Automatic Segmentation of Unconstrained Handwritten Numeral Strings

J. Sadri, C. Y. Suen, T. D. Bui
Center for Pattern Recognition and Machine Intelligence (CENPARMI)
Concordia University, 1455 de Maisonneuve Blvd. West, Montreal, Quebec, Canada H3G 1M8
Email: {j_sadri, suen, bui }@cs.concordia.ca

Abstract:

A new method of segmenting unconstrained handwritten numeral strings is proposed. It is based on the extracting of foreground and background features. In order to find foreground features for the first time an algorithm based on skeleton tracing is introduced. The skeleton of each connected component is traversed in clockwise and anti-clockwise directions, and intersection points which are visited in each traversal, are mapped on the outer contour to form foreground feature points. In order to find background features, another new algorithm is proposed. Considering vertical projections of top and bottom profiles, two background skeletons are found. After processing these two background skeletons, background feature points are extracted. Background and foreground feature points are assigned together to construct candidate segmentation paths. Finally each segmentation path is evaluated based on the properties of its left and right connected components. Our method can provide a list of good segmentation hypotheses for segmentation-based recognition systems. The NIST SD19 Database (Handwritten numeral strings) is used for evaluating of the method, and experiments show a very promising result.

Keywords: *Numeral String Segmentation, Numeral String Recognition, Foreground Features, Background Features, Skeleton Tracing*

1. Introduction

One of the most difficult problems in the area of Optical Character Recognition (OCR) is the segmentation and recognition of unconstrained handwritten numeral strings. It has been a popular topic of research for many years, and has many potential applications such as postal code reading, bank check processing, tax form reading, and recognition of many other special forms [1]. Generally numeral strings include isolated digits (with a lot of variations), touched, overlapped, and noisy or broken digits [2]. Some examples of these cases are shown in Figure 1. Segmentation of the connected numerals is one of the main challenges in handwritten numeral recognition systems [1].

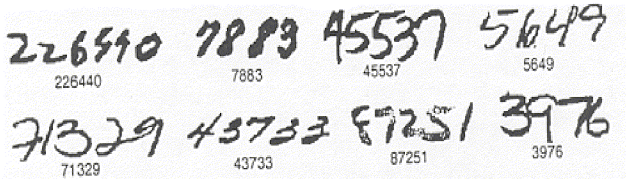

Figure 1. Some examples of numeral strings from NIST SD19 database.

For constructing segmentation paths many algorithms have been proposed in the past, and generally they can be classified into three categories. The first uses foreground features extracted from black pixels in the image [3], [4]. The second uses background features, extracted from white pixels in the image [5], [6], and the third category uses a combination of both these features [7]. As shown in the literature, approaches based on a combination of both foreground and background features have better results [7]. Also for the recognition of numeral strings, two approaches generally exist: segmentation-then-recognition, and segmentation-based-recognition [8]. In the former approach, the segmentation module provides a single sequence of hypotheses, where each subsequence should contain an isolated character, which is submitted to the recognizer. Since this technique does not use any contextual or recognition information for segmentation, it shows its limits rapidly when the correct segmentation does not fit the predefined rules of the segmentation module. In the later strategy, first the segmentation module provides a list of candidate segments, then each candidate segment is evaluated by the recognition module. Finally the list is post-processed by using contextual information [8].

Although many methods have been proposed for segmentation and recognition of connected handwritten numerals, much more improvement is still required to build robust systems for practical applications. Especially we need new methods to reduce the number of candidate

cutting paths to be tested, and decrease the rejection rate, while maintaining a good recognition performance.

In this paper, we present a new method of generating candidate segmentation paths for touched numeral strings, with unknown lengths. In this method segmentation is based on combining features from foreground and background of the image. The novelty of our method is that extracting features in the foreground is based on a new method called skeleton tracing, and in the background it is based on combining the vertical top and bottom projection profiles and their skeletons. Based on a combination of this information, and some global information from the string image, our algorithm tries to construct the best segmentation paths for separating the touched digits. Our proposed approach can be used for generating good hypotheses in segmentation-based recognition systems.

This paper is organized as follows: Section 2 presents the pre-processing steps, Section 3 presents our proposed method for feature extraction, Section 4 describes our algorithm for constructing segmentation paths, Section 5 presents the method for evaluating segmentation paths, Section 6 presents the experimental results, and finally we draw the conclusion.

2. Pre-processing

As seen in Figure 1, some images in our database need to be smoothed, by pre-processing steps. The method in [3] is used to remove the small spots of noise and to smooth the edges of the connected components. In the next step, to make the segmentation task easier and the segmentation paths as straight as much as possible, a method similar to [9] is used to correct the slant of each connected component. After slant correction, we may still need to smooth the edges of the images further, so another step of smoothing as mentioned earlier is taken (see Figure 2 for some results of pre-processing).

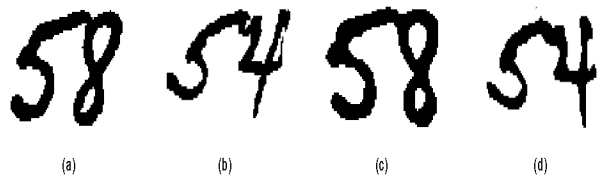

Figure 2. (a, and b) Original images. (c, and d) Slant corrected and smoothed images respectively.

3. Generating features points

3.1. Generating foreground features

To find feature points on the foreground (black pixels) of each connected component, a new algorithm based on skeleton tracing, is introduced. First by Zhang-Suen thinning algorithm [10], the skeleton of each connected component is extracted, (see Figure 3-b). Then on this skeleton, two points called starting and ending points (denoted by S, and E respectively) are found, as below. We start at the top-left corner of the skeleton image, scan each column of the pixels from the top going downward-starting from the leftmost column and proceeding to the right until we encounter a black pixel on the skeleton. We declare this pixel as "Starting point". In a similar way we start at the top-right corner of the skeleton, scan each column of the pixels from the top going downward-starting from the rightmost column, and proceeding to the left, until we encounter a black pixel. We declare this pixel as "Ending point", see S and E in Figure 3-b or 3-c. Then from the starting point (S), the skeleton is traversed in two different directions: first clockwise, and then anti-clockwise until both traverses reach the ending point (E), and stop. In figure 3-c, we define the traverse in the clockwise as top-skeleton, and the traverse in the anti-clockwise as bottom-skeleton. Actually top / bottom skeletons are subsets of pixels of the original (foreground) skeleton, which are visited during the traversals in clockwise / anti-clockwise directions. It is always possible to find these skeletons for any connected component uniquely. In the next paragraph, using these skeletons we explain how we can obtain important features for segmentation.

When the algorithm traverses the top / bottom skeletons, it keeps track of the intersection points (IPs) which are visited. IPs are points which have more than two connected branches (in Figure 3-b, they are denoted by ○). Some intersection points on top / bottom skeleton may be visited more than once. Corresponding to each visit of any intersection point in the skeletons, there is an angle which its bisector can be found (Figure 3-c). The intersections of these bisectors with the outer contour of the connected component are obtained, they are denoted by □ in Figure 3-d. These points form our foreground feature points.

In other words, during the tracing of the top or bottom skeleton, bisectors map intersection points on the outer contour to form feature points. As seen from Figure 3-d, if an intersection point is visited more than one time, it will have more than one corresponding feature point on the contour. However corresponding to each feature point (denoted by □) in Figure 3-d, there is just one intersection point on the skeleton. These feature points can be divided into two different subsets, one corresponding to top-skeleton and one corresponding to bottom-skeleton. These subsets are called top-foreground-features and bottom-foreground-features respectively. As shown in Figure 3-d each feature point, is very close to an intersection point on

the skeleton. Therefore these feature points can carry important information about the location of touching strokes, and segmentation regions.

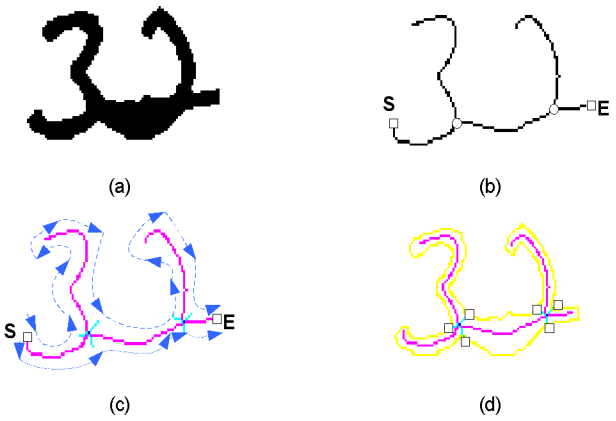

Figure 3. (a) Original image. (b) Original skeleton, starting point (S), ending point (E) are depicted by □. (c) From starting point (S), skeleton is traversed in two different directions (clockwise: dashed arrows, and anti-clockwise: dotted arrows) to the end point (E). (d) Mapping of intersection points on the outer contour by bisectors to form foreground-features (denoted by □).

3.2. Generating background features

Lu et. al. in [5], and Chen, and Wang in [7] used the skeleton of the background to extract features, and they used all white pixels of the background, including those inside the holes and inner parts of the digits. However as seen in Figure 4-c we do not consider all the white pixels as background. Here a new method of finding background features is introduced. First vertical top and bottom projection profiles of the connected component are found, as seen in Figures 4-d, and 4-e. Then skeletons of these images (black regions) are extracted, which are shown in Figures 4-f, 4-g, and they are called top-background-skeleton and bottom-background-skeleton respectively. These skeletons normally have long or extra branches, which sometimes do not lead us to reach to the cut regions, an example shown in Figure 5-a. We use a simple algorithm to remove these long or extra parts. As shown in Figure 5-b all parts of top-background-skeleton which are lower than the middle line, and all parts of bottom-background-skeleton which are higher than the middle line of the image are removed. Then on each background skeleton (top / bottom), end points (points which have just one black neighbor and they are denoted by □ in Figure 5-b) are found, and they are inserted into two different lists: one for end-points on top-background-skeleton, and another list for end-points on bottom-background-skeleton. The first and last end points in each list will not be considered. Compared to [5, and 7], our background feature points are more informative, and we also have a smaller number of feature points in the background, which makes decision making easier in the next step.

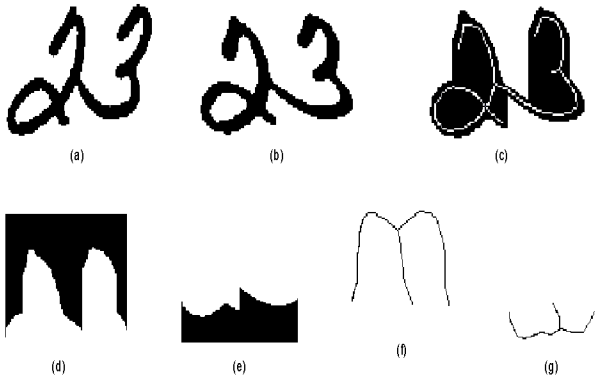

Figure 4. (a) Original image. (b) Pre-processed and slant corrected image. (c) Background region (White pixels outside of the black object). (d) Top projection profile. (e) Bottom projection profile. (f) Top-background-skeleton, (g) Bottom-background-skeleton.

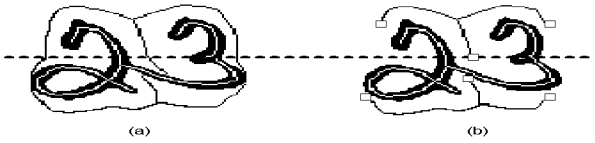

Figure 5: (a) Original top / bottom-background-skeletons. (b) Top / Bottom-background-skeletons after removing parts which are lower/higher than the middle line (Horizontal dashed line is the middle line, which divides the image into two parts with equal heights).

4. Constructing segmentation paths

In the previous steps (3.1, and 3.2), four different lists of features are obtained: two lists of foreground features (top/bottom-foreground-features), two lists of background features (top/bottom-background-features). Now two ways of searching are conducted: upward searching, and downward searching. If feature points in these lists from top to bottom, or from bottom to top, match with each other, they are assigned together to construct segmentation paths. Two feature points A, and B from two

different lists match with each other if condition (4-1) is met.

$$|x_A - x_B| \le \alpha \cdot H \quad , \quad \alpha \in [0.25, \quad 0.5], \qquad (4-1)$$

Here x_A , and x_B are the horizontal coordinates of A, and B respectively, α is a constant, which in our experiments we select it equal to 0.4, and H is the vertical height of the input connected component. The flowchart in Figure 6 shows the details, and an example is depicted in Figure 7.

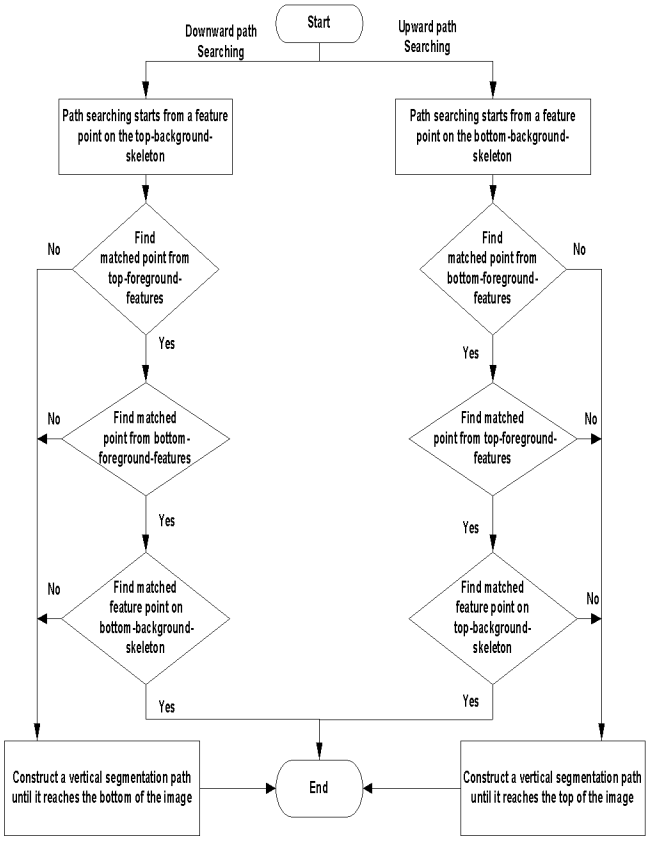

Figure 6. Flowchart of downward/ upward searching for constructing segmentation paths.

Figure 7. Feature points on the background , and foreground (from the top and bottom) are matched ,and assigned together to construct segmentation paths.

5. Evaluation of segmentation paths

Each segmentation path divides a connected component into two new connected components: one is on the left side and the other one is on the right side. Here each segmentation path is evaluated individually (without considering other candidate paths). Two constraints, on the left and right side connected components of each path, are applied as below. Paths which satisfy these two constraints are saved, and other paths are removed. (Figure 8 shows all the parameters which are used in these constraints)

- Overlap constraint: if horizontal overlap between the left and right side connected components of a segmentation path is above a threshold (see 5-1), the seg -mentation path is rejected. Here overlap is the horizontal overlap between the left and right side connected components, and W_l , W_r are their widths respectively (see Figure 8), and T_{ovl} is a threshold. We select T_{ovl} equal to 0.6 in our experiments.

$$\frac{Overlap}{Min\,(W_l, W_r)} \le T_{ovl} \qquad (5-1)$$

- Height constraint: if ratio of the heights of the left and right side components of a segmentation path, is below a threshold (see 5-2), then the segmentation path is rejected. Here H_l, and H_r are the heights of the left and right side connected components respectively (see Figure 8), and T_{Height} is a threshold. We select T_{Height} equal to 0.4 in our experiments.

$$\frac{Min\,(H_l, H_r)}{Max\,(H_l, H_r)} \ge T_{Height} \qquad (5-2)$$

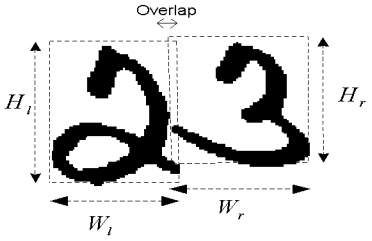

Figure 8. Illustration of all parameters which are used in evaluation of the segmentation paths.

6. Experimental results

To evaluate the proposed method, we have carried out some experiments on 835 images from NIST SD19 database. All images contain touching pairs of numeral digits, but the system does not know the length of the string as prior knowledge. After segmentation of the numerals by the system, we did a visual analysis and verified that in 95.3% of the cases, the best segmentation path is among the paths produced by the system. Because the system does not know the length of the input numeral string, for some images it produces more than one cutting path (see cases a, and b in Figure 9). In 88.6% of the cases the proposed method produces exactly one segmentation path, which is the correct segmentation path (see cases c, and d in Figure 9). In 2.1% of the cases the best segmentation path is not among those paths, produced by the system, so we consider these cases as errors (see cases e, and f). In 2.6% of the images the system cannot produce any segmentation path, which we consider those cases as rejected images (see cases g, and h). Most cases of error or rejection are due to high overlaps of connected digits. Figure 9 illustrates some results of our system.

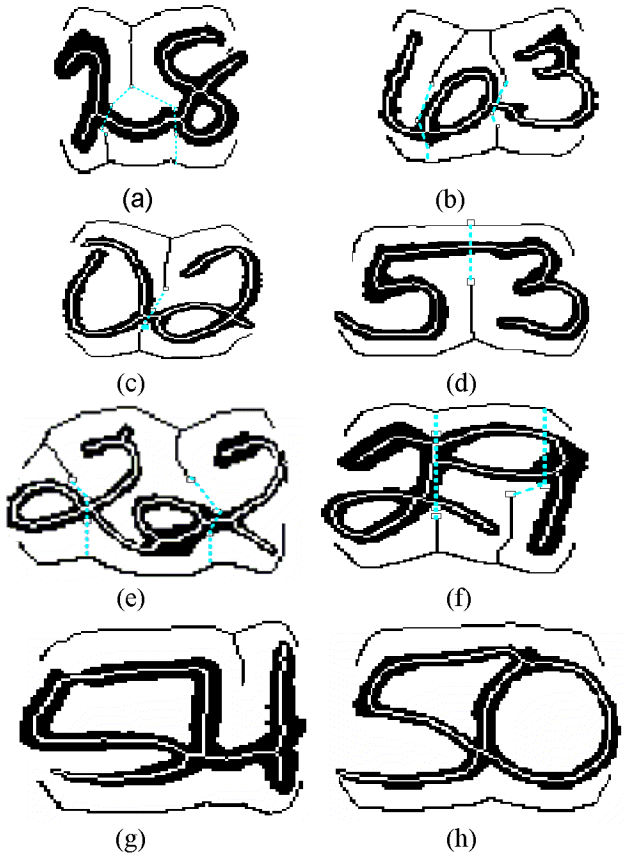

Figure 9. Some results of the system. (a-d) Segmentation candidates produced by the system ,and denoted by dashed lines. (e, and f) Cases of segmentation error. (g, and h) Cases of rejection.

7. Conclusions and future works

A new method of generating segmentation candidates for touching digits in numeral strings is presented. In order to find segmentation candidates, this method combines foreground and background features. New algorithms for finding feature points are presented. Due to the large variability of handwriting, we found that foreground and background features complement each other. We believe that, the method of skeleton tracing presented here will be helpful to the people in the pattern analysis community to solve other problems. Although in our system we did not use any prior knowledge about the length of the string, nor we included any recognition feedback for decision making, our experiments on touching digits from NIST SD19 Database show very encouraging results. In our future studies, we are going to add a recognition module (recognition feedback) to our system, to rank the multiple segmentation hypotheses by the recognition scores.

8. References

[1] U. Pal, A. Belaid, and Ch. Choisy, "Touching Numeral Segmentation Using Water Reservoir Concept," Pattern Recognition Letters, vol. 24, pp. 261-272, 2003.

[2] K.K. Kim, J.H. Kim, and C.Y. Suen, "Segmentation-Based Recognition of Handwritten Touching Pairs of Digits Using Structural Features," Pattern Recognition Letters, vol. 23, pp. 13-24, 2002.

[3] N.W. Strathy, C.Y. Suen, and A. Krzyzak, "Segmentation of Handwritten Digits Using Contour Features," Proc. Int. Conf. Document Analysis and Recognition, pp. 577-580, 1993.

[4] Z. Chi, M. Suters, and H. Yan, "Separation of Single–and Double Touching Handwitten Numeral Strings," Optical Eng., vol. 34, pp. 1,159-1,165,1995.

[5] Z. Lu, Z. Chi, W. Siu, and P. Shi, "A Background-Thinning-Based Approach, for Separating and Recognizing Connected Handwriting Digit Strings," Pattern Recognition, vol. 32, pp. 921-933, 1999.

[6] M. Cheriet, Y.S. Huang, and C.Y. Suen, "Background Region Based Algorithm for the Segmentation of Connected Digits," Proc. Int. Conf. Pattern Recognition, pp. 619-622, 1992.

[7] Y.K. Chen, and J.F. Wang, "Segmentation of Single or Multiple Touching Handwritten Numeral String Using Background and Foreground Analysis," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, pp. 1304-1317, 2000.

[8] L.S. Oliveira, R. Sabourin, F. Bortolozzi, and C.Y. Suen, "Automatic Segmentation of Handwritten Numerical Strings: A Recognition and Verification Strategy," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 11, pp. 1438-1454, Nov. 2002.

[9] T.Yamaguchi, Y.Nakano, M.Maruyama, H.,Miyao, and T.Hananoi, "Digit Classification on Signboards for Telephone Number Recognition", Proc. of Seventh Int. Conf. on Document Analysis and Recognition, pp. 359-363, 2003.

[10] T.Y. Zhang, and C.Y. Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns," Communication ACM, vol. 27, no. 3, pp. 236-239, 1984.