Recovering Dynamic Information from Static Handwritten Images

Yu Qiao and Makato Yasuhara
Graduate School of Information Systems, University of Electro-Communications,
1-5-1 Chofugaoka, Chofu, Tokyo, 182-8585, Japan
HT{qiaoyu,yas}@math-sys.is.uec.ac.jp

Abstract

This paper proposes an efficient method for recovering dynamic information from offline single-stroke hand drawing images. This method makes use of both the local analysis and global smoothness calculation. At first, a graph model is built from the skeleton. Then, odd degree nodes are resolved in a probability framework to detect the double-traced/terminal segments, and even degree nodes are analyzed by the Node Traversing Rule (NTR). We estimate the probability of two strokes being contiguous pair by PCA based angle calculation. Then, double-traced lines are identified. Finally, we calculate the smoothness for each of the possible paths by SLALOM approximation and select the smoothest one. Experiments show that our method works successfully on cursive hand drawing images.

1. Introduction

Handwriting analysis and recognition have received extensive attention in both the academic and production fields. Compared with offline handwriting recognition, online recognition makes use of dynamic information of a pen-tip movement and has shown better recognition performance [1]. The object of our research is to recover the dynamic handwriting information from offline static images. This can be seen to convert two-dimensional image to a sequence of one-dimensional vectors of pen-tip positions along time axis.

Researches in this field can be divided into two categories: (1) Skeleton based methods, which used thinning or medial axis transform algorithms to obtain the skeleton and then traced or searched a written order on the skeleton representation. Lee and Pan [2] traced the skeleton of offline signature by a set of heuristic rules. V. Govindaraju and N. Sriharo [7] presented an approach of separating handwritten text from interfering strokes based on Gestalt's segmentation and grouping principles. Liu, Huang and Suen [3] proposed a stroke segmentation method for Chinese characters by using polygonal approximation and certain rules. Both Jäger [4] and Huang, Yasuhara [5] used graph model to represent the skeleton and searched a Hamiltonian or an Euler path, which minimized certain smoothness cost functions. Kato and Yasuhara [6] avoided the combinatorial explosion of direct graph searching method and presented a graph-based approach by following 2-phase analysis to obtain labeling information. (2) Raster image based method without thinning. Rosenfeld [9] described a stroke recovery platform based on local regional and temporal clues. Plamondon and Privitera [10] developed a scanning method to find the natural course of strokes by calculating the curvature of contour.

2. Problem Description and Overview

In this paper, we proposed a hybrid approach to recover the natural drawing path (dynamic information) from static single-stroke handwritten images. We assume that each stroke is traced at most twice by pen-tip and there is at most one terminal or double-traced stroke connected to a node.

Our method has the advantages from both the efficiency of local node analysis and the robustness of global smoothness calculation. At first, we built the graph representation from the selection, which is obtained by using thinning algorithms. Then the problem is transformed to find a path, which covers all the edges in graph. Because exhaustive calculations of all the possible paths need large computation, we used node analysis algorithm to find the contiguous relations of whether segments connected to a node belong to the same stroke or not. The terminal and double-traced segments of odd nodes are detected by probability calculation and the contiguous relations of even nodes are obtained by using Node Traversing Rule (NTR). Finally, we calculate the global smoothness for each of the possible paths found by the SLALOM method and select the smoothest one.

3. Build Graph from Skeletons

The objective of this section is to transform the skeleton of input image into a geometrical graph representation G. Before applying thinning algorithm to obtain the skeleton, we use the smoothing method proposed in [13] to reduce the peaks and holes in the input image. We build graph G mainly based on the skeleton. This is because: the skeleton preserves the significant feature of the original pattern and permits a simpler structural analysis [11].

3.1 Extraction of Vertexes and Segments

To construct graph G, we need to extract vertexes set V and segments (edges) set E from the skeleton. A segment represents a part of the stroke in skeleton; while a vertex corresponds to a local geometrical configuration, where a segment terminates (terminal vertex) or multi segments joint (junction vertex). In the skeleton with 1 pixel width, the terminal vertex can be easily detected as it is a black pixel with only one 8-conneted neighbor; while the jointing vertex may be a pixel or a cluster of pixels; we call pixels belonging to a vertex as feature pixels. The configurations of feature pixels depend on the distortions caused by thinning procedure and on the local structure where segments meet. By following [8], we defined the set of feature pixels as:

$$S = \{P \mid N_b(P) = 1 \quad or \quad N_b(P) \geq 4 \quad or \quad N_c(P) \geq 3\}$$

Here, $N_c(P) = \sum_{i=0}^{7} |P_i|$ is the number of black pixels in its 8 neighbors and $N_c(P)$ is the Rutoviz crossing number which can be calculated by:

$N_c(P) = \frac{1}{2}\sum_{i=0}^{8} |P_{i+1} - P_i|$, where P_i i=(0,1,…,8) are the 8 adjacent neighbors of P and P_8=P_0.

After extracting feature pixels, we cluster adjacent feature pixels into vertexes, and then the non-feature pixels left are connected into line segments respectively.

3.2 Identification of Spurious Segments

The skeleton resulted from thinning process may include unwanted spurious outputs. The thinning procedures iteratively delete unnecessary contour pixels without altering the topology until 1-pixel width skeleton remains. The deletion or retention of a black pixel depends on the local configuration. Thinning procedures work well on the stroke area with good contours. But in area with noise or of the strokes' jointing, where contours are eroded or overlapped, the skeleton may be distorted and include artifact [11]. Hence the segments detected above can be classified into two types: real segment (r-segment) and spurious segment (s-segment). An r-segment corresponds to a part of real stroke; and s-segments are resulted from the thinning process and never exist in an original handwritten image. They are usually in the stroke jointing area. These undesirable s-segments will distort the structure of the original pattern. Therefore, it is necessary to differentiate these s-segments from the r-segments. We use a double threshold method. Details are as follows:

1) Estimate average stroke width w by w=2S/L, where S is the area of strokes (the number of stroke pixels) and L is the total length contour of input image.

2) Set two thresholds l_{th1}= k_1×w and l_{th2}= k_2×w, here l_{th1} is large enough so that the segment longer than l_{th1} must be r-segments; while l_{th2} is small enough so that the segment shorter than l_{th2} must be s-segments. In experiments, we chose k_1 = 4 and k_2 = 1.5.

3) For the segment with length between l_{th1} and l_{th2}, we examine it in the original image. We calculate the shortest distance $Dis(p_i, C)$ to the contour C for each pixel p_i in the segment. If $\sum_i Dis(p_i, C)/N < 0.65w$ and max{$Dis(p_i, C)$} <w, this segment should be a r-segment; otherwise, we label this segment as a spurious one.

After all the s-segments are identified, we can cluster connected s-segments with associated vertexes into a node. In this paper, we use vertex for skeleton and node for graph G. The terminal vertexes and individual crossing vertexes in the skeleton are transformed into nodes in G too; real segments are preserved as edges of G.

4. Node Analysis

By constructing graph G, the recovery problem of single stroke image can be reduced to a Traveling Salesman Problem (TSP), which tries to find the minimum cost path in G [4]. It is well known that TSP is NP hard and exhaustive search of all the possible paths may lead to huge computations in the final global optimization step. In order to reduce the computational complexity, it is necessary to analyze the contiguous relations among the segments connected to the same node. The contiguous relations can be represented by the information of whether or not two segments connected to a node belong to the same stroke; if so, we call that these two segments form a contiguous pair.

4.1 Estimation of the Probability of Two Segments Being a Contiguous Pair

Human normally writes characters in the smooth way, as it costs usually the least energy. Curvature, which measures the degree of the curve bending at certain position, has been proved effective for estimating the smoothness of the curve. Mathematically, curvature of a two-dimensional curve is defined as: $k(x, y) = \partial\varphi / \partial s$, where φ is the tangent angle and s denotes length along the line of travel.

In our problem, the objective is to determine the good continuity of the two segments, not the local curvature itself at certain point of the curve. These two segments are separated by the ambiguous node area (Fig. 1), and the skeletons near and inside nodes (s-segments) may be distorted and not reliable when estimating curvature.

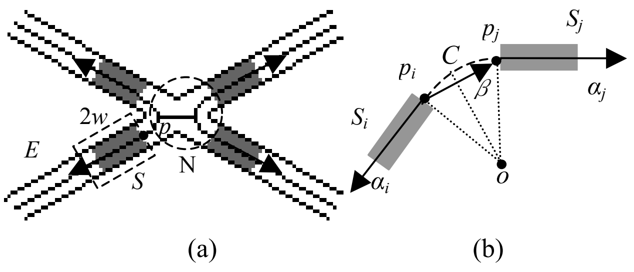

Figure 1. Examine contiguous pair segments

Suppose we have two segments E_i and E_j jointed at a node as shown in Fig. 1. p_i and p_j represent the two pixels of E_i and E_j connected to the node area; angle α_i and α_j denote the tangent directions at p_i and p_j of E_i and E_j, respectively. S_i and S_j denote a set of pixels near the node in the original stroke area, associated with E_i and E_j respectively. Because of the sensitivities of thinning algorithm to noise, we estimate angle α from the stroke part S near p (the gray regions in Fig. 1 (a)). Details are as follows:

At first, we trace L from p with length $2w$ along E; then we obtain the associated stroke part S of L in the original image by the following method: for each black pixel p_k in the original image, we define its nearest pixel $N_m(P_k)$ in skeleton as:

$$N_m(p_k) = \arg\min_{p_s}\{Dis(p_k, p_s) \mid p_s \in Skeleton\},$$

where $Dis(p_k, p_s)$ is the Euclidean distance between two pixels p_k and p_s. So the associated stroke area S can be represented by:

$$S = \{p_k \mid N_m(p_k) \in L\}.$$

Then we apply Principal Component Analysis (PCA) to the coordinates of the black pixels in S. It is well known that the first principal vector of PCA corresponds to the maximum-variance direction and can be the best linear summary of these pixels [12]. Hence angle α can be obtained as the direction of the first vector. Then we calculate the tangent direction: α_i and α_j respectively. β represents the direction angle from p_i to p_j.

The curvature we wanted to estimate is not a local curvature on a point but the connecting curvature whether or not these two segments are drawn contiguously by the same stroke. In this paper, we simply used angle difference to represent the curvature k:

$$k = |\pi - |\alpha_i - \beta|| + |\alpha_j - \beta|$$

The larger the curvature k is, the less likely E_i and E_j are contiguous pair. We define the probability of E_i and E_j being a contiguous pair as: $P_{ctg}(E_i, E_j) = e^{-mk}$. It is easy to see that $P_{ctg}(E_i, E_j) = P_{ctg}(E_j, E_i)$. We use MLE (Maximum likelihood estimation) algorithm to estimate m optimally based on a large number of curvature samples including both contiguous and dis-contiguous pairs of segments. The advantages of using probabilities are: 1) There is no need to apply any threshold in order to differentiate a contiguous pair from a dis-contiguous one. 2) By preserving the top multiple cases from the highest probability, it is possible to preserve more than one candidate for the global smoothness search.

4.2 Classification of Nodes and Segments

There are three possible connection relations for a segment E connected to node N:

1) Terminal type (Fig. 2 (a,b,e)): E terminates or comes and returns back at N; in other words, among all the other segments connected to N, there is no contiguous segment of E and we call this segment as a T-leg segment of N. If the degree of N is 3, the other two segments are regarded as T-hand segments (Fig. 2 (a,e)); if E is the only segment connected to N, we call N an F-node (Fig. 2 (b)) ;

2) Pair type (Fig. 2 (a, c, e)): E has one and only one contiguous pair segments at N.

3) Double-traced type (Fig. 2 (d, e, f)): E has two contiguous pair segments at N; hence E must be double-traced. N is a Y-node. If N is of degree 3, E is Y-leg segment and two other segments are Y-hand segments.

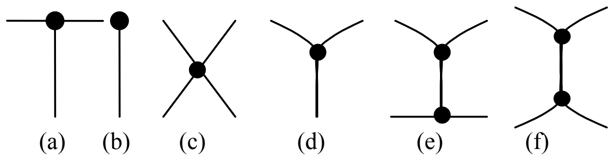

Figure 2. Examples of node types

4.3 Identification of Leg Segment connected to Odd Degree Node

Among the segments connected to an odd degree node, there must exist a segment, which is either double-traced or terminates. Following the definition in 4.2, this segment is a T/Y-leg segment. It is necessary to find out this leg segment before examining the contiguous relations among others.

At first, we consider the simplest and most popular case: the nodes of degree 3. Suppose that three segments connected to node N are E_1, E_2, E_3. If E_1 is a terminal segment, then E_2 and E_3 must be contiguous pair and the probability of this case can be calculated by:

$$P_{1T} = P_{ctg}(E_2, E_3)(1 - P_{ctg}(E_2, E_1))(1 - P_{ctg}(E_3, E_1)).$$

If E_1 is a double-traced segment, the probability can be calculated by:

$$P_{1Y} = (1 - P_{ctg}(E_2, E_3))P_{ctg}(E_2, E_1)P_{ctg}(E_3, E_1).$$

By flowing the same way, we can obtain the probability of E_2 being terminal segment (P_{2T}) or double-traced segment (P_{2Y}) and E_3 being terminal segment (P_{3T}) or double-traced segment (P_{3Y}). For all the 6 cases of connection relations, we take one case with the highest probability or the top two cases if their probabilities are near:

$$\max\{P_{1T}, P_{1Y}, P_{2T}, P_{2Y}, P_{3T}, P_{3Y}\}.$$

For an odd node connected by $2k$+1 segments: E_1, E_2,..,E_{2k+1}, generally the problem of detecting leg segment would be very complex if we take all the possible connection relations into consideration. For simplification, at first we define the Y/T probability for each segment being Y/T leg and then examine the leg segment from this Y/T probability. For a segment E_i, if it is a double-traced segment, the two associated hand segments can be detected by:

$$\{E_{il}^Y, E_{ir}^Y\} = \underset{(E_{k_1}, E_{k_2})}{\arg\max}\{P_{ctg}(E_i, E_{k_1})P_{ctg}(E_i, E_{k_2})(1 - P_{ctg}(E_{k_1}, E_{k_2}))\}\cdot$$

Then for each segment E_i, we define its Y probability as: $P_{iY} = P_{ctg}(E_i, E_{il}^Y)P_{ctg}(E_i, E_{ir}^Y)(1 - P_{ctg}(E_{il}^Y, E_{ir}^Y))$.

If E is a T-leg segment, we define its T probability as: $P_{iT} = (1 - P_{ctg}(E_{iT}, E_i))$

$$E_{iT} = \underset{E_k}{\arg\max}\{P_{ctg}(E_k, E_i) \mid i \neq k, k \in [1, 2k+1]\}\cdot$$

Next, we find the best candidate Y-leg as E_m:

$$m = \arg\max_i\{P_{iY} \mid i = 1, 2, ..., 2k+1\}$$

and the best candidate T-leg probability as E_n:

$$n = \arg\max_i\{P_{iT} \mid i = 1, 2, ..., 2k+1\}.$$

It is easy to see that it is meaningless to compare P_{mY} and P_{nT} directly. So we keep both T and Y candidate types, and resolve them by global analysis at last. If we take off all leg and hand segments, the number of segments left must be even and can be resolved by the method described in the 4.4.

4.4 Node Traversal Rule For Even Degree Node

For an even node with degree $2k$, which is created where k lines joint, we need to find the contiguous relations among these $2k$ segments: $E_1, E_2, .., E_{2k}$. The connection relations can be represented by k contiguous pairs denoted as $\{E_{11}, E_{12}\}$, $\{E_{21}, E_{22}\}$,.., $\{E_{k1}, E_{k2}\}$. It is easy to see that the number of possible connection relations are M=$2k(2k$-2) (2k-4)..2. If we use probability to obtain the connection relations, there will be two problems: 1) If k is big, the number M will be very large; 2) It is not reliable to multiply a large number of probability values as noise can be enlarged during multiplication. For this reason, we will provide the Node Traversing Rule, which efficiently solves this problem.

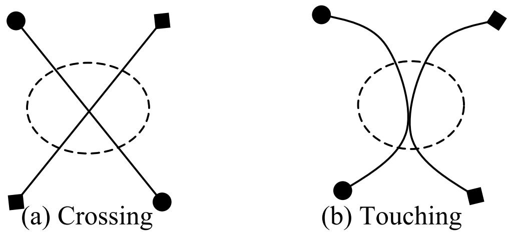

Figure 3 Configuration of two lines jointing (Segments with same end shape belong to the same stroke)

Before the detailed discussion, we classify a pair of lines, which traverse through a node, into two types: 1) crossing, each line passes through the other (Fig. 3 a); 2) touching, the two lines joint without crossing each other (Fig. 3 b). Kato et al. [6] introduced the Basic Tracing Algorithm (BTA) to selecting the middle path as contiguous pair segment in a node of degree 4. And the following Crossing Node Traversing Rule (CNTR) can be seen as a generalization of BTA.

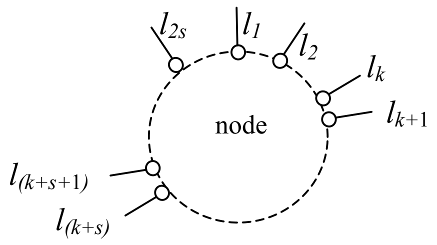

Figure 4 Crossing Node Traversing Rule

For a 2s-degree node through which s lines traverse, each two lines cross exactly once inside it. We label these $2s$ segments clockwise as shown in Fig. 4. For these crossing type nodes, we have the following Crossing Node Traversing Rule:

Take arbitrary one of the 2s r-segments as an incoming one to a node, the corresponding outgoing one from the node is determined uniquely as the s-th segment from the incoming one in counting either clockwise or counter-clockwise. In other words, segments $E_{(k+s)\%2s}$ and E_k form a contiguous pair.

For a node satisfying CNTR, we can prove the following theorem.

Theorem 1: *The node holds CNTR if and only if all the s lines traversing through it cross exactly once each other.*

CNTR stems essentially from the good continuity criterion that human normally write in the smoothest way.

In CNTR, to find the tracing relations, we need only to label the index of these segments clockwise. The next question is whether CNTR is true for all nodes or not. In the ideal situation: if s straight lines joint in a node, then two lines must cross each other once and only once and CNTR is true. In the real situations, however, it is difficult to know whether or not the lines are perfectly straight; in other words, it is possible that two lines touch inside the node. Thus CNTR should be extended for not only crossing junction but also touching junction. Thus we introduce the Node Traversing Rule (NTR) in more general sense:

Among all the s lines traversing a node, there is at most one touching pair of lines.

Under NTR, there are only s+1 possible cases (s touching cases and one crossing case) of connection relations. It is easy to see that this number is much less than M. And we can calculate the probability for each of the s+1 cases and keep acceptable ones as candidates.

5. Merge/Separate Operator

For both odd and even nodes, there may be more than one interpretation of the segments' connection relations for each of the nodes. Before using global calculation to select the smoothest path, it is necessary to decide the possible orders of these segments in G for all combinations of candidate relations. In spite of using tracing methods like many approaches before [2], [6], we employ the merge/separate operators: merge operators iteratively combine the two pair-segments into one segment, and separate operator separate terminal segment from a node.

For a T-type segment E_t at node N, we use a separator operator: add a new node M to graph G and replace E_t's end node N with new node M. For pair segments E_i, E_j at N, we combine these two segments into a new segment and replace E_i and E_j with this new segment in G (merge operator). There is a problem: node N, which connects E_i with E_j, may be a sub-graph. For this kind of node, we find the shortest path between V_i (end vertex of E_i) and V_j(end vertex of E_j) inside N by the Dijskra algorithm. After applying merge/separate operators on all T/P type segments, there are only Y-nodes and F-nodes left in G.

Double tracing is common in human writing. A double-traced line (D-line) may exist 1) between two Y-nodes (Y-Y pair, Fig. 2 f), or 2) between a Y-node and an F/T-node, where the segment arrives at a F/T-node and then return (Y-T pair, Fig. 2 e). A D-line may be divided into multiple segments by other strokes. Hence we use leg trace algorithm to find the whole trace line, which starts from the leg-segment of a Y-node and traces along the smooth path, until it reaches to another Y-node through leg segment, or reaches to F-node. Then segments along the path between start and end node are merged into one segment. If this segment is between Y-T pair, we clone this segment in graph G, and then merge two hand-segments of Y-node, this segment and its clone into one segment; else if this double segment is between Y-Y pair, there are two possible connection relations among all the four hands of two Y-nodes. (For example, assume the two Y-nodes are N_{Y1} and N_{Y2}, the double-traced line is L, the two hand segments connected to N_{Y1} are H_{L1}, H_{R1}, and hands with N_{Y2} are H_{L2}, H_{R2}; the two possible connection relations are: 1.{(H_{L1}, L, H_{L2}), (H_{R1}, L, H_{R2})}; 2{(H_{L1}, L, H_{R2}), (H_{R1}, L, H_{L2})}). After all of the Y-nodes are resolved by this processing, the order of the segments will be clear. The start node is selected as the most left-top one.

6. Calculate Global Smoothness by SLALOM

In section 4 and 5, we keep all possible contiguous relations as candidate cases. This may result in multiple tracing paths. To select the best one among all the single stroke paths, we used SLALOM approximation to estimate the global smoothness. The reason of using SLALOM is that skeleton resulted from the thinning process include too much noise. If we calculate the smoothness from skeleton itself, noise may influence the results too much. SLALOM was developed originally for inverse-quantization of digital signals; Huang and Yasuhara [5] used it first to calculate the smoothness of hand drawing curves.

For a curve denoted by $f(u)$, we introduce another curve $g(u)$ with a monotonously increasing index u. We obtain $g(u)$ by minimizing the sum of smoothness and the distance from $g(u)$ to $f(u)$ (error) along the 2D curve:

$$J(g) = \int \left(\frac{d^2}{dx^2} g(u)\right)^2 du + \alpha \int (g(u) - f(u))^2 du$$

where α is a coefficient.

For realization, we rewrite function $J(g)$ into two functions $J_x(g_x)$ and $J_y(g_y)$ for x and y coordinates respectively:

$$J_x(g_x) = \sum_i (g_x(i-1) - 2g_x(i) + g_x(i+1))^2 + \alpha \sum_i (g_x(i) - f_x(i))^2$$

$$J_y(g_y) = \sum_i (g_y(i-1) - 2g_y(i) + g_y(i+1))^2 + \alpha \sum_i (g_y(i) - f_y(i))^2$$

Then, we calculate the smooth functions $g_x(i)$ and $g_y(i)$ that minimize $J_x[g_x]$ and $J_y[g_y]$, respectively. This can be accomplished by solving linear equations. Based on the minimum value J^*_x and J^*_y obtained above, we define the global smoothness of a stroke path as: S=-(J_x+ J_y) and use S as the good continuity criterion to evaluate the single stroke paths obtained and choose the smoothest one. SLALOM smoothness calculation is the most time consuming computation in our system. To improve processing speed, we do sampling on pixels along the path and thus reduce the number of pixels solved by SLALOM.

7. Experiments

To examine the utility, we applied our method on a set of hand drawn scripts, which included words, letters and some artificial shapes. The experimental images include most possible combinations of touching/crossing nodes and double-traced lines. Some examples are show in Fig. 6. The written order is shown by arrows in the image. We can see that this method provides correct results for the single double-traced segments (Fig. 6 (a, b)), the multiple double-traced segments (Fig. 6 (d)), the odd nodes combination (Fig. 6 (c)), the 6-degree node (Fig. 6 (b)), the terminal segments (Fig. 6 (b)).

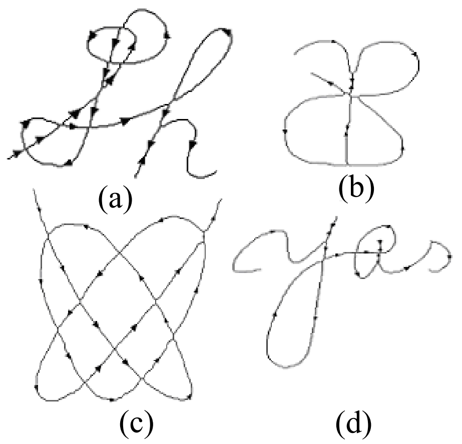

Figure 6 Experiment Results

8. Conclusions

In this paper, we propose a novel method for recovering the written order from static single stroke handwritten images. This method consists of two steps: 1) Local step: for odd nodes, we examine the double-traced /terminal segment in a probability framework. To estimate the probability of contiguous pair, a PCA based angle method was proposed. And for even nodes, we employ the Node Traversing Rule to identify the segments' contiguous relations. 2) Global step: we use merge/separate operators to recover the stroke order and finally apply SLALOM approximation to estimate the global smoothness for each possible path and select the smoothest one.

When compared with the results in [2], [3], [6], [7], our method is more efficient to deal with even nodes of degree more 4 by introducing NTR. Moreover, the identification of double-traced/terminal segment based on probability is more robust than that based on threshold [2], [3] and tracing method used in [6] where only nodes of degree 3 can be resolved. And when compared with the graph search method in [4], [5], our method can achieve the smoothness optimization with less computational cost. For further study, we consider the following questions: 1) Extend this method to multiple-stroke scripts; 2) Estimate the contiguous probability without angle calculation as far as possible, in order to be more insensitive to noise.

References

[1] R.Plamondon and S.N.Srihari, "On-Line and Off-line Handwriting Recognition: A Comprehensive Survey," *IEEE Trans. on PAMI*, Vol.22, No. 1, pp.63-84, 2000
[2] S. Lee and J.C. Pan, "Offline Tracing and Representation of Signatures," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 22, no. 4, pp. 755-771, 1992
[3] K Liu, Y S Huang, and C Y Suen, "Robust Stroke Segmentation Method for Handwritten Chinese Character Recognition ," *ICDAR*, vol., pp. 211-215,1997
[4] S. Jäger, "Recovering Writing Traces in Off-Line Handwriting Recognition: Using a Global Optimization technique," *ICPR*, pp. 150-154, 1996.
[5] T. Huang and M. Yasuhara, "A Total Stroke SLALOM Method for Searching for the Optimal Drawing Order of Off-Line Handwriting," *Proc. IEEE Systems, Man and Cybernetics Soc.*, pp. 2789-2794, 1995.
[6] Y. Kato and M. Yasuhara, "Recovery of Drawing Order from Single-Stroke Handwriting Images," *IEEE Trans. on PAMI*, vol. 22, no. 9, pp. 938-949, 2000.
[7] V. Govindaraju and S. Srihari, Separating Handwritten Text from Non-Textual Interference", *From Pixels to Features: II, J. Simon & S. Impedovo (editors), Elsevier Science Publishers,North-Holland*, pp. 17-28, 1992.
[8] Ke Liu, Yea S. Huang, Ching Y. Suen, "Identification of Fork Points on the Skeletons of Handwritten Chinese Characters, " *IEEE Trans. on PAMI*, vol. 21, no. 10, pp. 1095-1100, 1999
[9] D.S. Doermann and A. Rosenfeld, "Recovery of Temporal Information from Static Images of Handwriting," *Int'l J. Computer Vision*, vol. 15, pp. 143-164, 1995.
[10] R. Plamondon and C.M. Privitera, "The Segmentation of Cursive Handwriting: An Approach Based on Off-Line Re-covery of the Motor-Temporal Information, " *IEEE Trans. On Image Processing*, vol. 8, no. 1, pp.80-91, 1991.
[11] Lam. L, S-W. Lee, CY. Suen, "Thinning Methodologies- A Comprehensive Survey," *IEEE Trans. on PAMI*, vol. 14, no. 9, pp.869-885, 1992.
[12] I.T. Jolliffe, *Principle Component Analysis.* Springer-Verlag, 1986.
[13] H.D. Chang and J.F. Wang, "Preclassification for Handwritten Chinese Character Recognition by a Peripheral Shape Coding Method, " *Pattern Recognition*, vol. 26, no. 5, pp. 711-719, 1993.