Pergamon

Pattern Recognition, Vol. 31, No. 11, pp. 1601–1612, 1998

0031-3203/98 $19.00 + 0.00

PII: S0031-3203(98)00043-0

SPEEDING UP CHINESE CHARACTER RECOGNITION IN AN AUTOMATIC DOCUMENT READING SYSTEM

YI-HONG TSENG, CHI-CHANG KUO and HSI-JIAN LEE*

Department of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan 30050, R.O. C.

(*Received* 16 *April* 1997)

Abstract—In this paper, we present two techniques for speeding up character recognition. Our character recognition system, including the *candidate-cluster selection* and *modified branch-and-bound detail-matching* modules, is implemented using two statistical features: *crossing-counts* and *contour-direction counts*. In the training stage, we divide characters into different clusters by using reference characters. To have a very high recognition rate, the candidate-cluster selection module selects the top 60 clusters with minimal distances from among 300 predefined clusters. To further speed-up the recognition speed, we use a modified branch-and-bound algorithm in the detail-matching module. In the automatic document reading system, characters and punctuation marks are first extracted from printed document images and sorted according to their positions and the document orientation. The system then recognizes all printed Chinese characters between pairs of punctuation marks. The results are then spoken aloud by a speech-synthesis system. The character recognition system and the text-to-speech synthesis system are integrated in the Windows-based document reading system, which provides a user-friendly environment. © 1998 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved

Crossing-count features Contour-direction features Candidate-cluster selection
Branch-and-bound method Text-to-speech technique Automatic document reading system

1. INTRODUCTION

Techniques used for optical character recognition (OCR)[1,2] can be roughly grouped into two main categories: statistical methods and structural methods. Structural methods represent characters as compositions of structural units, usually called *primitives*. Techniques such as relaxation, and dynamic programming are applied to match input characters with all reference character models. Statistical methods usually consider characters as whole 2-D patterns, and extract sets of gray-scale-based features from them. A classifier is then used to match features extracted from the input patterns with features of all trained reference character models. Since the statistical features are usually highly dimensional and the number of characters is very large, the matching process takes considerable computation time. In this paper, a two-stage recognition system consisting of a candidate-cluster selection module and a modified branch-and-bound detail-matching module is proposed for speeding up character recognition.

Many methods, such as clustering and candidate selection, were used in different recognition engines to speed-up character matching. The clustering approach divided sample characters into several clusters and found the most possible results from the cluster that the input character fell in. The candidate selection approach pruned those characters having large matching distances and found the final one from the remaining candidates. Zhang *et al.*[3] proposed a clustering algorithm that minimized a cost metric. This algorithm can escape from local minima in clustering problems with large numbers of classes, a common hazard in conventional clustering methods. Tung *et al.*[4] introduced a multi-stage candidate selection module to speed-up the character recognition system. They extracted six statistical features and ordered them according to the reduction capability of each feature. The candidate selection used these features one by one to reduce the number of remaining candidate characters. We propose a new candidate-cluster selection module that divides 5401 Chinese characters into N clusters and selects n possible clusters for each input character. Characters in these selected clusters are recognized further by the detail-matching module.

A branch-and-bound algorithm is a tree search algorithm with hierarchical decomposition of a sample set of known patterns. The methods in references (5) and (6) focused on the decomposition of the sample set and design of rules for discarding less relevant nodes. In our modified branch-and-bound detail-matching module, we use feature stability to rearrange features and accumulated distances to reorder matching order. Feature decomposition and

*Author to whom correspondence should be addressed. E-mail: hjlee@csie.nctu.edu.tw.

node discarding were not significant in our detail-matching module.

Shuhi *et al.*[7] proposed a fast algorithm for the minimum distance classifier. Their algorithm omitted redundant calculations according to *Karhunen–Loeve expansion*. In the training stage, they found all eigenvalues and the corresponding eigenvectors from the variance–covariance matrix. The first k eigenvectors were used to transfer the first k feature dimensions of all prototypes z_i into the corresponding Karhunen–Loeve coefficients z'_{ij}, $i = 1, \ldots, n$ and $j = 1, \ldots, k$. In the recognition stage, the first k feature dimensions of an input pattern x were transferred into $x'_j, j = 1, \ldots, k$, and the prototypes were sorted in the order of the distance $|x'_1 - z'_{i1}|$. The accumulative distance $d^{(k)}(x, z_{N_i}) = \sqrt{\sum_{j=1}^{k} (x'_j - z'_{N_{ij}})^2}$ of the first k feature dimensions were calculated and the prototype z_{N_i} were omitted if $d^{(k)}(x, z_{N_i}) <$ the current minimum distance d_{cm}. Otherwise, the complete distance $d(x, z_{N_i}) = \sqrt{\sum_{j=1}^{n} (x'_j - z'_{N_{ij}})^2}$ was computed and compared with d_{cm}.

It is not easy to compute eigenvalues and eigenvectors from a high-dimensional matrix. In order to identify the more significant features, our modified branch-and-bound detail-matching module uses standard deviations, instead of the Karhunen–Loeve coefficients to rearrange all feature values of reference characters in the training phase. In the recognition phase, we first rearrange features extracted from input images according to the same sequence used in reference characters. To reduce the frequency of comparison between the accumulated distance and the upper bound, we check accumulated distances for every K features using the branch-and-bound algorithm.

A document image is generally composed of text blocks, graphic blocks and image blocks. Techniques for document analysis have been studied for many years.[8,9] In our automatic document-reading system, we assume that the input documents consist solely of text. Characters are first extracted from the document image by finding connected components and adjusted by applying component merging rules. The extracted characters are then used to determine the orientation of the document. We next obtain the arrangement order of characters from their positions, and the recognition engine recognizes those extracted characters in turn.

Among currently available communication media between humans and computers, speech is very natural and friendly. An automatic text-to-speech technique is used to generate fluent synthetic sounds based on input text. A text-to-speech system[10] will be very helpful in enabling blind persons to "read" documents. Hence, in this paper, we present a system that combines Chinese character recognition techniques and text-to-speech conversion in an automatic document-reading system.

The remainder of this paper is organized as follows. Section 2 describes features extracted from Chinese character images for use by our OCR system. Section 3 explains the character recognition system and the methods for speeding it up. Section 4 describes the flow and modules of the automatic document reading system, which consists of a character extraction process and a text-to-speech conversion module. Section 5 reports experimental results. Section 6 gives our concluding remarks.

2. EXTRACTION OF STATISTICAL FEATURES

In order to build a robust OCR system, we must extract significant features that tolerate variations among samples of the same character and differentiate variations among samples in different characters. We have evaluated some statistical features proposed in the literature: *stroke count features*,[11] *contour-pixel count features*,[12] *contour-direction features*,[13] *Oka's cellular features*,[14] *peripheral background area features*,[12] *crossig-count features*,[12] and *projection features*,[15] and selected two from among them to implement our OCR system.

After performance evaluation, the two features we selected are the *crossing-count feature (CCF)* and *contour-direction feature (CDF)*. In order to accommodate handwriting variations, a 2-D character image is first segmented non-uniformly into 8 strips in both the horizontal and vertical directions. These strips have the same numbers of black pixels, as shown in Fig. 1. The horizontal and vertical projections show the accumulated number of black points. From the intersections, we can obtain $64 (= 8 \times 8)$ sub-regions, in most of which different handwriting of certain characters has similar strokes. After segmenting the character images, we extract *CCFs* from each horizontal strip and vertical strip. Four uniform scan lines are defined in each strip, and the number of strokes, CC_k, intersecting each scan line k is counted. The feature f_i in strip i is the sum of the four numbers of intersected strokes. The dimension of this crossing-count feature is equal to 16 $(= 8 + 8)$. Formally, horizontal *CCFs* and vertical *CCFs*, denoted as $f_1, f_2, \ldots, f_8$ and $f_9, f_{10}, \ldots, f_{16}$, are defined as

$$f_i = \sum_{k=1}^{4} CC_{ik}, \quad i = 1, 2, \ldots, 16.$$

An example of extraction of cross count features (*CCFs*) is shown in Fig. 2.

The *contour-direction feature* (*CDF*) represents the number of contour points in four main directions: horizontal, vertical, diagonal and inverse diagonal. A 3×3 window defined over each black pixel, as shown in Fig. 3, represents feature patterns. There are 256 $(= 2^8)$ possible feature patterns, FP_1, FP_2, $\ldots, FP_{256}$. The direction feature value of each possible pattern can be precomputed and stored in an index table. Since the widths of strokes in handwritings differ greatly, only feature patterns located on the boundaries of strokes are considered. That is,

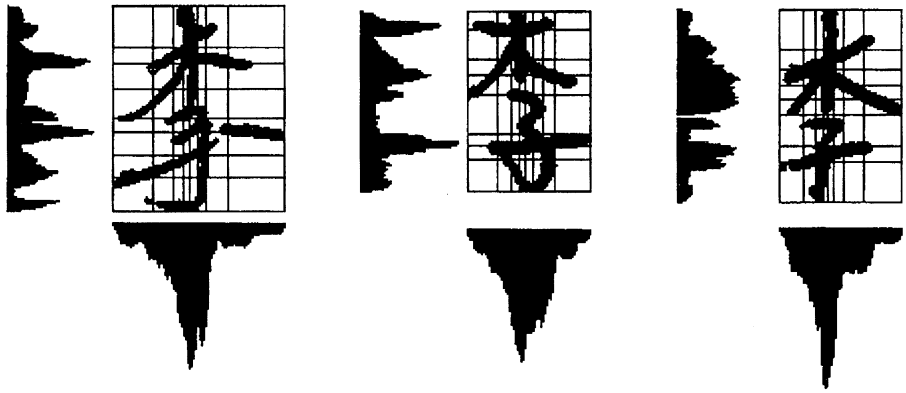

Fig. 1. Results of non-uniform segmentation of handwritten characters.

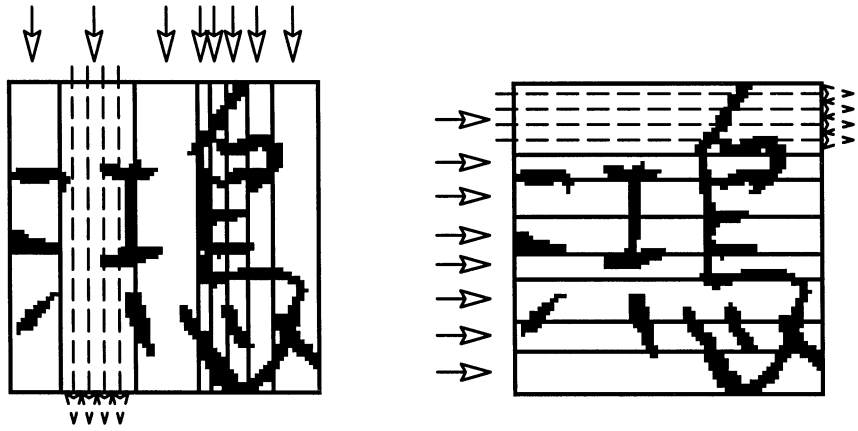

Fig. 2. Extraction of crossing-count features.

feature patterns are worthy noting if the middle black-pixels in Fig. 3 are contour-pixels of character images.

Let $a(x, y)$ represent the stroke-direction angle at the black pixel (x, y) with respect to the x-axis. We define $a(x, y) = \tan^{-1}(G_x/G_y)$, where G_x and G_y are derivatives on the x-axis and the y-axis, respectively. Based on the Sobel operator,[16] the derivatives are defined as

$$G_x = (z_6 + 2z_7 + z_8) - (z_1 + 2z_2 + z_3)$$

and

$$G_y = (z_3 + 2z_5 + z_8) - (z_1 + 2z_4 + z_6).$$

The ranges of stroke angles are from 0 to 180°. Since Chinese characters usually consist of horizontal strokes, vertical strokes, diagonal strokes and inverse diagonal strokes, the angle ranges are partitioned equally into four groups, G_1, G_2, G_3, and G_4, as shown in Fig. 4. The 256 feature patterns define over the middle black-pixel (x, y) can be classified into four categories, $\hat{FP}_1$ and $\hat{FP}_2$, $\hat{FP}_3$, and $\hat{FP}_4$, according to the angle $a(x, y)$. For each sub-region j, $a_{i,j}$ denotes

Z_1	Z_2	Z_3
Z_4	•	Z_5
Z_6	Z_7	Z_8

Fig. 3. Labels of a black pixel's neighbors.

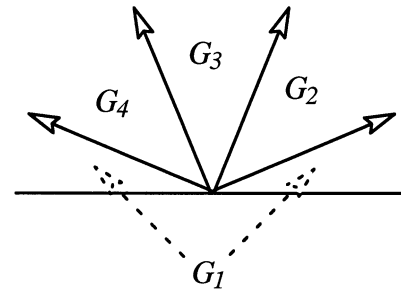

Fig. 4. Four angle groups.

the accumulation of feature patterns located in $\hat{FP}_i$. We calculate $a_{i,j}$ as follows:

```
For each sub-region j {
  For each feature pattern FP appears on the
    stroke boundary {
    classify FP into the category FP̂_i according to
      the stroke angle;
  increase a_{i,j}; }
}
```

where, $1 \leq i \leq 4$ and $1 \leq j \leq 8 \times 8$.

Let H and W denote the height and width of the character image, respectively. The perimeter of the character image is $2 \times (H + W)$. Because handwriting differs greatly in size, we normalize feature values by dividing them by the perimeter. The feature value is thus defined as $a_{i,j}/(H + W)$. Since there are four features in each sub-region, contour-direction features occur in a $8 \times 8 \times 4 = 256$-dimensional space.

3. CHARACTER RECOGNITION AND ITS SPEEDING UP

A Chinese character recognition system may take a long time to process feature vectors with high dimensionality, and handle large databases (over 5401 characters). This paper presents two new methods that reduce the number of candidate reference characters and the feature vector dimensionality quickly in two stages. The character recognition system, as shown in Fig. 5, contains two modules: candidate-cluster selection and modified branch-and-bound detail-matching. A candidate-cluster selection module first reduces the size of the candidate set from 5401 to a few hundreds by using simple statistical features. The reduced candidate set is then sent to the detail-matching module, in which we can prune incorrect candidate characters quickly by using parts of feature values dynamically.

3.1. Candidate-cluster selection module

Two approaches, clustering and candidate selection, for speeding-up character recognition have been proposed in the literature. The clustering approach divides sample characters into several clusters according to predefined features. The elements in the clusters may be exclusively disjoint or overlapping. After feature extraction, an input character is assigned into a specific cluster. Characters in this cluster were sent to the detail-matching module to find the correct result. The computation cost of a clustering module depends on the number of clusters. Thus, it is important to reduce the number of clusters for 5401 candidate Chinese characters. The candidate selection approach matches all characters to prune those characters having large matching distances. The size of the candidate set is reduced and the remaining candidates were then sent to the detail-matching module. Figure 6 shows the structure of a candidate selection module.

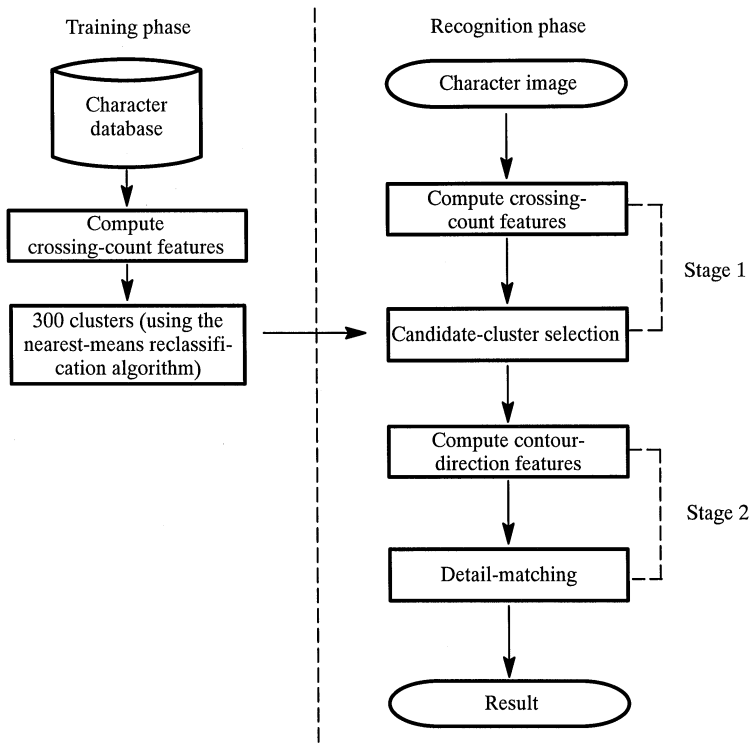

Fig. 5. Flow diagram of the character recognition system.

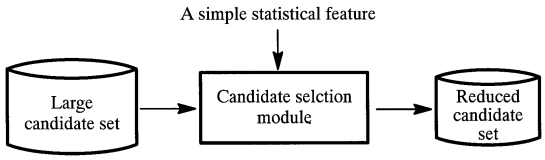

Fig. 6. Strucutre of a candidate reducing module.

Our candidate-cluster selection module divides 5401 Chinese characters into N clusters by the *nearest-mean reclassification algorithm*[17] in the training phase. About $5401/N$ characters were initially assigned to each cluster and the nearest-mean reclassification algorithm is used to adjust memberships and recompute the mean vectors of all clusters. We terminated the training phase when no character moved from one cluster to another.

During the recognition phase, the CCF feature vectors of input characters were compared with the mean vectors of N reference clusters to select candidate clusters. Since the hit rate, the probability of the correct character falls into the correct candidate cluster, affects detail-matching significantly, we selected the top n clusters with smaller distances to increase this probability. The unions of characters in these n clusters are taken as selected candidate characters. By adopting the candidate-cluster selection module, each input character need only be compared with N cluster centers instead of 5401 characters. Let N_{cand} denote the number of selected candidate characters in the candidate-cluster module. The computation cost of the character recognition system without the candidate-cluster selection module is $5401 \times Dim_{CCF+CDF}$, where $Dim_{CCF+CDF}$ denotes the total CCF and CDF feature dimensions. The computation cost of the character recognition system with candidate-cluster selection module becomes $N \times Dim_{CCF} + N_{cand} \times Dim_{CDF}$, where Dim_{CCF} and Dim_{CDF} denote the CCF and CDF feature dimension, respectively. In our system, $Dim_{CCF} = \frac{1}{17} Dim_{CDF+CDF}$. We need to select proper N and N_{cand} that can keep a high recognition rate and obtain a high speed-up ratio. These value are determined according to experimental results.

3.2. Metrics of distance

Let $x = (x_1, x_2, \ldots, x_{256})$ be the feature vector of an input pattern, $u_j = (u_{j1}, u_{j2}, \ldots, u_{j256})$ be the mean vector of the reference character j, and $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_{256})$ be the standard deviation along each feature. We evaluate the following five distances $d(x, u_j)$ for the 256-dimensional CDF between the input character and each of the 5401 handwritten Chinese characters.

- Minimum distance:

$$d(x, u_j) = \sum_{i=1}^{256} |x_i - u_{ji}|;$$

- Euclidean distance:

$$d(x, u_j) = \sum_{i=1}^{256} |x_i - u_{ji}|^2;$$

- Cross correlation distance:

$$d(x, u_j) = \frac{\langle x, u_j \rangle}{\|x\| \cdot \|u_j\|};$$

- Modified Mahalanobis distance:

$$d(x, u_j) = \sum_{i=1}^{256} \frac{(x_i - u_{ji})^2}{\sigma_i^2};$$

- L–Y distance:[18]

$$d(x, u_j) = \sum_{i=1}^{256} \left[\frac{(x_i - u_{ji})^2}{\sigma_i^2} + \log(\sigma_i^2) \right].$$

The recognition results are shown in Table 1. According to our experimental results, the method based on L–Y distance gives the best recognition results, either in top 1 or in top 10, while the method based on the minimum distance gives the worst results.

In the modified Mahalanobis distance, the Euclidean distance $(x_i, u_{ji})^2$ is refined according to the variance σ_i^2. The more variance σ_i^2 is, the smaller the distance $(x_i - u_{ji})^2$ will be. In the L–Y distance, the value of $\log(\sigma_i^2)$ is combined with the modified Mahalanobis distance. Since handwritten characters have a wide variety of writing styles among different writers, the variance σ_i^2 jis not very steady. The modified Mahalanobis distance and the L–Y distance are both affected by the value of σ_i^2. We replace the variance σ_i^2 by the standard deviation σ_i in the L–Y distance and evaluated the modified distance measure as follows:

$$M\_distance\text{: } d(x, u_j) = \sum_{i=1}^{256} \left[\frac{(x_i - u_{ji})^2}{\sigma_i} + c \times \log(\sigma_i) \right],$$

where c is a constant which will be set from experimental results. The recognition results using *M_distance* are shown as Table 2.

Table 1. Recognition results for 5401 handwritten Chinese characters by using five distance

Matching method	Recognition rate		
	Top 1	Top 3	Top 10
Minimum distance	79.72%	91.07%	96.11%
Euclidean distance	81.46%	91.92%	96.61%
Cross-correlation	81.13%	91.67%	96.22%
Modified Mahalanobis distance	83.70%	92.46%	96.04%
L-Y distance	86.44%	94.11%	97.00%

Table 2. Recognition results for 5401 handwritten Chinese characters by using *M_distance*

Matching method	Recognition rate		
	Top 1	Top 3	Top 10
M_distance	88.55%	96.39%	98.61%

3.3. Modified branch-and-bound detail-matching module

The candidate-cluster selection module reduces the candidate size significantly and keeps the hit-rate high. The detail-matching module uses a more complex statistical features to compare input character images with each character remaining in the candidate set. Since those statistical features have high dimensionality, considerable computation cost will be incurred calculating the distances between patterns. In our detail-matching module, we use the *CDF* whose feature dimensions are 256 to match input patterns with candidate characters. The distance between the input character X and the kth candidate character Z_k is defined as

$$d^{256}(X, Z_k) = \sum_{i=1}^{256} \left[\frac{(x_i - u_{ki})^2}{\sigma_{ki}} + c \times \log(\sigma_{ki}) \right],$$

where μ_{ki} and σ_{ki} are the mean and the standard deviation of the ith feature in the kth reference character, c is a constant and set from experimental results. We compare the feature vectors of input characters with those N_{cand} selected candidate characters.

We can also consider the detail-matching module as a means for minimizing a distance function. We can build a tree to represent the search problem; the number of terminal nodes is equal to the number of characters in the candidate set. The non-terminal node (i, j) in the tree denotes the jth feature of the ith candidate character and the only child of node (i, j) is node $(i, j + 1)$. Figure 7 shows the tree structure of the detail-matching module; it has a total of N_{cand} paths. We will traverse N_{cand} paths from root to terminal nodes to obtain N_{cand} distances and select the character with the minimum distance as the output.

In order to reduce the computation time required for traversing, a branch-and-bound method is proposed in this paper. Suppose that the minimum distance after traversing the first m paths is Min_{Dist}. When the accumulated distance in the $(m + 1)$th path is greater than Min_{Dist}, which is considered an upper bound, we can terminate the traversal for this reference character model. To make the branch-and-bound algorithm efficient, we need a good upper bound. If we can also find a good traversal order, we can find good solutions more quickly. Below, we present an algorithm that removes incorrect candidate characters quickly and thus reduces the detail-matching time.

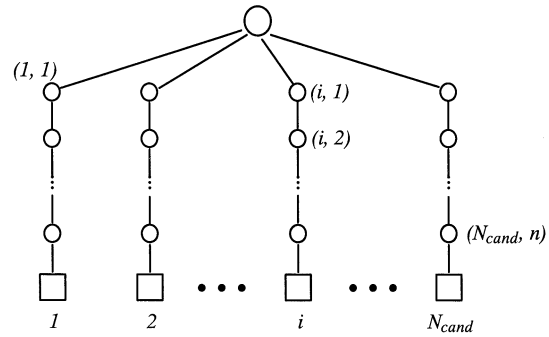

Fig. 7. Tree structure of the detail-matching module.

Let Z_m be the mean vector of the mth category. For the ith feature of 5401 character, we compute the total mean μ_i and the total variance σ_i^2 as follows:

$$\mu_i = \frac{1}{5401} \sum_{i=1}^{5401} Z_{mi}$$

$$\sigma_i^2 = \sum_{i=1}^{5401} Z_{mi} - \mu_i)(Z_{mi} - \mu_i)^{\mathrm{T}}$$

where $i = 1, 2, \ldots, n$.

In other words, the standard deviation σ_i represents the stability of the ith feature among the 5401 characters. Or say, the magnitude of σ_i represents the degree of difficulty involved in distinguishing a certain character from all the others. We rearrange the features in the descending order of standard deviations $\{\sigma_i\}$, i.e. $\sigma_{s_1} \geq \sigma_{s_2} \geq \cdots \geq \sigma_{s_n}$, and $\{s_1, s_2, \ldots, s_n\}$ is a permutation of the sequence $\{1, 2, \ldots, n\}$. According to this rearrangement, the more significant features that can distinguish among the 5401 characters more easily are placed ahead of the less significant ones. This rearrangement of features can prune incorrect characters more quickly when using the branch-and-bound technique.

In the detail-matching process, we first rearrange the features in the feature vectors of input character images according to the sequence $\{s_1, s_2, \ldots, s_n\}$ in the reference character set and obtain new feature vectors X. In order to reduce the frequency of comparison between the accumulated distance and the upper bound, we check accumulated distances per K features using our branch-and-bound method. We calculate the distance $d^P(X, Z_m)$ from the first P features of the input feature vector X and the mth reference feature vector Z_m, where $m = 1, 2, \ldots, 5401$, and then sort the 5401 distances in the ascending order. Let Z_l be the feature vector of the first character, which has a minimum distance of $d^P(X, Z_l)$. We then compute the distance $d^n(X, Z_l)$ from all n features. The distance $d^n(X, Z_l)$ is considered the upper bound in the branch-and-bound method, and is denoted as D_{cm}. We next calculate all distances $d^n(X, Z_m)$, where $m = 1, 2, \ldots, l - 1, l + 1, \ldots, 5401$, for the remaining features of all categories except the lth category. In

each calculation of $d^n(X, Z_m)$, we check the accumulative distance $d^{P+tK}(X, Z_m)$ for every K features, where and $t \in \mathbb{N}$ and $P + tK \leq n$. If the accumulative distance $d^{P+tK}(X, Z_m)$ exceeds the upper bound D_{cm} for a certain t, the mth candidate category is removed from the candidate character set. Otherwise, the upper bound D_{cm} is replaced by $D_{cm} = \min(D_{cm}, d^n(X, Z_m))$. The modified branch-and-bound algorithm adopted in our system is summarized as follows:

ALGORITHM: MODIFIED BRANCH-AND-BOUND FOR DETAILED MATCHING

Notation

m the number of Chinese character candidates.
n the feature vector dimensions.
Z_m the mean feature vector of candidate m.
σ_m the standard deviation vector of candidate m.

Training phase

1. Calculate total variances of n features among all mean feature vectors, $Z_1, Z_2, \ldots, Z_{5401}$, of 5401 reference characters and then arrange the n variances in descending order. Let $\{s_1, s_2, \ldots, s_n\}$ be the permutation of the sequence $\{1, 2, \ldots, n\}$
2. Rearrange the features of those 5401 feature vectors according to the sequence $\{s_1, s_2, \ldots, s_n\}$.

Recognition phase

1. Compute the feature vector X represented by $(X_1, X_2, \ldots, X_n)$, of an input character image and rearrange the features according to the sequence $\{s_1, s_2, \ldots, s_n\}$.
2. Compute the distance $d^n(X, Z_m)$ from the first P features of the input feature vector and feature vectors of all reference categories, where

$$d^P(X, Z_m) = \sum_{j=1}^{P} \left[\frac{\| X_j - Z_{mj} \|^2}{\sigma_{mj}} + c \times \log(\sigma_{mj}) \right], \quad 1 \leq m \leq 5401,$$

and then arrange the distances in descending order.
3. Compute the distances of n features for the lth candidate whose feature vector is Z_l and the minimum partial distance is $d^P(X, Z_l)$. Designate this distance as D_{cm}, which is defined as

$$D_{cm} = d^P(X, Z_l) + \sum_{j=P+1}^{n} \left[\frac{\| X_j - Z_{lj} \|^2}{\sigma_{lj}} + c \times \log(\sigma_{lj}) \right].$$

4. For all categories except the lth category {
   (a) Compute the distance $d^n(X, Z_m)$ using

$$d^n(X, Z_m) = d^P(X, Z_m) + \sum_{j=P+1}^{P+1+K} \left[\frac{\| X_j - Z_{mj} \|^2}{\sigma_{mj}} + c \times \log(\sigma_{mj}) \right];$$

   (b) For every K features, {
   If $d^n(X, Z_m) > D_{cm}$,
   goto next category;
   else calculate

$$d^n(X, Z_m) = d^n(X, Z_m) + \sum_{j=i+1}^{i+K} \left[\frac{\| X_j - Z_{mj} \|^2}{\sigma_{mj}} + c \times \log(\sigma_{mj}) \right];$$

   $i = i + K$;
   }.
   (c) If all features have been calculated and $d^n(X, Z_m) < D_{cm}$, set D_{cm} to $d^n(X, Z_m)$
   }.

4. AUTOMATIC DOCUMENT READING SYSTEM

This section introduces a useful application of Chinese character recognition. We implement an automatic document-reading system to recognize printed Chinese documents and translate the recognition results into speech output. In this system, characters and punctuation marks are first extracted from document images and sorted according to their positions and the document orientation. The recognition process for printed characters is the same as that for handwritten characters except that the reference models are trained from samples of printed Chinese characters. The text-to-speech technique developed by Hwang[10] is integrated with our document analysis and character recognition system. The flow diagram of the automatic document-reading system is shown in Fig. 8, which consists of three main modules: character extraction, character recognition, and text-to-speech conversion.

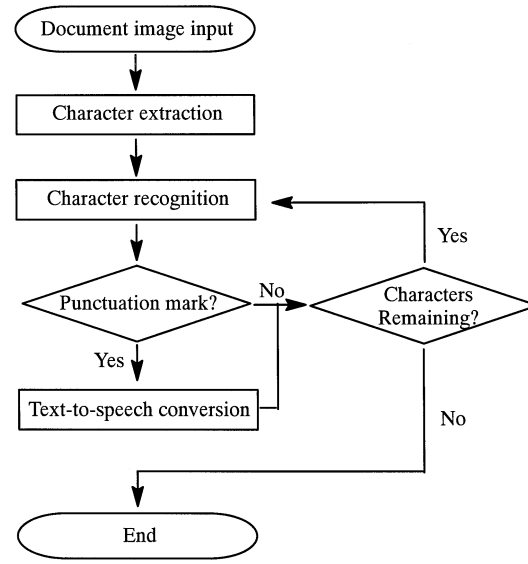

Fig. 8. Diagram of the automatic document-reading system.

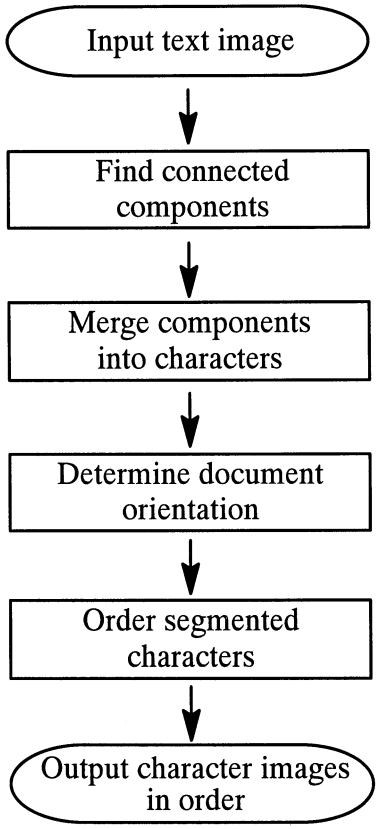

Fig. 9. Flowchart for character extraction.

4.1. Character extraction

In an OCR system, the performance of character extraction from a documant affects the character recognition results significantly. Projection is widely used for extracting printed characters.[19,20] But a skew document makes the projection method unsuitable for character extraction. In our system, we identify characters by finding connected components and then merging small broken components according to the distances between their centers. The flowchart for character extraction is shown in Fig. 9.

There have been many studies on skew detection of documents or text-blocks. These methods are based on the Hough transform, projection and Fourier transform. In this paper, we assumed that all test documents were deskewed.

There are some algorithms to extract connected components from an image. We use a method that saves computation time and memory space.[21] This method scans the document image from left to right and marks connected black pixels as components. We then check the previous scan line to detect overlapping projections on the x-axis for each pair of components and merge any overlapping components into new components. From this procedure, we obtain all the connected components represented as four corner points. Since these connected components may be parts of characters, we then need to merge some components.

We calculate the average area AA of all connected components. The components whose areas are smaller than $\frac{1}{25}\,AA$ are regarded as noise and deleted. The average height AH and the average width AW are then computed from those components whose areas are larger than $\frac{1}{3}AA$. The conditions for merging two components C_1 and C_2 are as follows:

Condition 1. One component encloses the other ($C_1 \subset C_2$ or $C_2 \subset C_1$).

Condition 2. Two components overlap ($C_1 \cap C_2 \neq \phi$).

Condition 3. The distance between the two centers of C_1 and C_2 is less than $\frac{3}{4}AH$.

These three conditions, as shown in Fig. 10, are checked repeatedly to obtain Chinese character components and punctuation marks.

In order to understand the context, we need to arrange extracted characters before they are sent to the character recognition system. In Fig. 11, we show the symbols used to determine the document orientation and character order. We define the distances between a certain extracted character C_i and its nearest neighboring components in the horizontal direction as the horizontal character space (HCS_i) and define the distances in the vertical direction as the vertical character space (VCS_i). The distances VCS and HCS are the average vertical and horizontal character spaces defined as $VCS = (1/n)\sum_{i=1}^{n} VCS_i$ and $HCS = (1/n)\sum_{i=1}^{n}, HCS_i$, where n denotes the number of characters in the document. After we obtain HCS and VCS for all components, the document orientation is determined to be vertical if the average VCS is smaller than the average HCS; otherwise the orientation is horizontal. If the document orientation is vertical, *max_RB* denotes the right boundary of the character C_r at the maximum x-coordinate of the

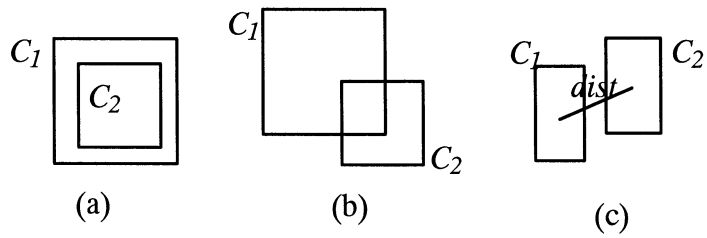

Fig. 10. Three conditions for merging two components C_1 and C_2. (a) $C_2 \subset C_1$; (b) $C_1 \cap C_2 \neq \phi$; (c) $dist < \frac{3}{4}\,AH$.

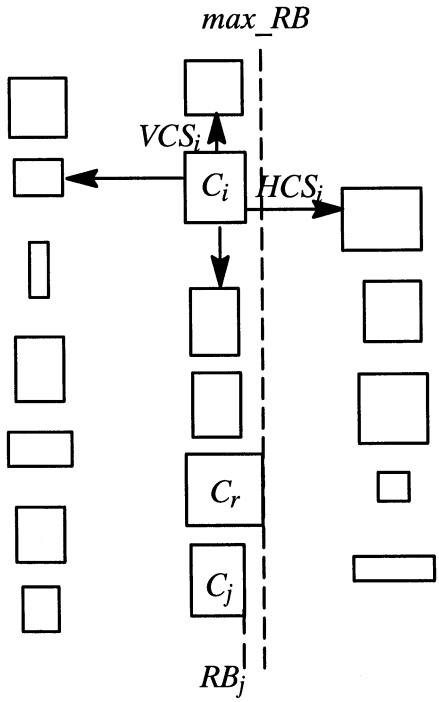

Fig. 11. Terms used to determine document orientation and character order.

document. For each character C_j with the right boundary RB_j, C_j is marked and classified into the same vertical text line as the character C_r if $max\_RB - RB_j < AW/2$. According to the vertical positions of characters we can determine the character order in the same vertical text line. Repeating the same process, characters in other text lines can be found. If the orientation of a document is horizontal, we perform a similar process.

4.2. Text-to-speech technique

To process general documents such as newspapers, magazines, and forms, the OCR engine has to combine with a document analysis system. The latter is not the main concern of this paper. We just want to show how the OCR engine can be used in the real world. The application system that combines the OCR engine with a text-to-speech system, which will be very helpful for the blinds, is a possible useful application we can develop. It can be omitted from this paper, but it will contribute to the real world. Since our OCR engine is especially used to recognize Chinese characters, the Mandarin text-to-speech system developed by Hwang[10] is employed in our system. The details of the text-to-speech system are given in reference[10] and are beyond the scope of this paper.

5. EXPERIMENTAL RESULTS

Our experiments with handwritten characters were performed on 5401 Chinese characters selected from

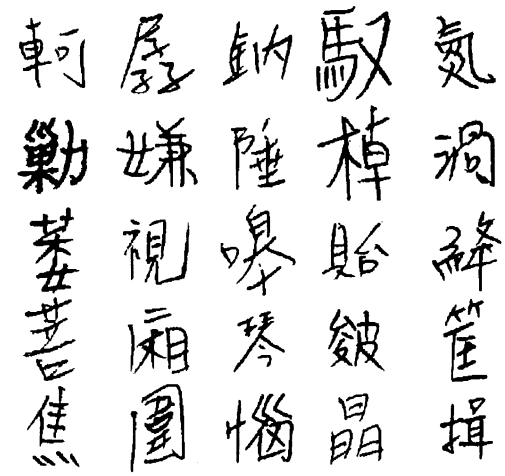

Fig. 12. Several test characters in the CCL/HCCR database.

the database CCL/HCCR. This database contains 5401 handwritten Chinese characters with 200 variants per character. Figure 12 shows some characters from this database. We selected 100 variants in even-numbered ranks to train reference character-models and tested all characters in odd-numbered ranks (100×5401 characters).

The candidate-cluster selection module selected the nearest n clusters for each input character from the N clusters defined over 5401 character categories. To keep a high hit rate and reduce the number of candidate characters, the relationship between N and n has to be concerned. The number of candidate characters would be reduced by increasing N and decreasing n, but the high hit rate must be kept by increasing n and decreasing N. Since N and n were correlative, we intended to find an optimal n/N to gain a good speed-up ratio. In our experiments, when the hit rate was taken as 99.5%, the value of n/N was about 0.25 when N was 300 and about 0.4 when N was 30. However, the value of n/N must be larger than 0.2 when N was larger than 300. Figure 13 shows the hit rate in the candidate-cluster selection module.

Figure 14 shows the speed-up ratios in the candidate-cluster selection module when the character recognition rate was above 90%. When N was 300 and n/N was about 0.2, the speed-up ratio exceeded 3.9. When N was 100, the speed-up ratio was only 3.24, in which n/N was 0.26. We found that when N became larger, the speed-up ratio also became higher. However, when N was larger than 300 and n/N was 0.2, the larger number of clusters N would increase the cluster matching time and reduce the speed-up ratio. To keep a high hit rate and obtain a high speed-up ratio, the value N was set at 300 and n/N was set at 0.2 in our candidate-cluster selection module.

Table 3 shows the recognition results of our OCR system for handwritten Chinese characters. The average recognition accuracy is 91.83%. In this paper, the

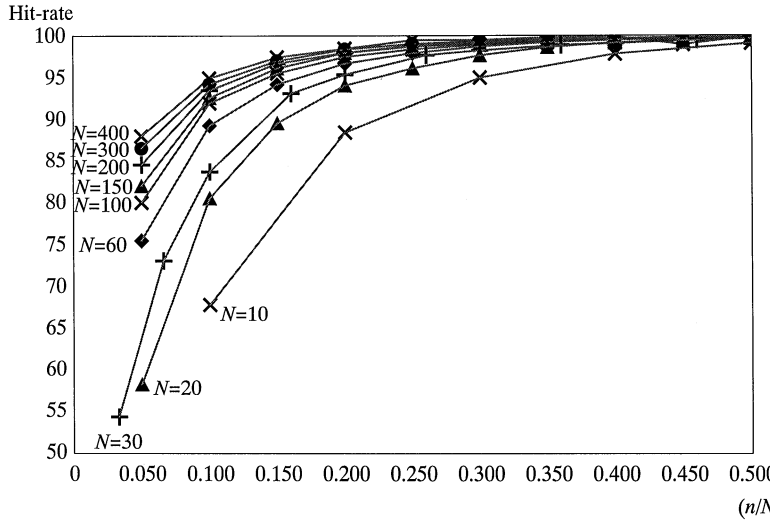

Fig. 13. Hit-rates in the candidate-cluster selection module with respect to the different values of n/N.

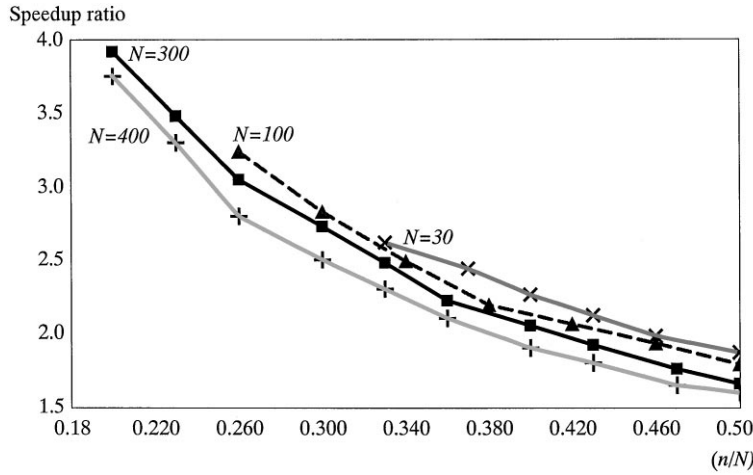

Fig. 14. Relationships between n/N and speedup ratio.

Table 3. Recognition results for handwritten Chinese characters

Rank 001–019	(95.41, 95.54, 95.16, 94.42, 94.96,	95.15, 95.00, 94.88, 94.91, 94.87)
Rank 021–039	(94.81, 94.74, 94.74, 94.75, 94.67,	94.58, 94.52, 94.43, 94.31, 94.25)
Rank 041–059	(94.19, 94.13, 94.07, 93.98, 93.90,	93.81, 93.75, 93.70, 93.59, 93.52)
Rank 061–079	(93.44, 93.39, 93.34, 93.28, 93.22,	93.16, 93.09, 93.01, 92.95, 92.88)
Rank 081–099	(92.83, 92.77, 92.71, 92.63, 92.55,	92.49, 92.41, 92.33, 92.26, 92.18)
Rank 101–119	(92.12, 92.05, 91.96, 91.88, 91.81,	91.75, 91.65, 91.57, 91.50, 91.42)
Rank 121–139	(91.32, 91.24, 91.15, 91.07, 90.99,	90.88, 90.77, 90.69, 90.61, 90.52)
Rank 141–159	(90.43, 90.33, 90.23, 90.12, 90.02,	89.92, 89.81, 89.70, 89.60, 89.49)
Rank 161–179	(89.37, 89.27, 89.16, 89.05, 88.93,	88.83, 88.71, 88.58, 88.46, 88.33)
Rank 181–199	(88.21, 88.09, 87.95, 87.83, 87.70,	87.56, 87.40, 87.26, 87.11, 87.02)

nominator of the speed-up ratio was the system without employing candidate-cluster selection and branch-and-bound. The speed-up ratio in candidate-clustering selection module was 3.92 when the top 60 clusters were selected from 300 pre-trained clusters. The speed-up ratio with the modified branch-and-bound detail-matching module was 2.02 corresponding to the system with the candidate-cluster selection module. The total speed-up ratio can be 3.92×2.02 on comparing with the nominator.

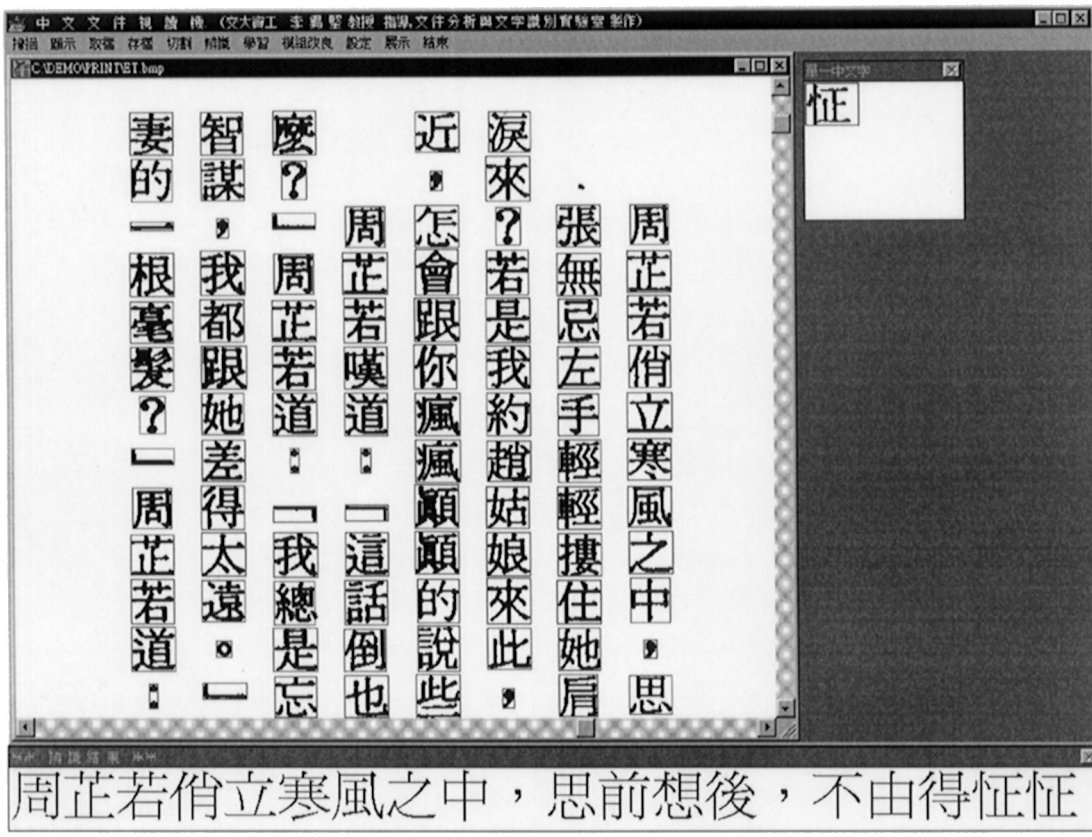

Fig. 15. Windows–based automatic document reading system.

Table 4. Experimental results for printed Chinese characters

Typeface	Song	Li	Kai	Round	Ming	Black	Total
Recognition rate	97.50%	93.75%	98.33%	95.83%	98.75%	97.33%	96.86%

The CPU processing time of our system is about 0.6 s per character on a Pentium 100 PC, which is not competent with commercial products. In this paper, we do not optimize codes and adjust parameters for reducing processing time. We point out the performance that a system, as a black box, integrating with our speeding-up modules can achieve.

The automatic document reading system is a Windows-based application implemented in C language on a Pentium 100 PC with a 16-bit Sound Blaster card. The tested document images were obtained using a Umax Vista S-8 color scanner at 300 dpi. Table 4 shows the recognition rates for our proposed recognition system on six typefaces tested using 5401 printed Chinese characters per typeface. The average recognition rate for the six typefaces was 96.86%.

The text-to-speech system was from Huang.[10] There is a speech database that contains 655 sentential and paragraphic utterances pronounced by a male speaker. Our document reading system extracted Chinese characters and punctuation marks from the document, recognized them, and sent the text strings between pairs of punctuation marks to the text-to-speech system. In order to evaluate the performance of our automatic document reader system, we selected the famous empire novel "*story of two swords, Yi-Tian and Tu-Long*" written by Chin Yung as the input document. Figure 15 displays the result of the Windows–based automatic document reading system. A subjective listening test conducted by graduate students in computer science confirmed that the synthetic speech sounded very fluent and natural.

6. CONCLUSIONS

The main contribution of this paper is to propose methods for speeding up the OCR engine via two modules: candidate-cluster selection and modified branch-and-bound detail-matching. Other less

important contributions include that we propose to integrate several document analysis techniques and a text-to-speech synthesis system in a Windows-based document-reading system.

A robust character recognition system for handwritten Chinese characters has been proposed in this paper. It first segmented character images non-uniformly into 8×8 sub-regions, and crossing-count features (*CCF*) and contour-direction features (*CDF*) are then extracted from each sub-region. The recognition system consists of two stages. The *CCF* is first used to determine the possible candidate clusters in the candidate-cluster selection module. Next, *CCF* and *CDF* are combined to match input character images with candidates in selected clusters in the detail-matching module, which uses a modified branch-and-bound algorithm to speed-up recognition and maintain a high recognition accuracy.

An automatic reading system for printed Chinese documents has been successfully implemented. A pure text document is first scanned, and then the characters are extracted by finding connected components. These extracted characters are arranged according to character positions and document orientation, which is determined from the average character spacing. After character recognition, recognized text strings are sent to the Mandarin text-to-speech system. Experimental results show high character recognition rates, low recognition times, and fluent speech output. We have shown an application system that combines an OCR engine with a text-to-speech module, which will be very helpful for the blind.

In the current system, the nearest-mean reclassification algorithm is not guaranteed to converge; it may be trapped in local minima and may fail to find the global minimum. Modification of this algorithm to obtain better performance is left for further study. The analysis of confusion characters to improve the character recognition accuracy is also left for further study. Document layout analysis, typeface identification, and character-set expansion are all directions for future research in automatic document-reading systems.

REFERENCES

1. V. K. Govindan and A. P. Shivaprasad, Character recognition – a review, *Pattern Recognition* **23**(7), 671–683 (1990).
2. T. H. Hildebrandt and W. Liu, Optical recognition of handwritten Chinese characters advances since 1980, *Pattern Recognition* **26**(2) 205–225 (1993).
3. Q. Zhang, Q. R. Wang and R. Boyle, A clustering algorithm for data-sets with a large number of classes, *Pattern Recognition* **24**(4), 331–340 (1991).
4. C. H. Tung, H. J. Lee and J. Y. Tsai, Multi-stage pre-candidate selection in handwritten chinese character recognition systems, *Pattern Recognition* **27**(8), 1093–1102 (1994).
5. Kamgar-Parsi B. and L. N. Kanal, An improved branch and bound algorithm for computing *k*-neighbours, *Pattern Recognition Lett*. **3,** 7–12 (1985).
6. H. Niemann, R. Goppert, An efficient branch-and-bound nearest neighbour classifier, *Pattern Recognition Lett*. **7,** 67–72 (1988).
7. S. Shuji, M. Michihiko and I. Katsuo, A fast algorithm for the minimum distance classifier and its application to Kanji character recognition, *Proc. 3rd ICDAR*, 283–286 (1995).
8. F. M. Wahl, K. Y. Wong, and R. G. Casey, Block segmentation and text extraction in mixed text/image documents, *Computer Graphics Image Process.* **20,** 375–390 (1982).
9. L. A. Fletcher and R. Kasturi, A robust algorithm for text string separation from mixed text/graphics images, *IEEE Trans. Pattern Recognition Mach. Intelli.* **10,** 910–918 (1988).
10. S. H. Hwang, A study on information generator for mandarin text-to-speech, Ph. D. dissertation, Dept. of Electronics, National Chiao Tung University, Taiwan, R.O.C. (1996).
11. J. Tsukumo and K. Asai, Machine printed Chinese and Japanese characters recognition method and experiments for reading Japanese pocket books, *IEEE Comput. Society Conf. on Computer Vision and Pattern Recognition*, pp. 162–167 (June 1986).
12. L. Tu, *et al*., Recognition of handprinted Chinese characters by feature matching, *Int. Conf. on Computer Processing of Chinese and Oriental Languages*, pp. 154–157 (1991).
13. C. H. Tung, H. J. Lee, Increasing character recognition accuracy by detection and correction of erroneously-identified characters, *Pattern Recognition* **27**(9), 1259–1266 (1994).
14. R. Oka, Handwritten Chinese–Japanese characters recognition by cellular feature, *Proc. 6th Int. Conf. on Pattern Recognition*, pp. 783–785, IEEE, New York (October 1991).
15. S. H. Kim and J. I. Doh, Off-line recognition of Korean scripts using distance matching and neural network classifiers, *Proc. 3rd ICDAR*, pp. 34–37 (August, 1995).
16. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, pp. 418–419. Addison-Wesley, Reading, MA (1992).
17. K. Fukunaga, *Introduction to Statistical Pattern Recognition*. Academic Press, New York (1990).
18. T. F. Li and S. S. Yu, Handprinted Chinese character recognition using the probability distribution feature, *Int. J. Pattern Recognition Artificial Intell.* **8**(5), 1241–1258 (1994).
19. Yeong-Shuenn Lin, Character boundary identification method and system, U.S. Patent, No. 5, 253, 305.
20. Jar-Long Wang, Method and system for identifying lines of text in a document, U.S. Patent, No. 5, 307, 422.
21. S. M. Yang, Multi-thresholding character extraction in a map, Master's thesis, Dept. of Comp. Sci. & Info. Engr., National Chiao Tung University, Taiwan, R.O.C. (1995).