Locating Segmentation Regions of Connected Handwritten Digits

Jianming Hu and Hong Yan

Department of Electrical Engineering
The University of Sydney, NSW 2006, Australia
{jianming, yan}@ee.usyd.edu.au

Abstract. Segmentation of handwritten touching characters is an important research topic in Optical Character Recognition (OCR). Finding the initial candidate segmentation points is the first important step. A method is described to locate the initial candidate segmentation regions based on the boundary analysis of a character string. This method can reduce the redundant searching regions significantly. Therefore, the segmentation reliability can be increased and the segmentation time can be reduced.

1 Introduction

Handwritten character segmentation is one of the most difficult problems in character recognition because of the rather complicated touching patterns of handwritten characters. Various methods for handwritten character segmentation have been reviewed in [1] and [2]. A number of algorithms have been proposed to solve this problem [3]-[8].

The first important step in a segmentation algorithm is to choose candidate points for a partitioning path. In a boundary based segmentation algorithm, one often divides the outer boundary into regions from which the candidate cut points are searched out. In many segmentation algorithms, the boundary is simply partitioned into upper and lower profiles at the leftmost and rightmost points. A more sophisticated method in [8] divided an outer boundary into open regions, mountain regions, and valley regions.

If we examine the regions divided by the above algorithms, we can find that there are many "redundant" regions which can be excluded from the candidate searching regions. Locating the candidate segmentation regions reliably and accurately is an important step in a segmentation algorithm. However, this problem is not trivial. Without considering the specific structures of the digits, it would be very hard to solve.

This paper describes a new approach to locate candidate segmentation regions based on boundary analysis of a character string. The boundaries are first smoothed and features are extracted from each boundary. The touching patterns are then classified according to the extracted feature vectors, and various touching patterns are considered in each class. Because the boundary structures can

be interpreted globally and the domain-specific knowledge of each numeral can be implemented conveniently, this method has a high reliability and can reduce the number of candidate partitioning paths substantially.

2 Boundary Smoothing and Feature Extraction

The image is first smoothed to remove small spurs and fill up small holes, and then a stroke width compensation algorithm is applied [9]. The boundaries of the smoothed image are then extracted and smoothed using the iterative smoothing algorithms in [10] and [11]. Three kinds of boundary features are extracted [11]: the main feature vectors (v and a) for the even chain code directions, the secondary feature vectors (w and b) for the odd chain directions, and the supplementary feature vectors (x and c). A feature vector fv consists of 5 components:

$$fv[j] = (label, s, t, dir, l) \tag{1}$$

where j is the index, *label* is the symbol of the feature vector, s and t are the starting point and end point, *dir* is the direction of the feature vector which is the chain code of the points in the feature vector except the last point, and l is the length of the feature vector.

The main feature vectors are the most important descriptors of a boundary, and therefore a global code is given to describe them. The global code consists of 6 elements which are defined as follows:

$$gc[i] = (gc^{(i)}[0], gc^{(i)}[1], gc^{(i)}[2], gc^{(i)}[3], gc^{(i)}[4], gc^{(i)}[5]) \tag{2}$$

where i is the index, $gc^{(i)}[0]$ and $gc^{(i)}[1]$ denote the number of v and a feature vectors respectively, $gc^{(i)}[2]$, $gc^{(i)}[3]$, $gc^{(i)}[4]$, and $gc^{(i)}[5]$ denote the number of feature vectors whose directions are chain code 0, 2, 4, and 6 respectively. For the convenience of matching, the main feature vectors are stored before all other feature vectors. Figure 1 shows an example, where (a) is the boundary with extracted feature vectors, and (b) is the codes of the feature vectors.

3 Boundary Regions

In the following, the jth main feature vector in the outer boundary is denoted as $ov[j]$ $(j = 0, 1, ..., n-1)$, where n is the number of main feature vectors in the boundary. The height and width of the image are denoted as h and w respectively. The components of $ov[j]$ are denoted as $ov[j].label$, $ov[j].s$, $ov[j].t$, $ov[j].dir$, and $ov[j].l$ respectively (Eq. (1)). The x and y coordinates of $ov[j].s$ ($ov[j].t$) are denoted as $ov[j].s_x$ ($ov[j].t_x$) and $ov[j].s_y$ ($ov[j].t_y$) respectively. Similar symbols can be specified for an inner boundary except that we change the symbol ov to iv.

The proposed boundary region location algorithm consists of three steps. In the first step, we search for the v feature vectors which are visible from one

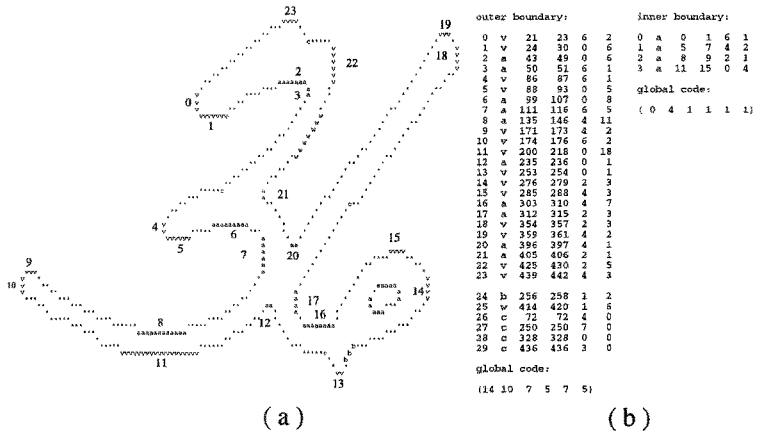

Fig. 1. Boundary smoothing and feature extraction algorithms: (a) the smoothed boundary with extracted feature vectors; (b) the codes for the feature vectors.

of the four (left, right, top, and bottom) sides, and label them as l, r, u, and d, respectively. If the jth v vector satisfies the following conditions, then it is labeled as l (left profile feature vector): (a) $ov[j].dir = 6$; (b) $ov[j].s_x < 2w/3$; (c) $ov[j].s_x = min$ for the scan line $y = ov[j].s_y$ and $ov[j].t_x = min$ for the scan line $y = ov[j].t_y$.

Similar to the above labeling methods, we can label r, u and d vectors in the top and bottom profiles. The candidate regions in the lower profile are then the mountain regions between two d vectors, and the candidate regions in the upper profile are the valley regions between two u vectors. Figure 2(a) shows the l, r, u, and d vectors, the mountain region and the valley region for Fig. 1(a). For comparison, Fig. 2(b) shows the mountain regions and valley regions for Fig. 1(a) using the methods in [8], where the mountain and valley regions are denoted as "M" and "Y" respectively. We can see that the new method is successful in excluding the redundant regions.

However, there are still redundant regions for some images as shown in Fig. 3(a), where the valley region between two u vectors in digit "4" and the mountain regions between two d vectors in digits "4" and "9" are redundant. Therefore, the candidate regions need to be reduced further. In the second step, the specific models of some numerals are constructed according to the feature vectors of the individual numerals. If the corresponding feature vectors of an input boundary image match the feature vectors of a model, one or more candidate regions can be eliminated. At the same time, the information about the specific (left or right) numeral is recorded and passed to the separation procedure.

The redundant regions caused by the left numeral are mainly due to numerals "2", "3", "4", "5", and "6". The redundant regions caused by the right numeral are mainly due to numerals "2", "4", "6", "8", and "9". We have constructed 5 models for the left numeral and 9 models for the right numeral which are shown in Figs. 4 and 5 respectively, where a main feature vector is labeled by its type

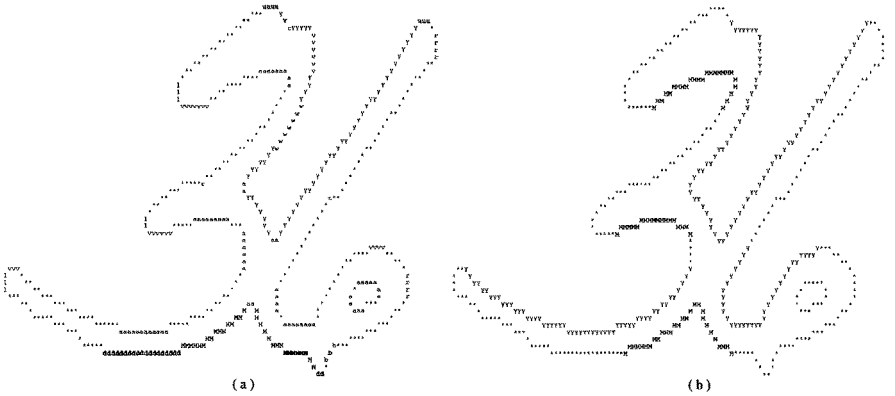

Fig. 2. Comparison of the located mountain and valley regions: (a) the proposed algorithm; (b) the method by Strathy, *et al.*

and direction. The dotted arrow represents a possible pair of v and a vectors, and the dashed arrow represents the link position to the other numeral. The dotted loop indicates that the loop may be filled up.

The models are constructed according to the possible writing styles of the numerals and the possible feature vectors. To match a model, we check the type and direction of each main feature vector, where the leftmost l vector is chosen as the starting feature vector. The geometrical positions of the main feature vectors are specified approximately.

When a model in Figs. 4 and 5 is matched, the mountain region between the two d vectors and the valley region between the two u vectors are cancelled. For example, both the left numeral and the right numeral in Fig. 3(a) are matched with the models, and therefore the two mountain regions and the one valley region are cancelled. Next, the $(a, 0)$ vector on the lower profile and the $(a, 4)$ vector on the upper profile are relabeled. If they are in the cancelled regions, they are labeled using the symbol of the numeral. Otherwise, they are labeled as $(p, 0)$ and $(q, 0)$ respectively. Figure 3(b) shows the result for Fig. 3(a).

In the final step, we record the region information of the image, which is described as follows:

$$ig = (nt, ny, np, nq, ni) \tag{3}$$

where nt and ny denote the number of mountain and valley regions respectively, np and nq denote the number of p and q vectors respectively, and ni is the number of loop regions.

For a mountain region or a valley region, 9 elements are recorded as follows:

$$rg[i] = (rg^{(i)}[0], rg^{(i)}[1], rg^{(i)}[2], rg^{(i)}[3], rg^{(i)}[4], rg^{(i)}[5], rg^{(i)}[6], rg^{(i)}[7], rg^{(i)}[8]) \tag{4}$$

where i is the index of the region, $rg^{(i)}[0]$ and $rg^{(i)}[1]$ are the indices of the starting and end main feature vectors in the boundary respectively, $rg^{(i)}[2]$ and $rg^{(i)}[3]$ are the numbers of v and a (including p or q) vectors in the region

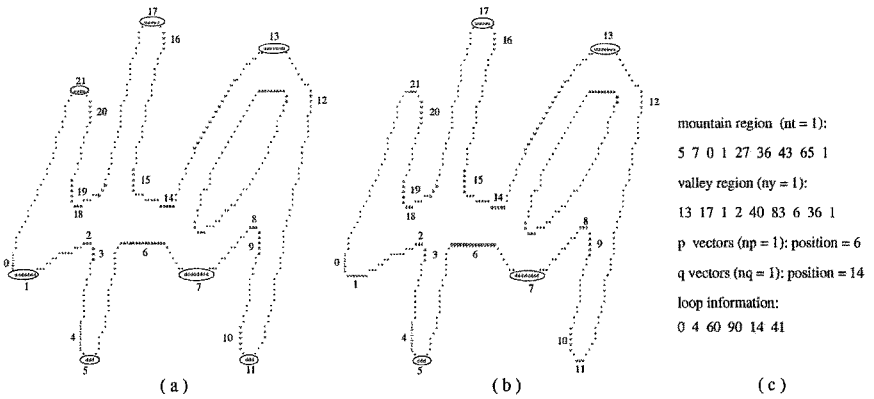

Fig. 3. Further reduction of redundant regions: (a) boundary with redundant mountain and valley regions; (b) boundary with redundant mountain and valley regions being cancelled; (c) region information.

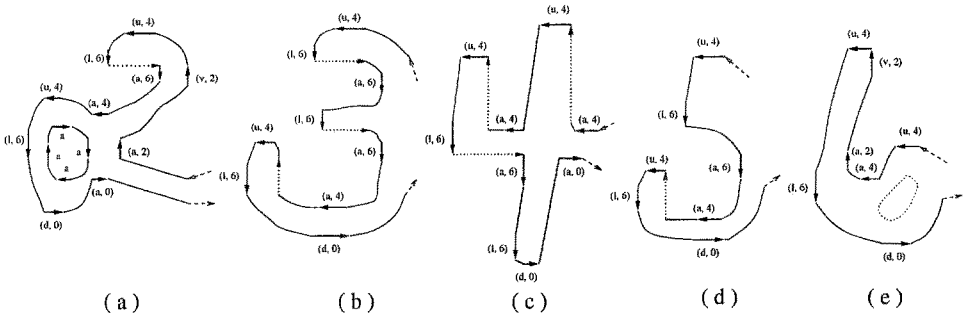

Fig. 4. Models for the left digit in the touching pairs.

respectively, $rg^{(i)}[4]$ and $rg^{(i)}[5]$ are the minimum and maximum x coordinates of the main feature vectors in the region respectively, $rg^{(i)}[6]$ and $rg^{(i)}[7]$ are the minimum and maximum y coordinates of the main feature vectors in the region respectively, and $rg^{(i)}[8]$ is the number of p (q) vectors in the region.

For the calculation of the minimum and maximum x and y coordinates, the end point of the starting feature vector and the starting point of the end feature vector are used. For other main feature vectors, we can use either the starting point or the end point of each vector.

For a loop region, we record the following information:

$$lp[i] = (lp^{(i)}[0], lp^{(i)}[1], lp^{(i)}[2], lp^{(i)}[3], lp^{(i)}[4], lp^{(i)}[5]) \tag{5}$$

where i is the index of the loop, $lp^{(i)}[0]$ and $lp^{(i)}[1]$ are the numbers of v and a vectors in the region respectively, $lp^{(i)}[2]$ and $lp^{(i)}[3]$ are the minimum and the maximum x coordinates of the main feature vectors in the region respectively, $lp^{(i)}[4]$ and $lp^{(i)}[5]$ are the minimum and the maximum y coordinates of the main feature vectors in the region respectively.

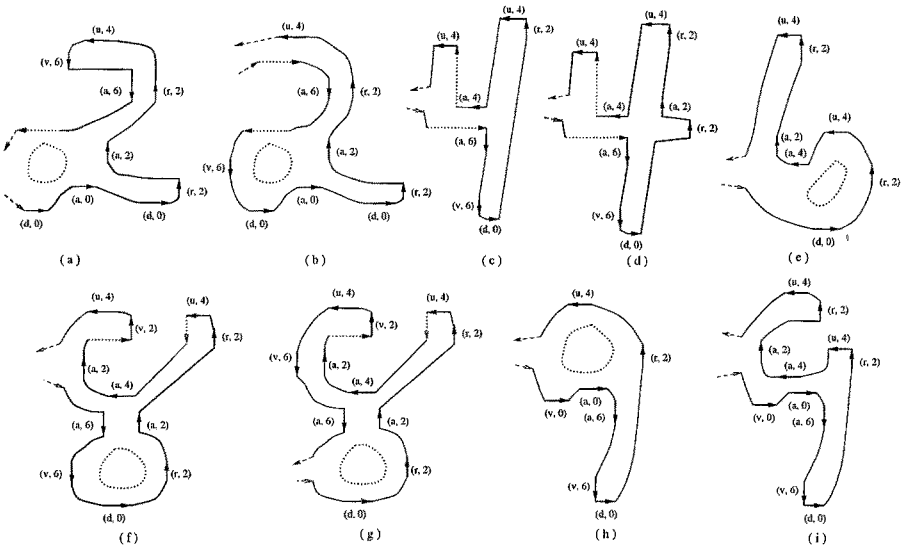

Fig. 5. Models for the right digit in the touching pairs.

If there are two or more loops, they are sorted. The loops are first sorted into two groups by their x coordinates: $x_{min} < w/2$ for the first group and $x_{min} \geq w/2$ for the second group. If there are still more than one loops in a group, they are sorted again by their y coordinates: $y_{min} < h/2$ or $y_{min} \geq h/2$.

For the image in Fig. 3(b), the redundant mountain regions and valley regions are excluded and there is only one candidate mountain region and one candidate valley region. The corresponding region information is shown in Fig. 3(c).

4 Experimental Results

After the mountain regions and the valley regions have been determined, a partitioning path will be constructed from a point in the mountain regions and a point in the valley regions. The partitioning path can be determined by analyzing the p and q feature vectors. First, the numeral strings are classified into 8 subclasses according to (nt ny np nq). Second, a number of models are constructed to deal with some common touching patterns in each subclass. When no model information can be used, we specify a small number of candidate points and use a recognition algorithm to verify the partition by "trial and error". Figure 6 shows some examples, where all the partitioning paths attempted by the algorithm are shown.

The data used in the experiment are two-digit strings extracted from the NIST database. The training group consists of 1,200 images, and the testing group consists of 3,355 images. The success rate is 89.66% when both of the separated digits are recognized correctly, of which 84.4% pairs are partitioned in the first attempt. There are another 6.02% pairs where a correct partitioning

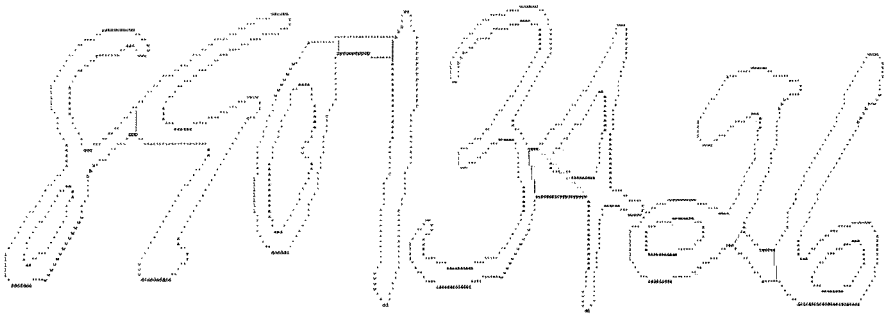

Fig. 6. Examples of the resulting partitioning paths.

path has been located, but one or two digits have been rejected by the recognition algorithm.

The proposed algorithm can be extended to a numeral string of more than two characters, where we can match the first and last digits in the string each time. Figure 7 shows two examples, where the candidate segmentation regions are the regions between two "u" and two "d" vectors. The partitioning paths can be determined in a similar way as for two-digit strings.

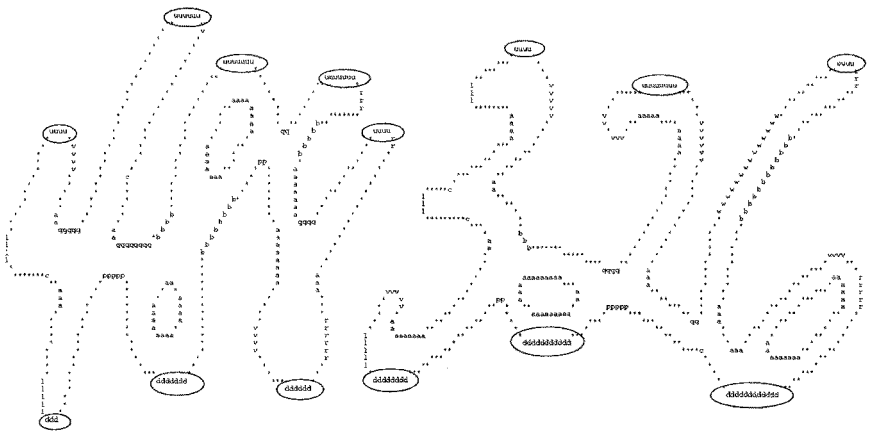

Fig. 7. Examples of three-digit string after the proposed algorithm.

5 Conclusion

Based on the boundary smoothing and feature extraction algorithms, an algorithm is proposed to locate the candidate partitioning regions. Various models are constructed based on the feature vectors to locate the candidate partitioning

regions. This method is more reliable and accurate than some other algorithms where no specific touching patterns are considered. Some models in this paper can be used for the leftmost and rightmost digits in a string of more than two characters directly. In addition, for the models where the full information about a digit is described, this digit can be recognized directly without the need to perform the boundary filling and recognition algorithms.

References

1. Casey, R. G., Lecolinet, E.: A survey of methods and strategies in character segmentation. IEEE Trans. Anal. and Mach. Intell. **18** (1996) 690-706
2. Lu Y., Shridhar, M.: Character segmentation in handwritten words – an overview. Pattern Recognition **29** (1996) 77-96
3. Casey, R. G., Nagy, G.: Recursive segmentation and classification of composite patterns. in *Proc. Sixth Int'l Conf. Pattern Recognition,* (1982) 1023-1026
4. Shridhar, M., Badreldin, A.: Recognition of isolated and simply connected handwritten numerals. Pattern Recognition **19** (1986) 1-12
5. Bayer T. A., Kreßel, U. H.-G.: Cut classification for segmentation. in *Proc. Int'l Document Analysis and Recognition,* (1993) 565-568
6. Westall, J. M., Narasimha, M. S.: Vertex directed segmentation of handwritten numerals. Pattern Recognition **26** (1993) 1473-1486
7. Fujisawa, H., Nakano, Y., Kuriro, K.: Segmentation methods for character recognition: From segmentation to document structure analysis. Proc. of the IEEE **80** (1992) 1079-1092
8. Strathy, N. W., Suen, C. Y., Krzyzak, A.: Segmentation of handwritten digits using contour features. in *Proc. 2nd Int'l Conf. Document Analysis and Recognition,* (1993) 577-580
9. Hu, J., Yu, D., Yan, H.: Algorithm for stroke width compensation of handwritten characters. Electronics Letters **32** (1996) 2221-2222
10. Yu, D., Yan, H.: An efficient algorithm for smoothing, linearization and detection of structural feature points of binary image contours. Pattern Recognition **30** (1997) 57-69
11. Hu, J., Yu, D., Yan, H.: A multiple point boundary smoothing algorithm. Pattern Recognition Letters, accepted.