A Novel Character Segmentation Method for Text Images Captured by Cameras

Hsin-Te Lue, Ming-Gang Wen, Hsu-Yung Cheng, Kuo-Chin Fan,
Chih-Wei Lin, and Chih-Chang Yu

Due to the rapid development of mobile devices equipped with cameras, instant translation of any text seen in any context is possible. Mobile devices can serve as a translation tool by recognizing the texts presented in the captured scenes. Images captured by cameras will embed more external or unwanted effects which need not to be considered in traditional optical character recognition (OCR). In this paper, we segment a text image captured by mobile devices into individual single characters to facilitate OCR kernel processing. Before proceeding with character segmentation, text detection and text line construction need to be performed in advance. A novel character segmentation method which integrates touched character filters is employed on text images captured by cameras. In addition, periphery features are extracted from the segmented images of touched characters and fed as inputs to support vector machines to calculate the confident values. In our experiment, the accuracy rate of the proposed character segmentation system is 94.90%, which demonstrates the effectiveness of the proposed method.

Keywords: Webcam-based OCR, character segmentation, typographical structure, periphery features, dynamic programming.

Manuscript received Mar. 13, 2010; revised July 21, 2010; accepted Aug. 2, 2010.

Hsin-Te Lue (phone: +886 3 422 7151 ext. 35328 email: lapolas@fox1.csie.ncu.edu.tw), Hsu-Yung Cheng (email: chengsy@csie.ncu.edu.tw), Kuo-Chin Fan (email: kcfan@csie.ncu.edu.tw), and Chih-Wei Lin (email: chihway@ms65.url.com.tw) are with the Institute of Computer Science and Information Engineering, National Central University, Taoyuan, Taiwan, R.O.C.

Ming-Gang Wen (email: mgwen@nuu.edu.tw) is with the Department of Information Management, National United University, Miaoli, Taiwan, R.O.C.

Chih-Chang Yu (corresponding author, tacoyu@mail.vnu.edu.tw) is with the Department of Computer Science and Information Engineering, Vanung University, Tao-Yuan, Taiwan, R.O.C.

doi:10.4218/etrij.10.1510.0086

I. Introduction

With advances of transportation, people often travel to other countries without familiarity of domestic languages. If translation tools could be provided for travelers to help recognize shop or road signs, it would be of great help. Translation tools require proper text input systems. The existing text input systems include voice input, handwriting input, and keyboard input. The voice input systems are hard to use for the users without familiarity of the language because they cannot pronounce the words correctly. Keyboard input would not be feasible for such users if the languages are not constructed based on alphabets. Users can try drawing the contour of the words so that handwriting input may get a response. However, users may get wrong responses due to using the wrong stroke order of the characters. It would be more convenient for users to use text images directly as the input of the translation tools. With the help of character recognition techniques, such an idea is feasible since personal mobile devices, such as mobile phones and PDAs, are often equipped with low resolution cameras.

Compared to traditional optical character recognition (OCR), the text images captured by users using mobile devices may contain objects of interest with complex backgrounds. It is a challenge for researchers to detect objects of interest from a complex and low contrast background. Instead of discussing the character recognition techniques, we focus on the new challenges imposed by the new application mentioned above in this work, that is, how to detect foreground texts and segment single character from an image correctly.

The current text detection research is roughly divided into rule-based and classifier-based methods. Rule-based methods

[1]-[5] formulate rules with prior-knowledge to distinguish text and non-text blocks. The rules formulated by experienced experts would perform classification efficiently, but may not be robust. Classifier-based methods [6]-[9] utilize the extracted features from images as input of classifiers, such as neural networks, to classify text and non-text blocks. The classifiers usually need enough samples for training to improve the accuracy. Moreover, the extracted features and parameters of a classifier are often tuned to improve the classification rate.

In traditional page segmentation, X-Y Cut [10], [11] is a top-down method which uses article layouts to find paragraphs, text lines, and characters, and then segments them by horizontal and vertical projections. Therefore, it is infeasible to segment the document when the image is skewed. The run-length smearing algorithm (RLSA) [12] is another simple but efficient page segmentation algorithm. RLSA can extract text blocks in the document images by manipulating a run smearing operation. However, RLSA also suffers from similar problems encountered in X-Y Cut.

After finding text blocks, these text blocks are linked to form meaningful text lines, that is, words and sentences, according to their reading order. Text lines are constructed according to the distance between two text blocks. Note that row spacings are often larger than character spacings in most documents.

Traditional character segmentation techniques are divided into projection methods, recognition-feedback-based methods, and classifier-based methods. Projection methods [13], [14] assume that the locations with no projection are the locations of spacing. However, it is risky to confirm segmentation locations using only a projection method in our application because touched characters and broken characters would often be generated when images are captured by cameras.

Recognition-feedback-based methods [15] can provide a recovery mechanism for error segmentation. The segmented text blocks are inputs of the recognition kernel. This method is more reliable than projection methods, but the computational cost is also larger.

Classifier-based methods [16] select segmentation points using classifiers trained by correct segmentation samples. The drawback of classifier-based methods is that classifiers require enough training samples to extend the classification scope.

The flowchart of the proposed system is illustrated in Fig. 1. Firstly, candidate text blocks are detected after inputting a text image. The text blocks will be used to construct text lines by properties of text clusters and a distance-based method. Next, we find typographical structures in each text line. Finally, a character segmentation mechanism combined with a touched character filter is proposed. In the experiments, we evaluate the performance of our proposed system by the recognition rates of the segmented text blocks.

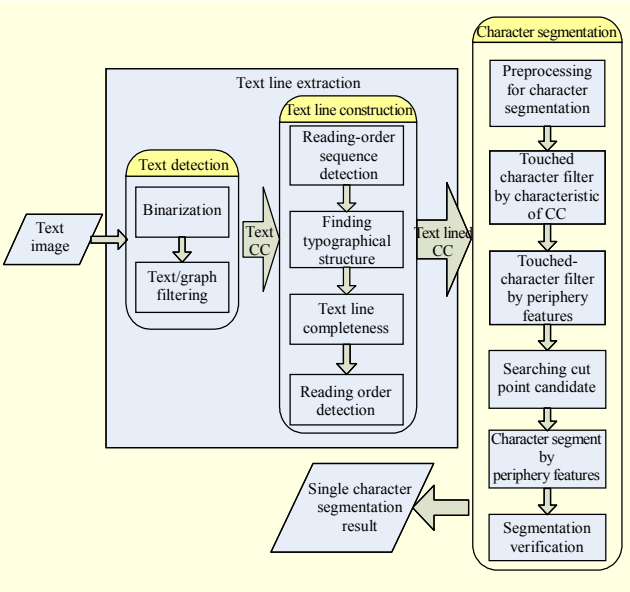

Fig. 1. Flowchart of proposed system.

II. Text Line Extraction

1. Text Detection

If the background includes some patterns or water-markings, it would also add difficulty to the application. The pictures and trademarks are often classified as the foreground. Therefore, the proposed text detection system needs to classify the foregrounds into texts and pictures.

A. Binarization

In general, text documents are often done with high contrast. Therefore, foregrounds and backgrounds can be separated using a binarization method. Considering the performance of the binarization and inherent nature of the documents, we propose a two-stage binarization mechanism. The well-known Otsu method [17] is adopted in the proposed two-stage binarization mechanism.

In the first stage, the proposed mechanism will look for foreground candidates using a global threshold found by the Otsu algorithm. The second stage binarization will be operated after text line construction. The region of a text line will be divided into several regions according to the average character width. Each of these small regions will be considered as a processing unit in the second stage binarization which is binarized by the Otsu method again.

B. Graphic/Text Segmentation

It will raise the performance substantially if the pictures and noise can be filtered from the foreground candidates as much as possible in this module. After the first stage binarization, the

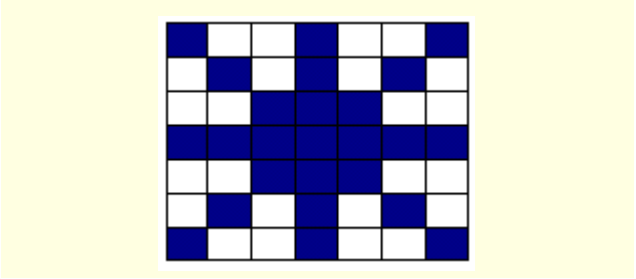

Fig. 2. 7×7 mask of auto-regressive feature.

foreground objects are labeled utilizing the connected component (CC) method. These CCs are also the text candidates. The CCs will be normalized to a fixed size. Then, auto-regressive features [18], [19] are extracted from the CCs as the inputs of neural network for text noise classification as illustrated in Fig. 2.

2. Text Line Construction

Users often try to cover as much of an image's document area as possible when capturing documents using cameras. Moreover, users may use the zoom-in function to capture texts clearly. As a result, the image may contain the document contents without the margin of the document. This will add challenges when rotating the document images without the margin information.

Compared to traditional OCR, the images captured by cameras are often askew and the text lines may not be straight. We propose a bottom-up CC-based method to construct text lines.

A. Reading-Order Sequence Detection

In this paper, a distance-based method is designed to construct text lines according to the observation that the row spacing is often greater than the word spacing in most document layouts. Instead of calculating the distance between central points of two text candidates, the distance between two text candidates is estimated by the out-length proposed in our previous work [20] as illustrated in Fig. 3. Here, out-length is defined as the length of the segment between the boundaries of two text candidates.

The advantage of out-length is that its use avoids additional distance estimations when the widths of some text candidates are too large. In Fig. 4, we can find the neighboring CCs for each CC by using an out-length. For example, we can determine that CC 3 is closer to CC 2 than CC 1 using distances measured by out-length. If we consider the distances between central points of CCs, we will connect CC 1 and CC 2, and the text line will thereby be constructed in the wrong direction.

A two-stage statistical method is proposed to find the text

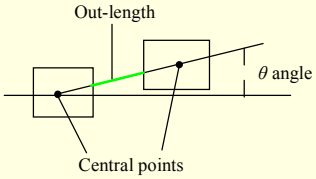

Fig. 3. Illustration of out-length and slant angle between two CCs.

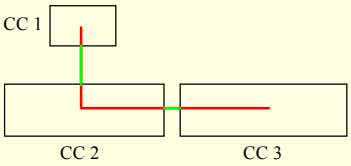

Fig. 4. Illustration showing that CC 3 is closer to CC 2 than CC 1 by using out-length, but CC 1 is closer than CC 3 estimated by using the distance between the central points of the CCs.

line reading order. In the first stage, for each text candidate, a neighboring candidate which has the smallest out-length is chosen. Then, the angle θ between the horizontal line and the line linking the central points of these two neighboring CCs is computed (see Fig. 3). For the time being, a histogram is constructed, and the angle θ_m with the majority votes is utilized to determine the coarse reading order orientation of the document. The reading order orientation will be refined by the typographical structures of the text lines in the second stage described in II.2.C.

The coarse reading order orientation estimated in the first stage is temporally assumed as the correct reading order orientation to construct the text lines. Suppose there are n text CCs denoted as $CC_1,\ldots, CC_n$. The text line construction algorithm is stated as follows:

Step 1. For an unvisited text candidate area CC_i and its neighboring CC_j, angle θ_{ij} which is the angle between CC_i and CC_j can be evaluated by the following equation:

$$\theta_m - \varepsilon < \theta_{ij} < \theta_m + \varepsilon, \quad (1)$$

where θ_m is the document reading order orientation. The purpose of (1) is to link several CCs into a text line along a straight direction. Here, ε in (1) is a tolerant threshold to link CCs into a straight text line.

If the inequality in (1) is satisfied for θ_{ij}, go to step 2. Otherwise, select another neighboring CC_k with the second smallest out-length, and check the inequality again using angle θ_{ik}. If the inequality in (1) is satisfied for θ_{ik}, go to step 3. If (1) is not satisfied for both θ_{ij} and θ_{ik}, go to step 4.

Step 2. Link CC_j to CC_i. Go to step 1, and check the next text candidate CC_j.

Step 3. Link CC_i to CC_k. Go to step 1, and check the next

text candidate CC_k.

Step 4. CC_i cannot be connected with any other CC at this stage. Find another unvisited CC_p and go to step 1. If all CCs have been visited, terminate the algorithm.

After performing the algorithm, most CCs will be linked to some text lines. If the documents are captured with a slanted angle, the images can be deskewed by utilizing the slope of typo-lines.

B. Typographical Structure

Typographical structures [21] have been designed since the era of typewriter and are still preserved in the printed fonts today. Figure 5 illustrates the four lines (called typo-lines) which bound all printed English characters in three areas. The four lines are named the top line, upper line, baseline, and bottom line. The three areas are called the upper, central, and lower zones.

According to the location of typographical structure, the printed alphanumeric characters and punctuation marks can be classified into seven categories, called typo-classes.

i. Full: the character occupies three zones, such as 'j,' left parenthesis, and right parenthesis.

ii. High: the character is located in both upper and central zones, such as capital letters, numerals, 'b,' 'd,' and so on.

iii. Short: the character is only located in the central zone, such as 'a,' 'c,' 'e,' and so on.

iv. Deep: the character appears in central and lower zones. Only the four lowercase letters 'g,' 'p,' 'q,' and 'y' belong to this typo-class.

v. Subscript: the punctuation mark is closer to the baseline, such as comma, period, and so on.

vi. Superscript: the punctuation is closer to the upper line, such as quotation marks, double quotation marks, and so on.

vii. Unknown: the class is given when the typo-class cannot be confirmed due to lack of certain typo-lines.

To determine the typographical structure, a typo-line extraction algorithm is proposed which integrates *k*-means and least mean square error (LMSE). In our proposed algorithm, a rough baseline is first extracted. Then, all CCs along a text line are classified by the distance from their top edges to the baseline using the *k*-means algorithm. Two clusters are generated by the *k*-means algorithm. Next, we apply LMSE algorithm on the clusters to determine the corresponding typo-lines (top line and upper line). If the distance between top line and upper line is smaller than certain ratio of the average character height, two typo-lines will be merged into one. As shown in Fig. 6(a), four of five text lines lack one typo-line because there is no character occupying all of the three areas. Similarly, baseline and bottom lines can also be extracted by the same procedure.

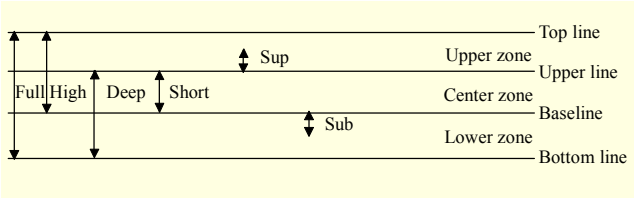

Fig. 5. Diagram of typographical structure and typo-class.

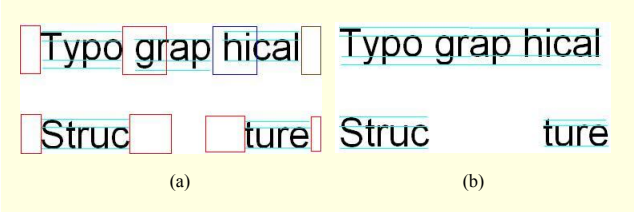

Fig. 6. Illustration showing several partial text lines merged into a complete text line: (a) five partial text lines in two rows and (b) three partial text lines merged in first row.

In the previous text-line extraction process, some extracted text lines may be the temporary text lines as depicted in Fig. 6(a). To further refine the text lines, we also iteratively combine neighboring text lines which have similar typo-lines to make the text lines more complete. The rationale of similarity lies mainly on the idea that a text line can be extended in a certain region along the reading order direction and checked as to whether or not the neighboring text line overlaps and exceeds over certain ratio of areas in the extended region. In (2) and (3), the typo-line similarities are defined as

$$\begin{aligned} &\left|TL_1ofTailCCx - TL_2ofHeadCCx\right| > \text{Threshold}_1 \quad \text{when} \quad ReadingOrder = H, \\ &\left|TL_1ofTailCCy - TL_2ofHeadCCy\right| > \text{Threshold}_1 \quad \text{when} \quad ReadingOrder = V, \end{aligned} \tag{2}$$

$$\begin{aligned} &\frac{Y\_Area\_of\_overlap(TL_1, TL_2)}{Y\_Projection(TL_1)} > \text{Threshold}_2 \\ &\frac{Y\_Area\_of\_overlap(TL_1, TL_2)}{Y\_Projection(TL_2)} > \text{Threshold}_2 \end{aligned} \quad \text{when} \quad ReadingOrder = H, \\$$

$$\begin{aligned} &\frac{X\_Area\_of\_overlap(TL_1, TL_2)}{X\_Projection(TL_1)} > \text{Threshold}_2 \\ &\frac{X\_Area\_of\_overlap(TL_1, TL_2)}{X\_Projection(TL_2)} > \text{Threshold}_2 \end{aligned} \quad \text{when} \quad ReadingOrder = V, \tag{3}$$

where the reading order will be either horizontal or vertical, and the operator will change from x to y or from y to x. $TL_1ofTailCCx$ is the x value of the right boundary of the tail CC in text line 1, and $TL_2ofHeadCCx$ is the x value of the left boundary of the head CC in text line 2. $Y\_Projection(TL_1)$ is the distance between the cross points of the top and bottom lines of TL_1 on x set as $(TL_1ofTailCCx+TL_2ofHeadCCx)/2$. $Y\_Area\_of\_overlap(TL_1, TL_2)$ is the overlapping area between the cross points of the top and bottom lines of TL_1 and TL_2 on x set as $(TL_1ofTailCCx + TL_2ofHeadCCx)/2$. Threshold_1 in (2) is

Table 1. Appearance rate of lower letters.

Letter	App. rate	Letter	App. rate	Letter	App. rate	Letter	App. rate
a	8.2	h	6.1	o	7.5	v	1.0
b	1.5	i	7.0	p	1.9	w	2.4
c	2.8	j	0.2	q	0.1	x	0.2
d	4.3	k	0.8	r	6.0	y	2.0
e	12.7	l	4.0	s	6.3	z	0.1
f	2.2	m	2.4	t	9.1		
g	2.0	n	6.7	u	2.8		

Table 2. Appearance rate of typo-class.

Typo-class	Full	High	Short	Deep
Appearance rate	0.2	35	59.1	6

to determine the degree of closeness of two text lines. In our work, we consider that two text lines can be merged into one text line if the distance between two text lines is less than 1.5 to 3 times of the average character width. Threshold_2 in (3) checks the degree of similarity in the typographical structure of two text lines, which is important in determining the merging of two text lines. It is set as 0.5 in our work.

The pictorial illustration of the rationale is shown in Fig. 6. If (2) and (3) are satisfied, the two temporary text lines can be linked. For the example in Fig. 6, the overlapping area in the right extended region (red rectangle in Fig. 6(a)) of "typo" and the partial region of "grap" is large enough. Moreover, the difference between the distances between the heights of the two typo-lines is also small enough. Therefore, "grap" will be linked to "typo," and "hical" will also be linked by the same mechanism (blue rectangle in Fig. 6(a)).

C. Reading Order Confirmation

The proposed reading-order confirmation process is designed by analyzing the typo-classes of the characters. By observing typo-classes of alphanumerical characters, we can find that the appearance rates of high type and deep type characters are significantly different. The appearance rate of each letter varies in articles of different domains. Baker and others [22] calculated the appearance rates of 100,362 letters in newspapers and novels (Table 1). The appearance rates of the typo-classes are listed in Table 2. The coarse reading order orientation can be confirmed according to the observation that the appearance rates of the high type are significantly larger than the appearance rates of the deep type.

III. Character Segmentation

Comparing the images captured by the camera, it is more difficult to select a proper segmentation point because the characters are touched severely due to the blurred character boundaries. Hence, the images are usually enhanced to reduce the external effects before character segmentation. In this section, a character segmentation mechanism with the touched character filter is proposed.

1. Mis-filtered Text Recovery

In the previously mentioned text/noise filter (see II.1.B), the text CCs may be classified as the noise due to low-quality image. The meanings of the text lines or the words will be wrong due to the mis-filtered texts missing in the text line. The mis-filtered text CCs often are located near or in the text lines. Hence, these CCs would be recovered by comparing the scope of the text lines which are surrounded by the typo-lines. If the mis-filtered text CCs fall in the boundary, they are merged with the overlapping characters or inserted into the text lines. The boundaries of the text lines are extended the width of two characters in order to recover the CCs which are near the text lines.

2. Touched Character Filtering Utilizing a Characteristic of CC

In this subsection, a character segmentation mechanism with a touched character filtering is proposed. The text CCs will be classified as a single character or touched characters by the devised filter. The proposed mechanism consists of two stages including the touched character filtering utilizing a characteristic of CC and touched character filtering using peripheral features. In the first stage, seven intrinsic features of CCs are grouped to form a vector $\boldsymbol{C}=\{\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3, \boldsymbol{c}_4, \boldsymbol{c}_5, \boldsymbol{c}_6, \boldsymbol{c}_7\}$ for the touched character filter. The intrinsic features are described as follows:

$\boldsymbol{c}_1$: the height-width ratio of CC,

$\boldsymbol{c}_2$: the X-axis position of the maximum vertical projection relative to the left boundary in the CC (see Fig. 7),

$\boldsymbol{c}_3$: the Y-axis position of the maximum horizontal projection

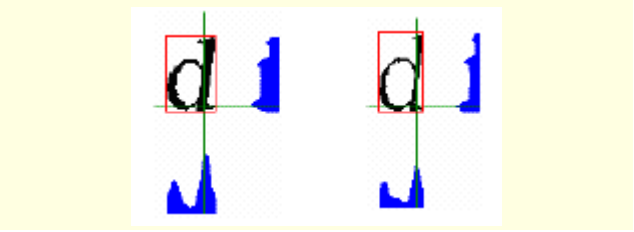

Fig. 7. Relative position diagram illustrating $\boldsymbol{c}_2$ and $\boldsymbol{c}_3$.

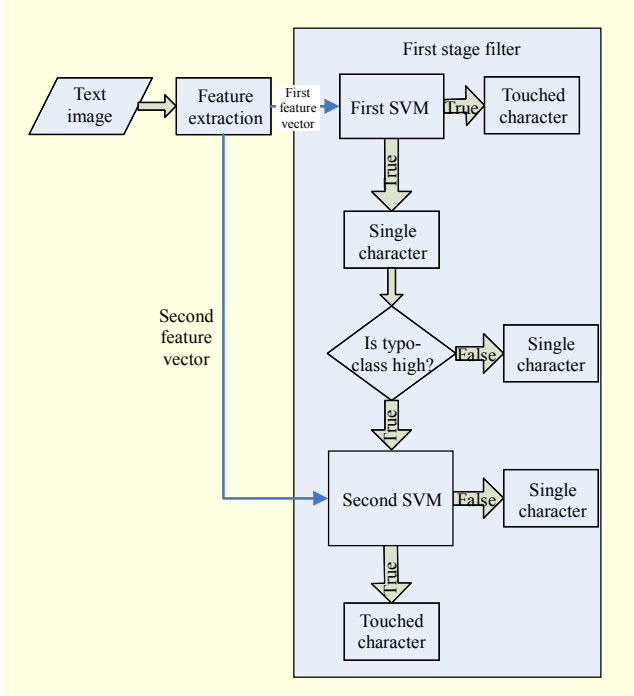

Fig. 8. Flowchart of the first stage touched character filtering.

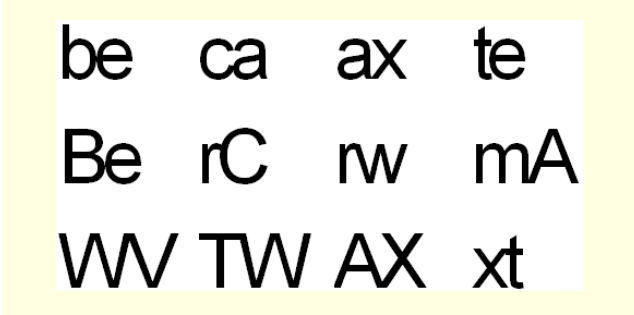

Fig. 9. Illustration of negative samples of SVM database.

relative to the upper boundary of the CC (see Fig. 7),

$\boldsymbol{c}_4$: the maximum value of vertical projection/the height of CC,

$\boldsymbol{c}_5$: the maximum value of horizontal projection/the width of CC,

$\boldsymbol{c}_6$: the maximum value of the 5 bins average vertical projection/the height of CC,

$\boldsymbol{c}_7$: the maximum value of the 5 bins average horizontal projection/the width of CC,

where the 5 bins average projection in $\boldsymbol{c}_6$ and $\boldsymbol{c}_7$ is defined in (4), and x is the referred bin:

$$Project = \frac{\sum_{i=x-2}^{x+2} Project[i]}{5}. \quad (4)$$

Figure 8 shows the flowchart of the first stage touched character filtering. The extracted feature vector $\boldsymbol{C}=\{\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3, \boldsymbol{c}_4, \boldsymbol{c}_5, \boldsymbol{c}_6, \boldsymbol{c}_7\}$ is trained by two SVMs to classify CCs into a single character or touched characters. The feature set $\{\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3, \boldsymbol{c}_4, \boldsymbol{c}_5\}$ is treated as the input for the first SVM, and $\{\boldsymbol{c}_1, \boldsymbol{c}_6, \boldsymbol{c}_7\}$ is fed to the second SVM.

The SVM database includes positive samples and negative samples with 7 font types (Arial, Arial Narrow, Courier New, Times New Roman, Mingliu, PMingliu, and KaiU) and 4 font sizes (32, 48, 52, and 72). The positive samples consist of alphanumerical characters and punctuations. The negative samples are composed of two connected alphanumerical characters. The illustration of negative samples in SVM database is shown in Fig. 9.

3. Touched Character Filtering by Periphery Features

After the first stage filtering, the periphery features will be extracted from the touched character CCs for use in the second stage filtering. The periphery features are composed of 32 character contour values f_i, where $i = 1, 2, \ldots, 32$ as shown in Fig. 10. In Fig. 11, the closer the peripheral feature to the central position, the larger the weight is assigned. Comparing to the contour in the central zone of CC, the contour of CC in those non-central zones will be influenced easily by image distortion. Hence, the periphery features in the boundary of central zone will be more reliable than the periphery features in the boundaries of non-central zones.

$$f_i = w_{i \bmod 8} \frac{p_i}{l_i}, \quad (5)$$

where the weight $w_{i \bmod 8}$ can be obtained by referring to Fig. 11. If $0<i<9$ or $16<i<25$, l_i is the character width. Otherwise, l_i is the character height. P_i is the distance between the boundary and the contour, that is, the length of the blue band in Fig. 10. The 32 periphery features and the feature of height-width ratio of CC are grouped to form a feature vector $F=\{f_1, f_2, \ldots, f_{33}\}$.

Before the second stage touched character filter algorithm beginning, the definitions of these terms are given firstly:

S_i: periphery feature vector of the character template, $S_i=\{S_{if1}, S_{if2}, \ldots, S_{if33}\}$, $i=1, \ldots, n$, where n is the number of character templates.

t: periphery feature vector of the character template to be analyzed.

thA: thresholds in determining positive weight value.

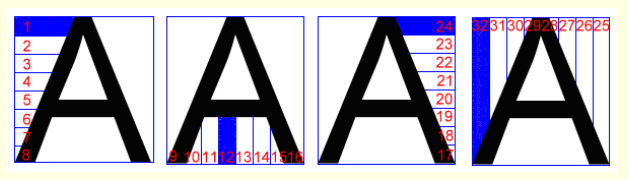

Fig. 10. Illustration of 32 periphery features.

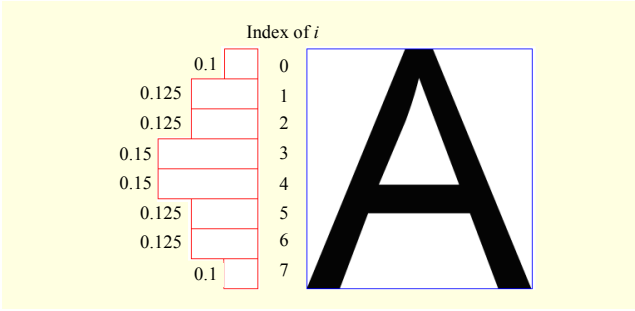

Fig. 11. Illustration of the weight of periphery feature.

thB: thresholds in determining negative weight value.

thC: threshold in determining 33rd feature value.

The second stage touched character filtering process using periphery features can then be described as follows:

Step 1. If $\left|t_{f_j} - s_{if_j}\right| < thA$ and $\left|t_{f_{33}} - s_{if_{33}}\right| < thC$, $d_j = 1,\ j = 1,...,32,\ i = 1,...,n.$

Step 2. If $\left|t_{f_j} - s_{if_j}\right| > thB$ and $\left|t_{f_{33}} - s_{if_{33}}\right| < thC$, $d_j = -1,\ j = 1,...,32,\ i = 1,...,n.$

Step 3. If the condition in step 1 and step 2 is not satisfied, then $d_j = 0$.

Step 4. $PV = \sum_{j=1}^{32} d_j W_j, \forall d_j > 0,\ NV = \sum_{j=1}^{32} d_j W_j, \forall d_j < 0$.

Step 5. Repeat step 1 to step 4 until the whole character templates have been compared, and record the template with the largest *PV*. Here, *PV* indicates a positive weight value and *NV* is a negative weight value.

As to the second touched character filter, we hope that the filter can tolerate slight skew and significant feature displacement. The threshold values of *thA*, *thB*, and *thC* are set as 0.05 to 0.1, 0.05 to 0.1, and 0.5.

If the *PV* is larger than a threshold and *NV* is smaller than another threshold, the CC is considered as a single character. Otherwise, the CC is considered as a touched character.

4. Cut Point Candidate Searching

If the CCs are still considered as touched characters after the two-stage filtering, then the CCs will be inputted to the character segmentation mechanism. The character segmentation mechanism consists of three steps:

i. Cut point candidate searching.

ii. Character segmentation using periphery features.

iii. Segmentation result verification.

There are three features to be utilized in searching cut point candidates: (1) the vertical projection, (2) the vertical profile, and (3) the gray level vertical projection, as shown in Figs. 12 and 13.

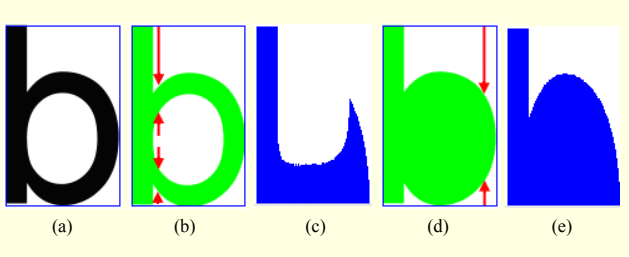

Fig. 12. Illustration of vertical projection and vertical profile: (a) original image, (b) accumulation of pixels in obtaining vertical projection, (c) vertical projection of (b), (d) accumulation of pixels in vertical profile, and (e) vertical profile of (d).

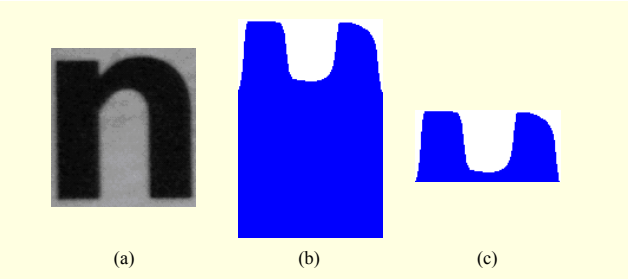

Fig. 13. Illustration of gray level projection: (a) original image, (b) gray level projection, and (c) normalized gray level projection.

The vertical profile, also called Caliper distance [23], is the distance between the top contour pixel and the bottom contour pixel in each bin. For example, shown in Fig. 12(e) is the vertical profile obtained from the character "b" in Fig. 12(d).

We define *G* as the set of the gray level projection of CC, that is, $G = \{g(0), g(1),\ldots, g(w-1)\}$. The gray level projection can be formulated as follows:

$$g(x) = \sum_{y=0}^{h} \left| I(x,y) - 255 \right|, \tag{6}$$

where $I(x, y)$ is the gray level value in pixel (x, y), and h is the height of the image. Figure 13 illustrates the process in obtaining the gray level projection in a gray level image. Shown in Fig. 13(b) is the result obtained by performing (6) on the image in Fig. 13(a). Figure 13(c) is the final result after normalizing the gray level projection in Fig. 13(b) by performing (7a) and (7b).

$$g_n(x) = \frac{g(x)}{\mathrm{Max}(G)} \times 255, \tag{7a}$$

$$g_N(x) = g_n(x) - \mathrm{Min}(G_n), \tag{7b}$$

where $G_n = \{g_n(0), g_n(1), \ldots, g_n(x-1)\}$.

The following equation is used to evaluate the goodness of the cut points obtained from the three features:

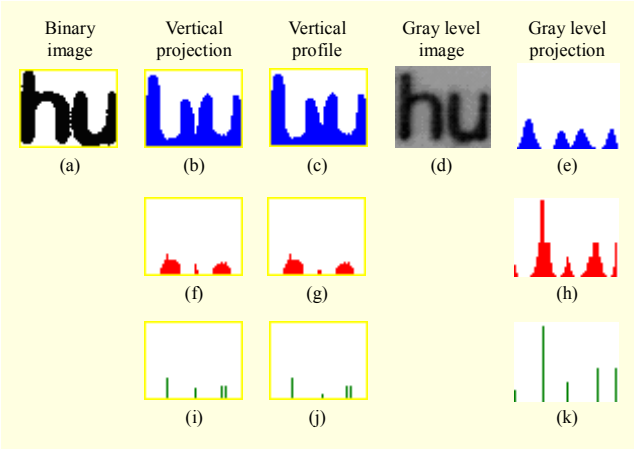

Fig. 14. Example illustrating cut point candidate searching: (a) binary image, (b) vertical projection, (c) vertical profile, (d) gray level image, (e) gray level projection, (f)-(h) results are obtained by performing (8) on (b) and (c), and (i)-(k) results after cut points selection.

$$p(x)=\frac{V(lp)-2V(x)+V(rp)}{V(x)+1}, \tag{8}$$

where $V(lp)$ is the first peak to the left of x, $V(rp)$ is the first peak to the right of x, and $V(x)$ denotes the value of x. The larger the value of $p(x)$ in the histograms of the three features, the better the point is for x to be a cut point. A cut point selection rule can be designed according to the following two criteria. The first criterion states that the number of cut points can be chosen according to the width-height ratio of a CC for a complete segmentation. The larger the width-height ratio is, the more characters the CC will contain. Hence, those points with larger value of $p(x)$ will have a higher tendency to be chosen as cut points. The second criterion is that cut points near a selected cut point within a certain distance can be ignored to reduce computation cost because the distance between two cut points has minimum bound due to the minimum width restriction of a character. Shown in Fig. 14 is an example illustrating cut point searching.

5. Character Segmentation by Periphery Features

If the found cut-point candidates can include all of the real cut points in the touched character image, there exists only one combination of the cut points which will segment the touched character image correctly. If there are n cut point candidates, there will be 2^n combinations of cut points. It is too difficult to find the correct combination without a certain efficient pruning mechanism. Here, the periphery features of a character image are utilized as the inputs of SVM to output a confidence value for evaluating the goodness of segmentation. A combination of the cut points with the maximum confident value which is estimated by dynamic programming (DP) is considered the final segmentation result. Suppose the number of cut point candidates together with the left boundary and right boundary of the touched character image is n, $0 \leqq i \leqq i+k \leqq j < n$, where i, j, k, n, a, and b are integers, and i, $i+k$, and j are the indexes of cut points. The boundary conditions of DP are described as

$$m(i,j)=\begin{cases}0, & \text{if} \quad i=j,\\ \text{Max}\{(m(i,i+k)\times a+m(i+k,j)\times b)/(a+b), m(i,j)\}, & \text{if} \quad i<j,\end{cases} \tag{9}$$

where $m(i, j)$ is the confident value of the image between cut points i and j. a and b are the counters accumulating the number of segmented characters in the image between i, $i+k$ and $i+k$, j, respectively. Here, an average strategy is adopted to calculate the confident values formed by the combinations of cut points when a character image is divided into several segmented images. For each $m(i, j)$, its value can be obtained

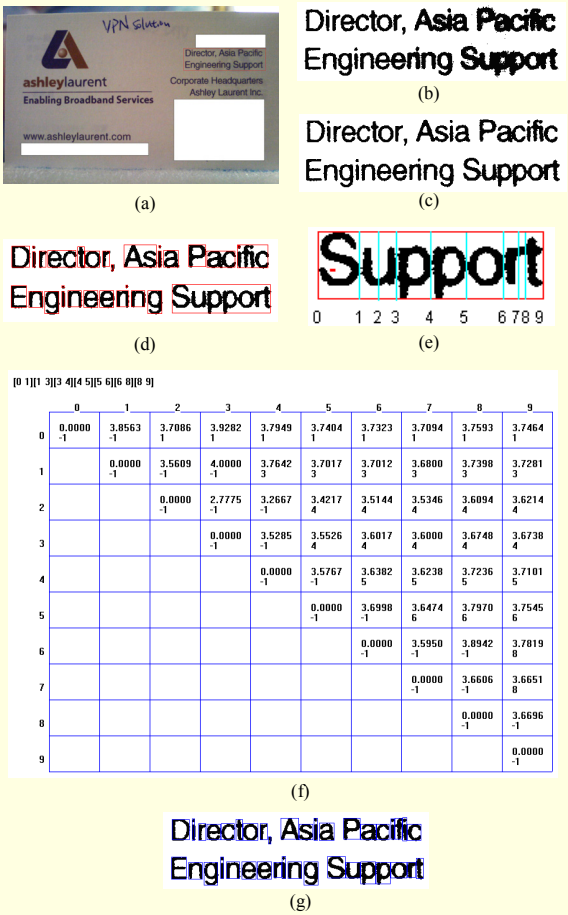

Fig. 15. Example illustrating cut point selection using DP: (a) original image, (b) partial binarized image obtained by Otsu binarization, (c) result of text-line-based binarization, (d) result of CC labeling, (e) cut points in a CC "Support," (f) DP table of (e), and (g) final character segmentation result.

by comparing the average confident value of the segmented image parts within cut points i, j and the confident value of the whole image without segmentation. The larger of these two values is chosen as the final value $m(i, j)$.

The diagram of the character segmentation is shown in Fig. 15. Take one touched character, "Support," in Fig. 15(d) as an example. The number of cut points n, which are selected by the proposed method, is 10 in the CC where cut point 0 is the left boundary of the CC, and cut point 9 is the right boundary of the CC. In this example, $m(i, j)$ is given a confident value from 0 to 4 obtained by SVM. There are two values in each block of the DP table as shown in Fig. 15(f). The upper value is the confident value of the character image between row i and column j, whereas the lower value indicates the selected cut point index in the character image between row i and column j. If the confident value of the whole image between row i and column j is larger than the average confident value of the image divided into 2 parts between $(i, i+k)$ and $(i+k, j)$, then the cut point index will be set to –1. For instance, the numerical values in block (0, 3) which is (3.9282, 1) indicates that a larger average confident value will be obtained in (0, 3) when cut point 1 is selected to form segmented images (0, 1) and (1, 3). The numerical value in block (1, 3) conveys that the confident value of the whole image without segmentation is larger than the average confident value of all segmentation combinations in (1, 3). Following the same procedure, the final segmentation combination of the CC (0, 1) (1, 3) (3, 4) (4, 5) (5, 6) (6, 8) (8, 9) derived by DP can be obtained as shown in the upper left corner of Fig. 15(f).

6. Segmentation Verification

To reduce the broken character and over-segmentation effects, the segmentation verification module will be performed after character segmentation for correcting the segmentation errors caused by merely using the typo-class information. For example, the character 'm' is usually segmented into two parts, recognized as 'r n' or 'n 7', which is an unreasonable situation because the typo-class of 'm' is short and the typo-classes of 'n 7' are short and high. Table 3 tabulates our designed character segmentation check table with each element representing one unreasonable situation. If a character is segmented into the specific combination as listed in Table 3, the segmentation of the character will be ignored to preserve the original character by not performing the segmentation task.

Table 3. Segmentation combination verification

Text	Segmentation combination	Type	Text	Segmentation combination	Type
M	r,n、n,7、n,1	3	C	C,:	1
N	r,1、r,7	3	c	c,:	3
W	v,v	3	B	I,3	1
W	V,V	1	D	I,3	1
H	I,7、t,1、t,7	1			

IV. Experimental Results

Experiments were conducted by executing a character recognition process on testing images to verify the performance of our proposed segmentation method. The character recognition method proposed in [24] was implemented to evaluate the performance of our method.

In the experiments, text images captured from fifty business cards by a two-million-pixel webcam with resolution 1600 × 200 are inputted as testing images. Testing images include the business cards with simple binary backgrounds and complex color backgrounds. There are 9,550 characters and 419 touched characters (1,050 single characters in the touched characters) for a total of 10,600 characters in the testing images. Here, the accuracy rate is adopted to evaluate the performance of segmentation which is defined by

$$\text{Recall} = \text{Num}_{\text{cor}} / \text{Num}_{\text{total}}. \quad (10)$$

The average accuracy rate of our proposed touched character filtering in detecting touched characters is 92.14%. The worst case of touched character filtering for a single text image is 81.20% because the illumination varies drastically on the image as illustrated in Fig. 16. The overall accuracy rate of touched character segmentation is 98.57%. The worst case of touched character segmentation for a single text image is 90.00%. One example illustrating the phenomenon of severe touched characters is shown in Fig. 17. It is difficult to find

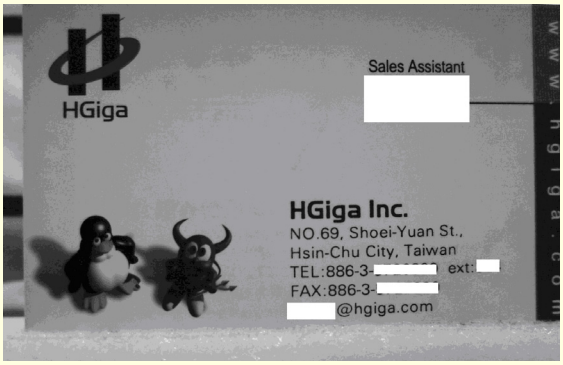

Fig. 16. Sample image of worst case in touched character filtering. Lightness varies severely in image.

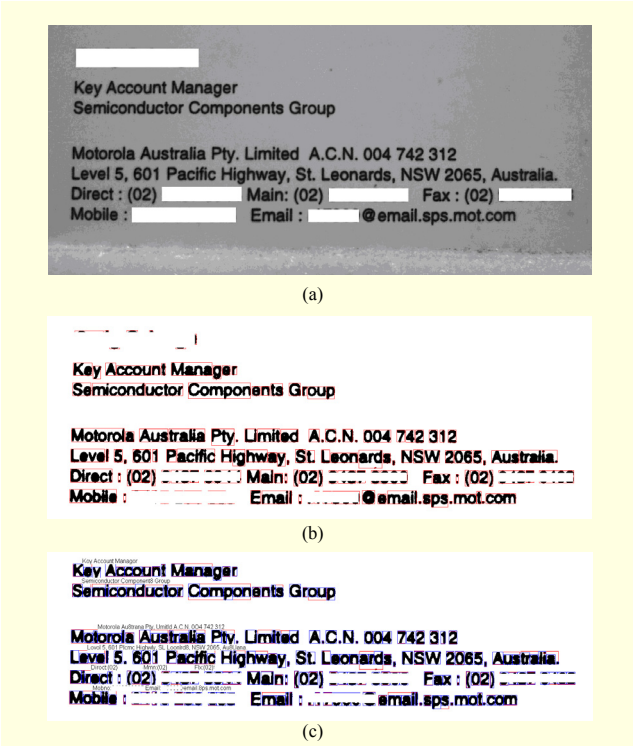

Fig. 17. Sample image of worst case in touched character segmentation. Both lighting variation and character appear severely blurred in image: (a) original image, (b) binarization image, and (c) character segmentation and recognition result.

good cut points to segment touched characters precisely, especially when the text image is blurred. After executing the character recognition process, the accuracy rate of our segmentation result is 94.90%. It is difficult to segment and recognize blurred touched characters due to the influences of illumination variation, and out of focus due to depth.

V. Conclusion

In this paper, a text image preprocessing and character segmentation system dedicated to text images captured by cameras was proposed. Since captured text images are usually accompanied with complex backgrounds and severe environments, the traditional top-down method would be inappropriate to accomplish the task of character segmentation. To remedy these problems, a novel character segmentation method was presented. Experimental results demonstrated the feasibility and validity of our proposed method in the segmentation of text images captured by cameras.

References

[1] X. Chen et al., "Automatic Detection and Recognition of Signs from Natural Scenes," *IEEE Trans. Image Process.*, vol. 13, no. 1, 2004, pp. 87-99.

[2] R. Lienhart and A. Wernicke, "Localizing and Segmenting Text in Images and Videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 4, 2002, pp. 256-268.

[3] M.R. Lyu, J. Song, and M. Cai, "A Comprehensive Method for Multilingual Video Text Detection, Localization, and Extraction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 2, 2005, pp. 243-255.

[4] V. Wu, R. Manmatha, and E.M. Riseman, "Textfinder: An Automatic System to Detect and Recognize Text in Images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 11, 1999, pp. 1224-1229.

[5] W. Wu, X. Chen, and J. Yang, "Detection of Text on Road Signs from Video," *IEEE Trans. Intell. Transport. Syst.*, vol. 6, no. 4, 2005, pp. 378-390.

[6] S. Hu and M. Chen, "Adaptive Fréchet Kernel Based Support Vector Machine for Text Detection," *IEEE Int. Conf. Acoustics, Speech, Signal Process.*, vol. 5, 2005, pp. 365-368.

[7] K.C. Kim et al., "Scene Text Extraction in Natural Scene Images Using Hierarchical Feature Combining and Verification," *Proc. 17th Int. Conf. Pattern Recognition*, vol. 2, 2004, pp. 679-682.

[8] K.L. Kim, K. Jung, and J.H. Kim, "Texture-Based Approach for Text Detection in Images Using Support Vector Machines and Continuously Adaptive Mean Shift Algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, 2003, pp. 1631-1639.

[9] T. Yamguchi and M. Maruyama, "Character Extraction from Natural Scene Images by Hierarchical Classifiers," *Proc. 17th Int. Conf. Pattern Recognition*, vol. 2, 2004, pp. 687-690.

[10] K.C. Fan and L.S. Wang, "Classification of Machine-Printed and Handwritten Texts Using Character Block Layout Variance," *Pattern Recognition*, vol. 31, no. 9, 1998, pp.1275-1284.

[11] J.L. Meunier, "Optimized XY-Cut for Determining a Page Reading Order," *Proc. 8th Int. Conf. Document Anal. Recog.*, vol. 1, 2005, pp. 347-351.

[12] K.C. Fan, L.S. Wang, and Y.K. Wang, "Page Segmentation and Identification for Intelligent Signal Processing," *Signal Process.*, vol. 45, no. 3, 1995, pp. 329-346.

[13] M.C. Jung, Y.C. Shin, and S.N. Srihari, "Machine Printed Character Segmentation Method Using Side Profiles," *IEEE Int. Conf. Syst., Man, Cybernetics*, vol. 6, 1999, pp. 863-867.

[14] S. Liang, M. Shridhar, and M. Ahmadi, "Efficient Algorithms for Segmentation and Recognition of Printed Characters in Document Processing," *IEEE Pacific Rim Conf. Commun., Comput. Signal Process.*, vol. 1, 1993, pp. 240-243.

[15] C.L. Liu, H. Sako, and H. Fujisawa, "Effects of Classifier Structures and Training Regimes on Integrated Segmentation and Recognition of Handwritten Numeral Strings," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 11, 2004, pp.1395-1407.

[16] S. Marinai, M. Gori, and G. Soda, "Artificial Neural Networks for

Document Analysis and Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 1, 2005, pp. 23-35.
[17] N. Otsu, "A Threshold Selection Method from Gray Level Histograms," *IEEE Trans. Syst., Man Cybern.*, vol. 9, no. 1, 1979, pp. 62-66.
[18] K.I. Kim et al., "Support Vector Machines for Texture Classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 11, 2002, pp. 1542-1550.
[19] V. Vapnik, *The Nature of Statistical Learning Theory*, New York: Springer-Verlag, 1995.
[20] L.S. Wang, *Document Analysis for Lossless Reproduction*, Dissertation, Institute of Computer Science Information Engineering, National Central University, 1997.
[21] A. Zramdini and R. Ingold, "Optical Font Recognition Using Typographical Features," *IEEE Trans. Pattern Anal. Mach. Intelligence*, vol. 20, no. 8, 1998, pp. 877-882.
[22] H. Beker and f. Piper, *Cipher System: The Protection of Communication*, John Wiley & Sons, 1983.
[23] C. M. Thillou et al., "A Multifunctional Reading Assistant for the Visually Impaired," *EURASIP J. Image Video Process.*, vol. 2007, no. 4, 2007. Available athttp://downloads.hindawi.com/journals/ivp/2007/064295.pdf.
[24] F. Chang, et al., "A Prototype Classification Method and its Application to Handwritten Character Recognition," *IEEE Int. Conf. Syst., Man Cybern.*, vol. 5, 2004, pp. 4738-4743.

Hsin-Te Lue received the BS in computer science and information engineering from Tamkang University, Taiwan, in 2000, and the MS from the National Central University, Taiwan, in 2002. Currently, he is a PhD candidate in the National Central University, Taiwan. His research interests include pattern recognition, document analysis, and optical character recognition.

Ming-Gang Wen received the BS in computer science and information engineering from National Chiao-Tung University, Taiwan, in 1989, and his MS in computer science and electronic engineering from National Chiao-Tung University, Taiwan, in 1993. He received his PhD in computer science and information engineering from the National Central University, Taiwan, in 2003. In 1993, he joined the Department of Information Management at National Lien-Ho Institute of Technology. He is currently an associate professor in the Department of Information Management, National United University, Taiwan. His research interests are in the areas of 3D optical character recognition, data hiding, and digital watermarking.

Hsu-Yung Cheng received the BS and MS in computer science and information engineering from National Chiao-Tung University, Taiwan, in 2000 and 2002, respectively. She earned the PhD from the University of Washington in electrical engineering in 2008. She became an assistant professor at the Department of Computer Science and Information Engineering, National Central University, Taiwan, in 2008. Her research interests include image and video analysis and intelligent systems.

Kuo-Chin Fan received the BS in electrical engineering from the National Tsing-Hua University, Hsinchu, in 1981, and the MS and PhD from the University of Florida, Gainesville, in 1985 and 1989, respectively. In 1983, he joined the Electronic Research and Service Organization (ERSO), Taiwan, as a computer engineer. From 1984 to 1989, he was a research assistant with the Center for Information Research, University of Florida. In 1989, he joined the Institute of Computer Science and Information Engineering, National Central University, Jhongli, Taiwan, where he became a professor in 1994. From 1994 to 1997, he was the chairman of the Department. Currently, he is the vice-president of Fo Guang University, I-Lan, Taiwan. His current research interests include image analysis, optical character recognition, and document analysis.

Chih-Wei Lin received the double BE in civil engineering and computer science and information engineering from the Tamkang University, Taipei, Taiwan, in 2004, and the double MS in civil engineering and computer science and information engineering from National Central University, Jhongli, Taiwan, in 2007. He is currently pursuing the PhD in computer science and information engineering at National Taiwan University, Taipei, Taiwan. His current research interests include image analysis, computer vision, surveillance, and pattern recognition.

Chih-Chang Yu received the BS in computer science and information engineering from National Central University, Taiwan, in 2002. He received the PhD from the same university in 2009. He became an assistant professor in the Department of Computer Science and Information Engineering at Vanung University in 2009. His research interest includes video analysis and pattern recognition.