Dynamic Text Line Segmentation for Real-Time Recognition of Chinese Handwritten Sentences

Da-Han Wang, Cheng-Lin Liu
National Laboratory of Pattern Recognition (NLPR)
Institute of Automation of Chinese Academy of Sciences
95 Zhongguancun East Road, Beijing 100190, P.R. China
Email: {dhwang, liucl}@nlpr.ia.ac.cn

Abstract—**Real-time recognition of handwritten sentences enables fast text input but the dynamic nature of writing makes reliable text line segmentation difficult. This paper proposes a method for real-time dynamic text line segmentation of online Chinese handwriting. The core of the method is a statistical classifier for modeling the geometric relationship between an ongoing stroke and the previous text lines, to assign the stroke into a previous line or form a new line. The method can deal with delayed strokes and therefore enables robust real-time recognition. We evaluated the segmentation performance on a dataset of online Chinese handwriting by simulating the real-time writing and recognition process. The experimental results demonstrate the effectiveness and robustness of the proposed method.**

Keywords-real-time recognition; dynamic text line segmentation; stroke-line relationaship

I. INTRODUCTION

For handwriting-based text input, real-time sentence recognition is a better alternative to character recognition since sentence writing is more natural and enables faster and more accurate input by utilizing contexts. Handwritten sentence (character string) recognition is a difficult contextual classification problem involving character segmentation and recognition, and has been attacked by many researchers [1-6]. A feasible approach is the over-segmentation-based recognition fusing character recognition scores, linguistic context and geometric context [5,6].

While handwritten sentence recognition is more complicated and computationally intensive than character recognition, it can be accelerated by real-time recognition: the characters are segmented and recognized while they are being written. To meet this goal, we formerly proposed a real-time handwritten sentence recognition method by implementing over-segmentation-based recognition using a dynamically maintained candidate segmentation-recognition lattice [7]. Since the recognition of candidate characters consumes the majority of computing and is performed during writing, sentence recognition is obtained immediately after a long pen lift (probable end of sentence).

In sentence-based input, due to the limitation of writing area, a sentence or several sentences are often written in multiple lines. This makes text line segmentation difficult because the lines are short, the strokes are dynamically produced, and there are often delayed strokes, which are inserted into previous characters or even previous lines. Unlike previous text line segmentation methods that mostly group strokes into lines after all strokes are produced, the dynamic segmentation during writing can only utilizes the information of part of strokes.

Among the previous online document segmentation methods, some segment text lines using heuristics or simple features like horizontal projection [8,9] and off-stroke distances [10]. The methods based on optimizing line-fitting objectives [11-13] yield more reliable line partitioning. They usually take a hypothesis-and-test strategy to generate candidate line partitioning and seek for the optimal partitioning by heuristic search. To generate text line hypotheses, however, these methods require all the strokes have been written. In dynamic text line segmentation for real-time recognition, it is impossible and inappropriate to construct candidate segmentation hypotheses because the ongoing strokes dynamically change the document structure. For real-time recognition, line segmentation is performed on each stroke rather than on the whole page. So, optimization-based methods cannot be directly applied to real-time recognition.

In our previous system [7], we segmented ongoing strokes into lines using some simple heuristic rules. Despite its promise in our preliminary experiments, the simple line segmentation is not robust enough. Assignment of strokes into wrong lines will cause misrecognition of characters and even whole lines.

In this paper, we propose a robust method for dynamic text line segmentation in real-time recognition of Chinese handwritten sentences. We adopt a statistical classifier, support vector machine (SVM), to model the geometric relationship between the ongoing stroke and the existing text lines. By classification based on extracted features of a line-stroke pair, the classifier judges whether to assign the stroke to a previous line or it starts a new line. The method can deal with delayed strokes by grouping into previous lines, and therefore, it makes the real-time recognition system more stable. To evaluate the dynamic segmentation performance, we generated multi-line sentences data from an online Chinese handwriting dataset containing ink pages. The experimental results demonstrate the effectiveness and robustness of the proposed method.

II. System Overview

Fig. 1 illustrates the flowchart of the real-time recognition system. As introduced in [7], it consists of four main modules: real-time recognition module, sentence recognition module, sentence edition module and language association module. While the modules of real-time recognition and sentence recognition are the core of the system, the other two modules make the system more usable.

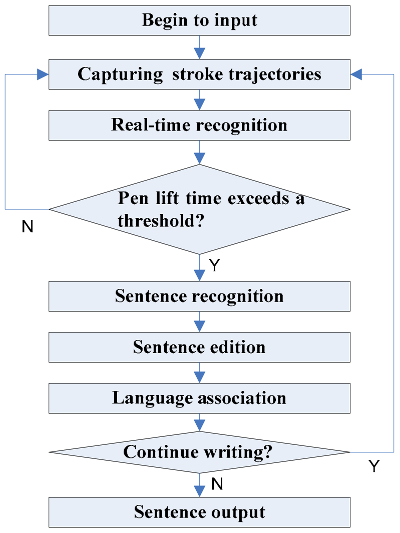


Figure 1. Flow chart of real-time sentence recognition.

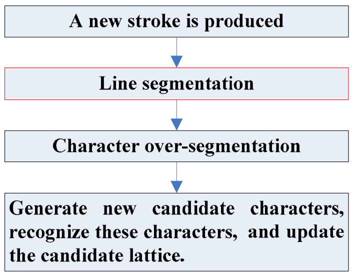

A new stroke is produced
Line segmentation
Character over-segmentation
Generate new candidate characters, recognize these characters, and update the candidate lattice.

Figure 2. Flow chart of the real-time recognition of a stroke

The real-time recognition module acts whenever an ongoing stroke is produced. Fig. 2 details the real-time recognition module. In line segmentation, the system judges which line the stroke belongs to. If the stroke belongs to one previous line, then the line is updated and over-segmentation is performed on the line. If no previous line is found to contain the stroke, the stroke is considered to start a new line and compose the first segment (a stroke block) of the line. The dynamic process of text line segmentation will be our focus in this paper, and will be described in Section III.

The method of character over-segmentation is similar to that in [7] except that for assigning strokes into segments, we later use an SVM classifier for decision making. If a stroke belongs to one previous segment of the line, the system updates the segment, otherwise creates a new segment using the stroke and finds the position of the new segment in the line according to the left boundaries.

After assigning the new stroke, updated segments or newly created segments are merged with previous segments to generate new candidate characters, which are recognized by a character classifier to assign candidate classes, as introduced in [7]. The new candidate characters and assigned classes are added to the candidate segmentation-recognition lattice. Fig. 3 shows an intermediate candidate lattice and its updated form due to a new stroke.

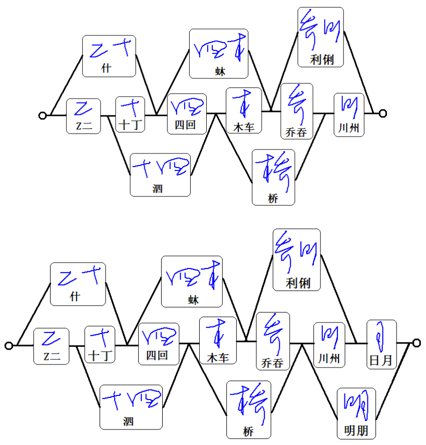


Figure 3. A candidate lattice (upper) and the updated one due to a new stroke (lower).

After real-time recognition on a new stroke, if the pen lift time exceeds a threshold (adjustable by the user, e.g., 0.5s), the result of sentence recognition is obtained by path search in the updated candidate lattice, performed by the sentence recognition module. The sentence edition module is designed to correct character segmentation errors and recognition errors by manual operations using the pen. The language association module is aimed to accelerate the writing process by automatically entering successive characters associated with the recognized partial sentence.

III. Line Segmentation

As this paper concentrates on text line segmentation, we will not present the character string recognition implemented in the system. For more details, please refer to [7]. In fact, any over-segmentation-based character string recognition system can be implemented in the real-time recognition system if using a dynamic candidate lattice. In the following, we focus on the method of dynamic text line segmentation.

A. Line Segmentation Algorithm

Algorithm 1 illustrates the process of real-time line segmentation and over-segmentation of an ongoing stroke, corresponding to those in Fig. 2. Denote the stroke by *strk*, and suppose that there are *m* previous lines, denoted as *lines*. In the algorithm, *lineIdx* is the index of the text line that the new stroke belongs to, and $lineIdx = -1$ indicates that the stroke starts a new line. The function **LineStrokeFeature**($strk, line_i$) extracts geometric features characterizing the relationship between the stroke and the *i*-th line. Based on the features, if the classifier judges that *strk* belongs to $line_i$, then update $line_i$ and perform over-segmentation on the updated $line_i$. Otherwise, the process continues until one line containing the stroke is found. If there is no line containing the stroke, the stroke will be considered to start a new line and form the first segment of the line.

Algorithm 1. Line segmentation on a new stroke

```
Input: Existing lines : lines
       Line number: m
       A new stroke: strk
Initialization: set $lineIdx = -1$
For $i=m$ to 1
    feature = LineStrokeFeature(strk, $line_i$),
    Classifier(feature),
    if strk belongs to $line_i$
        lineIdx=i;
        break;
    else
        continue;
End for.
If (lineIdx>0)
    Merge strk into the lineIdx-th line,
    OverSegmentation(updated lineIdx-th line)
Else
    Create a new line $line_{m+1}$ using strk,
    Create the first segment of the line using strk.
    m=m+1.
End if.
End.
```

B. *Training Sample Collection*

We adopt a statistical classifier to model the geometric relationship of a line-stroke pair, and to judge whether the stroke belongs to the line or not.

To collect training samples for the two-class classifier, we extract samples from a stroke and its temporally previous lines. If the stroke belongs to the line, the sample is considered to be a positive one, otherwise a negative one. Samples can be extracted from online documents containing multiple text lines by simulating the real-time writing process and extracting features from strokes each paired with both its genuine line (forming positive sample) and the previous lines (forming negative samples).

C. *Line-Stroke Feature Extraction*

We extract geometric features from a pair of line and stroke to characterize the relationship between them, which are input to a statistical classifier for decision making. We do not rely on temporal feature such as the off-stroke distance so as to cope with delayed strokes. Before feature extraction, a pair of a line l and a stroke s are tentatively merged and fitted by linear regression. Denote the merged line as l_t. The line height that will be used for feature normalization is estimated by computing the average height of strokes, as in [7]. We extract 22 features from a line-stroke pair, as listed in Table I. The features can be divided into four categories:

1) Five features related to the line l (No.1-5 in Table I).
2) Two features related to the stroke s (No.6-7).
3) Four scalar features related to the line l_t (No.8-11).
4) Eleven scalar features related to the geometric relationship between the stroke s and the line l as well as the line l_t (No. 12-22 in Table I).

Table I. Line-Stroke Geometric Features (the last column denotes whether normalized w.r.t. the text line height or not).

No.	Feature	Norm
1-2	Height and width of l	Y
3	The number of strokes in l	N
4	Average regression error of l : σ_1^2	Y
5	Horizontal direction of the line l	N
6	Height of s	Y
7	Aspect ratio of s	N
8-9	Height and width of l_t	Y
10	Average regression error of l_t : σ_2^2	Y
11	Horizontal direction of the line l_t	N
12	Growth of line height	Y
13	Change of horizontal direction	N
14	Change of average regression error	Y
15	Distance between l and s, as the minimum distance between s and the strokes in l	Y
16	Common area of l and s	Y
17-18	Distances of upper/lower bound of s to vertical center of l along the norm direction of l	Y
19-22	Distances between the upper bounds, lower bounds, upper-lower bounds, and lower-upper bounds of l and s	Y

IV. Experiments

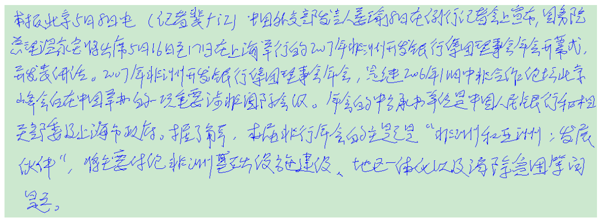
To evaluate the performance of the proposed method, we conducted experiments using a dataset of online handwritten Chinese texts: CASIA-OLHWDB2.1 (called DB2.1 for short in this paper) [14], which contains 1,500 pages in total, which are segmented into 17,282 text lines. The dataset is partitioned into a training set (1200 pages including 13,758 lines) and a test set (300 pages including 3,524 lines).

In sentence-based input, due to the limitation of writing area, users tend to write 3 to 5 text lines and each line contains only a few characters, say, not more than eight characters. To simulate this situation, we used the DB2.1 to generate pages with 3 to 5 text lines, each line consisting of six to eight characters. We then simulated the real-time writing process and performed dynamic text line segmentation on the generated data.

A. *Dataset Generation*

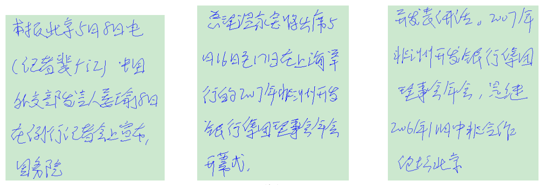
The text lines in DB2.1 contain about 30 characters in each line. So we split each line into multiple lines as one generated page by making the width of each line not larger than five times of the average height of the original lines. Fig.

4 shows an example of data generation: (a) is a text page, (b) shows three new pages derived from the first three lines in (a). Table II provides the details of database DB2.1 and the generated database (called GDB2.1 for short in this paper).



本报北京5月8日电（记者裴广江）中国外交部发言人姜瑜8日在例行记者会上宣布，国务院
总理温家宝将出席5月16日至17日在上海举行的2007年非洲开发银行集团理事会年会开幕式，
并发表讲话。2007年非洲开发银行集团理事会年会，是继2006年11月中非合作论坛北京
峰会后在中国举办的另一项重要涉非国际会议。年会的中方承办单位是中国人民银行和相
关部委及上海市政府。据了解，本届非行年会的主题是“非洲和亚洲：发展
伙伴”，将重点讨论非洲基础设施建设、地区一体化以及治理等问
题。

(a)



本报北京5月8日电
（记者裴广江）中国
外交部发言人姜瑜8日
在例行记者会上宣布，
国务院

总理温家宝将出席5
月16日至17日在上海举
行的2007年非洲开发
银行集团理事会年会
开幕式，

并发表讲话。2007年
非洲开发银行集团
理事会年会，是继
2006年11月中非合作
论坛北京

(b)

Figure 4. (a) A text page; (b) Three generated pages from the first three lines.

Table II. Details of DB2.1 and generated GDB2.1.

Databases		#page	#line	#line/page
DB2.1	train	1200	13,758	11.46
	test	300	3,524	11.74
GDB2.1	train	13,758	52,316	3.80
	test	3,524	13,497	3.83

B. *Performance Metrics*

To evaluate the performance of real-time line segmentation, we simulated the real-time process. Given an online page, we input strokes in writing order, and whenever a stroke is input, the system performs line segmentation and updates text lines. After the last stroke is processed, the result of line segmentation is obtained.

Many metrics have been defined for evaluating performance of line segmentation [13, 15-16]. We adopt some of them and define a new metric for performance of real-time line segmentation.

To define metrics, matches are predefined. A **one-to-one match** is a match where a detected line and a ground-truthed line contain identical strokes. And **g_one-to-many match** occurs when the union of two or more result lines equal to a ground-truthed line. Similarly, a **d_many-to-one match** means the union of two or more ground-thuthed lines equals a detected line.

Among the performance metrics presented in [13], we chose detection rate **(DR)**, recognition accuracy **(RA)** and entity detection metric **(EDM)** that are defined as follows:

$$DR = w_1 \frac{one2one}{N} + w_2 \frac{g\_one2many}{N},$$

$$RA = w_3 \frac{one2one}{M} + w_4 \frac{d\_many2one}{M},$$

$$EDM = \frac{2 \cdot DR \cdot RA}{DR + RA},$$

where N is the number of ground-truthed lines, M is the number of detected lines, and $w_1 \sim w_4$ are all set to 1. DR, RA and EDM are similar to recall, precision and F-rate, respectively. Page recognition rate (**PRR**), defined as the percentage of pages with no segmentation error, is used to measure the page level performance.

C. *Experimental Results*

We used a linear SVM classifier to model the geometric relationship of a line-stroke pair because it performs efficiently. The two-class SVM is trained on the training set of GDB2.1, and line segmentation performance is evaluated on the test set of GDB2.1. We compare the proposed method with other methods that apply heuristic rules. Experimental results on GDB2.1 are listed in Table II. The off-stroke distance method is to segment a line when off-stroke distance is larger than a threshold (3**lineHeight* in our experiments). Another rule considered is the vertical overlap degree, and the overlap degree is defined as $0.5 * (\frac{overlap}{hei1} + \frac{overlap}{hei2}) - \frac{dis}{span}$, similar to that in [17]. The threshold for overlap degree is set as 0.4 empirically. The off-over method is to relax thresholds for both the off-stroke method and the overlap method and then combine them to make decisions.

In fact, the pages in GDB2.1 have very few delayed strokes. In this case, Table III shows that the off-stroke method performs fairly well and is comparable to the proposed classifier-based method. To evaluate the performance on pages with delayed strokes, we produce delayed strokes for GDB2.1 by changing the writing order of a stroke in each page. Specifically, we randomly chose a stroke and place it randomly after its original position, keeping coordinates unchanged. Experimental results on the newly generated GDB2.1 with delayed strokes are listed in Table IV.

Table III. Performance on GDB2.1 WITHOUT delayed strokes.

Methods	DR	RA	EDM	PRR
Off-stroke	0.9225	0.9640	0.9428	0.9225
Overlap	0.6453	0.4903	0.5572	0.2415
Off-over	0.9084	0.9456	0.9266	0.6853
SVM	**0.9272**	0.9644	0.9455	0.8918

Table IV Performance on GDB2.1 WITH delayed strokes.

Methods	DR	RA	EDM	PRR
Off-stroke	0.7492	0.7256	0.7372	0.4492
Overlap	0.2455	0.0971	0.1392	0.0235
Off-over	0.9074	0.9401	0.9235	0.6960
SVM	**0.9276**	0.9616	0.9443	0.8995

We can see that the performance of the off-stroke method deteriorates significantly while the proposed method performs stably with delayed strokes. The performance of the off-over method remains good because the off-stroke constraint is relaxed to tolerate delayed strokes. On both datasets, the inferior performance of the overlap method can be attributed to the instability of overlap degree calculation.

Fig. 5 shows some examples of line segmentation by different methods. Fig. 6 shows some examples of incorrect segmentation by the proposed method, where the first two errors result from the inaccurate estimation of line height in the beginning of writing, and the third one maybe due to the special shape of the dot stroke.

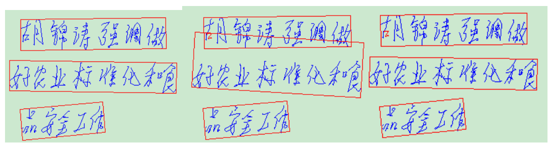

(a) (b) (c)

Figure 5. (a) A page without delayed strokes is correctly segmented by the off-stroke method; (b) a delayed stroke deteriorates the off-stroke method; (c) the page with delayed stroke is correctly segmented by the proposed method.

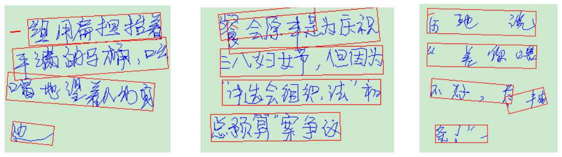


Figure 6. Examples of incorrect segmentation by the proposed method.

We also compared the performances of line segmentation using different classifiers: SVM, linear discriminant function (LDF) and single-layer neural network (SLNN). The results in Table V show that the SVM and SLNN perform comparably well.

Table V. Performance of different classifiers on data WITH delayed strokes.

Methods	DR	RA	EDM	PER
SVM	**0.9276**	0.9616	0.9443	0.8995
SLNN	0.9262	0.9624	0.9440	0.8873
LDF	0.5860	0.6196	0.6023	0.4580

V. Conclusion

To enable users write multiple lines of texts and delayed strokes in sentence-based input system, we proposed a robust dynamic text line segmentation method for real-time recognition of Chinese handwritten sentences. By using a statistical classifier (SVM) on geometric features characterizing the relationship between an ongoing stroke and the previous text lines, the ongoing stroke can be reliably classified into a previous line or decided to form a new line, even when there are delayed strokes. The dynamic text line segmentation algorithm has been demonstrated effective in experiments, and will be integrated into the real-time sentence recognition system.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (NSFC) under grants no.60825301 and no.60933010.

References

[1] H. Murase, Online recognition of free-format Japanese handwritings, *Proc. 9th ICPR*, 1988, Vol.2, pp.1143-1147.

[2] M. Nakagawa, B. Zhu, and M. Onuma, A model of on-line handwritten Japanese text recognition free from line direction and writing format constraints, *IEICE Trans. Information and Systems*, vol.E88-D, no.8, pp.1815-1822, 2005.

[3] X.-D. Zhou, J.-L. Yu, C.-L. Liu, T. Nagasaki, and K. Marukawa, Online handwritten Japanese character string recognition incorporating geometric context, *Proc. 10th ICDAR*, Curitiba, Brazil, 2007, pp. 48–52.

[4] B. Zhu, X.-D. Zhou, C.-L. Liu, and M. Nakagawa, A robust model for on-line handwritten Japanese text recognition, *Document Recognition and Retrieval XVI (DRR)*, San Jose, USA, 2009, pp. 1–10.

[5] X.-D. Zhou, C.-L. Liu, and M. Nakagawa, Online handwritten Japanese character string recognition using conditional random fields, *Proc. 11th ICDAR*, Barcelona, Spain, 2009, pp. 521–525.

[6] M. Cheriet, N. Kharma, C.-L. Liu, and C.Y. Suen, *Character Recognition Systems: A Guide for Students and Practitioners*, John Wiley & Sons, 2007.

[7] D.-H. Wang and C.-L. Liu, An approach to real-time recognition of Chinese handwritten sentences, *Proc. 2th Chine-Japan-Korea Joint Workshop on Pattern Recognition, (CJKPR)*, Fukuoka, Japan, 2010, pp. 203–208.

[8] A. K. Jain, A. M. Namboodiri, and J. Subrahmonia, Structure in on-line documents, *Proc. 6th ICDAR*, Seattle, WA, 2001, pp. 844-848.

[9] E. H. Ratzlaff, Inter-line distance estimation and text line extraction for unconstrained online handwriting, *Proc. 7th IWFHR*, Nijmegen, Netherlands, 2000, pp. 33-42.

[10] M. Nakagawa and M. Onuma, On-line handwritten Japanese text recognition free from constrains on line direction and character orientation, *Proc. 7th ICDAR*, Edinburgh, Scotland, 2003, pp. 519-523.

[11] M. Liwicki, E. Indermuhle, and H. Bunke, On-line handwritten text line detection using dynamic programming, *Proc. 9th ICDAR*, Curitiba, Brazil, 2007, pp. 447-451.

[12] M. Ye, P. Viola, S. Raghupathy, H. Sutanto, and C. Li, Learning to group text lines and regions in freeform handwritten notes, *Proc. 9th ICDAR*, Curitiba, Brazil, 2007, pp. 28-32.

[13] X.-D. Zhou, D.-H. Wang, and C.-L. Liu, A robust approach to text line grouping in online handwritten Japanese documents. *Pattern Recognition*, vol.42, No.9, pp. 2077-2088, 2009.

[14] C.-L. Liu, F. Yin, D.-H. Wang, and Q.-F. Wang, CASIA online and offline Chinese handwriting databases, *Proc. 11th ICDAR*, Beijing, China, 2011.

[15] I. Phillips and A. Chhabra, Empirical performance evaluation of graphics recognition systems, *IEEE Trans. PAMI*, Vol. 21, No. 9, pp. 849-870, Sep. 1999.

[16] A. Antonacopoulos, B. Gatos, and D. Bridson, ICDAR2007 page segmentation competition, *Proc. 9th ICDAR*, Curitiba, Brazil, 2007, pp. 1279-1283.

[17] C.-L. Liu, M. Koga, and H. Fujisawa, Lexicon-driven segmentation and recognition of handwritten character strings for Japanese address reading. *IEEE Trans. PAMI*, Vol.24, No.11, pp. 1425-1437, 2002.