Proceedings of International Conference on Image Processing and Character Recognition

A Chinese-Character Stroke Extraction Algorithm Based on Contour Information

Chungnan Lee and Bohom Wu

Institute of Computer and Information Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan, ROC

Phone: (07) 5252000~4313
Fax: (07)5254301
email: cnlee@cie.nsysu.edu.tw

Abstract

In this paper, we present a new stroke extraction algorithm that integrates all levels of contour information including boundary points, dominant points, corner points, segments, cross section sequence graph, character structure to extract strokes of Chinese characters. Unlike the traditional bottom-up approach, which the current step of information extraction is only based on the previous step, we use and accumulate all extracted information until the final extraction of strokes is done. The longer contour segment of stroke tends to preserve better stroke information and is robust to the interfering contour. Hence, stroke extraction in the proposed algorithm is based on the longer contour segment of stroke instead of the medial axis. Experimental results show that the proposed algorithm can correctly extract the strokes up to 95% from a printed and handwritten test samples based on the human perception. This research provides a solid basis for the structural matching of Chinese characters.

1. Introduction

The stroke is an essential information to the structural matching of Chinese characters[1, 2]. The structure of Chinese character that can be characterized by the relative position and orientation of strokes is invariant to size and orientation. The major difference between the Chinese character and the Latin family of languages is that the most of strokes of Chinese characters are composed of vertical or horizontal line segments. Many algorithms have been developed to extract strokes either by thinning or non-thinning approaches [3-21]. The thinning approach has been intensively studied by [3-14]. The main problem for the thinning is the deformation of the skeletons, as a result, it produces distorted strokes. Many authors have proposed algorithms to improve the stroke segmentation, such as a maximum circle technique by Liao and Huang [14], a modified Hough transform by Cheng et al.[15], etc. Liao and Huang [8] use Bernstein Bezier curve and Wang et al. use quadratic equation to do curve fitting [9]. The non-thinning approach has been studied by [15-21]. Mori and Sakura [26] uses line filtering to extract strokes. Chang and Lai use boundary properties at junction points and the relaxation techniques to extract a stroke. Huang et al. [18] use cross section sequence to extract strokes. Jiang et al. [20] use a rubber band to extract small components, called parts, and then merge part into strokes. However, the strokes is not necessary corresponding to those of human perception or writing.

In this paper, we focus on the stroke extraction of the Chinese characters and propose a new algorithm integrating all levels of contour information. We first extract the boundary points. Second, the corner points are detected. Then these corner points are used to find contour segments, Fourth, we modified the cross section sequence graph to find singular regions which are defined as an end point region or a junction region. Finally, the Bezier curve are used to check the continuity of segments of connection regions and determine whether they belong to the same stroke. Each step of extracted information are preserved through the process. Unlike the traditional bottom-up approach, which the current step of information extraction is only based the previous step, we accumulate all extracted information until the final extraction of strokes is done. The longer contour segment of stroke tends to preserve better stroke information and is robust to the interfering contour. Hence, stroke extraction in the proposed algorithm is based on the longer contour segment of stroke instead of the medial axis. Results show the new approach can extract correct strokes up to 97.5% from 200 printed characters and 95% from 64 handwritten characters. The rest of the paper is organized as follows. In Section 2 we describes the contour extraction and gives the corner points detection. Section 3 presents the detection of the structure of characters. In Section 4 we describe the stroke extraction. The experimental results are given in Section 5, Finally, the conclusion is given in the last section. Before we go to the next section, we give the information flow of the algorithm in Figure 1.

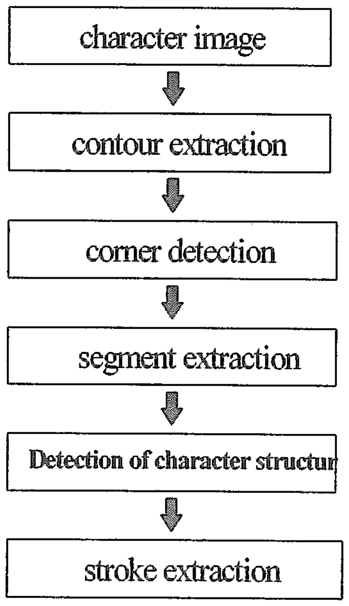


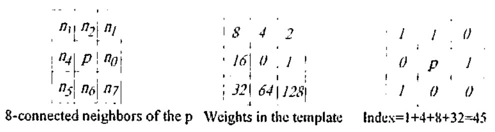
Figure1. The information flow of the algorithm.

2. Contour Information Extraction

Contour is one of the most fundamental information after image segmentation. A set of border pixel position is extracted and recorded in an appropriate data structure. To ensure the correct contour extraction we follow the traditional assumption that the contour is a simple cycle form. In other word, different contours cannot exactly share a single pixel.

2.1 Contour points detection

To detect contour points we use a 3 by 3 template T with different weights as shown in Figures 2 (a) and (b) to operate on the image f(x, y).



(a) (b) (c)

Fig. 2. (a) 8-neighbors.(b) The 3x3 template with weights.(c) An example of the index calculation.

Pixels in f(x, y) are classified into three categories – interior, noise, and contour points. They are defined as follows.

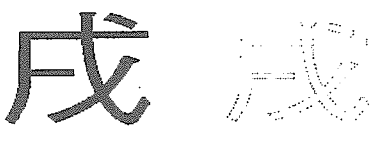
1. Interior point: when (n0 and n2 and n4 and n6) = 1 and (n1 or n3 or n5 or n7) = 1. Those pixels are defined as interior points.
2. Noise point: when n_i ,... , $n_{(i+m) mod 8}$ is 1, where 0<=i<=7 , m>=1, then (n_i ,... , $n_{(i+m) mod 8}$) it is defined as a set of consecutive points. If the neighborhood combinations satisfies the following two criteria, then the pixel is a noise point.
   a) The set of consecutive points is zero.
   b) The sets of consecutive points are equal to two and doesn't exist a simple cycle.
3. Contour point: f(x,y) \ (interior and noise points).

2.2 Contour Following

After contours are extracted, we trace the contour of same region by using the Freeman chain code. The exterior boundary is traced by the clockwise direction and the exterior boundary is traced by the counter-clockwise direction.

2.3 Corner Detection

The intersection of strokes will result a dramatic change in the contour. The corner detection is to detect such change. We first use the Teh and Chin's one curvature algorithm [25] to detect the dominant point from the chain code. In order to speed up the further calculation, the category, direction code, and one curvature are stored as an attribute look-up table. Totally, there are 255 combinations. Only when the pixel's category value is 3, it has the values for direction code and one curvature. Second, the region of support is calculated. Finally, the four passes in [25] are used to detect the dominate points. The character image after dominate points detection is shown in Figure 3. Though the Teh and Chin's algorithm works excellent on dominant points, the further processing is required to extract segments for stroke extraction. We extract corner points from the dominant points by detecting the dramatic change. Figure 4 shows the results after corner detection.



(a) (b)

Figure 3. (a). The character image. (b). The result after the detection of dominant points

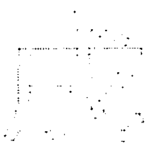


Figure 4. The corner points after the corner detection process.

3. Detection of Character Structure

In order to connect two sub-strokes, a crossing region has to be detected first. The common approach for crossing region extraction are thinning, run length coding, and cross section sequence graph. Due to working with regions, the CSSG gives a better result among these three approaches. We apply the CSSG approach, but make a couple of major changes. We use contour segments and information to find the cross section sequence which is more stable and memory efficient than the traditional CSSG approach.

3.1 Contour Segment Extraction

Instead of using medial axis, as most papers do, we use the contour segments. Following the contour, the segments can be extracted. The first corner encountered is the start point and the second corner encountered is the end point of the segment. The definition of a segment is as follows. Let segment Sg be an order sequence. Sg = $<s, c_1, \ldots, c_k, e>$, *where s* is the start point, *e* is the end point, and $c_{1\ldots k}$ are the contour point. *s* and *e* are determined by the corner points. L(Sg) is the length of the segment. L(Sg) = k, where k is the number of contour points between *s* and *e*.

3.2 Construction of CSSG

Cross section is defined as a path between an order pair of contour points, cs(c_i, *opp*(c_i)), where c_i and *opp*(c_i) belongs to opposite contour segments of the same region. A cross section sequence is a sequence of cross section with no crossing [26]. A character image may contain many cross section sequences. Before we described the methods to construct the CSSG, we give a definition to normal direction as follows.

For a segment *seg* , *s* is the start point , *e* is the end point, c_i is the ith contour point at the segment . Let *dir(p,q)* be the direction from point *p* to point *q* and *dist(p,q)* be the Euclidean distance from point *p* to point *q*. Let $\Delta\theta$ be the angle deviation between *dir*(c_i,*e*) and *dir*(*s*, c_i) and $\Delta\phi = [dist(c_i,e) / (dist(s,c_i) + dist(c_i,e))] * \Delta\theta$.

The approximation tangent direction *tan_dir* at c_i is computed as

$$tan\_dir = (dir(s, c_i) + \Delta\phi) \bmod 360$$

If the angle deviation between *tan_dir* and *dir*(c_i,*e*) is greater than $\Delta\theta$, then

$$tan\_dir = (dir(s, c_i) - \Delta\phi + 360) \bmod 360$$

The normal direction at c_i is *tan_dir* + 90 as shown in Figure 5.

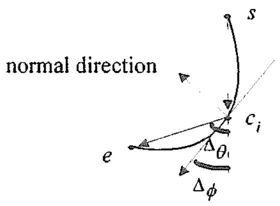


Figure 5. The normal direction at c_i.

To construct the CSSG we use the following procedures.

Step 1. Suppress those segments whose length are less than a tenth of the image size.

Step 2. Use those contour points of the remain segment, seg_j , j=1,..., m to construct cross section. Because points that close to the end region might be distorted by noise, we use $c_1,\ldots,c_{k-l}$ to construct the cross section, where *l* is set to 0.1**k*, and use the normal direction of point c_i to find a point, *opp*(c_i), on the opposite side. The opposite segment is represented by *oppseg*(c_i , seg_j). If the opposite segment is not suppressed and the angle between the tangent direction of c_i and the tangent direction of *oppseg*(c_i, seg_j) is larger than a threshold, ang_thr, then the cross section is valid, otherwise the cross section is invalid. The ang_thr is set to 155.

Step 3. Group those cross sections that belong to seg_j and *oppseg*(c_i , seg_j) as a cross section sequence which is represented by { *cs*(c_1 , *opp*(c_1)) ,..., *cs*(c_m , *opp*(c_m)), where *m* is the number of cross sections. When m is larger than a threshold, then the cross section sequence is valid, otherwise it is invalid. The threshold is set to a twentieth of the image size. We use the the first and last cross sections *cs*(c_1 , *opp*(c_1)) and *cs*(c_m , *opp*(c_m)) to index the sequence and remove the rest of cross sections to save the memory.

Step 4. Find all cross section sequences and sort them from small to large by length. If an intersection occurs, then remove the longer one. Figure 6 shows the intersection of cross sections.

After applying the above four steps, one can get the cross section sequence graph as shown in Figure 7.

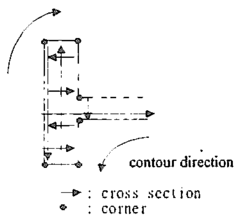


Figure 6. An example showing the intersected cross sections.

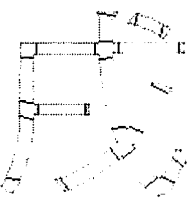


Figure 7. The cross section sequence graph. Only the first and last cross sections are used to represent the cross section sequences.

3.3 Contour Segments Pairing

The purpose of contour pairing is to find meaningful segments for each stroke. Usually, there exists two longer segments that are more significance than others, which are called meaningful segment. Other segments of the stroke are called meaningless segment, such as end region segment, singular region segment, etc. Meaningful segments are used to extract stroke. The meaningful segments generated by pairing are shown in Figure 8.

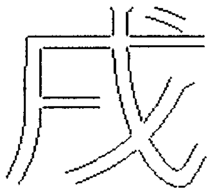


Figure 8. The meaningful segments after pairing.

3.4 Extraction of Singular Regions

As shown in Figure 7 there are many partitioned regions in the CSSG. For each region it may connect to several other regions. The number of regions connected to a region is called region adjacency number. For those regions with end region, cross section, and junction region, are called singular regions. Singular regions are important information to evaluate the continuity of a stroke. The method to extract a singular region is as follows.

Step 1. Remove those corner points that do not belong to the meaning segments.

Step 2. Start from a cross point, (c_i), of an arbitrary cross section. If the cross section has not been marked, then set its traced number to one, where the traced number is defined as how many time the cross section has been traced, and then use $opp(c_i)$, as a new starting point. If all the cross section have been searched, then go to Step 4.

Step 3. Follow the contour from the starting point to find the next point until the following conditions are satisfied.

(a) If the point is a corner point, then record the segment that contains the corner point and check its direction. If the point is an end point, then the segment is an in-degree, otherwise the segment is an out-degree as shown in Figure 9.

(b) If a cross point of the cross section is encountered, then check whether the cross section is an starting cross section and how many the cross section has been traced.

(b-1) If the traced number is 2, then it is not the starting cross section and the search is fail.

(b-2) If the point is the starting point, then the search succeeds and the region is established. The segments encountered found in Step 3 (a) are connected to the region, then set the traced number of the cross section to 2.

Continue the process (a) and (b) until the stack is empty and then return to Step 2.

(c) If the traced number of the cross section is zero, then the cross section has not encountered, then set its traced number to 1, push the cross point into stack, and jump to the opposite cross point to continue the next cross point.

(d) If the traced number is one, then jump to the opposite cross point to continue the next point.

Figure 10 shows the process of Steps 2 and 3.

Step 4. Remove those regions without any connected segments.

The result for the singular region extraction is shown in Figure 11.

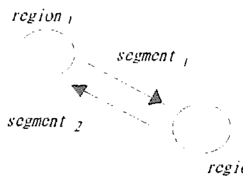


Figure 9. An example showing the two kinds of degree type.

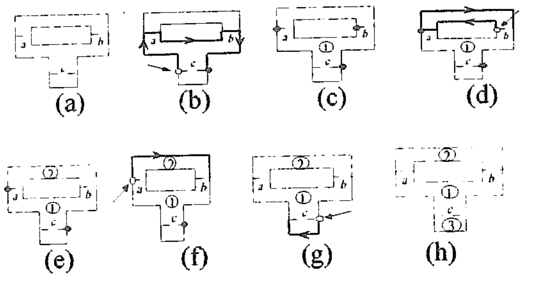


Figure 10. An example of the extracting regions process. (a) A CSSG with three dash lines as cross sections. (b) The cross section *c* is used as the starting cross section, the cross point is pushed into stack and is marked as a black dot, then jump to the opposite cross point as a new starting point. Follow the contour one encounters the cross sections *a* and *b* that are pushed into stack and marked as black dots. (c) Result of a successful search to obtain region 1. (d) The cross section *b* is popped from stack to start another search. (e) Region 2 is a result of a successful search. (f) The cross section *a* is popped to start another search. Since the cross section *b* has been successfully searched, its traced number is 2. Therefore, the search fails. (g) The cross section *c* is popped to start another search. (h) Region 3 is a result of another successful search.

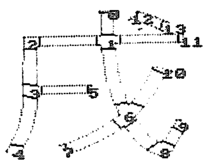


Figure 11. The singular regions.

3.5 Region Connection

After singular regions are extracted, we have to find the connection between singular regions and regular regions. For all pairing meaningful segments, we take the longest segment of the stroke. The longer segment tends to preserve more information. Figure 12 shows the survival segments. Based on the singular region and survival segment the relationship of regions can be extracted. The number of regions connected to a region is called the region connection number. The result is shown in Figure 13.

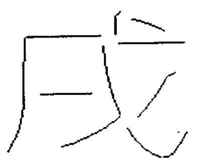


Figure 12.The survival segments

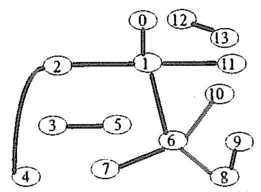


Figure 13. The region connection graph

4. Stroke Extraction

`To extract the stroke one has to calculate the continuity of segments. In the literature, most of papers focus on direction deviation between two medial axes. In here, we focus on the continuity of the contour.

4.1 Continuity Checking

If the curvature change of two segments are quite uniform, then one can use some sampled points to represent the segments and to fit a curve. This curve will close to the segments. Otherwise the curve will not close to the segments. For example, let seg_1 and seg_2 be two segments with s_1 、e_1 、s_2 、e_2 as the starting points and the end points, respectively. As shown in Figure 14 (a) where two segments do not belong to the same stroke, the error distance between the fitted curve will be large. On the contrary, as shown in Figure 14 (b) the curve fits the segments uniformly.

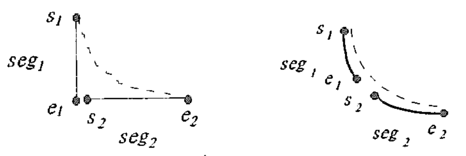


Figure 14. (a) The two segments are not consistent . (b) The two segments are consistent.

The Bezier polynomial is a popular representation model in 3D graphics. The application of the Bezier polynomial in stroke segmentation can be found in [14]. The Bezier polynomial is represented as follows.

$$B(t)=\sum_{i=0}^{m}\binom{m}{i}P_i t^i (1-t)^{m-i} \quad \textit{for } 0 \le t \le 1 \textit{ where } P_0 \textit{ and } P_m \textit{ are end points,}$$

$P_1, P_2, \ldots, P_{m-1}$ are control points.

where

$\binom{m}{i}t^i(1-t)^{m-i}$ *is Bernstein polynomial basis function*

, $i = 0,\ldots,m$, and m the order of polynomial. We use m+1 contour points to find m order of Bezier curve, then generate n+1 point with equal incremental interval, where n = ((m+1)/2) +1.

4.2 The Process of Stroke Extraction

There are two phases in stroke

extraction process. In the first phase the T junction is processed. The T junction can be detected from the region connection number and the region adjacent number. If the two numbers are not equal, then it is a T junction. In second phase we calculate the continuity of segments. The procedure can be summarized as follows.

Step 1. When region connection number is larger than 1, we pair all possible segments. Each pair of segment is a segment group.

Step 2. For each segment group the continuity of the segments is calculated as follows.

(a) the in-degree segment should be paired with the out-degree segment

(b) The dominant points of segments are used as control points to generate Bezier curve. The input of dominant points is in order without repetition.

(c) The corner points are used to check the fitting of Bezier curve. For each corner it measures the distance to the Bezier curve.

(d) When the distance value is smaller than a threshold, then two segments are connectable and redraw the region connection graph.

Step 3. After all regions are processed, we can obtain the region connection graph as illustrated in Figure 15.

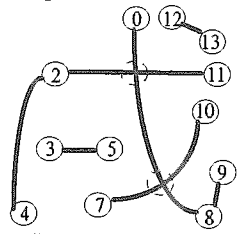


Figure 15. The final region connection graph.

5. Experimental Results and Discussions

5.1 Results

To test the effectiveness of our algorithm we apply the algorithm to 200 Gothic style characters that are randomly selected from the Big-5 code and are shown in Figure 16. The image size for each characters is 128 x 128.

乙丁乃九了入刀力卜又下丈丫凡久么也乞叉
口土夕大女子小尸山川工已干弓才丑丐不丰
丹之尹井仁今元允內兮公冗凶分切勾勿匹升
厄友及反壬太夭孔少尺屯巴幻弔引心戈手支
文什斤方日月木止毋比毛氏水父爻片牙牛王
且丘主乍以兄冉冊凹凸刊加包匆北半卡去只
史台四失尼布幼弗斥本未正民氐永犯瓦生田
甲皿伐矢禾立企光兕先全共再刑劣印同吊合
后回夸奸字安尖并年忙戎戍托收早曳有朽死
氛聿美庭痔粕貫雩湃犄鈍閑朕男我尾除陛黑

Figure 16. Test samples of printed characters.

The classification of the stroke extraction is based on the human perception, not based on the definition in the dictionary. For example, the character '乙' as shown in Figure 17 (a) is defined as one stroke in the dictionary. However, from the human perception it should be divided into three strokes as shown in Figure 17 (b). When all strokes of a character are completely extracted, it is called a correct extraction.

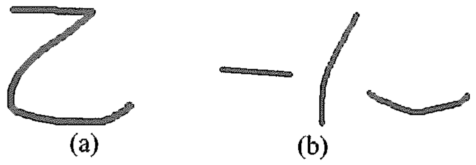
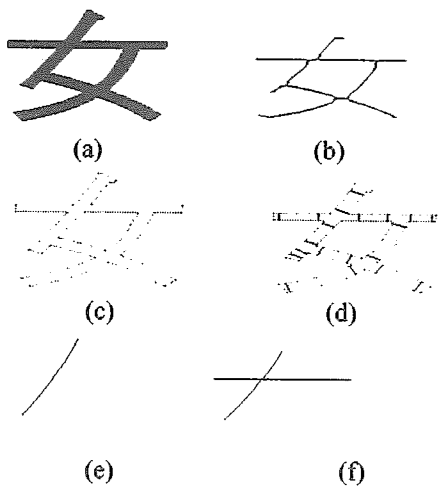
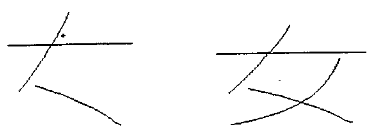


Figure 17. (a) Character image. (b) The character is separated into three strokes from the human perception.

We use one character as an example to show the steps of the algorithm. In Figure 18(b) the thinned character is done by using the algorithm proposed by Zhang and Suen [29]. As one can see that the thinned character has distortion on the junction regions which is the common problem for thinning approach. Figure 18 (c) shows the results after corner detection, Figure 18 (d) shows the results of modified CSSG. Figures 18 (e)-(h) show the sequence of stroke extraction by the algorithm. The stroke is the contour of large boundary for each stroke image. Among the 200 characters tested the algorithm has extracted the stroke correctly for 195 characters. Only five characters, 什, 仁, 除, 陛, and 生, are ambiguously extracted on a specific stroke. For example, Figure 19 shows that the stroke '亻' has extracted as '亻' due to the undetected corner which is marked by circle in Figure 19(b).





(g) (h)

Figure 18. An example of tested characters. (a) Character image. (b) Thinned image. (c) Result after corner detection. (d) CSSG. (e)-(h) The stroke extraction of the character.

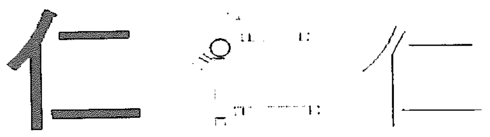


Figure 19. (a) Character image. (b) CSSG. (c) Result after stroke extraction.

The algorithm is also applied to 64 handwritten characters as shown in Figure 20. Among 64 characters tested, the algorithm has extracted the stroke correctly for 62 characters. Two characters, 生 and 而, are extracted ambiguously on '生' and '而'.

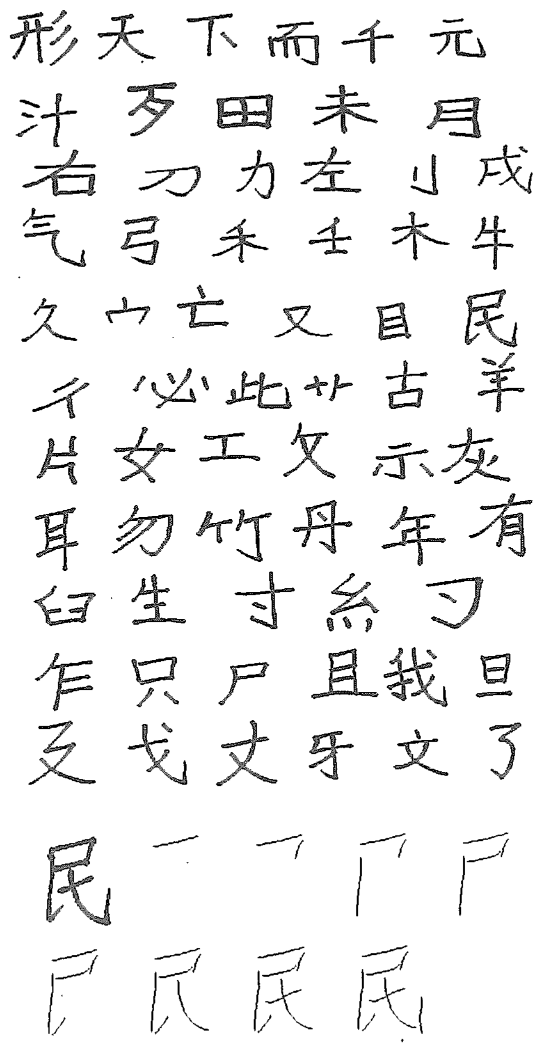


Figure 20. Test samples of handwritten characters and stroke extraction sequences of one character.

5.2 Discussions

As one can see the extracted strokes are those longer contour of the strokes. The longer contour of a stroke tends to preserve more information. Furthermore, it is more robust during the stroke extraction. Especially, for the T junction where one side of contour clearly represents the stroke. But the other side of contour may be interfered by the intersection of other stroke. For the medial axis or the skeleton approach the intersection region may cause the problem. Because the proposed algorithm uses the contour information, it is very efficient for the character with wide strokes. On the contrary, the thinning algorithm needs to spend many iterations to obtain the skeleton.

Though the algorithm works very effective on stroke extraction, there are still several points to be improved. First, for those smooth curves the corner detection may fail to detect the right corner, as a result, the stroke may be extracted wrongly as shown in Figure 20. Second, when applying the algorithm to different fonts the parameters used in the algorithm have to be tuned. Third, The strokes cannot severely distorted, otherwise the extra strokes will be extracted.

6. Conclusions

We have developed a new stroke extraction algorithm that integrates all levels of the contour information. The contour information including boundary points, dominant points, corner points, contour segments, cross section sequence graph, and character structure. The boundary points, dominant points, corner points, and contour segments are extracted first. Then, the boundary point, corner points, contour segment are used to find the CSSG. Third, we use contour segments and the CSSG to extract the character structure. Finally, a Bezier curve taking dominant points and corner points as inputs is used to check the continuity and to extract the stroke. Both printed and handwritten characters are used to test the efficiency of the algorithm. Experimental results show the proposed algorithm can correctly extract the strokes up to 95% for both test samples. In the future the algorithm will be incorporated into a structural matching recognition system.

Acknowledgment

This research was partially supported by the National Science Council of Taiwan, R.O.C., under contract NSC-86-2213-E-110-029.

References

1. T. H. Hildebrandt and W. Liu , Optical Recognition of handwritten Chinese Characters : Advances since 1980, *Pattern Recognition*, Vol.26, No.2, .205-225, (1993).
2. S. Mori, K. Yamamoto, and M. Yasuda, Research on machine Recognition of Handprinted Characters, *IEEE Transactions on Pattern Analysis Machine Intelligence*, Vol.6, No.4,.386-405, (1984).
3. L. Lam, S. W. Lee, and C. Y. Suen, Thinning Methodologies - A Comprehensive Survey, *IEEE Transactions on Pattern Analysis Machine Intelligence*, Vol.14, No.9, .869-885, (1992).
4. L. Lam and C. Y. Suen, Evaluation of thinning algorithm from an OCR viewpoint, *International Conference on Document Analysis and Recognition*, .35-40, (1993).
5. H. Ogawa and K. Taniguchi, Thinning and Stroke Segmentation for Handwritten Chinese Character Recognition, *Pattern Recognition*, Vol.15, No.4, .299-308, (1982).
6. P.N. Chen, Y. S. Chen, and W.H. Hsu, Stroke Relation Coding - A New Approach to the Recognition of Multi-Font Printed Chinese Characters, *International Journal of Pattern Recognition and Artificial Intelligence*, Vol.2, No.1, .149-160, (1988).
7. S. L. and W. H. Tsai, Recognizing Handwritten Chinese Characters by Stroke-Segment Matching Using an Iteration Scheme, *International Journal of Pattern Recognition and Artificial Intelligence*, Vol.5, No.1&2, .175-197, (1991).
8. C. W. Liao and J. S. Huang, A Transformation Invariant Matching Algorithm for Handwritten Chinese Character Recognition , *Pattern Recognition*, Vol.23, No.11, .1167-1188, (1990).
9. A. B. Wang, K. C. Fan, and J. S. Huang, Recognition of Handwritten Chinese Characters by Modified Relaxation Methods, *Image and Vision Computing*, Vol.12, No.8, (1994).
10. A. J. Hsieh, K. C. Fan, T. I. Fan, and Chin-Wen Ho, A Stroke-Based Handwritten Chinese Character Recognition System Using Greedy Matching Method, *Journal of Information Science and Engineering 11*, .1-22, (1995).
11. H. D. Chang and J. F. Wang, A Robust Stroke Extraction Method for Handwritten Chinese Characters, *International Journal of Pattern Recognition and Artificial Intelligence*, Vol.8, No.5, .1223-1239, (1994).
12. C. C. Hsieh and H. J. Lee, A Probabilistic Stroke-Based Viterbi Algorithm for Handwritten Chinese Characters Recognition, *International Journal of Pattern Recognition and Artificial Intelligence*, Vol.7, No.2, .329-352, (1993).
13. K. W. Gan and K. T. Lua, A New Approach to Stroke and Feature Point Extraction in Chinese Character Recognition, *Pattern Recognition Letters* 12, .381-387, (1991).
14. C. W. Liao and J. S. Huang, Stroke Segmentation by Bernstein - Bezier Curve Fitting, *Pattern Recognition*, Vol.23, No.5,.475-484, (1990).
15. F. H. Cheng, W. H. Hsu, and M. Y. Chen, Recognition of Handwritten Chinese Characters by Modified Hough Transform Techinques, *IEEE Transactions on Pattern Analysis Machine Intelligence*, Vol.11, No.4, .429-439, (1989).
16. L. Y. Tseng and C. T. Chuang, An Efficient Knowledge-Based Stroke Extraction Method for Multi-Font Chinese Characters, *Pattern Recognition*, Vol.25, No.12,.1445-1458, (1992).
17. C. T. Chuang and L. Y. Tseng, A Stroke Extraction Method for Multifont Chinese Characters Based on the Reduced Special Interval Graph, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.25, No.7, .1171-1178, July (1995).
18. J. S. Huang, Tzung Rern Jang, Fu Chang, Extracting Strokes of a Chinese Character Based on Cross-Section Sequence Graph, *Journal of Information Science and Engineering* 8, .512-524, (1992).
19. F. Chang, Y. C. Chen, H. S. Don, Stroke Segmentation as a Basis for Structural Matching of Chinese Characters, *International Conference on Document Analysis and Recognition*, .35-40, (1993).
20. J. Jiang, W. Kim, and H. Tominaga, Recognition and Representation of Text Characters Using Rubber Band , *International Conference on Document Analysis and Recognition*, .955-958, (1993).
21. J. H. Wang and Shinji Ozawa, Automated Generation of Chinese Character Structure Data Based on Extracting the Strokes, *International Conference on Document Analysis and Recognition*, .806-809, (1993).
22. S. M. Ali and R. E. Burge, A New Algorithm for Extracting the Interior of Bounded Regions Based on Chain Coding, *Computer Vision, Graphics, and Image Processing*, Vol.43,.256-264, (1988).
23. F. Y. Shih and W.T Wong, A New Safe-Point Thinning Algorithm Based on the Mid-Crack Code Tracing, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.25, No.2, (1995).
24. H. Freeman, Computer processing of line drawing images, *Comput. Survey*, Vol.6, No.1,.57-97, (1974).
25. C.H. Teh and R. T. Chin, On the Detection of Dominant Points on Digital Curves, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol.11, No.8, .859-872, (1989).
26. T. Suzuki and S. Mori, Structureal Description of Line Images by the Cross section Sequence Graph, *International Journal of Pattern Recognition and Artificial Intelligence*, Vol.7, No.5,.1055-1076, (1993).