Segmenting Characters from License Plate Images with Little Prior Knowledge

Wenjing Jia, Xiangjian He and Qiang Wu
iNEXT, Faculty of Engineering and Information Technology
University of Technology, Sydney (UTS), PO Box 123, Broadway, NSW, 2007 Australia
Wenjing.Jia-1@uts.edu.au, Xiangjian.He@uts.edu.au, Qiang.Wu@uts.edu.au

Abstract

In this paper, to enable a fast and robust system for automatically recognizing license plates with various appearances, new and simple but efficient algorithms are developed to segment characters from extracted license plate images. Our goal is to segment characters properly from a license plate image region. Different from existing methods for segmenting degraded machine-printed characters, our algorithms are based on very weak assumptions and use no prior knowledge about the format of the plates, in order for them to be applicable to wider applications. Experimental results demonstrate promising efficiency and flexibility of the proposed scheme.

1. Introduction

As an important element for intelligent transportation system (ITS), automatic license plate recognition (ALPR) not only recognizes and counts vehicles, but also distinguishes each as unique by recognizing the characters in the license plates via computer vision techniques. ALPR has found many real-world applications in various areas and has also been widely studied for many years.

As one of the most crucial modules in an ALPR system, segmenting characters properly from the extracted license plate regions plays an important role towards the success in efficiently recognizing the license plates. As it has been pointed out in a recent survey [1], most of the recognition errors in ALPR systems are not due to missing recognition power but due to errors or inaccuracy during segmentation stage. The research on fast and robust ALPR systems by Zhang et al. [13] also indicates that, for an ALPR system that works in a flexible manner under little restricted conditions, the success on character segmentation becomes even more significant, as license plate regions detected under such situation tend to have great variety in their appearances. This poses a challenging problem for OCR software to correctly recognize them. For an ALPR system working in complex environment, this is still a challenging task.

In spite of its importance for ALPR, character segmentation failed to attract as much attention as other modules in an ALPR system did (e.g. license plate detection module) until recent years when people's research interest on ALPR starts shifting towards developing fast and robust ALPR systems working in more flexible environment than previous ones. Such systems require efficient strategies for robust character segmentation in order to assure accurate character recognition. More and more research work focusing on segmenting characters for ALPR are reported in literatures [3] [4]. Methods dealing with problems such as rotation correction [4], dirt in plates [4], clustering character blocks [3], etc. for license plate character segmentation have been proposed. However, the adaptability of most reported methods is very limited due to the fact that nearly all methods rely on prior knowledge of license plates such as the format, spatial arrangement, the number of characters, etc. of the license plates [3]. Although using prior knowledge can improve the performance of segmentation algorithms, it however greatly limits the adaptability of the algorithms. As it is shown in Figure 1, these assumptions do not hold true for the license plates we deal with which are captured from (anonymous). It can be imagined that for a license plate character segmentation scheme to be adaptable to wider areas, the greater variety of license plates' appearance will only make the task more difficult. Although in many cases the format of targeted license plates are predefined and can become available for algorithm design, the variety of the format from plate to plate is still a big trouble for segmentation software unless frequent manual adjustment is allowed.

In this paper, we propose a set of simple but effective algorithms towards segmenting characters from extracted license plate image regions. Our final goal is to find an efficient and adaptable solution which does not rely on the prior knowledge of license plates. To our knowledge, similar solutions have not been seen anywhere in literatures. Our algorithms are based on two weak assumptions that, within a single license plate, 1) the spatial intervals between most (instead of all) adjacent characters are approximately equal;

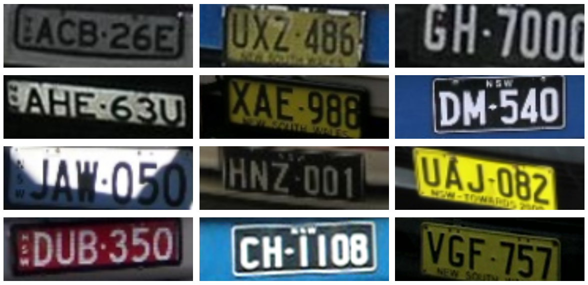


Figure 1. Examples of the extracted license plate images.

2) the width of the most strokes of characters is approximately the same. However, as it can be seen from the sample images shown in this paper, the spatial intervals between characters and the stroke width of characters of difference plates can be very different from plate to plate. Particularly, our focus has been put on segmenting characters from background license plates when characters are connected with the frames of the plates on one or more sides. We observe that when the number of characters are not fixed, this has become a challenge for nearly all existing systems.

The remaining of the paper is organised as follows. Section 2 discusses related work in literatures. Section 3 gives the details of our solution to license plate character segmentation step by step. Examples of experimental results are given in Section 4. The paper concludes in Section 5.

2. Related Work

Many methods for segmenting degraded machine-printed characters have been reported in literatures [2] [7]. These methods can be roughly classified into two major groups: straight segmentation methods and recognition-based segmentation methods [2]. Straight segmentation methods attempt to segment an image region that mainly contains characters into individual subregions purely based on analyzing the image without recognizing each component as a certain character and are preferable than the latter in most systems where recognition can be too computationally expensive to be extensively used. This group of methods can be further classified into projection-based methods [14] [15] and clustering-based methods [4].

It seems that character segmentation for license plate images should not be a challenging problem any more. However, it is still a big challenge for accurate and fast character segmentation under the situations when license plate boundaries are connected to the inside characters, characters are connected to each other, and characters are broken [5]. The efficiency, adaptability, and insensitivity to noise of some existing algorithms are low; and some algorithms, though with high robustness and adaptability, are computationally complex and hence are in unfavorable situation for being used in real-time applications. Moreover, nearly all existing methods for license plate character segmentation apply some assumptions about license plates which do not apply to our case. Prior knowledge that have been used for license plate character segmentation include: the number, the font and the size of characters, the spatial intervals between each characters, etc. For instance, the algorithm proposed in [5] used the character width as an important parameter for enhancing edges, and the number of characters when merging two adjoining sub-characters or splitting connected characters. As it is shown below, none of these assumptions holds true for the license plates we deal with. In our case, the number, fonts, formats, sizes of the characters in the license plates we are investigating can be very different, and cannot be assumed to be the same for all license plates.

Figure 1 shows some examples of the license plate images extracted from vehicle images. As it can be seen from the figure, 1) the color pattern of the foreground and background of a license plate, as well as the background color of the vehicles, can be very different from case to case; 2) although many license plates we deal with are with six major characters, it is not rare that they contain more or less characters; 3) the format of different license plates is also not unified, which results in the width and height of each character in license plates are not unified, neither are the spatial intervals between characters; 4) the view angles of each license plate can be very different; 5) the light intensities of each license plate is also unstable and unpredictable. All of the above facts have in fact made it a very challenging task to correctly segment characters from license plate images. To our knowledge, none of existing methods is able to be used directly to solve our problem.

In the following section, we present the details of our solution to segmenting characters from license plate images after they are extracted from still vehicle images or frames of video sequences.

3. Character Segmentation from Extracted License Plate Image Regions

As shown in Figure 2, the procedure of our character segmentation algorithms includes the following major steps: skew and slant correction, horizontal and vertical boundary removal, color polarity computation, and binarization.

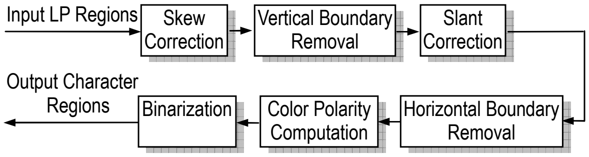


Figure 2. The flow of our proposed character segmentation algorithm.

3.1. Skew and Slant Correction

For an ALPR system operating in a flexible manner such as with moving instead of fixed cameras, it can be very common that a license plate appears to be tilted in images, as either *skew* (the deviation of the baseline from the horizontal direction) or *slant* (deviation of average near-vertical strokes from the vertical direction), as described in Figure 3. The skew and slant is typically introduced due to the fact that the license plate is not in horizontal direction and the plate plane is not vertical to the camera projection. Correcting such skew and slant distortion is very important for accurate character segmentation.

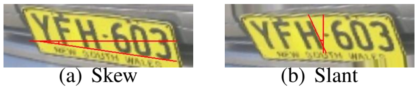

(a) Skew (b) Slant

Figure 3. Illustration of skew and slant of a license plate.

For correcting skew and slant distortion, the key is to detect the skew and slant angles correctly. In the literatures, several methods have been proposed to deal with this problem. Basically, there are methods based on the Hough transform [4] [15], the Radon transform [5] [11], the clustering technique [3] and so on. In our case, we found that methods based on Radon transform take relatively longer time to compute than the Hough transform based methods. Also, when characters are connected with the boundaries, clustering-based methods may fail to correctly cluster the characters. In [10], a fast algorithm is proposed for skew and slant correction in printed document images, where only the histogram of gradient orientation of the input gray-level image is employed. However, it turns out in most license plate images the frames of the plates may dominate the histogram of gradient orientation. Hence, we propose a simple and fast skew and slant correction method that combines the advantages of the traditional Hough line detection methods and the gradient direction based methods.

The Hough transform provides a robust technique for line detection in images. In most cases, there are either boundaries surrounding the license plate characters, or significant color difference between a license plate and its background, linear edges can be found in the detected license plate images. We apply the Hough transform based method to detect the angle of the longest near-horizontal lines and correct the license plate images accordingly. A complete skew detection and correction process is given as below, where the algorithm $VerticalCutting$ is given in the following subsection.

Algorithm 3.1: DESKEWDESLANT(I)

```
Given a 2-D gray-level image I.
1. Get the Otsu threshold [9] $T$ of $I_B$.
2. Apply Canny edge detector on $I_B$ to get its edge map $I_E$.
3. Use the Hough transform to detect the longest line and
its angle $\theta$ in $I_E$.
4. Rotate the source image $\theta$ degree.
5. Remove the top and bottom margins with Algorithm 3.2
and copy the resultant image into $I'$.
6. Calculate the HOG of $I'$ and locate the index
$\phi$ corresponding to the peak value of HOG.
7. Correct the slant so that for each pixel $(x, y)$
in the original image, the new coordinate $(x', y')$
in the deslanted image $I_{dst}$ is:
$\begin{cases} x' = x - (H - y) \times \tan(\phi) \\ y' = y \end{cases}$ ,
return $(I_{dst})$
```

For license plates where there are no such clear boundaries, i.e., the background color of license plates is similar with that of the vehicle body where the plate is mounted, the Hough space is accumulated from the "border line" edge points only. Edge pixels that can be viewed as border line edge pixels do not have connected edge pixels below their positions.

Two examples of Hough transform based license plate deskew are shown in Figure 4. In the figure, the red lines indicate the detected skew angles via Hough transform.

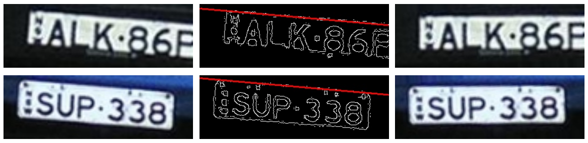


Figure 4. Examples of Hough transform based license plate deskewing.

3.2. Vertical Boundary Removal

Unlike the character segmentation problem in some machine-printed documents, where adjacent characters tend to touching or overlapping each other, the characters in license plates are rarely touched and overlapped with adjacent characters, because additional spacing is built into the fonts used for the purpose of being easily identified with human eyes. Though there could be such noise as mounting nails or dirt which connect two adjacent characters, the major obstacle, nevertheless, is to cut characters from the background when they appear to touch the frames of plates on one or more sides.

Figure 5 shows some of such examples. As it is shown in the figure, given the situation that characters of license plates touch either the frame of the plates or the backgrounds of vehicle bodies, character segmentation methods that are based on connected component analysis (CCA) and clustering are likely to fail.

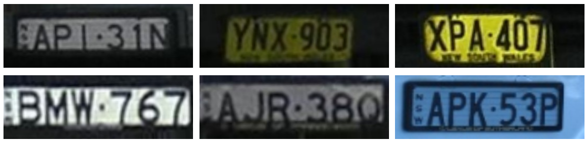


Figure 5. Examples of license plate images where characters appear to touch one or more sides of license plate frames. All license plate images have been corrected from skew distortion.

Hence, towards accurate license plate recognition it is essential to remove the top, bottom, left and right boundaries of detected license plates properly while keeping the character strokes intact. In this subsection, we first present a simple but effective algorithm to remove the top and bottom boundaries (or margins) surrounding the characters based on the outcomes of skew correction. The removal of left and right margins, i.e., horizontal boundary removal, is based on a different idea and will be detailed in the next subsection.

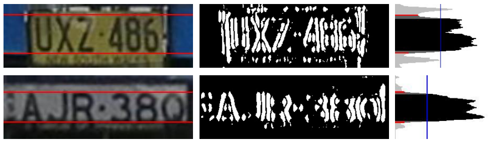


Figure 6. Using the horizontal projection of binarized vertical edge map to locate the top and bottom borderlines of characters in license plates. **Left** : original detected license plate images with located top and bottom borderlines marked as red lines. **Middle** : binarized vertical edge maps of the images in left column. **Right** : the horizontal projection of the binary image in the middle row, where the m values are marked as blue lines and the indices corresponding to the top and bottom borderlines are plotted in red color.

Our algorithm to locate the up and bottom borderlines of characters contains two major steps: vertical edge detection and horizontal projection. We firstly propose our definition and method for detecting *vertical edges* on the deskewed license plate images. Then, the Otsu's method [9] is used to binarize the resultant vertical edge map in order to obtain a binary image for projection. In the binarized vertical edge map, the edge pixels are represented by white pixels and other pixels by black pixels. Then the horizontal projection of this binary image is obtained. A simple strategy is used to find the top and bottom borderlines (and the middle line for multi-row plates) of characters. The detailed procedure of vertical boundary removal is given in the following algorithm. The horizontal projection histograms of the two images shown in the left column of Figure 6 are shown in the right column, where the up and bottom borderlines located are marked as red lines.

Algorithm 3.2: VERTICALCUTTING(I)

```
Given a deskewed 2-D gray-level image $I$ :
$I = I(x,y), x \in [0,W], y \in [0,H], I \in [0,255]$,
perform the following operations:
1. Obtain vertical edge map $I_{VE}$ according to $VerticalEdge$).
2. Get binarised vertical edge image $I_{BW}$ as:
$I_{BW}(x,y) = \begin{cases} 1, & \text{if } I_{VE}(x,y) \geq Th \\ 0, & \text{otherwise} \end{cases}$ ,
where $Th$ is the Otsu threshold of $I_{VE}(x,y)$.
3. Collect the horizontal projection $h(y)$ as:
$h(y) = \sum_x I_{BW}(x,y), y \in [0,H]$
4. Calculate the mean value of $h(y)$ as:
$m = \sum_y yh(y)$.
5. Determine the top and bottom bounds as:
$y_T = \max_y\{h(y) | \frac{m}{3} \leq h(y) \leq m\}, y = 0, \cdots, H/2-1$
$y_B = \min_y\{h(y) | \frac{m}{3} \leq h(y) \leq m\}, y = H/2, \cdots, H-1$
6. Copy the image bounded by $y_T$ and $y_B$ into $I\prime$:
$I\prime(x,y) = I(x,y)$, $0 \leq x \leq W$ and $y_B \leq y \leq y_T$.
```

Note that, the use of horizontal projection to separate the characters from background images is not a new concept. It however contains two innovative ideas here. The first is the projection in our method is computed from our newly defined vertical edge maps. Whether the edge at a pixel is vertical or not is judged by its edge angle $\alpha(x,y)$. Image pixels with edge angles greater than 45^o and less than 135^o are kept and recorded as vertical edges. As other non-character edge above and below the characters produce nearly no vertical edge information after the plate being rotated to horizontal direction, by doing so, we can successfully suppress the upper and lower margins that surround the characters. This can be seen from the binarized vertical edge maps in the middle column of Figure 6. The second innovation is our strategy of locating the upper and lower bounds from the horizontal projection profile. As it is in traditional methods that use horizontal projection to separate objects from background in vertical direction, the indices corresponding to zero points of the profile are taken as the bounds. However, the results can be quite inaccurate as there may be noise present which results in no zero points exit. Traditional solutions to this problem include introducing various fixed thresholds. We instead use an adaptive strategy where the bounds are defined as the indices corresponding to the two valley points of the projection that are closest to the indices producing the mean value, as shown in the right column of Figure 6. Using this strategy, the top and bottom bounds of all testing license plates have been correctly located. Also note that, this method can also be used for license plates which have more than one rows of major characters.

3.3. Horizontal Boundary Removal

Following our solution to the removal of top and bottom margins from the deskewed license plate images, in this subsection, we present an adaptive method to remove left and right boundaries of license plates. Our method is based on the analysis on the vertical projection of the binarized images of the vertically cut license plates. To obtain a binary image, Otsu's thresholding method is again used at first. Then the vertical projection of the binary image is computed. In order to correctly remove margins at the left and right side, it is needed to estimate the width of characters and the intervals between characters. To do so, traditional methods search valley points in the vertical projection histogram. This again can be affected by the noise that connects adjoining characters. We use a statistic way to find the "valley" points in the projection histogram. The value corresponding to valley points is then used to help to estimate the width and interval of characters. We first count the running-length of "peaks" and "valleys". Their mean length is then taken as the width of normal characters and the normal intervals between characters. Then, blob analysis is used on the vertical projection profile images to determine if a projection is generated from a character, based on the density, width and intervals with its neighbors. The detailed procedure is given as below.

Figure 7 shows three examples. As it can be seen from the figure, characters that are with different widths, at different interval length, and even connected with the background have been segmented correctly.

Algorithm 3.3: HORIZONTALCUTTING(I_{cut1})

```
Given a gray-level image $I_{cut1}$.
1. Get the binary image of $I_{BW}$.
2. Get the vertical projection profile $P_V$,
$P_V(x) = \sum_y I_{BW}(x,y)$, $x \in [0, W]$
3. Get the histogram of the values of $P_V$ as
for each $x \in [0, W]$,
  do $h[P_V(x)] \leftarrow h[P_V(x)] + 1$.
and find the valley value as the first valley of $h(i)$.
4. Deduct valley $val$ from the $P_V$ as:
$P_V(x) \leftarrow P_V(x) - val$.
5. Perform blob analysis on $P_V$
and remove non-character blobs.
6. Return the cut license plate images.
```

Note that there are two major differences between our method with the traditional methods. Firstly, when using vertical projection profile to determine the width of each character, instead of simply looking for zero points, we introduce a mechanism to determine the bin value for the "valley". Secondly, different from the other works that also use the blob analysis method for character segmentation, our blobs are blobs of vertical projection profile. This avoids the problem that noise may connect neighboring characters. Moreover, as we use the estimation of character widths and the intervals, our method can separate characters from the connected backgrounds or plate frames. In this method, there is only a very weak assumption that the strokes of characters in a single license plate are approximately the same and the intervals between most characters are the same.

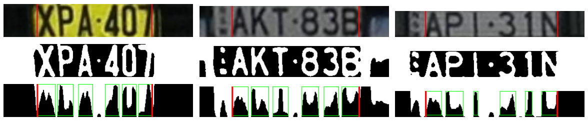


Figure 7. Using the vertical projection of binarized image and blob analysis to locate the left and right bounds of characters in license plates. **Top** : license plate images after vertical cutting with located left and right bounds marked as red lines. **Middle** : binarized images of the images in left column. **Bottom** : the vertical projection of the binary images in the middle row, where each blob is circled using green lines and the left and right bounds are plot in red color.

3.4. Color Polarity Computation

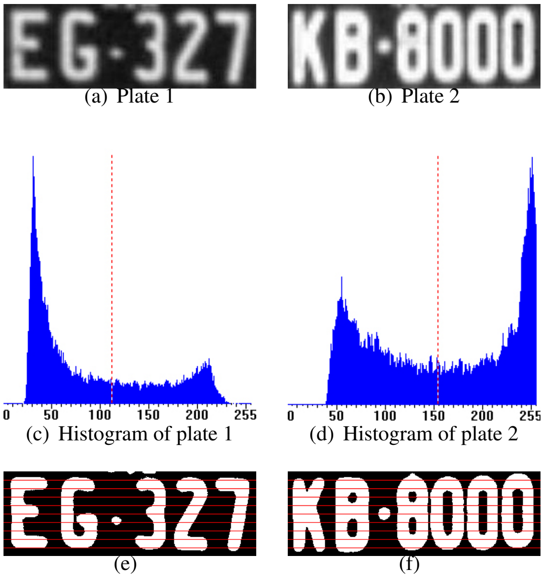
Before applying OCR software to recognize characters segmented, the character regions with boundaries removed need to be converted into a binary image, where pixels belonging to characters are highlighted and other pixels suppressed. As it has been mentioned earlier that vehicle license plates of different countries or regions, even different areas in a single country, can be of various foreground (text) colors and background colors. Each combination of foreground color and background color is referred as a "foreground-background color scheme" in this paper. When converted to grey-level images, all license plates appear as either dark foreground on a light background or light foreground on a dark background. Note that there are still two possibilities, that the foreground intensity can be either darker or lighter than that of the background, and therefore correspondingly the background must be lighter or darker, and the intensities of pixels in the lighter and darker areas, which originate from the different colors, still have a large variance.

In many OCR applications, it usually requires that binarized character images to be fed into the character recognition model, typically with a black foreground on a white background. Though the inverse style, i.e., a white foreground on a black background, may also be an acceptable setting for some OCR softwares, they unexceptionally have the requirement that all testing license plate images are with the common foreground-background color polarity.

For images that foreground and background intensities can be of two major trends (i.e., darker foreground on brighter background, and brighter foreground on darker background), this requires an efficient scheme to determine the color polarity of a license plate and then to invert colors from one style to the common one if necessary, before or after the binarization operation.

Among several algorithms proposed to conduct this task such as [8][6], there is one effective method, proposed by Lyu et al. [8], consists of color polarity classification and color inversion (if necessary), followed by adaptive thresholding, dam point labeling and inward filling. In this section, we propose a fast and efficient overlay text extraction technique, which is based on Lyus approach with some modifications for better performance. The overall procedure of proposed extraction method is shown as below. However, to our knowledge, all methods to solve this problem are based on the assumption that the number of background pixels are greater than that of the foreground. Some used the assumption that the percentage that the total area of characters occupies the entire area is standard, for instance 20% in the cases studied in [12]. Using histogram analysis to count the number of darker pixels and brighter pixels (or black pixels and white pixels if after binarization), people may easily determine the color polarity of a given binarized image region.

Note that, however, we cannot assume that the fonts and formats of characters from different license plates are the same. As its consequence, the percentage that the total area of characters occupies the entire license plate area will not always be in the same range. Figure 8 gives such an example, where both of the two license plates are from (an anonymous place).


(a) Plate 1

(b) Plate 2

(c) Histogram of plate 1

(d) Histogram of plate 2

(e)

(f)

Figure 8. Determining foreground and background of detected license plates. Both of the two examples of detected license plates are from the state of NSW of Australia.

As it can be seen from the Figure 8 that, even though it has been taken for granted by most people that the the number of foreground pixels is less then that of the background pixels, this is however not always the case. The foreground pixels of the license plate in Figure 8(a) takes about 34% of the total area, it is however 52% in the plate shown in Figure 8(b). Hence, in our system, we develop a more reliable and also fast mechanism to determine foreground and background which is detailed as below. Our goal in this subsection is to check polarity of the refined character regions and inverse the pixel intensities if needed so that the output character regions always contain black foreground (characters) on white background.

Algorithm 3.4: COLORPOLARITY(I)

```
Given a binary image $I$ and 11 scanning lines $\vec{L} = \{l_i\}$.
for each $l_i \in \vec{L}$
    do { if $l_i(x, y)$is black;
           do $\vec{B} = \{b_m\}$
         else if $l_i(x, y)$is white.
           do $\vec{W} = \{w_n\}$,
$b\star = \max \text{prob}(b_m)$
$w\star = \max \text{prob}(w_n)$
if $b\star >= w\star$ return $(foregroundisblack.)$
    else return $(foregroundiswhite)$.
```

Our method is based on the fact that in any license plate, the width of the strokes of different characters are relatively stable. This method operates on binarized results of detected license plate images. Otsu's method for computing an adaptive global threshold for binarization is utilized. The computed threshold is then used to binarize the detected license plates into two colors: black for pixels which grey-level intensity is less than the threshold, and white for pixels which grey-level intensity is greater than the threshold. We then investigate the color transition pattern of pixels on each of 11 scanning lines which pass through the detected area at an equal interval (one tenth of the total height of the detected area). Figures 8(e) and (f) show the scanning lines. The detailed procedure is given as follow.

4. Experimental Results

This section gives some examples of segmentation results using the method described above, which are shown in Figure 9.

Using OCR software such as ABBYY for recognizing characters on segmented results, all of the characters can be correctly recognized.

5. Conclusions

In this paper, towards developing an efficient and adaptive solution for segmenting characters from extracted li-

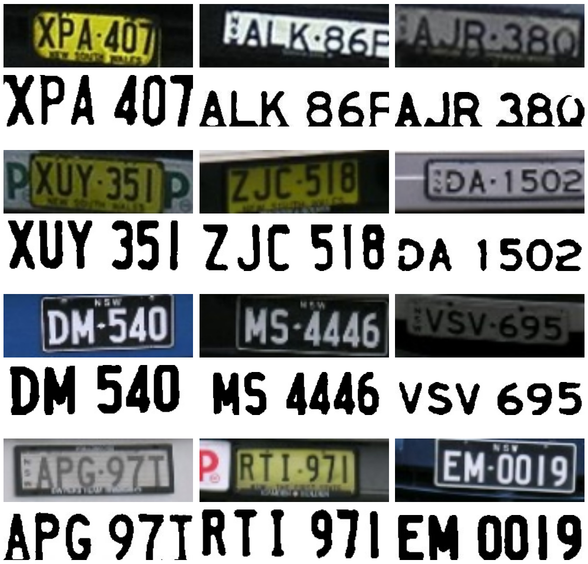


Figure 9. Results of character segmentation on various license plates images.

cense plate images, a set of simple but effective algorithms are proposed. Experiments have demonstrated that, without using the prior knowledge on the format of characters in a license plate, the character region still can be accurately segmented out from license plate background. Particularly, we looked at the segmentation problem when characters are connected with the background on one or more sides of license plates. The segmentation was performed without using the prior knowledge of the format of license plates such as the number of characters, the size of characters etc. Our solution proposes an effective and adaptive solution to license plate character segmentation, which in turn can contribute to enable a fast and robust ALPR system that operates in a flexible manner and under less restricted conditions.

6. Acknowledgements

The work in this paper was supported by UTS Chancellor's Postdoctoral Research Fellowship program.

We also appreciate valuable comments received from the anonymous reviewers.

References

[1] C.-N. Anagnostopoulos, I. Anagnostopoulos, I. Psoroulas, V. Loumos, and E. Kayafas. License plate recognition from still images and video sequences: A survey. *Intelligent Transportation Systems, IEEE Transactions on*, 9(3):377–391, Sept. 2008. 1

[2] R. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(7):690–706, 1996. 2

[3] H. Deng and X. Song. License plate characters segmentation using projection and template matching. *Information Technology and Computer Science, 2009. ITCS 2009. International Conference on*, 1:534–537, July 2009. 1, 3

[4] J.-M. Guo and Y.-F. Liu. License plate localization and character segmentation with feedback self-learning and hybrid binarization techniques. *Vehicular Technology, IEEE Transactions on*, 57(3):1417–1424, May 2008. 1, 2, 3

[5] X. Jia, X. Wang, W. Li, and H. Wang. A novel algorithm for character segmentation of degraded license plate based on prior knowledge. In *Automation and Logistics, 2007 IEEE International Conference on*, pages 249–253, Aug. 2007. 2, 3

[6] W. Kim and C. Kim. A new approach for overlay text detection and extraction from complex video scene. *IEEE Transactions on Image Processing*, 18(2):401–411, Feb. 2009. 6

[7] Y. Lu. Machine printed character segmentation – an overview. *Pattern Recognition*, 28(1):67–80, 1995. 2

[8] M. R. Lyu, J. Song, and M. Cai. A comprehensive method for multilingual video text detection, localization, and extraction. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(2):243–255, 2005. 6

[9] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–69, 1979. 3, 4

[10] C. Sun and D. Si. Skew and slant correction for document images using gradient direction. *ICDAR'97: Proceedings of the 4th International Conference on Document Analysis and Recognition*, pages 142–146, 1997. 3

[11] J. xiong Dong, P. Dominique, A. Krzyyzak, and C. Suen. Cursive word skew/slant corrections based on radon transform. *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, pages 478–483, Aug.-1 Sept. 2005. 3

[12] F. Yang, Z. Ma, and M. Xie. A novel approach for license plate character segmentation. *Industrial Electronics and Applications, 2006 1ST IEEE Conference on*, pages 1–6, 2006. 6

[13] H. Zhang, W. Jia, X. He, and Q. Wu. Learning-based license plate detection using global and local features. *Proceedings of the 18th International Conference on Pattern Recognition*, pages 1102–1105, 2006. 1

[14] J. Zhang, X. Fan, and C. Huang. Research on characters segmentation and characters recognition in intelligent license plate recognition system. In *Control Conference, 2006. CCC 2006. Chinese*, pages 1753–1755, Aug. 2006. 2

[15] Y. Zhang and C. Zhang. A new algorithm for character segmentation of license plate. In *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, pages 106–109, June 2003. 2, 3