A BACKGROUND-THINNING BASED ALGORITHM FOR SEPARATING CONNECTED HANDWRITTEN DIGIT STRINGS

Zhongkang Lu†, Zheru Chi† and Pengfei Shi‡

†Department of Electronic Engineering
The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong

‡Institute of Pattern Recognition & Image Processing
Shanghai Jiaotong University
Shanghai, 200030, P. R. of China

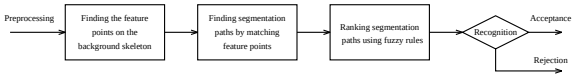

Figure 1: Flowchart of our background-thinning based approach for segmenting handwritten digit strings.

ABSTRACT

Most algorithms for segmenting connected handwritten digit strings are based on the analysis of the foreground pixel distributions and the features on the upper/lower contours of the image. In this paper, a new approach is presented to segment connected handwritten two-digit strings based on the thinning of background regions. The algorithm first locates several feature points on the background skeleton of the digit image. Possible segmentation paths are then constructed by matching these feature points. With geometric property measures, these segmentation paths are ranked using fuzzy rules generated from a decision-tree approach. Finally, the top ranked segmentation paths are tested one by one by an optimized nearest neighbor classifier until one of these candidates is accepted based on an acceptance criterion. Experimental results on NIST special database 3 show that our approach can achieve a correct classification rate of 92.4% with only 4.7% of digit strings rejected, which compares favorably with the other techniques tested.

1. INTRODUCTION

The segmentation and recognition of connected characters is a key problem in the development of Optical Character Recognition (OCR) systems. Many algorithms have been proposed in the recent years. However, the performance of an existing technique is less satisfactory due to the problem in segmenting handwritten characters. Lu has given an overview on various techniques for the segmentation of machine printed characters [1] and another overview, together with Shridhar, on handwritten characters [2].

In this paper, a new segmentation approach is presented based on the analysis of the background (regions excluding the characters) skeleton of a digit string image. Figure 1 shows the flowchart of our background-thinning based digit segmentation approach. The algorithm first locates several feature points on the background skeleton of the digit image. Possible segmentation paths are then constructed by matching these feature points. With geometric property measures, these segmentation paths are ranked using fuzzy rules generated from a decision-tree approach. Finally, the top ranked segmentation paths are tested one by one by an optimized nearest neighbor classifier until one of these candidates is accepted based on an acceptance criterion.

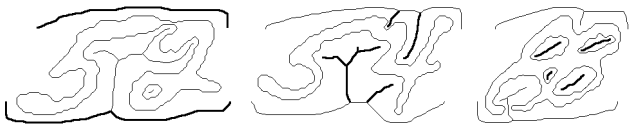

Figure 2: Examples of background skeletons with thick lines indicating (a) base segments; (b) branch segments; and (c) hole segments.

2. FEATURE POINT EXTRACTION

In order to get a complete background skeleton, the foreground should be placed in an rectangle frame with a small space to each of the four sides. The background regions of a digit string image is thinned by using a skeletonization algorithm adapted from Hilditch's thinning algorithm [3]. For explaining our algorithm better, the definitions of different segments on the background skeleton are given first in the following:

- *Base segment:* a segment generated between the foreground and the top or bottom of the image, as the thick lines shown in Fig. 2(a). The upper one is named as the upper-base segment and the lower one as the lower-base segment.
- *Side segment:* a segment generated between the foreground and the left or right image frame, which are not used in the subsequent processing.
- *Branch segment:* a segment connected to a base segment (excluding side segments) as the thick lines shown in Fig. 2(b). Like a base segment, a branch segment connected to the upper-base segment is named as the

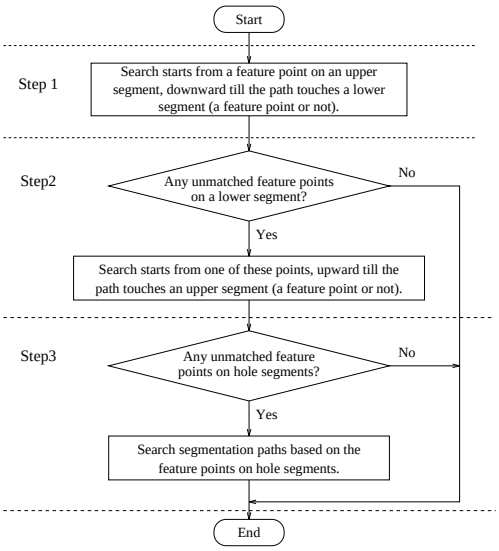

Figure 3: Flowchart for the construction of segmentation paths.

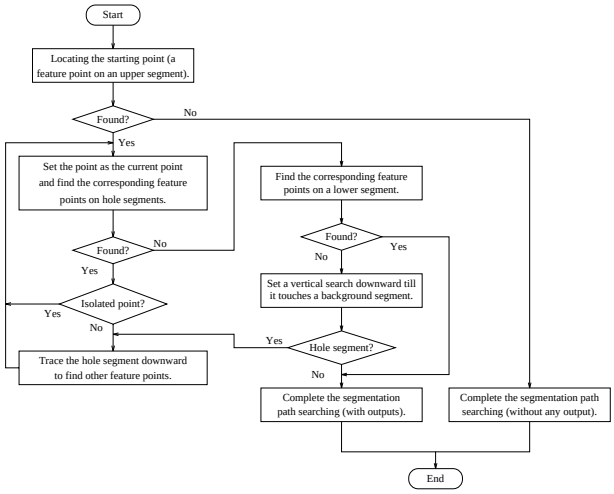

Figure 4: Flowchart for the top-down searching of segmentation paths.

upper-branch segment and a branch segment connected to the lower-base segment as the lower-branch segment.

- *Hole segment:* a segment generated from a hole region of the background as the thick lines shown in Fig. 2(c).

The feature points include end points, fork points and corner (high curvature) points in the background skeletons. The extraction of feature points includes two steps. The first step is to find all fork points and end points on the background skeleton, and then LSEA (Least Square Error Approximating) method is used to find the corner points.

3. CONSTRUCTION OF SEGMENTATION PATHS BY MATCHING FEATURE POINTS

A segmentation path is constructed by connecting several feature points on the background skeleton with possible extension to the top and/or bottom of an image. A three-step searching scheme shown in Fig. 3 is adopted to search the feature points on upper segments, lower segments and hole segments and to construct segmentation paths. The first step is to search the feature points from the top to bottom (top-down searching). If there exist unmatched feature points on lower segments, conduct a bottom-up searching from these points in the second step. If there exist unmatched feature points on hole segments, another search is conducted in the third step in which searching is first done upward and downward separately and the two traced segments are integrated together into a segmentation path. As a result, all possible segmentation paths can be identified by this searching scheme.

Figure 4 shows the flowchart of the top-down search in the first step. It starts from a feature point on an upper segment and end at a point, a feature point or not, on a lower segment. The important part of the searching is to find the matched feature points on hole segments and lower segments.

Quite often, more than one feature points are matched to the current feature point, as Example 1 shown in Fig. 5(a).

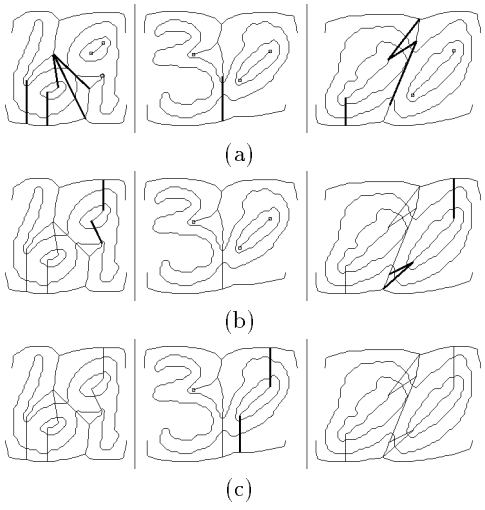

Figure 5: Examples of segmentation paths (feature points marked by '□' are unmatched points): (a) segmentation paths found (dark lines) by Step 1 (top-down searching); (b) segmentation paths found by Step 2 (bottom-up searching); and (c) segmentation paths found by Step 3 (searching from hole).

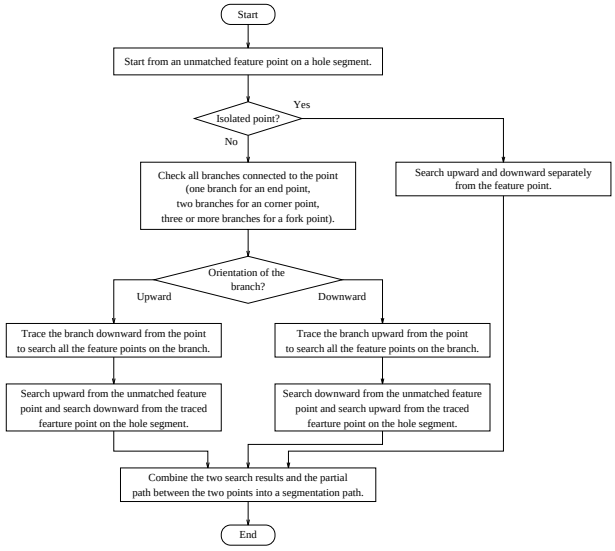

Figure 6: Flowchart for the searching beginning at a feature point on a hole segment.

All possible segmentation paths should be recorded for further processing. On the contrary, sometimes no feature point can be found, as Example 2 shown in Fig. 5(a). In this case, a default vertical searching path is constructed downward till it touches a background segment (either a hole segment or a lower segment).

The bottom-up searching in the second step is similar to the top-down searching in the first step. One difference is that the former is searching upward and the latter is searching downward.

Figure 6 is the flowchart for Step 3 of our searching scheme. The searching process begins from the unmatched feature points on hole segments and search in two directions. The tracing and searching are same as those in the first two steps. The key problem here is to determine which point should be to search upward and which to search downward. At present, the orientation of the branch connected to the feature point is used to judge the search direction. After the searching, two searched segments and the segment between the unmatched feature points are integrated into a segmentation path. As Example 2 shown in Fig. 5(c), there are two unmatched points on the hole segment after Steps 1 and 2, the searching starts from the two points, one upward and one downward, the two traced segments and the segment between the two points represents a possible segmentation path.

4. RANKING SEGMENTATION PATHS USING FUZZY RULES

The degree to which a candidate is a good segmentation path is determined by fuzzy rules with the nine properties associated with the path and separated parts by the path.

An image of a digit string is split into the left part and the right part by a segmentation path. Segmentation paths are ranked based on the nine properties associated with paths and the separated parts by the paths. The nine properties are adapted from [4], which have been successfully used to ranking segmenting paths for recognizing handwritten single- and double-touching digit strings.

Interactive Dichotomizer decision trees and rules were first present by Quinlan [5]. Decision rules work well when input data is noise-free. However, its performance degrades when input data is uncertain or noisy. Chi *et al* proposed to use fuzzified decision rules to deal with the uncertain problems in handwritten digit recognition [6].

By using the membership grades instead of binary values 0 and 1 for the degree to which a rule is fired, the decision rules becomes a set of fuzzy rules.

5. EXPERIMENTS AND DISCUSSION

Decision rules were generated and tested using 6,697 manually classified segmentation paths, which were extracted from 1,134 two-digit strings from NIST special database 3. With 4,000 as training samples, 28 decision rules were generated. Among them, 17 rules were fuzzified and the others were discarded because they have little impact on training samples. The correct classification rate is 98.2% on the 4,000 training paths and 97.3% on the remaining 2,697 test paths.

The separated digits were recognized using an optimized nearest-neighbor classifier proposed by Yan [7] with sixty-four intensities on the 8×8 image. The classifier returns both the assigned class and the distance of the image from the closest class prototype. The distance is termed as the "recognition measure" and used as the estimation of the reliability of a classification [4].

The classifier was trained using isolated digits extracted from NIST special database 3. In total, 53,449 isolated digits were extracted, and classified for training the classifier, and other 53,185 samples for testing. The correct rate is 98.9% for training samples and 97.8% for test samples, without rejection. The mean values M_i $(i = 0, 1, \ldots, 9)$ of the recognition measures for the correctly classified digits in the training samples were obtained and used to decide whether a classification was accepted or not. On the other hand, 3,355 two-digit strings, which were also extracted from NIST special database 3, were used as the test data for the whole system. Note that the digits from these 4,489 two-digit strings (3,355 for testing the whole system and the 1,134 for generating and testing fuzzy decision rules) were not included in training the optimized nearest-neighbor classifier.

Suppose that the tolerate radius is a. If the recognition measure is smaller than aM_i, then the classification is accepted. Table 1 shows the experimental results with different recognition measure radius. With the increasing recognition measure radius, both the rejection rate and the correct classification rate decrease.

For a comparison, we also applied a few other algorithms to separate the same two-digit strings used in our experiments. These algorithms include the MCM (Modified Curvature Method) of Chi *et al* [4], and those of Fenrich [8], Fujisawa *et al* [9] and Zhao *et al* [10]. Table 2 is the experimental results of these algorithms together with ours.

We can see from Table 2 that our background-thinning based approach for segmenting and recognizing connected digit strings can produces results that compare favorably with those from the other techniques tested. Moreover, our approach can deal with both single- and multi-touching problems in digit string segmentation and achieve a correct rate of 92.4% with only 4.7% of digit strings rejected.

Table 1: Experimental results of test samples with different recognition measure radius ($M_i, i = 0, 1, ..., 9$, are the mean values of recognition measures for digits 0, 1, ..., 9).

a	Rejected	Correct	Wrong
1.2	958 (28.6%)	2324 (97.0%)	73 (3.0%)
1.6	353 (10.5%)	2825 (94.1%)	177 (5.9%)
1.8	216 (6.4%)	2919 (93.0%)	220 (7.0%)
2.0	157 (4.7%)	2954 (92.4%)	244 (7.6%)
2.2	127 (3.8%)	2949 (91.4%)	279 (8.6%)

Table 2: A performance comparison of our approach with other digit separation algorithms.

Techniques	High rejection rate		Low rejection rate	
	Rejection (%)	Wrong (%)	Rejection (%)	Wrong (%)
Our algorithm	28.6	3.0	4.7	7.6
MCM of Chi [4]	32.7	4.9	3.7	10.8
Fenrich algorithm [8]	53.3	5.8	5.3	27.1
Fujisawa algorithm [9]	33.5	7.7	0.7	21.9
Zhao *et al* algorithm [10]	38.3	4.4	3.4	13.3

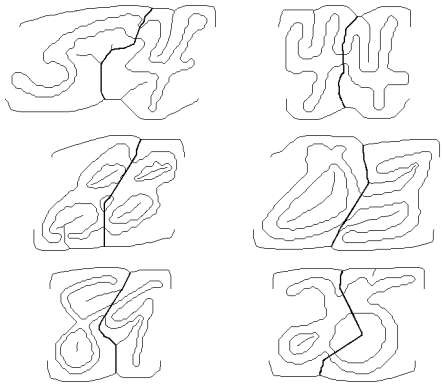

Figure 7: Examples of separated connected digit strings

6. CONCLUSION

In this paper, a background-thinning based approach is presented to separate and recognize connected two-digit strings. In this approach, a set of feature points on the background skeleton of an digit string image are used to trace all possible segmentation paths based on a three-step searching process. Fuzzified decision rules, which are generated from training samples, are used to rank segmentation paths. The top ranked paths are then tested by an optimized nearest neighbor classifier and a good enough segmentation path is determined based on pre-set acceptance criteria. Experimental results on NIST Special Database 3 show that our technique can successfully separate a large proportion of connected digit strings of single- or double-touching with a performance which is compared favorably with those of other techniques tested.

7. REFERENCES

[1] Y. Lu. Machine printed character segmentation – an overview. *Pattern Recognition*, 28(1):67 – 80, 1995.

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

[3] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-wesley Publishing Company, Boston, 1992.

[4] Z. Chi, M. Suters, and H. Yan. Separation of single- and double-touching handwritten numeral strings. *Optical Engineering*, 34(4):1159 – 1165, 1995.

[5] J. R. Quilan. Introduction of decision trees. *Machine Learning*, 1(1), 1986.

[6] Z. Chi, M. Suter, and H. Yan. Handwritten digit recognition using combined id3-derived fuzzy rules and Markov chains. *Pattern Recognition*, 29(11):1821 – 1833, 1996.

[7] H. Yan. Handwritten digit recognition using optimized prototypes. *Pattern Recognition Letters*, 15, 1994.

[8] R. Fenrich. Segmentation of automatically located handwritten numeric strings. In *From Pixels to Features III: Frontiers in Handwriting Recognition*, pages 47 –59. Elsever Science, 1992.

[9] H. Fujisawa, N. Yasuaki, and K. Kurino. Segmentation methods for character recognition: from segmentation to document structure analysis. *Proc. IEEE*, 80(7):1079 – 1092, 19922.

[10] Z. Zhao, M. Suters, and H. Yan. Connected handwritten digit separation by optimal contour partition. In *Proc. DICTA-93 Conference on Digital Image Computing: Techniques and Applications*, pages 786 – 793, 1993.