 (2) (2001) 266–277.