A Recognition Based System for Segmentation of Touching Handwritten Numeral Strings

Yun Lei, C.S. Liu, X.Q. Ding, Qiang Fu
State Key Laboratory of Intelligent Technology and Systems,
Tsinghua University, Beijing 100084, P.R. China
{leiyun, lcs, dxq, fuq}@ocrserv.ee.tsinghua.edu.cn

Abstract

A novel recognition-based system for segmentation of touching handwritten numeral strings is proposed. In this paper, we combine external contour analysis and projection analysis to find candidate segmentation points With internal contour analysis, the candidate segmentation points is utilized to determine the corresponding candidate segmentation lines with which the numeral string is over-segmented. Each subimage of the over-segmented string is defined as a ***fragment***. The combination of one or more adjacent fragments is defined as a ***clique***. Thus, each candidate segmentation result is composed of one or more cliques. Subsequently, all the candidate segmentation results are described in a probabilistic model, and a classifier is embedded to recognize each clique. Finally, with the maximum a posterior (MAP) criterion, the optimal segmentation result is selected from all candidate segmentation results. This scheme is effective and robust for both single and multiple touching numerals. Experiment results on collection of samples from NIST SD19 show that our system can achieve a correct rate of 97.72% without rejection, which compares favorably with those reported in the literature.

1. Introduction

Segmentation of handwritten numerals is a key step for automatic recognition of handwritten numeral strings since the segmentation results have great effect on the recognition rate. At present, many approaches have been proposed on segmentation of touching handwritten numeral strings. According to the methods by which candidate segmentation points are determined, all the popular segmentation schemes can be classified into two categories: topological structure based scheme [1, 2, 3] and contour based scheme [4, 5]. In the topological structure based scheme, thinning or skeleton of the foreground and background are usually implemented first. However, the instable performance of thinning and skeleton algorithm needs to be improved [10], and thinning or skeleton is also time-consuming. For these two reasons, the performance of topological structure based scheme is limited. In contour based scheme, the potential information in original strings is fully employed because the original string can be reconstructed from its contours without distortion. Moreover, the complexity of extraction and processing contours is very low, while the contour based scheme fails to find the segmentation points which exist in the smooth intervals of extracted contours.

In some cursive handwritten word recognition systems [6, 7], the touching characters are over-segmented firstly, and then all candidate segmentation results are ranked according to their respective evaluations given by a classifier and a lexicon. Finally, the optimal segmentation result can be obtained from the ranked list. By this strategy, the recognition rate has been able to be improved greatly. Inspiration drawn from this strategy leads to a novel recognition-based system for segmentation of touching handwritten numeral strings proposed in this paper. Our scheme combines the contour analysis and projection analysis to find candidate segmentation lines with which a numeral string is over-segmented. Subsequently, all the candidate segmentation results are described in a probabilistic model. With an imbedded classifier, the optimal segmentation result can be obtained according to the maximum a posterior (MAP) criterion.

The outline of the paper is as follows: Section 2 explicitly explains each block of the proposed system. Section 3 compares the experiment results of various schemes. Section 4 summarizes the work presented in this paper.

2. System overview

The flowchart of the proposed system is illustrated in **Fig. 1**. The input of our system is a binary image of a handwritten numeral string and the output are the sub-images resulting from segmentation.

2.1. Preprocessing

Preprocessing of the proposed system includes four steps. First, the height of the input image is normalized to 64 pixels and its width is scaled with the same factor. In practice, the input images contain noise and broken strokes. In order to eliminate the small isolated noise and

Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)
0-7695-2187-8/04 $20.00 © 2004 **IEEE**

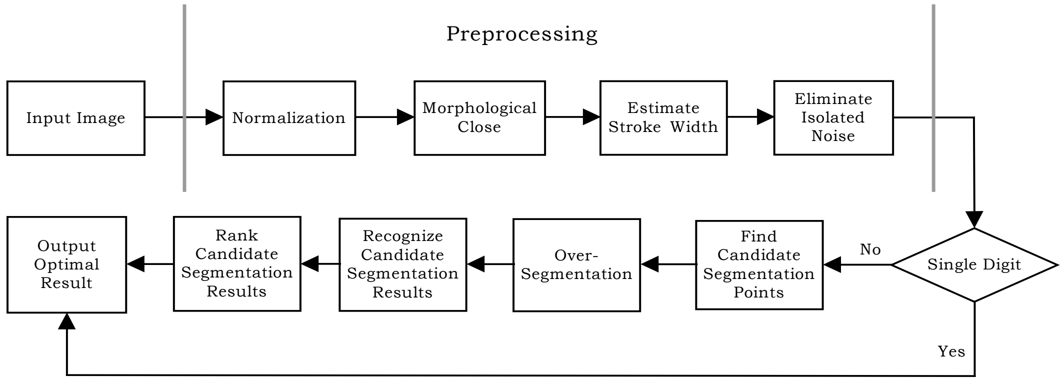

Figure 1. **Flowchart of our segmentation system**

connect the broken strokes, morphological close operation is implemented in the second step of preprocessing. With the methods documented in [8], we manage to estimate the average stroke width, denoted by λ, of the input numeral string in the third step and eliminate some isolated noise further in the fourth step of preprocessing.

2.2. Single-digit classifier

Before segmentation, the preprocessed image is input into a single-digit classifier to judge whether it is a single digit or not. The single-digit classifier employed in our system is the nested-subset classifier [9], which extracts the multi-scale directional element features of input samples and output their corresponding nested-subset Mahalanobis distances.

Let A denote the figure set $\{0,1,2,...,9\}$, and I denote the preprocessed image with height h and width w. Since the output of the embedded classifier is the distance, $d(I,W)$, between I and $W \in A$, a fuzzy membership function can be used to map $d(I,W)$ to the posterior possibility $P(W|I)$ according to (1).

$$U(I,W)=\begin{cases} e^{-\alpha d(I,W)}, & \alpha d(I,W)<1 \\ 0, & \alpha d(I,W)\geq 1 \end{cases} \tag{1}$$

If $$\max\{P(W|I)\simeq U(I,W) \mid W\in A\}\geq\theta \tag{2}$$

then I is considered as a single digit. Otherwise, I is considered as a numeral string. Where, α and θ are two positive constants and $0<\theta<1$. If the string only contains a single digit, it will be output directly. Otherwise, it will be sent to the following blocks to be segmented.

2.3. Candidate segmentation points

Usually, touching of adjacent numerals in a string brings about valleys and hills in the string's external contour and horizontal projection. Thus, contour analysis and projection analysis are combined in this paper to find the candidate segmentation points.

2.3.1. Contour analysis

Assume the origin of image coordinates is located at the top left point. Let $I(m,n)$ denote the value of pixel (m,n). If $I(m,n)=1$, pixel (m,n) is a foreground pixel. Otherwise, it is a background pixel.

Before contour analysis, the external and internal contours of a string are extracted. According to the leftmost and rightmost points in the string, the external contour can be divided into two parts: top contour and bottom contour. As shown in **Fig. 2(b)**, the leftmost and rightmost points of the string are marked by "□".

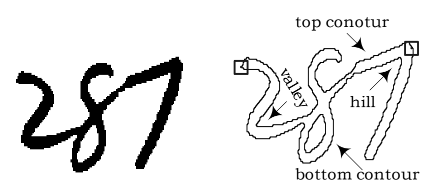

Figure 2. (a) **Touching string** (b) **Contours**

In most cases, segmentation points of a touching string appear in the corners of the top contours' valleys and the bottom contours' hills. Therefore, these kinds of corner points in the external contour are candidate segmentation points. In this paper, the top contour of a touching string is represented by a list of clockwise chain codes $T=\{(T_x(i),T_y(i))|1\leq i\leq L_t\}$, where $(T_x(i),T_y(i))$ is the i th node of T, and L_t is the length of the top contour. Similarly, the bottom contour is represented by a list of anti-clockwise chain codes $B=\{(B_x(i),B_y(i))|1\leq i\leq L_b\}$, where $(B_x(i),B_y(i))$ is the i th node of B, and L_b is the length of the bottom contour. The starting nodes of list T and list B are the leftmost pixel of the given touching string, as illustrated in **Fig. 2(b)**.

Next, in order to eliminate noise, the top and bottom contours are smoothed by the method documented in [7]. With the scheme in [11], all the corner points in the external contour are obtained, as depicted in **Fig. 3(a)**.

Figure 3. (a) (b)
(a): **Corner points** (b): **Candidate segmentation points from contour analysis**

Let the set of corner points in the top contour be $TC=\{(TC_x(i),TC_y(i))|1\le i\le n_{tc}\}$, where $(TC_x(i),TC_y(i))$ is the i th node of TC, and n_{tc} is the number of corner points in TC. Likewise, let the set of corner points in the bottom contour be $BC=\{(BC_x(i),BC_y(i))|1\le i\le n_{bc}\}$, where $(BC_x(i),BC_y(i))$ is the i th node of BC, and n_{bc} is the number of corner points in BC. $\forall\ 1\le i\le n_{tc}$, let t_i denote the index of pixel $(TC_x(i),TC_y(i))\in TC$ stored in list T. If $(TC_x(i),TC_y(i))$ is located in the valley of the top contour, it must satisfy

$$TC_y(i)=\max\{T_y(j)|t_i-\lambda\le j\le t_i+\lambda\} \qquad (3)$$

where λ is the estimated stroke width. Similarly, $\forall\ 1\le i\le n_{bc}$, let b_i denote the index of pixel $(BC_x(i),BC_y(i))\in BC$ stored in list B. If $(BC_x(i),BC_y(i))$ is located in the hill of the bottom contour, it must satisfy

$$BC_y(i)=\min\{B_y(j)|b_i-\lambda\le j\le b_i+\lambda\} \qquad (4)$$

For a string with stroke width larger than a single pixel, the nearest pixel under a segmentation point in the top contour must belong to foreground, and the nearest pixel above a segmentation point in the bottom contour also must belong to foreground. Therefore, $\forall\ 1\le i\le n_{tc}$, if $(TC_x(i),TC_y(i))\in TC$ is a candidate segmentation point, it must satisfy

$$I(TC_x(i),TC_y(i)+1)=1 \qquad (5)$$

$\forall\ 1\le i\le n_{bc}$, if $(BC_x(i),BC_y(i))\in BC$ is a candidate segmentation point, it must satisfy

$$I(BC_x(i),BC_y(i)-1)=1 \qquad (6)$$

Thus, according to $(3)\sim(6)$, all corner points which are not candidate segmentation points can be deleted from TC and BC. After deleting, all the remaining corner points are candidate segmentation points, as shown in **Fig. 3(b)**. Let all the remained corner points in TC denoted by $TS=\{(TS_x(i),TS_y(i))|1\le i\le n_t\}$, where $(TS_x(i),TS_y(i))$ is the i th candidate segmentation point in TS, and n_t is the number of points in TS. Let all the remained corner points in BC denoted by $BS=\{(BS_x(i),BS_y(i))|1\le i\le n_b\}$, where $(BS_x(i),BS_y(i))$ is the i th candidate segmentation point, and n_b is the number of points in BS. The contour-based scheme can find all possible candidate segmentation points in most cases. However, it fails to find the segmentation points which exist in the smooth intervals of extracted contours. For example, in **Fig. 3(a)**, the contour between "8" and "7" goes smoothly, so the contour-based analysis fails to find the segmentation points between "8" and "7". To solve this problem, an adaptive projection analysis is proposed in this paper.

2.3.2. Projection analysis

Let $P_x(j)$ be the horizontal projection of I, where $1\le j\le w$ and w is the width of I. With the estimated average stroke width λ, an adaptive projection-based algorithm is stated as follows:

Step 1: Let $j\leftarrow 0$, $k_t\leftarrow 0$, $k_b\leftarrow 0$.

Step 2: Let $j\leftarrow j+1$. If $\lambda\le j\le w-\lambda$, then go to **Step 3**. Otherwise, exit.

Step 3: If $(P_x(j)<2\lambda)\cap((P_x(j-1)\ge 2\lambda)\cup(P_x(j+1)\ge 2\lambda))$, then go to **Step 4**. Otherwise, go to **Step 2**.

Step 4: $k_t\leftarrow \arg\max\limits_{1\le i\le L_t}\{T_y(i)|T_x(i)=j\}$,
$k_b\leftarrow \arg\min\limits_{1\le i\le L_b}\{B_y(i)|B_x(i)=j\}$.

Step 5: since $\forall\ 1\le i\le n_{tc}$, t_i denote the index of pixel $(TS_x(i),TS_y(i))\in TS$ stored in list T, let $k_t^1=\arg\min\limits_{t_i}\{|t_i-k_t|\}$. If $|k_t^1-k_t|>2\lambda$, then add pixel $(T_x(k_t),T_y(k_t))$ into TS, and $n_t\leftarrow n_t+1$

Step 6: since $\forall\ 1\le i\le n_{bc}$, b_i denote the index of pixel $(BC_x(i),BC_y(i))\in BC$ stored in list B, then let $k_b^1=\arg\min\limits_{b_i}\{|b_i-k_b|\}$. If $|k_b^1-k_b|>2\lambda$, then add pixel $(B_x(k_b),B_y(k_b))$ into BS, and $n_b\leftarrow n_b+1$.

Step 7: Go to **Step 2**.

2.4. Candidate segmentation lines

For determining a segmentation line l, its two end-points should be found firstly. If the straight line connecting the coupled end-points does not intersect any internal contour in image I, then l is the straight line.

Otherwise, l is a piecewise line. The method with which l is determined will be stated in details in the latter part of this section. The approach to finding coupled end-points of a candidate segmentation line is as follows: $\forall\ 1 \le i \le n_t$, the corresponding end-point of the segmentation line determined by the candidate segmentation point $\left(TS_x(i), TS_y(i)\right) \in TS$ is located in the bottom contour. Let $c_t = \arg\min_j \left\{ B_y(j) \middle| B_x(j) = TS_x(i) \right\}$, where $\left(B_x(j), B_y(j)\right) \in B$, $1 \le j \le L_b$. Then $\left(B_x(k_t), B_y(k_t)\right)$ is considered as the coupled end-point of $\left(TS_x(i), TS_y(i)\right)$, where

$$k_t = \arg\min_{|j-c_t| \le 4\lambda} \left\{ \left[TS_x(i) - B_x(j)\right]^2 + \left[TS_y(i) - B_y(j)\right]^2 \right\} \quad (7)$$

So, add point $\left(B_x(k_t), B_y(k_t)\right)$ into BS. Likewise, $\forall\ 1 \le i \le n_b$, the corresponding end-point of the segmentation line determined by $\left(BS_x(i), BS_y(i)\right) \in BS$ is located in the top contour.

Let $c_b = \arg\max_j \left\{ T_y(j) \middle| T_x(j) = BS_x(i) \right\}$, where $\left(T_x(j), T_y(j)\right) \in T$ and $1 \le j \le L_t$. Then $\left(T_x(k_b), T_y(k_b)\right)$ is considered as the coupled end-point of $\left(BS_x(i), BS_y(i)\right)$, where

$$k_b = \arg\min_{|j-c_b| \le 4\lambda} \left\{ \left[BS_x(i) - T_x(j)\right]^2 + \left[BS_y(i) - T_y(j)\right]^2 \right\} \quad (8)$$

So, add point $\left(T_x(k_b), T_y(k_b)\right)$ into TS. Sort all the points among TS, in ascending order, according to their corresponding indexes in list T, and let $S_t = \left\{ (x_i, y_i) \middle| 1 \le i \le n_t \right\}$ represent the sorted result of TS. Similarly, sort all the points in BS, in ascending order, according to their indexes in list B, and let $S_b = \left\{ (u_i, v_i) \middle| 1 \le i \le n_b \right\}$ represent the sorted result of BS. Obviously, n_t is equal to n_b.

Therefore, $\forall\ 1 \le i \le n_t$, $(x_i, y_i) \in S_t$ and $(u_i, v_i) \in S_b$ are the coupled end-points of the i th candidate segmentation line.

Let $Q_i = (x_i, y_i) \in S_t$ and $Q_i^{'} = (u_i, v_i) \in S_b$, where $1 \le i \le M = n_t = n_b$. If line segment $\overline{Q_i Q_i^{'}}$ does not intersect any internal contour in I, then the i th candidate segmentation line is $\overline{Q_i Q_i^{'}}$. Otherwise, let C_i represent the set composed of all corner points in the internal contours that intersect $\overline{Q_i Q_i^{'}}$. Let C_i^1 represent the corner point, among C_i, which is nearest to Q_i, and C_i^2 represent the corner point, among C_i, which is nearest to $Q_i^{'}$. Then the i th candidate segmentation line is the piecewise line composed of $\overline{Q_i C_i^1}$, $\overline{C_i^1 C_i^2}$ and $\overline{C_i^2 Q_i^{'}}$.

According to the positions of all candidate segmentation lines, they can be denoted by $l_1, l_2, ..., l_M$ in left-to-right order, and let $L_0 = \left\{ l_1, l_2, ..., l_M \right\}$. $\forall\ 1 \le i \le M-1$, let $t^{(i)}$ be the index of (x_i, y_i) in list T, where $(x_i, y_i) \in S_t$ is the upper end-point of l_i. Let $b^{(i)}$ be the index of (u_i, v_i) in list B, where $(u_i, v_i) \in S_b$ is the lower end-point of l_i. If $\left| t^{(i)} - t^{(i+1)} \right| \le 2\lambda$ and $\left| b^{(i)} - b^{(i+1)} \right| \le 2\lambda$, then the longer line of l_i and l_{i+1} is deleted from L_0. **Fig. 4** shows all candidate segmentation lines of the numeral string in **Fig.2(a)**.

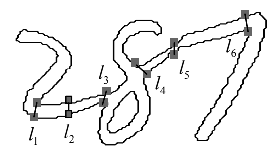

Figure 4. **Candidate segmentation lines**

2.5. Optimal segmentation result

Assume image I describes a numeral string Z which is composed of N digits $Z_1, Z_2, ..., Z_N$, where N and $Z_1, Z_2, ..., Z_N$ are random variables. Assume I is segmented into N subimages, $H_1, H_2, ..., H_N$ by method L, where L is the set of some segmentation lines. In our recognition-based scheme, the segmentation result with the maximum recognition probability is considered as the optimal segmentation result. According to MAP criterion, the optimal segmentation method L satisfies

$$\hat{L} = \arg\max_L \left\{ \max_Z \left[P\left(Z, L \middle| I\right) \right] \right\} \quad (9)$$

Since $P\left(Z, L \middle| I\right) = P\left(Z \middle| L, I\right) P\left(L \middle| I\right)$, we have

$$\hat{L} = \arg\max_L \left\{ \max_Z \left[P\left(Z \middle| L, I\right) P\left(L \middle| I\right) \right] \right\} \quad (10)$$

Assume that $L = \left\{ l^{(1)}, l^{(2)}, ..., l^{(N-1)} \right\}$ contains $N-1$ lines, then

$$\begin{aligned} P\left(L \middle| I\right) &= P\left(N, l^{(1)}, l^{(2)}, ..., l^{(N-1)} \middle| I\right) \\ &= P\left(N \middle| I\right) P\left(l^{(1)}, l^{(2)}, ..., l^{(N-1)} \middle| N, I\right) \quad (11) \end{aligned}$$

Since M segmentation lines have been obtained from contour analysis and projection in the preceding part of this section, image I can be over-segmented into $M+1$ subimages with $L_0 = \left\{ l_1, l_2, ..., l_M \right\}$, as shown in **Fig. 4**. Each of these subimages is defined as a *fragment*, and each combination of one or more adjacent fragments is defined as a *clique*. Thus, $M+1$ fragments can produce $(M+1)(M+2)/2$ cliques. Define $L^{(k)}$ as the k th subset of L_0, and L_0 has 2^M different subsets, denoted by $L^{(1)}, L^{(2)}, ..., L^{(M_0)}$, where $1 \le k \le 2^M = M_0$. Since different subsets of L_0 can segment I into different subimages, there exists M_0 different candidate segmentation results. Define

I_k as the k th segmentation result of I by $L^{(k)}$ and I_k is composed of n_k cliques, $E_1, E_2, ..., E_{n_k}$, where $1 \le n_k \le M+1$. According to (10) , the optimal segmentation is

$$\hat{L} = \arg\max_{L} \left\{ \max_{Z} \left[P\left(Z \middle| L, I\right) P\left(L \middle| I\right) \right] \right\}$$
$$= \arg\max_{L^{(k)} \in SA} \left\{ \max_{Z} \left[P\left(Z \middle| L = L^{(k)}, I\right) P\left(L = L^{(K)} \middle| I\right) \right] \right\} \qquad (12)$$

where $SA = \left\{ L^{(1)}, L^{(2)}, ..., L^{(M_0)} \right\}$ and $1 \le k \le M_0$ .Given image I and segmentation method L , $H_1, H_2, ..., H_N$ are N isolated cliques resulting from segmentation. Thus, they are all independent random variables when given I and L . Then,

$$P\left(Z \middle| L = L^{(k)}, I\right) = P\left(Z_1, ..., Z_N \middle| H_1 = E_1, ..., H_N = E_N, N = n_k\right)$$
$$= \prod_{i=1}^{n_k} P\left(Z_i \middle| H_i = E_i\right) \qquad (13)$$

With (1) , expression (13) can be written as

$$P\left(Z \middle| L = L^{(k)}, I\right) \simeq \prod_{i=1}^{n_k} U\left(E_i, Z_i\right) \qquad (14)$$

In (11) , $P\left(N \middle| I\right)$ is the probability of existing N digits in string Z when image I is given, and $P\left(l^{(1)}, l^{(2)}, ..., l^{(N-1)} \middle| N, I\right)$ is the co-occurrence probability of segmentation lines $l^{(1)}, l^{(2)}, ..., l^{(N-1)}$ when I and N are given. In essence, $P\left(l^{(1)}, l^{(2)}, ..., l^{(N-1)} \middle| N, I\right)$ depicts the probability of each writing style produced by the corresponding segmentation method, where the writing style means the position and size properties of each numeral pattern in image I . In this paper, fuzzy membership functions are incorporated to estimate $P\left(N \middle| I\right)$ and $P\left(l^{(1)}, l^{(2)}, ..., l^{(N-1)} \middle| N, I\right)$. If the width to height ratio of a handwritten digit is 1, the most likely number of digits in image I is $\frac{w}{h}$. Thus, $P\left(N \middle| I\right)$ can be approximated by

$$P\left(N = n_k \middle| I\right) \simeq e^{-\beta_1 \left| n_k - \frac{w}{h} \right|} \qquad (15)$$

Let h_i and w_i denote the height and width of E_i respectively, and g_i denote the vertical coordinate of the center of E_i . Usually, in the normalized image I , g_i is most likely near the middle of I , and h_i is most likely equal to h , and w_i is most likely equal to $\frac{w}{n_k}$. For simplification, $P\left(l^{(1)}, l^{(2)}, ..., l^{(N-1)} \middle| N, I\right)$ is approximately computed as

$$P\left(l^{(1)}, l^{(2)}, ..., l^{(N-1)} \middle| N, I\right)$$
$$\simeq \prod_{i=1}^{n_k} e^{-\beta_2 \left| h_i - h \right| - \beta_3 \left| w_i - \frac{w}{n_k} \right| - \beta_4 \left(g_i - \frac{h}{2} \right)^2} \qquad (16)$$

With (14) , (15) and (16) , expression (12) can be expended as

$$\hat{L} \simeq \arg\max_{L^{(k)} \in SA} \left\{ \max_{Z} \left[e^{-\beta_1 \left| n_k - \frac{w}{h} \right|} \prod_{i=1}^{n_k} U\left(E_i, Z_i\right) e^{-\beta_2 \left| h_i - h \right| - \beta_3 \left| w_i - \frac{w}{n_k} \right| - \beta_4 \left(g_i - \frac{h}{2} \right)^2} \right] \right\}$$
$$= \arg\max_{L^{(k)} \in SA} \left\{ e^{-\beta_1 \left| n_k - \frac{w}{h} \right|} \prod_{j=1}^{n_k} e^{-\beta_2 \left| h_j - h \right| - \beta_3 \left| w_j - \frac{w}{n_k} \right| - \beta_4 \left(g_j - \frac{h}{2} \right)^2} \prod_{i=1}^{n_k} \max\left[U\left(E_i, Z_i\right) \right] \right\} \qquad (17)$$

where $Z_i \in A = \{0, 1, ..., 9\}$, $1 \le k \le M_0$.

Expression (17) shows that searching for the optimal segmentation can be implemented by three steps: firstly, recognize all cliques; secondly, find the optimal recognition result of each clique; finally, find the optimal segmentation method maximizing (17) . $\forall\ 1 \le i \le n_k$, $\max\left[U\left(E_i, Z_i\right)\right] = 0$ means that the value of the objection function in (17) is zero. Thus, any searching path containing a clique satisfying $\max\left[U\left(E_i, Z_i\right)\right] = 0$ can be pruned. By this strategy, the time and space complexity of our system can be decreased greatly.

3. Experiment results and discussion

The samples in our experiments are all collected from NIST SD19 (including hsf_0, hsf_2, hsf_3 ,hsf_4, hsf_6 and hsf_7), and **Fig. 5** shows some typical samples in our collection. In order to obtain the parameters α , θ , β_1 , β_2 , β_3 and β_4 in the above probabilistic model, 500 two-digit-touching strings are used to optimize (17) . The testing results on 3359 other samples of two-digit-touching strings are listed in **Tab.1.** In this paper, the rejection rate and correct rate are defined as follows:

$$\text{rejection rate} = \frac{\text{\# of rejected images}}{\text{\# of all test images}} \times 100\%$$

$$\text{correct rate} = \frac{\text{\# of correctly segmented images}}{\text{\# of accepted images}} \times 100\%$$

The data in **Tab.1** shows that our scheme has achieved better performance than others in segmenting two-digit-touching strings. In addition, we also collect 525 samples of three-digit-touching strings, and the accuracy of segmenting has reached 93.33%. Although the accuracy of Oliveira et al [12]'s approach on 2385 three-digit strings from NIST SD19 has reached 95.38%, yet not all of the three digits in each of their three-digit samples are touched together. In contrast with their samples, the three digits in each string of our three-digit-touching samples are touched together. Thus, our collection of samples is more difficult for segmentation. If the conditional probability $P\left(L \middle| I\right)$ is not considered, the correct rate of segmenting two-digit strings is 96.99% and that of three-digit strings is 91.05%. Therefore, the proposed probabilistic model gives obvious improvement in segmentation performance. **Fig. 6** illustrates some examples of wrong segmentation, where “20” is segmented into “020”, “551” is segmented into “537”, and “52” is segmented into “02” because these false segmentations optimize the objection function in (17) .

4. Conclusions

In this article, we propose a novel recognition based system for segmentation of touching handwritten numeral strings. Our system combines contour analysis and projection analysis to find candidate segmentation lines. All candidate segmentation results and their corresponding writing styles are described in a probabilistic model. With an embedded single-digit classifier, the optimal segmentation result is found according to the MAP criterion. It should be noted that the single-digit classifier in our system is independent of segmentation. This is a good strategy for the fusion of segmentation and recognition. Thus, the improvement on the performance of the embedded classifier will bring about higher accuracy of the whole system. When searching for the optimal segmentation result, pruning algorithm is employed to greatly decrease the time and space complexity of our system for real-time requirement.

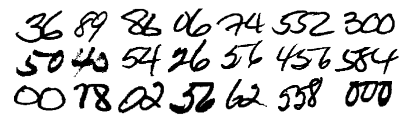

Figure 5. **Typical samples in our collection**

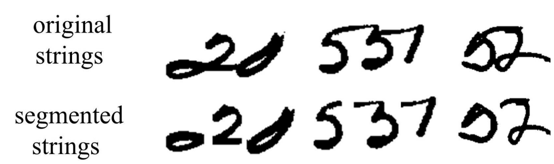

Figure 6. **Instances of wrong segmentation**

Table 1. **Performance of various methods**

Segmentation methods	Sample collection [sample number]	Correct rate	Rejection rate
Chen & Wang [1]	NIST[4178] + Unknown [322]	96%	7.8%
Pal, et al [2]	French Bank Check [2250]	94. 8%	3.4%
Oliveira, et al [12]	NIST SD19 [2370]	96.88%	0%
Our scheme	NIST SD19 [3359]	97.72%	0%

5. References

[1] Y. K. Chen and J. F. Wang. Segmentation of Single- or Multiple- Touching Handwritten Numeral String Using Background and Foreground Analysis. IEEE Trans on PAMI, 2000, Vol. 22, pp. 1304-1317.

[2] U. Pal, A. Belaïd, and Ch. Choisy. Touching Numeral Segmentation Using Water Reservoir Concept. Pattern Recognition Letters, 2003, Vol. 24, pp. 261-272.

[3] Z. Chi, M. Suters and H. Yan. Separation of Single- and Double-Touching Numeral Strings. Optical Engineering, 1995, Vol. 34, pp. 1159-1165.

[4] H. Fujisawa, Y. Nakano and K. Kurino. Segmentation Methods for Character Recognition : from Segmentation to Document Structure Analysis. Proc. IEEE, 1992, Vol. 80, pp. 1079 – 1092.

[5] N. W. Strathy, C. Y. Suen and A. Krzyzak. Segmentation of Handwritten Digits Using Contour Features. Proc. 2th ICDAR, Tsukuba, Japan, 1993, pp. 577-580.

[6] H. Yamada and Y. Nakano. Cursive Handwritten Word Recognition Using Multiple Segmentation Determined by Contour Analysis. IEICE Trans on Information & System, 1996, Vol. E79-D, pp. 464-470.

[7] G . Kim and V. Govindaraju. A Lexicon Driven Approach to Handwritten Word Recognition for Real-Time Applications. IEEE Trans on PAMI, 1997, Vol. 19, pp. 366-379.

[8] M. D. Garris. Component-Based Handprint Segmentation Using Adaptive Writing Style Model. NIST Internal Report 5843, 1996.

[9] J. Y. Zhang and X. Q. Ding. Multi-Scale Feature Extraction and Nested-Subset Classifier Design for High Accuracy Handwritten Character Recognition. Proc. 15th ICPR, Barcelona, Spain, 2000, pp. 581-584.

[10] L. Lam, S.W. Lee, and C. Y. Suen. Thinning Methodologies-a Comprehensive Survey. IEEE Trans on PAMI, 1992, Vol. 14, pp. 869-885.

[11] K .Jack and P. Stephen. Corner Detection for Chain Coded Curves. Pattern Recognition, 1995, Vol. 28, pp.843-852.

[12] L. S. Oliveira, R. Sabourin, F. Bortolozzi and C. Y. Suen. Automatic Recognition of Handwritten Numerical Strings: a Recognition and Verification Strategy. IEEE Trans on PAMI, 2002, Vol. 24, pp. 1438- 1454.

Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)
0-7695-2187-8/04 $20.00 © 2004 **IEEE**