Chinese Handwritten Character Segmentation in Form Documents

Jiun-Lin Chen, Chi-Hong Wu and Hsi-Jian Lee
Department of Computer Science and Information Engineering
National Chiao Tung University,
Hsinchu, Taiwan 30050

Abstract. This paper presents a projection based method for segmenting handwritten Chinese characters in form documents with known structures. In the preprocessing phase, a noise removal method is proposed that preserves stroke connections and character edge points. In the character segmentation phase, the projection profile analysis method is used to segment a text line image into projection blocks. In addition, projection blocks are classified into one of four types: mark, half-word, single-word, and two-word. Large blocks are then split and small blocks are merged. In addition, an OCR system is adopted to eliminate errors resulting from the inappropriate merging of Chinese numerical characters with other characters. As for 1319 Chinese characters are tested during our experiments, the correct segmentation rates of 92.34% and 91.76% are obtained with and without the OCR module.

Keywords: Noise removal, Projection profile analysis, Form document processing, Character segmentation, Optical character recognition.

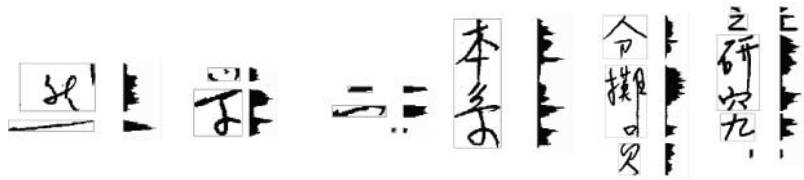
1. Introduction

Organizations receive many form documents daily, and processing these form documents requires much human effort. Automatic form document processing systems can reduce the human effort involved and increase efficiency. A typical automatic document processing system typically consists of six modules: preprocessing, document layout analysis, character segmentation, feature extraction, character recognition, and content understanding [5].

Character segmentation has long been a critical topic in document analysis. The performance of a character segmentation process significantly affects recognition results of an OCR (optical character recognition) system. A typical form document normally consists of three components: form frames, label fields, and data fields. In this paper, we present a projection based character segmentation algorithm for extracting handwritten characters from input form data fields. The form structures are assumed here to be known, i.e., form frames and label fields with printed characters have been extracted.

Although many systems have been proposed for recognizing printed documents [1] [2] [3] [4], only a few were designed for recognizing handwritten documents. The difficulties of the segmentation of handwritten Chinese characters include the following: (1) skewed and curved text lines, (2) different sizes of characters mixed in a text line, and (3) ill-separated characters. Figure 1 presents several examples which are difficult for character segment.

S.-W. Lee and Y. Nakano (Ed.): DAS'98, LNCS 1655, pp. 348-362, 1999.



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

Fig. 1 Sample charactres which are difficult for character segment.

Three strategies have been proposed for character segmentation [6]: dissection, recognition-based segmentation, and holistic methods. Among the methods proposed for character extraction, Lu [7] presented an overview of machine printed character segmentation. The investigation described various methods for segmenting machine printed characters. Seni and Choen [8] proposed an external word segmentation method for separating lines of unconstrained handwritten texts into words. Chiang and Yu [9] presented an iterative character segmentation method for irregularly formatted Chinese documents. In their method, they estimated parameters using an iterative merging algorithm. Lecolinet and Moreau [10] proposed an approach adopting characters pre-recognition, which detects connected components of single, connected, or broken characters. Lu and Shridhar [11] surveyed literature related to character segmentation for handwritten words.

Projection profile analysis and connected component analysis are two widely used dissection methods. By considering the speed and complexity, we adopt the projection profile analysis method for segmenting handwritten characters. The proposed system consists of four phases: preprocessing, document layout analysis and text line extraction, character segmentation, and post-editing.

In the preprocessing phase, we perform the Niblack [12] binarization operation and remove noises in the image. Data fields are then extracted according to the input form structure. In the character segmentation phase, either the horizontal or the vertical projection operation is performed according to the form structures of the extracted data fields. Next, the parameters are estimated for classifying projection blocks into different categories. The splitting and merging operations are then performed on blocks which are too large and blocks which are too small to satisfy the defined constraints. After all data fields are processed, users can examine the segmentation results through a graphical user interface. If any mis-segmentation has occurred, the users can modify the segmentation results through the user interface.

This study makes the following assumptions. (1) The data fields of input form structures are defined with respect to registration points. (2) The structures of all the form documents to be processed are known in advance. (3) Each data field contains only one text line. Situations in which more than one text line appears in a data field, are left for future research.

The rest of this paper is organized as follows. Section 2 describes the methods used for preprocessing and text line extraction. Section 3 presents the proposed character segmentation method. Next, Section 4 provides a revision of the segmentation process and the re-segmentation function. Section 5 summerizes the experimental results, and those results are discussed as well. Conclusions are finally drawn in Section 6.

2. Form Document Preprocessing

2.1 Form Image Binarization

We use the adaptive method proposed by Niblack [12] to binarize input documents. The mean and standard deviation of an $n \times n$ window ($n = 5$ in our experiments) centered on pixel i in an image are initially computed, denoting them as $mean_i$ and $stddev_i$ respectively. A threshold T for this window is then calculated according to the formula: $T = k \times mean_i + stddev_i$, where k is 0.02. If the gray level of the pixel is greater than the threshold T, the pixel is set to white. Otherwise, the pixel is set to black. This operation is executed on every pixel in the gray scale image. Figure 2 illustrates the differences between the global thresholding and adaptive methods.

(a) (b) (c)

Fig. 2. Binarization results. (a) Original images. (b) Global thresholding results. (c) Niblack results.

2.2 Noise Removal

Mask filtering and small connected component removal are two commonly used noise removal methods. Mask filtering checks the black pixels count in an $n \times n$ window and removes the black pixels if the count is smaller than a given threshold. Mask filtering is limited in that image edge points are frequently deleted.

To eliminate the drawbacks of mask filtering, we present a novel noise removal operation. The whole image is scanned from top to bottom, and left to right. For each black pixel in the binarized image, we check the pixels in a 3×3 window around it. If none of the 8-neighbors of the target pixel are black, we mark the target pixel as an isolated noise and remove it. If more than one of the 8-neighboring pixels are black, the following operations are performed in a 5×5 window.

(1) For each black pixel, P, if there are fewer than six black pixels in the 8-neighborhood of P and these black pixels have no black 8-neighbors in the outer ring, the black pixels in the central 3×3 window are denoted as noise and then removed. Figure 3 provides an illustrative example.

(2) For a black 8-neighbor, nb_8, of P which has a black 8-neighbor in the outer ring, we make nb_8 as the center of a side of a new 5×5 window as illustrated in Fig. 3.

If the black pixel count is smaller than 1/3 of the pixels in the new 5×5 window, then P is considered to be noise and is removed.

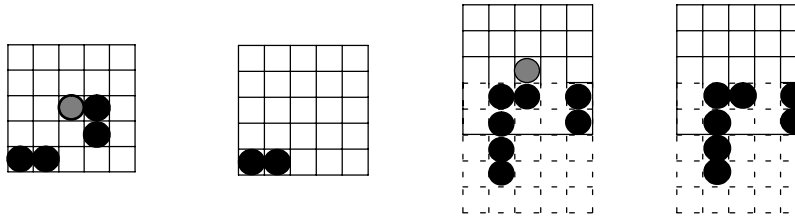


Fig. 3. The 5×5 window used for noise removal. (a) (c) Sample noise. (b) (d) Th noise removal results.

2.3 Text Line Extraction

To extract text lines from an input form document, we initially apply the method of Chen and Lee [13] to obtain the document structure. According to the coordinates of the registered corner point, $P_c = (x_c, y_c)$, and the relative coordinates of data fields $\{(x_{i_l}, y_{i_t}), (x_{i_r}, y_{i_b}) \mid i = 0, ..., n\}$ from the extracted form structure as shown in Fig. 4, the absolute coordinates of each data field can be obtained as follows:

$$P_{i_{l-t}} = P_c + P_{i_{l-t}} = (x_c, y_c) + (x_{i_l}, y_{i_t}),$$
$$p_{i_{r-b}} = P_c + P_{i_{r-b}} = (x_c, y_c) + (x_{i_r}, y_{i_b}).$$

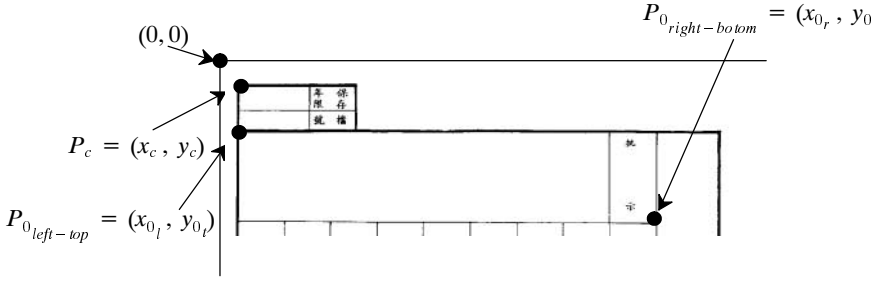


Fig. 4. The form structure and data field coordinates.

The text lines can then be extracted according to the absolute coordinates of the data fields. Next, the projection profiles of the data fields are analyzed to remove the field lines. Figure 5 presents an example of detecting left and right boundaries on a vertical text line. The positions of left and right field lines can be detected at the peaks of the projection profile and are denoted as f_l and f_r. The closest valley points are then found at both sides of each field line and denoted as v_{l_l}, v_{l_r}, . v_{r_l} and v_{r_r}. If the projection value at v_{l_r} is 0, which indicates a gap, we locate the left boundary of the text line at v_{l_r}. Otherwise, the text data touch f_l, and we locate the left boundary of the text line at v_{l_l}. If the projection value at v_{r_l} is 0, which indicates a gap, we locate the right boundary of the text line at v_{r_l}. Otherwise, the text data touch f_r, and we locate the right bound-

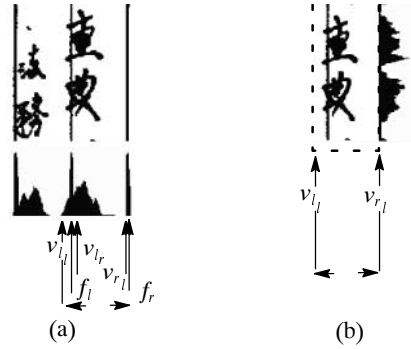


Fig. 5. (a) Locating left and right boundaries in vertical text lines extraction. (b) The extracted text line from (a).

ary of the text line at v_{r_r}. The top and bottom boundaries of the text line can be detected by a similar algorithm.

3. Chinese Handwritten Character Segmentation

Projection profile analysis and connected component analysis are two conventional methods for dissection strategy. In this paper, we adopt the projection profile analysis method for the following reasons:

(1) Segmentation performance. If characters in a text line are well spaced, both methods can separate these characters correctly. However, the connected component method generates more blocks and the operations required to merge these blocks together are much more complicated. On the other hand, if characters touch each other, both the projection and connected component methods are unable to separate these characters without further processing.

(2) Processing speed. Experimental results, as shown in Table 1, indicate that projection profile analysis is faster than the connected component analysis method.

3.1 Character Segmentation using Projection Profile Analysis

Since the structures of input form documents are known in advance, the writing direction in each data field is known before performing the projection operation. Text lines are then segmented by performing vertical and horizontal projection on horizontal and vertical text lines, respectively. While the following discussion is limited to character segmentation of vertical text lines, a similar procedure can segment horizontal text lines.

The input text lines are segmented at those positions where the projection values are smaller than a given threshold, $T_{projection}$. The bounding rectangles for each segmented area are then detected to obtain projection blocks. The threshold value, $T_{projection}$, is estimated as follows:

$$T_{projection} = w_{f_l} + w_{f_r},$$

$$where\ w_{f_l} = \begin{cases} v_{l_r} - f_l & \text{if characters touch the left form line.} \\ 0 & \text{otherwise.} \end{cases}$$

$$w_{f_r} = \begin{cases} v_{r_l} - f_r & \text{if characters touch the right form line.} \\ 0 & \text{otherwise.} \end{cases}$$

Table 1. Computation time of connected component analysis and projection profile analysis methods.

Image size / Method	Connected component analysis		Projection profile analysis (milli–seconds)
	Time (milli–seconds)	Number of components	
121 x 1948	874.670	237	41.363
142 x 2251	780.081	178	58.267
136 x 2002	309.138	43	34.027
2550 x 3162	30037.625	885	574.450

3.2 Projection Block Feature Extraction

For each detected projection block, the following features are extracted as illustrated in Fig. 6.

(1) Block width (W_i) : The width of a projection block.

(2) Block height (H_i) : The height of a projection block.

(3) Block gap (G_{ij}) : The vertical or horizontal distance between projection blocks i and j in horizontal or vertical text lines respectively.

(4) Block aspect ratio ($\frac{W_i}{H_i}$): The ratio of block width to block height.

(5) Density ($\frac{B_i}{W_i \times H_i}$): The ratio of the number of black pixels to the area of a projection block.

Since the aspect ratio of most handwritten Chinese characters is nearly 1, another two related features can be derived, i.e. average character width ($avgC_W$) and average character height ($avgC_H$).

- The average block width in a text line:

$$avgB_W = \sum_{i=1}^{N} w_i/N, \text{ where } N \text{ is the number of projection blocks.}$$

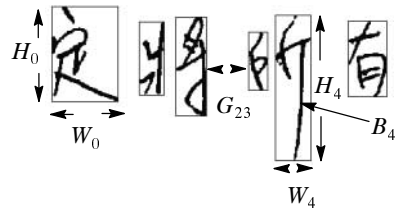


Fig. 6. Projection block features in a horizontal text line.

- The average character width of a vertical text line is obtained from the projection blocks whose widths are larger than $avgB_w$, since smaller blocks are caused by noise and partial characters in a text line.

$$avgC_W = \sum_{i=1}^{M} w_i/M, \ \forall\ W_i > avgB_w,$$

where M is the number of blocks whose widths are larger than $avgB_w$.

- The average character height of a vertical text line is calculated from those projection blocks whose heights are close to $avgC_W$.

$$avgC_H = \sum_{i=1}^{K} h_i/K, \ \forall\ h_i \in \{h_i \mid \frac{h_i}{avgC_W} \approx 1\ , i = 1...K\}.$$

If several characters in the text line touch one another, a large and unreasonable value for $avgC_H$ will be obtained. Under such circumstances, we set $avgC_H$ equal to $.avgC_W$

According to the above features, projection blocks can be classified into four categories: mark, half-word, single-word, and two-word as follows.

- Mark: The height of a projection block in a vertical text line is smaller than $T_{mark} \times avgC_H$, where $T_{mark} = 0.3$. Figure 7 shows ten projection blocks classified as marks.

Fig. 7. Projection blocks classified as marks (denoted by *). (a) (b) (c) Examples form vertical text lines. (c) (d) Examples from horizontal text lines.

- Half-word: Projection blocks of vertical text lines satisfy

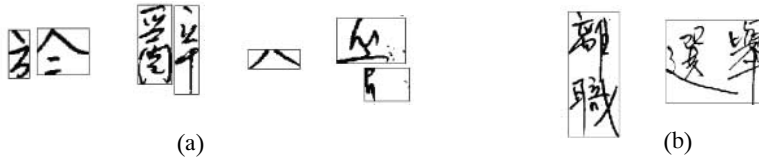
$$T_{mark} \times avgC_H < H_i < T_{half-word} \times avgC_H, \text{ where } T_{half-word} = 0.7.$$

Figure 8 (a) presents some half-word projection blocks.

- Single word: Projection blocks of vertical text lines satisfy

$$T_{half-word} \times avgC_H < H_i < T_{two-word} \times avgC_H, \text{ where } T_{two-word} = 1.5.$$

- Two-word: If the height of a projection block in a vertical text line is greater than $T_{two-word} \times avgC_H$, the block is classified as a two-word block. Figure 8 (b) depicts two examples of two-word projection blocks.



(a) (b)

Fig. 8. (a) Half-word projection blocks. (b) Two-word projection blocks.

After classifying all blocks of a text line, we estimate two other features: average inter-character gap (*avgInterG*) and average intra-character gap (*avgIntraG*), according to the category and spatial information of all projection blocks.

(1) Average inter-character gap (*avgInterG*) : The average distance between characters.

$$avgInterG = \frac{\sum G_{ij}}{M},\ G_{ij} = B_{j_{top}} - B_{i_{bottom}},\ \text{or}$$
$$G_{ij} = B_{j_{left}} - B_{i_{right}}\ \forall B_i \in \{half\text{-}word,\ single\text{-}word,\ two\text{-}word\}.$$

(2) Average intra-character gap (*avgIntraG*): The average distance between small projection blocks that may be single Chinese characters.

$$avgIntraG = \frac{\sum G_{ij}}{N},\ G_{ij} = B_{j_{top}} - B_{i_{bottom}},\ \text{or}\ G_{ij} = B_{j_{left}} - B_{i_{right}},$$
$$\text{and}\ G_{ij} < avgIntraG,\ \forall B_i \in \{mark,\ half\text{-}word,\ single\text{-}word\}.$$

Figure 9 summerizes the distances between different categories of projection blocks used herein to estimate intra-character and inter-character gaps.

	Mark	Half-word	Single-word	Two-word
Mark	●	●	●	
Half-word	●	✓ ●	✓ ●	
Single-word	●	✓ ●	✓	
Two-word			✓	✓

● Distances used to estimate average intra-character gaps.

✓ Distances used to estimate average inter-character gaps.

Fig. 9. The consideration of distances between different categories of projection blocks for estimating average intra-character and inter-character gaps.

3.3 Splitting large Projection Blocks

The two-word projection blocks may contain more than one character, and such blocks must be split. To locate the appropriate cut position for splitting, we search the possible cut area defined as follows.

$$(B_{i_{top}} + avgC_H \times T_{upper-bound},\ B_{i_{top}} + avgC_H \times T_{lower-bound}),$$
$$\text{where } T_{upper-bound} = 0.5 \text{ and } T_{lower-bound} = 1.5.$$

The cut position is located inside the possible cut area according to the following criteria. (1) The projection value and the number of crossing strokes at the cut position must be a minimum in the possible cut area. (2) The heights of the split blocks must be as close as possible to the average block height, $avgB_H$.

An example of the possible cut area and the located cut position is illustrated in Fig. 10. Examples of split results are shown in Fig. 11.

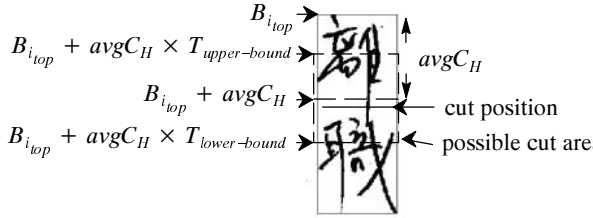


Fig. 10. The possible cut area and the located cut position for a two-word block.

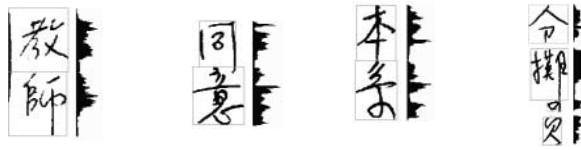


Fig. 11. Examples of split two-word projection blocks.

3.4 Merging Small Projection Blocks

As general known, a handwritten character may be segmented into more than one projection block, accounting for why these blocks must be merged together to extract correct characters. Herein, a two phase process is applied to merge mark blocks, half-word blocks and single-word blocks.

Phase One: Merging Mark Blocks and Single-word Blocks. Mark projection blocks which satisfy one of the following criteria, are classified as punctuation marks.

- The inter-character distances between a given block and its neighboring blocks are all greater than the average inter-character gap, *avgInterG*.
- The block id located at the down-right direction of its predecessor block.

A mark projection block is considered as a primitive stroke and should be merged with its successor block by applying the following heuristic rules.

- The width of the mark block is smaller than the width of its successor block.
- The overlapping length in the horizontal direction, of the mark block and its successor block, is larger than a predefined threshold value.

- The distance between the mark block and its successor block is smaller than the average intra-character gap, *avgIntraG*.
- The height of the merged result is close to the average character height, $avgC_H$.

Figure 12 presents examples of merged results. A merging probability is also provided for each merge operation. These probability values can be used in the manual post-editing (Section 4.2). The merging possibility is formally defined below.

$$PM_{i,i+1} = \begin{cases} 1 - \dfrac{G_{i,i+1}}{avgInterG}, & \text{if } avgInterG > G_{i,i+1} \\ 0.1, & \text{otherwise} \end{cases}$$

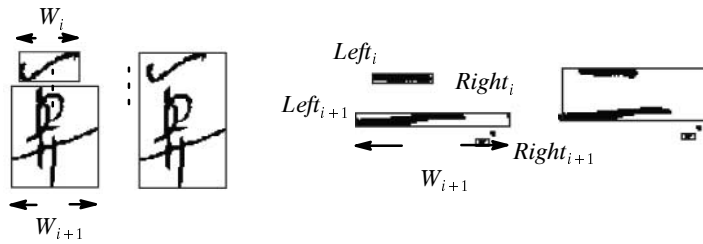


Fig. 12. Examples of merging mark blocks as primitive strokes in vertical text lines.

The single-word projection block is merged with its successor block if the distance between them is smaller than *avgInterG*, and the height of the merged result is close to $avgC_H$. Figure 13 depicts two examples. The merge probability is defined as:

$$PS_{i,i+1} = \begin{cases} 1 - \dfrac{G_{i,i+1}}{avgIntraG} - ((\dfrac{avgC_H}{H_i + G_{i,i+1} + H_{i+1}}) - 1), \\ \qquad\qquad \text{if } avgC_H > H_i + G_{i,i+1} + H_{i+1} \\ 1 - \dfrac{G_{i,i+1}}{avgIntraG} - (1 - (\dfrac{avgC_H}{H_i + G_{i,i+1} + H_{i+1}})), \textit{otherwise.} \end{cases}$$

Fig. 13. Two single-word block merged results in vertical text lines.

Phase Two: Merging Half-word Blocks. Before phase two is performed, the values of $avgC_W$, $avgC_H$, *avgInterG*, and *avgIntraG* are re-estimated according to the merged results of phase one. A half-word block is merged with its successor block if the distance between them is smaller than *avgInterG*, and the height of merged result is close to

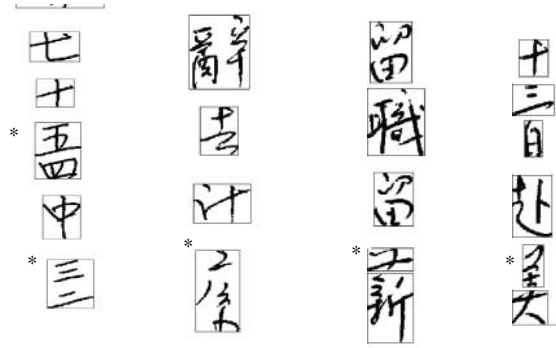
$avgC_H$. Examples are shown in Fig. 14. The formula for merging probability $PH_{i,i+1}$ is the same as the formula of $PS_{i,i+1}$.

(a) (b)

Fig. 14. Examples of merging half-word blocks in vertical text lines.

4. Character Segmentation Revision

After the above processes have been completed, most characters can be segmented correctly. However, some characters may be segmented erroneously for the following reasons. (1) Character sizes in a text line varies widely. (2) Inter-character and intra-character gaps vary widely. (3) Characters are written abnormally.



(a) (b) (c) (d)

Fig. 15. Examples of incorrect segmentation results (denoted by *). (a) (b) Merging error. (c) (d) Segmentation error.

4.1 Character Segmentation Revision with the OCR Module

In our experiments, we found that consecutive half-word projection blocks are probably individual Chinese characters. In a vertical text line, consecutive half-word blocks may result from the following situations:

(1) The blocks may be "一", "二", "三", "四", "五", "六", "七", "八", "九", or "十" (i.e., Chinese characters for numbers 1-10).

(2) The blocks may be individual characters written smaller than normal characters.

(3) The blocks may actually be parts of characters.

In order to reduce the segmentation errors generated from these ambiguities, we include an OCR module in our segmentation process. The recognition results can be used to improve the segmentation process. However, recognizing every possible pro-

jection block is time-consuming and impractical. Thus, the OCR module is designed only for recognizing Chinese numeric characters which introduced most of the segmentation errors in our experiments.

When two consecutive half-word blocks satisfy the merging criteria described in section 3.4, we apply the OCR module to these two blocks for recognition. For a situation in which the OCR module recognizes the half-word block as a Chinese numerical character, the merge operation should be abandoned and this half-word block made into a single character block. Figure 16 illustrates an example.

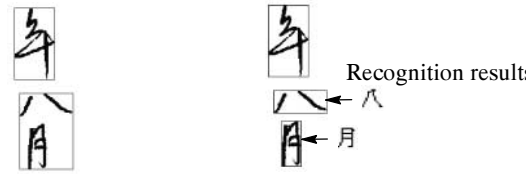


Fig. 16. Segmentation results with and without OCR module.

4.2 Character Segmentation with User Intervention

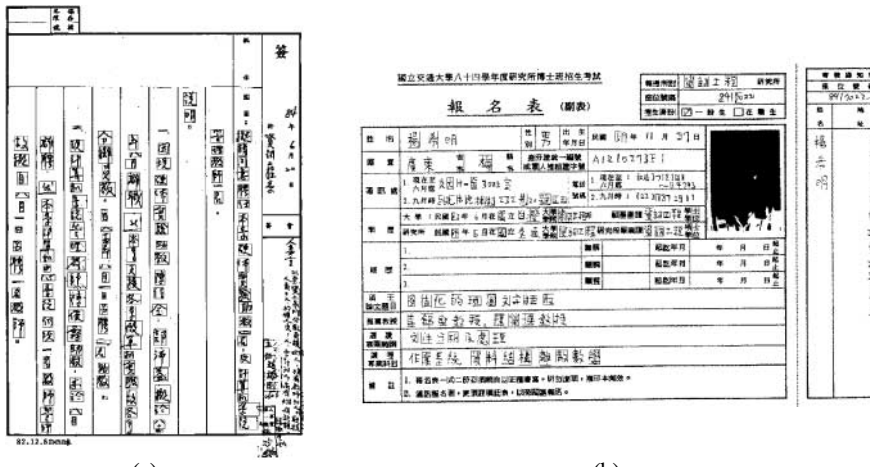
The proposed system provides a user interface for users to correct some segmentation errors. Users can drag the mouse pointer to merge blocks that are not merged in the above process. To correct erroneously merged blocks, users can double click on the merged blocks, causing the system to separate the blocks at the point of lowest merge probability.

5. Experimental Results and Discussions

The proposed system was implemented in Visual C++ with MFC (Microsoft Foundation Class) on an IBM Pentium90 personal computer. The test sample documents were scanned at a resolution of 300 dpi (dots per inch).

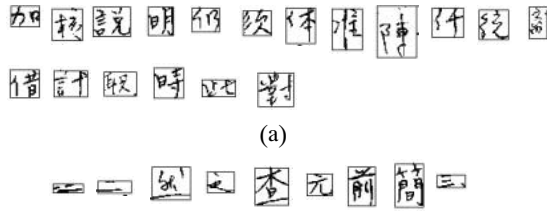
Figure 17 show the segmentation results of vertical and horizontal form documents. Figure 18 presents correctly segmented characters with left-right and top-bottom structured extracted from horizontal and vertical text lines, respectively. Such characters are usually mis-segmented in projection based character segmentation methods.

Herein, we tested a total of nine sample form documents with seventy six text lines and 1319 Chinese characters. Table 2 displays the segmentation rates and processing times obtained without using the OCR module. Table 3 summerizes the results obtained with the OCR module involved. 1206 Chinese characters were correctly extracted without the aid of the OCR module, The correct segmentation rate is 91.43%. With the aid of the OCR module, 1218 Chinese characters were correctly extracted with a correct segmentation rate of 92.34%. The results of the proposed character segmentation method are satisfactory.



(a) (b)

Fig. 17. Segmentation results. (a) A vertical form document. (b) A horizontal document.



(a)

(b)

Fig. 18. Correct segmented characters. (a) Left-right structured characters from horizontal text lines. (b) Top-bottom structured characters from vertical text lines.

Table 2 Correct segmentation rates and processing times without the OCR module.

	Total words	Correctly segmented words	Correct segmentation rates	Processing time(ms)
Fig. 1	164	157	95.73 %	2831.634
Fig. 2	144	132	91.67 %	2182.325
Fig. 3	118	114	96.61 %	2176.854
Fig. 4	125	121	96.81 %	2543.087
Fig. 5	175	157	89.71 %	2611.529

In our experiments, erroneous segmentation results were due to the following.

(1) The heights of characters in a vertical text line vary widely, as shown in Fig. 19 (a).

(2) Under some circumstances, the cut position with the smallest crossing stroke number and projection value was not the correct split location, as displayed in Fig. 19 (b)
(3) Sparsely written characters, as shown in Fig. 19 (c).
(4) Characters are written too close, as shown in Fig. 19 (d).

(a) (b) (c) (d)

Fig. 19. Examples of erroneously segmented characters.

Table 3 Correct segmentation rate and processing time with the aid of OCR system.

	Total words	Correctly segmented words	Correct segmentation rate	Processing time(ms)
Fig. 1	164	158	96.34 %	4595.960
Fig. 2	144	138	95.93 %	3354.280
Fig. 3	118	115	97.45 %	3782.252
Fig. 4	125	121	96.81 %	2875.377
Fig. 5	175	157	89.71 %	5619.013

6. Conclusions

This paper presents a projection based character segmentation system for separating handwritten Chinese characters in form documents. Handwritten characters vary greatly in style and appearance making it difficult to segment characters accurately in text lines. Our system uses a projection based segmentation method with the aid of an OCR module to achieve better performance. The proposed method can successfully segment characters with left-right or top-bottom structures which cannot be correctly segmented in typical projection based methods. Experimental results demonstrates the validity of our proposed system. The correct segmentation rate is 92.34%. The average process time for segmenting a character is about 29.45ms.

Future research aimed at to enhancing the performance of the proposed system should (1) find more accurate features, (2) segment data fields containing more than one text line, (3) remove form frames from text lines, (4) segment text line characters with editing symbols, and (5) incorporate the system as a module in a complete document analysis system.

References

[1] Srihari, S.N., "Document Image Understanding," Proc. IEEE Computer Society Fall Joint Computer Conf., pp.87-96, 1986.

[2] Wang, D. and S.N. Srihari, "Analysis of Form Images," Proc. 1st Internat. Conf. Document Anal. Recognition, pp.181-191, 1991.

[3] Casey, R.G., D.R. Ferguson, K. Mohiuddin and E. Walach, "Intelligent forms processing system," Machine Vision and Applications, Vol. 5, pp. 143-155, 1992.

[4] Lam, S.W., L. Javanbakht, and S.N. Srihari, "Anatomy of a form reader," Proc. 2nd Intern. Conf. on Document Analysis and Recognition, pp. 506-509, 1993.

[5] L. A. Fletcher and R. Kasturi, "A robust algorithm for text string separation from mixed text/graphic images," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 10, No. 6, pp. 910-918, 1988.

[6] R. G. Casey and E. Lecolinet, "Survey of methods and strategies in character segmentation," IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 18, No. 7, pp.690-706, July 1996.

[7] Y. Lu, "Machine printed character segmentation - An overview," Pattern Recognition, Vol. 28, No. 1, pp. 67-80, 1995.

[8] G. Seni and E. Cohen, "External word segmentation of off-line handwritten text lines," Pattern Recognition, Vol. 27, No. 1, pp. 41-52, 1994.

[9] C. C. Chiang and S. S. Yu, "An iterative character segmentation method for irregularly formatted Chinese documents," in Proceedings of the 5th Optical Character Recognition and Document Analysis, Chung Li, Taiwan, 1996, pp. 61-67.

[10] E. Lecolinet and J. V. Moreau, "A new system for automatic segmentation and recognition of unconstrained zip codes," in Proceedings Sixth Scandinavian Conference Image Analysis, Oulu, Finland, June 1989, pp. 585.

[11] Y. Lu and M. Shridhar, "Character segmentation in handwritten words - An overview," Pattern Recognition, Vol. 29, No. 1, 1996, pp. 77-96.

[12] W. Niblack, An Introduction to Digital Image Processing, Prentice Hall, 1986.

[13] J. L. Chen and H. J. Lee, "An efficient algorithm for Form Structure Extraction using Strip Projection," Pattern Recognition, Vol. 31, No. 9, pp. 1353-1368, 1998.