Touching Character Splitting of Chinese Handwriting Using Contour Analysis and DTW

Liang XU, Fei YIN, Cheng-Lin LIU

National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences, Beijing 100190, P.R. China
E-mail: {lxu,fyin,liucl}@nlpr.ia.ac.cn

Abstract: The splitting of touching characters remains a challenge in over-segmentation, which is crucial to the performance of integrated segmentation-recognition of handwritten character strings. In this paper, we propose a new method based on contour analysis for touching character splitting in Chinese handwriting. To reliably locate splitting points on the contour of touching pattern, we pair upper and lower contour points using DTW (dynamic time warping) such that a corner point in upper/lower contour can be always paired with a proper contour point of opposite side for forming a splitting path. The splitting paths can then be verified by incorporating character recognition and contextual information. Our preliminary experiments on an image database of unconstrained Chinese handwriting demonstrate that the proposed method can give high recall rate of segmentation point detection.

Key Words: Chinese handwriting, touching character splitting, contour analysis, DTW, splitting paths

1. INTRODUCTION

Character string (word, sentence or text line) recognition is one of essential tasks in document image analysis. Due to the irregularity of between-character spacing, touching characters and the variability of character shapes, the recognition of handwritten character strings remains a great challenge. To overcome the difficulty of character segmentation, the strategy of integrated segmentation-recognition is widely adopted for handwritten character string recognition. By this strategy, the character string image is over-segmented into primitive segments each composing a character or sub-character, and the primitive segments are grouped into characters in string recognition incorporating character recognition and contextual information. The string recognition performance crucially relies on the over-segmentation, which is hoped to detect all the between-character boundaries and generate few redundant boundaries (false alarms).

Many methods have been proposed for alphanumeric character string (word) segmentation [1, 2], and some works have focused on the splitting of touching characters (multiple characters in a connected component, also called a touching pattern), which appear frequently in both printed and handwritten images. After connected component labeling, the connected components possibly containing multiple characters are analyzed to split at possible between-character boundaries. According to the features analyzed, the methods of touching pattern splitting can be categorized into foreground-based methods (including contour analysis [3] and skeleton analysis), background-based methods (including background thinning and concavity analysis [4]), and hybrid methods combining foreground and background information [5].

Compared to alphanumeric handwriting, the segmentation of handwritten Chinese characters poses some new challenges. First, Chinese handwriting must be recognized at sentence or text line level because there is no extra between-word space. Second, many Chinese characters consist of multiple components (radicals), and so, within-character gaps and between-character gaps are mixed. Third, Chinese characters have complicated structures and form complicated types of between-character touching. Some touching patterns of Chinese handwriting are shown in Fig. 1, where some touching points are difficult to detect(e.g.(e) and (f)).

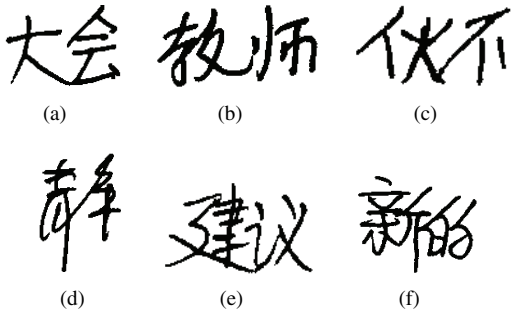


Fig. 1: Examples of Chinese touching patterns.

Chinese handwriting segmentation has been attacked by some researchers in different ways. Some methods ([6, 7, 8]) try to separate characters before recognition and thus can be called as dissection-based methods. Such methods locate between-character boundaries merely according to geometric features,

978-1-4244-7210-9/10/$26.00 ©2010 IEEE

and would lead to good performance if combined with character recognition and linguistic context. The method of Tseng and Lee [9] combines candidate splitting path generation and character recognition. The character separation techniques in these works, stroke extraction and grouping [6], probabilistic Viterbi algorithm [9], background and foreground skeleton analysis [7, 8], have some demerits despite their merits. Stroke extraction itself is an unsolved problem, the probabilistic Viterbi algorithm is computationally expensive and generates too many splitting paths, and background/foreground thinning generates distorted and spurious skeleton branches and some splitting paths do not correspond to skeleton branches. The method of Liu et al. [10] segments seven touching types in Japanese handwriting but fails to segment more complicated, though infrequent touching types.

In this paper, we focus on touching pattern splitting in Chinese handwriting for over-segmentation. We aim at a high recall rate of between-character boundary (splitting point) detection but allow low precision, i.e., a character is allowed to be segmented into multiple fragments. Provided that the between-character boundaries are included in splitting points, the characters can be correctly segmented in string recognition incorporating character recognition and contextual information. Our method is based on contour analysis and can be viewed as an improvement of the method in [10]. To reliably locate splitting points on the contour of connected component of touching pattern, we pair upper and lower contour points using DTW (dynamic time warping) such that a corner point in upper/lower contour can be always paired with a proper contour point of opposite side. The splitting paths formed by paired contour points can then be verified by character recognition and contextual information. Our experimental results on an image database of unconstrained Chinese handwriting show that the proposed method can detect the between-character splitting paths for most touching characters.

The rest of this paper is organized as follows. Section 2 gives an overview of the splitting method. Section 3 and 4 illustrates the main procedures of touching pattern splitting: contour analysis and matching, splitting paths generation and validation. Experiment results are presented in Section 5 and concluding remarks are offered in Section 6.

2. OVERVIEW OF SPLITTING METHOD

In this work, we focus on the over-segmentation of touching characters, particularly those of single touching. We choose the contour analysis-based method because the contour of touching pattern has prominent shape information of splitting points, which mostly lie at corner points. However, some splitting points do not have corner points on both upper and lower contour, such as the ones in Fig. 1(a) and 1(b). In the case of Fig. 1(b), both foreground skeleton and background skeleton cannot locate the splitting point. Foreground thinning and background thinning are also computationally expensive.

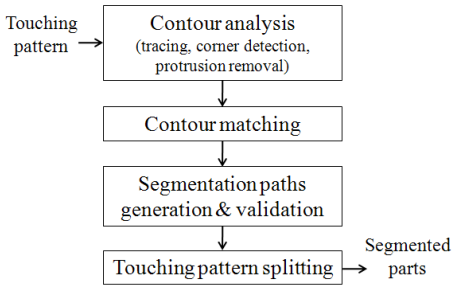


Fig. 2: Touching pattern splitting process.

Except the ligature-type touching as shown in Fig. 1(d), almost all splitting points have at least one corner point at upper or lower contour, and the splitting path can be formed by connecting a corner point on upper/lower contour with a contour point (not necessarily corner) on the opposite side. Hence, the task of splitting path formation is reduced to a problem of point matching between upper and lower contours.

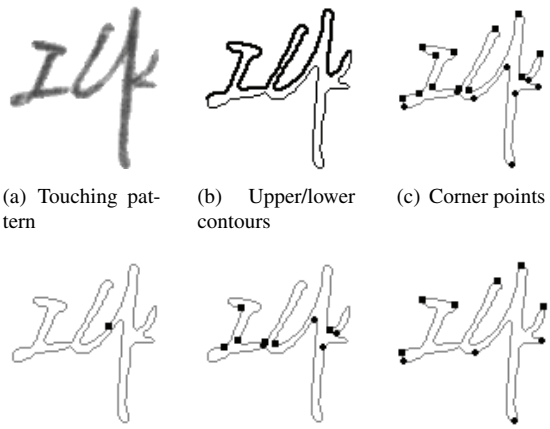
Our over-segmentation process consists of two stages: connected component analysis and touching pattern splitting. During connected component analysis, connected components are separated, and for subsequent touching pattern splitting, the stroke width (SW) and text line height (LH) are estimated as done in [10]. After merging connected components that are highly overlapping in horizontal direction, the components with large width or width-to-height ratio are possible touching patterns and undergo the process of touching pattern splitting. In cases of merged components, contour analysis is performed on the widest connected component.

The touching pattern splitting process consists of four steps as shown in Fig. 2. In contour analysis, the upper and lower contours of touching pattern are traced, corner points are detected, and the contour parts of protruding strokes are removed. In contour matching, the points of upper and lower contours after protrusion removal are paired by DTW. Splitting paths are generated based on contour matching, and finally, the touching pattern is separated at splitting paths.

3. CONTOUR ANALYSIS AND MATCHING

In the binary image of touching pattern, both the upper contour and lower contour are traced from the left-most stroke point to the right-most stroke point. From both upper and lower contours, corner points are detected using the algorithm of Rosenfeld and Johnston [11]. For long contour segments without prominent corner points, polygonal approximation [12] is ap-

plied to find extra break points. A break point is found when the maximum point-to-chord distance is greater than a threshold (set equally to estimated SW in our experiments). For a touching pattern, Fig. 3(c) and Fig. 3(d) show the corner points and break points, respectively. The corner points are classified into concave points (Fig. 3(e)) and convex points (Fig. 3(f)). Concave points are the corner points turning counter-clockwise in upper contour and those turning clockwise in lower contour, while convex points are the corner points turning clockwise in upper contour and those turning counter-clockwise in lower contour. The concave points and break points are feature points for locating splitting paths.



(a) Touching pattern (b) Upper/lower contours (c) Corner points

(d) Break points (e) Concave points (f) Convex points

Fig. 3: Contour analysis on a touching pattern.

For locating splitting points and generating splitting paths, the feature points need to be matched with contour points on the opposite side. The matched points, one on the upper contour and one on the lower contour, are expected to be on two sides of a stroke such that the connected line between them cuts a stroke or a horizontal bridge. A vertical stroke protruding upward or downward does not form a bridge because its two sides are both on the upper contour or lower contour, and therefore, cannot be matched by upper-lower contour matching. We thus remove such protruding strokes (Fig. 4(a)) before contour matching.

For identifying a protruding stroke, we examine three consecutive corner points CP_0, CP_1, CP_2 on the upper or lower contour. If CP_1 is a convex point and two lines (CP_0, CP_1) and (CP_1, CP_2) are approximately parallel (the angle of CP_1 is nearly zero), the three points correspond to a protruding stroke. To remove the protruding stroke, the contour points between CP_0 and CP_2 are deleted from the edge point sequence of upper or lower contour. Fig. 4(b) shows the modified upper and lower contours after removing protruding strokes. Removing protruding strokes helps reduce the computation cost of contour matching and improve the reliability of contour points pairing.

(a) Protruding strokes (b) Modified contours

Fig. 4: Identification and removal of protruding strokes.

After removing protruding strokes, the upper and lower contours are matched by dynamic time warping (DTW). DTW is a dynamic programming algorithm minimizing a string edit distance for matching two sequences of points, and has been widely used in matching time sequences such as speech signals ([13]). After DTW, the points of two sequences are corresponded (alignment). The effect of alignment largely depends on the edit costs (substitution, deletion and insertion costs). To encourage upper/lower contour points with similar horizontal coordinates to be matched such that segmentation paths are nearly vertical, we set the substitution cost for two points as

$$dist = \theta \times |dx| + |dy| \tag{1}$$

where dx and dy are differences of horizontal and vertical coordinates, respectively, and θ is empirically set as 3. The deletion cost and insertion cost are set the same as the substitution cost.

4. SPLITTING PATHS FORMATION

Candidate splitting paths are formed by connecting feature points (corner points and extra break points) with their paired points on the opposite side of contour found by DTW (if multiple points are paired with a feature point, the one of minimum distance is selected). The candidate splitting paths are ordered from left to right, and some redundant splitting paths are removed according to simple geometric rules.

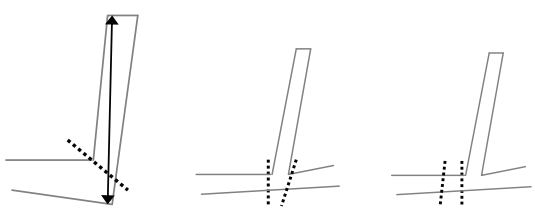
For describing the validation rules for splitting paths, some parameters are defined as below:

- Path$_i$: the i-th splitting path, formed by a pair of upper-lower contour points (at least one of them is a feature point);
- dist$_i$: modified city block distance as in Eq. (1) between the paired contour points;
- L_i: length of the i-th splitting path in number of pixels;
- L_i^f: number of foreground pixels on the i-th splitting path;
- x_i^u, x_i^l: horizontal coordinates of the upper and lower contour points of i-th splitting path;

- y_i^u, y_i^l: vertical coordinates of the upper and lower contour points of i-th splitting path;
- x_i^c, y_i^c: horizontal and vertical coordinates of the center of i-th splitting path.

We have four validation rules as follows.

- RULE1: If the distance between the paired contour points is too long ($dist_i > T_1$), the splitting path is invalid. T_1 is set as $\theta_1 \times SW$.
- RULE2: If $dist_i > T_2$ ($T_2 < T_1$) and $L_i^f / L_i < \theta_3$, the splitting path is invalid. T_2 is set as $\theta_2 \times SW$, and $0 < \theta_3 < 1$.
- RUEL3: If a splitting path crosses a long vertical stroke, the path is invalid. An example is shown in Fig. 5(a), where the dashed line crosses a long vertical stroke, which is identified by the maximum vertical run length (denoted by doube arrow).
- RULE4: If two splitting pathes are close to each other overlapped ($|x_i^c - x_j^c| < T_3$, $T_3 = \theta_4 \times SW$), one of them is selected to be invalid as follows
  (a) If the distance of one path is much larger than the other, this path is invalid.
  (b) If two paths have similar distances and vertical coordinates, we check whether there is a vertical stroke between them or not. If there is not a vertical stroke, the paths are considered "visible" to each other, and the one with larger distance is invalid; otherwise, the two paths are "non-visible", and both are valid to be verified by postponed character recognition and context. Fig. 5(b) and Fig. 5(c) show examples of non-visible and visible splitting paths, respectively.



(a) RULE3 (b) Non-visible (c) Visible

Fig. 5: Schematic illustration of validation rules

In summary, the above four rules are applied to remove redundant splitting paths. The former three rules examine splitting paths individually considering the path distance and foreground pixel ratio. The last rule examines the relationship between two neighboring splitting paths to see whether they are confusing (non-visible) or not.

5. EXPERIMENTAL RESULTS

We evaluated the effects of the proposed splitting method on an dataset of touching characters collected from the public Chinese handwriting database HIT-MW [14] database. We extracted 4,472 samples of single-touching Chinese character pairs for our experiment, some examples are shown in Fig.1. All the touching character images have correct (ground-truth) splitting paths annotated manually. We can decide whether a splitting path located by an over-segmentation algorithm is correct or not by comparing with the ground-truth. We decide a splitting path to be correct if it is the closest to a ground-truth path among all the candidate splitting paths and there is not long vertical stroke between them (i.e., they are visible to each other). The overall performance of over-segmentation is measured by the recall and precision rates as well the harmonic average (F-measure):

$$Recall = \frac{\#\text{ of correct splitting paths}}{\#\text{ of ground-truth paths}} \times 100\% \quad (2)$$

$$Precision = \frac{\#\text{ of correct splitting paths}}{\#\text{ of candidate splitting paths}} \times 100\% \quad (3)$$

$$F\text{-}measure = \frac{2 \times Recall \times Precision}{Recall + Precision} \times 100\% \quad (4)$$

In our experiments, the empirical parameters of our splitting algorithm were set as $\theta_1 = 4.8$, $\theta_2 = 2$, $\theta_3 = 0.9$, $\theta_4 = 1.5$.

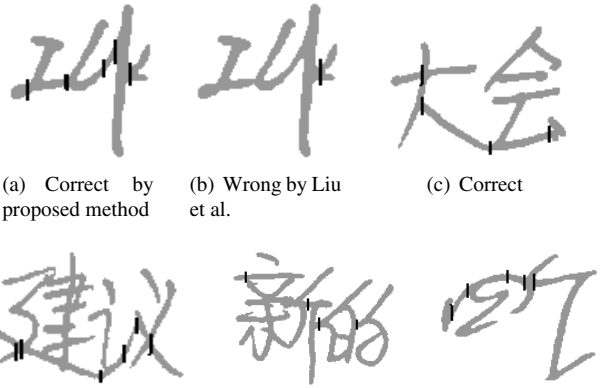
We compare the performance of the proposed method with an existing method that has shown success in Japanese handwritten character string recognition (Liu et al. [10]). Table 1 lists the correct rates of the proposed method and the one of Liu et al. It is evident that the proposed splitting algorithm achieves a much higher recall rate than the one of Liu et al. It should be noted that the method in [10] splits ligatures (which have no corner points) by analyzing single stroke regions, but the proposed method does not treat ligatures. By considering single stroke regions, the proposed method can give higher recall rate.

The high recall rate implies most between-character boundaries of touching patterns are correctly located. Though there are many extra candidate splitting paths (low precision), they can be verified in postponed string recognition incorporating character recognition and contextual information. On the contrary, if the between-character boundary is not located, the mis-segmentation cannot be restored in string recognition. Thus, a high recall rate is essential for character string recognition.

Fig.6 shows some examples of over-segmentation. Fig. 6(a) shows the splitting paths located by the proposed method, which include the correct one. Fig. 6(b) shows that the existing method of Liu et.al.[10] fails to locate the correct splitting path. This is because the touching portion does not conform to the conditions of one of seven touching types in [10]. Figs. 6(c), 6(d) and 6(e) show touching patterns with correct splitting paths

Table 1: Performance of touching character splitting on HIT−MW data.

Method	Recall	Precision	F-measure
Liu et al. [10]	62%	60%	61%
Proposed method	84%	16%	27%



(a) Correct by proposed method (b) Wrong by Liu et al. (c) Correct

(d) Correct (e) Correct (f) Wrong by proposed method

Fig. 6: Examples of touching character over-segmentation.

successfully located by the proposed method. In Fig. 6(f), the proposed method fails to locate the annotated slant splitting path. This is due to the improper matching between upper and lower contours, which we need to investigate further in the future.

6. CONCLUSION

We presented a new over-segmentation method for splitting touching characters in Chinese handwriting. The main feature of the method is that we use DTW to pair contour feature points with the best matched points on the opposite upper/lower side. This enables the locating of splitting paths even when there is no feature point on upper or lower contour. Our preliminary experiments demonstrate that the proposed method can locate between-character boundaries in touching patterns with high recall rate. Future efforts will be made to refine the filtering of splitting paths for improving the precision, and to verify splitting paths incorporating character recognition and contextual information.

ACKNOWLEDGEMENTS

This work is supported by the National Natural Science Foundation of China (NSFC) under grants no. 60775004, 60825301 and 60933010. The authors thank Tonghua Su for helpful discussions.

References

[1] R.G. Casey, E. Lecolinet. A survey of methods and strategies in character segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(7): 690-706, 1996.

[2] Y. Lu, M. Sridhar. Character segmentation in handwritten words: an overview. *Pattern Recognition*, 29(1): 77-96, 1996.

[3] N.W. Strathy, C.Y. Suen, A. Kryzyzak. Segmentation of handwritten digits using contour features. *Proc. 2nd ICDAR*, Tsukuba, Japan, 1993, pp.577-580.

[4] U. Pal, A. Belaid, Ch. Choisy. Touching numeral segmentation using water reservoir concept. *Pattern Recognition Letters*, 24(2): 261-272, 2003.

[5] Y.-K. Chen, J.-F. Wang. Segmentation of single- or multiple-touching handwritten numeral string using background and foreground analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11): 1304-1317, 2000.

[6] L.Y. Tseng, R.C. Chen. Segmenting handwritten Chinese characters based on heuristic merging of stroke bounding boxes and dynamic programming. *Pattern Recognition Letters*, 19(10): 963-973, 1998.

[7] S. Zhao, Z. Chi, P. Shi, H. Yan. Two-stage segmentation of unconstrained handwritten Chinese characters. *Pattern Recognition*, 36(1): 145-156, 2003.

[8] Z. Liang, P. Shi. A metasynthetic approach for segmenting handwritten Chinese character strings. *Pattern Recognition Letters*, 26(10): 1498-1511, 2005.

[9] Y.-H. Tseng, H.-J. Lee. Recognition-based handwritten Chinese character segmentation using a probabilistic Viterbi algorithm.*Pattern Recognition Letters*, 20(8): 791-806, 1999.

[10] C.-L. Liu, M. Koga, H. Fujisawa. Lexicon-driven segmentation and recognition of handwritten character strings for Japanese address reading. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(11): 1425-1437, 2002.

[11] A. Rosenfeld, E. Johnston. Angle detection on digital curves. *IEEE Trans. Computers*, 22: 875-878, 1976.

[12] U. Ramer. An iterative procedure for the polygonal approximation of plane closed curves. *Comput. Graphics Image Process*, 1: 244-256, 1972.

[13] H. Sakoe, S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 26(1): 43-48, 1978.

[14] T. Su, T. Zhang, D. Guan. Corpus-based HIT-MW database for offline recognition of general-purpose Chinese handwritten text. *Int. J. Document Analysis and Recognition*, 10(1): 27-38, 2007.