JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 19, 555-570 (2003)

A Coarse Classification Scheme on Printed Chinese Characters by Encoding the Feature Points

MING-GANG WEN[*+], CHIN-CHUAN HAN[++], KUO-CHIN FAN[+]
AND DA-WAY TANG[+]
[*]*Department of Information Management*
National Lien-Ho Institutes of Technology
Miaoli, 360 Taiwan
[+]*Department of Computer Science and Information Engineering*
National Central University
Chungli, 320 Taiwan
[++]*Department of Computer Science and Information Engineering*
Chung-Hua University
Hsinchu, 300 Taiwan

In this paper a coarse classification scheme is proposed to speed up the recognition process of machine printed Chinese character. Simple and stable features are extracted by encoding feature points into a codeword of length 16. Geometrically, the codeword represents the distribution of feature points among character strokes. Using these simple features to do coarse classification can eliminate a large number of impossible candidates. Only few surviving candidates need be re-checked in the following more complex recognition algorithms. This scheme will simplify the design of the classifier algorithm and reduce the recognition time. Some experimental results are given to show the validity and efficiency of our proposed methods.

Keywords: coarse classification, feature point extraction, codeword encoding, optical character recognition, threshold-based selection

1. INTRODUCTION

OCR (optical character recognition) is an essential process in converting paper documents into electronic files for office automation and digital libraries. Recently, several commercial products have been marketed to recognize commonly used Chinese characters of a specific machine-printed font. In order to reconstruct the original documents in the digital library, three critical functions should be considered in the OCR process. They include (1) the capacity to recognize a large number of characters, (2) a quick response time, and (3) the ability to recognize multi-font characters. In this paper we will focus on the second topic of improving the efficiency of the recognition process. The most common and efficient approach to achieve this goal is to match reference characters by using features of low dimensionality. Stable and simple features will be extracted in the coarse classification process. Using simple and stable features to do the coarse classification task can eliminate a large number of impossible candidates. Only

Received April 19, 2002; accepted September 25, 2002.
Communicated by H. Y. Mark Liao.

few surviving candidates need be re-checked in the furthermore complex recognition. This scheme will reduce the search time, and can simplify the design of the classifier algorithm in the later recognition process.

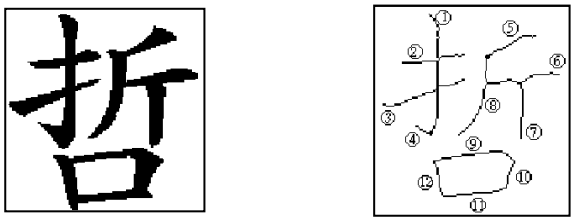
Most of the coarse classification algorithms are based on these key elements: stable and simple features. Tseng *et al.* [1] selected contour direction and crossing count of a character to be the statistical features in their coarse classification method. Pei *et al.* [2] used morphological techniques to extract the stable stroke at four margins of a character. Lin and Cheng [3], for example, designed a two-stage clustering scheme to recognize printed Chinese characters. They utilized the stable features for pre-classification and re-checked the surviving candidates by using detailed features of characters. Moreover, Cheng and Chen [4] defined the sub-structure of characters and extracted them in a pre-defined sequence. The sequence is encoded and classified to a specific cluster by the hashing approach. Chang and Wang [5] extract the radical structure from the four corners of characters to do the pre-classification task. Thirty-seven kinds of basic structure were defined and classified into 25 clusters. The radical structures located at the four corners and belonging to the 25 clusters encode a codeword. The codewords are clustered in the coarse classification process. For the coarse classification Han *et al.* [6] proposed a stroke-based method to cluster printed Chinese characters into three types. Fan *et al.* [7] used the information of pseudo skeleton for the coarse classification. Wang *et al.* [8, 9] used this technique on the radical extraction scheme for hand-written Chinese characters.

In this paper we propose an automatic coarse classification approach for the recognition of machine printed Chinese characters by encoding four clusters of feature points. The main contribution of this scheme is to eliminate a large number of impossible candidates and thereafter simplify the design of the later classification algorithm. In this approach, the coarse classification process is accomplished in four main modules: (1) *pre-processing*, (2) *feature point extraction*, (3) *codeword encoding* and (4) *candidate selection*. First, the character images are normalized to a pre-defined size. The strokes and feature points are extracted and encoded to obtain a codeword of length 16 by using the *k*-mean clustering algorithm. Finally, the codeword of an input character is matched with codewords of reference characters in the database to obtain a candidate list of small size by using four threshold-based selection rules.

2. PRE-PROCESSING

The pre-processing module includes image thinning and stroke segmentation operations. A character image is captured from a flat-form scanner which transfers the paper documents into electronic image data. In our approach we are concerned with the feature points among character strokes. Before encoding a feature point into a codeword, each stroke in the character should first be extracted to facilitate the supply of stroke information for later feature point extraction and encoding.

Strokes embedded in a Chinese character are extracted from the thinned image to obtain the skeleton of the character [10]. After thinning, the feature points, including 1-fork point (end point), 2-fork point (turning point), 3-fork point (T-junction point), 4-fork point (cross point), and other points of degree larger than 4, in the thinned image are extracted according to topological properties [11-13]. Fig. 1 illustrates the processing results of stroke extraction. More details are given in [6].



(a) (b)

Fig. 1. The pre-processing. (a) the original pattern, and (b) the segmented pattern.

3. FEATURE POINT EXTRACTION

The end points, corner points, and crossing points (called *feature points*) are segmented. In this section the types of feature points are defined and encoded to obtain the reference codes. After carefully analyzing the structure of Chinese characters, the fork degree of most Chinese character strokes is found to be less than or equal to four. For example, the fork points of strokes in Fig. 2 are drawn by dot bullets. The numbers besides the bullets denote the degree of fork points. These fork points whose degrees are less than 5 are defined to be the *stable feature points* and represented as $f(x, y, d)$, where (x, y, d) denotes the location (x, y) and degree d of feature point f.

If the degree of fork points is larger than four, the decomposition rules should be designed to decompose the fork points into several feature points of degree less than five. In general, feature points of degree 4, i.e., a crossing of two strokes, are first separated from fork points of degree larger than 4. Next, the connecting points of the other strokes are defined to be the feature points of degree 3. That is, a fork point of degree d larger than 4, for example $f(x, y, d)$, can be decomposed into a feature point of degree 4 (e.g., $f(x, y, 4)$) and $(d - 4)$ feature points of degree 3 (e.g., $f(x, y, 3)$). Four examples are shown in Fig. 3. In this figure, all degrees of fork points at the left-most part are larger than 4. After the decomposition process, the feature points are obtained as shown at the right-hand side.

Therefore, the generating rules G for the stable feature points embedded in the decomposition rules are

$$\mathsf{G}(f(x, y, d)) = \begin{cases} f(x, y, d) & \text{if } d <= 4, \\ f(x, y, 4) + (d-4) f(x, y, 3) & d = 5,6,8 \\ \textit{ignored} & \textit{otherwise.} \end{cases} \tag{1}$$

In Fig. 2 there are ten feature points of degree 1, six of degree 2, two of degree 3, and two of degree 4.

4. CODEWORD ENCODING

The stable feature points $f(x, y, d)$ of character C_r have successfully been extracted in the previous section. In this section these feature points are assigned to four clusters

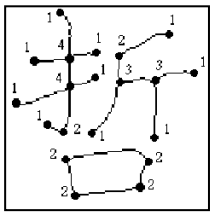


Fig. 2. The extracted feature points of the character in Fig. 1.

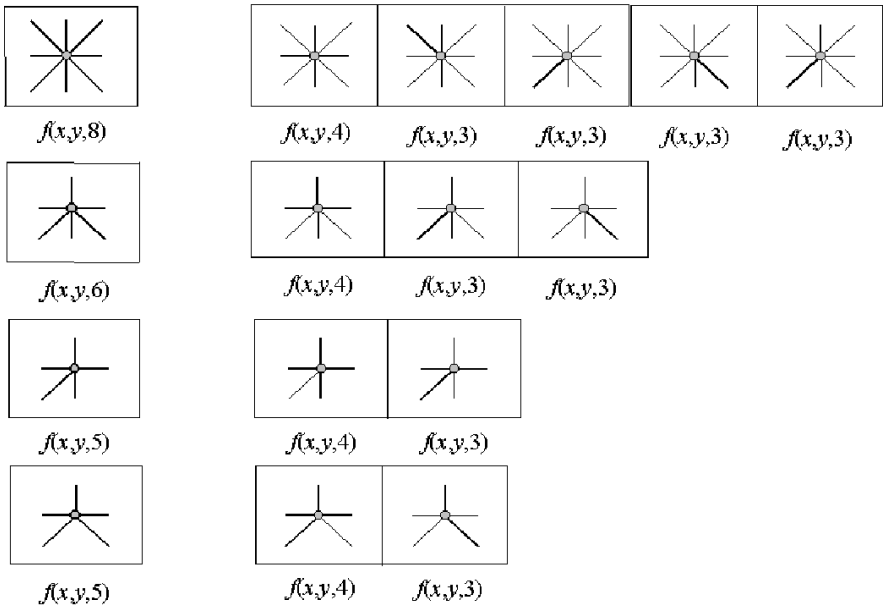


Fig. 3. The decomposition of Feature points of degree larger than 4.

and encoded to obtain codeword W_r in three steps. First, four cluster centers $\bar{c}_i$, where i = 1, 2, 3, 4, are calculated from the stroke pixels by using k-mean clustering. Second, each feature point $f(x, y, d)$ is assigned to its corresponding cluster $\bar{c}_i$. And finally, the feature points in each cluster are counted to obtain a codeword W of length 16.

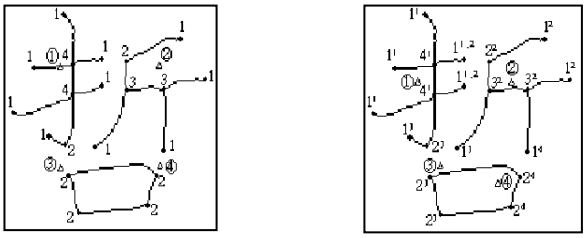
In the first step, four important factors which will inference the clustering results should be considered in the k-mean clustering algorithm. They are (1) the class number k, (2) the initial value of each cluster center, (3) the ordered sequence of clustering samples, and (4) the distribution of samples. It is very hard to automatically determine these values. The first three are manually and heuristically determined by users based on their experience. In our proposed scheme, the first three parameters are determined in the following context according to characteristics of Chinese characters. Carefully analyze the structure of Chinese characters. More than 80% of characters comprise 4 clustering centers by applying the *maximin-distance* clustering algorithm [14]. Thus, the class number k in our proposed scheme is set to be 4. We can calculate the cluster centers $\bar{c}_i$ in four

divided portions of a character image. Next, the initial values of four cluster centers (e.g., the second parameter in the *k*-mean clustering algorithm) are set to (w/4, h/4), (3w/4, h/4), (w/4, 3h/4), and (3w/4, 3h/4). Here, values w and h denote the width and height of a character image. Third, the sequence of clustering samples is reordered from left to right, and top to bottom because Chinese characters are frequently written from left to right, and from top to bottom. For example, four cluster centers located near the triangles in Fig. 4(a) are initialized. The stroke pixels and four cluster centers are clustered by *k*-mean clustering. After clustering, the new cluster centers are modified as shown in Fig. 4(b).

In the second step, the Euclidean distance $| f_j - \bar{c}_i |$ between each feature point f_j and the cluster center $\bar{c}_i$ is calculated. Feature point f_j is assigned to cluster $\bar{c}_i$, if the distance $| f_j - \bar{c}_i |$ is smaller than a given range which is the value of the shortest distance of feature point f_j to four cluster centers plus a small positive value δ. That is,

$$f_j \to \bar{c}_i, \quad \text{if } | f_j - \bar{c}_i | < \min_{k=1}^{4} | f_j - \bar{c}_k | + \delta. \tag{2}$$

Here, i = 1, 2, 3, 4. In Fig. 4(b), the feature points are assigned to four clusters. The superscript of each feature point represents the cluster number.



(a) (b)

Fig. 4. The clustering process using the proposed clustering algorithm.

Table 1. The codeword of the character in Fig. 2.

Cluster No.	degree = 1	degree = 2	degree = 3	degree = 4
1	5	0	0	2
2	4	1	2	0
3	2	3	0	0
4	1	2	0	0
codeword	$W_{r,k,i}$ = (5, 0, 0, 2), (4, 1, 2, 0), (2, 3, 0, 0), (1, 2, 0, 0)			

Finally, count the number of feature points in each cluster to generate the codeword for character C_r. As stated in the previous section, all fork points are converted into feature points of degree less than 5. Thus, the feature points are counted in four clusters from degree 1 to 4 to obtain the codeword of length 16. The number of feature points in each cluster of Fig. 4(b) are listed in Table 1. The codeword W_r = (5, 0, 0, 2), (4, 1, 2, 0),

(2, 3, 0, 0), (1, 2, 0, 0) for the character C_r in Fig. 4(b) is easily and quickly generated. The algorithm to encode the codeword of character C_r is as follows:

Algorithm for Encoding

Input: The stroke pixels s_i and feature points f_j of character C_r.
Output: The codeword W_r of character C_r.
Step 1: Reorder the sequence of stroke pixels s_i from left to right and top to bottom in the character image.
Step 2: Set k in the k-mean clustering algorithm to 4.
Step 3: Initialize four cluster centers $\bar{c}_1, \bar{c}_2, \bar{c}_3$, and $\bar{c}_4$ to (w/4, h/4), (3w/4, h/4), (w/4, 3h/4), and (3w/4, 3h/4), respectively. Here, values w and h represent the width and height of a character image.
Step 4: Perform k-mean clustering until the algorithm converges. The Euclidean distance criterion is used to compute the distances among samples.
Step 5: Assign feature point f_j to cluster $\bar{c}_i$, if $| f_j - \bar{c}_i | < \min_k | f_j - \bar{c}_k | + \delta$, where δ is a small positive value.
Step 6: Count the number of feature points in each cluster to generate the codeword W_r for character C_r.

5. CANDIDATE SELECTION RULES

From the above section, the codewords of characters of a specific font are obtained from the extracted strokes and feature points. However, the structure of extracted strokes will be distorted due to corruption of the image. This problem will generate various codewords for a particular character from various corrupted images of the character. In the following we devise four threshold-based selection rules to select the candidates which should be re-checked in the following recognition stages. The threshold value is calculated from two values. One is a pre-determined value θ which is determined from human knowledge, and the other ξ is an adaptive value that is statistically generated from some training samples. In our experiments, the city block distance (i.e., the absolute error) is adopted to evaluate the similarity between two codewords. This distance function is defined as

$$d(W_1, W_2) = \sum_{k=1}^{4} \sum_{i=1}^{4} | w_{1,k,i} - w_{2,k,i} |. \quad (3)$$

Here, value $w_{r,k,i}$ denotes the number of feature points of degree i in the k^{th} cluster of codeword W_r. Additionally, some examples are illustrated to show the selected results in each selection process.

Rule I: Selecting via fixed threshold

In this scheme the distance between the test and reference characters is compared with a fixed and pre-determined threshold θ. The adaptive value ξ is set to be 0 in this

selection process. Two reference characters C_{1561} and C_{2120}, for example, possessing better image quality are displayed in Figs. 5(a) and (b). Two test characters C_{t1} and C_{t2} of different image sizes are shown in Figs. 5(c) and (d). The codewords of reference characters W_{1561} = (6, 0, 0, 2), (4, 1, 2, 0), (2, 3, 0, 0), (1, 2, 0, 0) and W_{2120} = (5, 1, 0, 2), (2, 2, 1, 0), (4, 1, 0, 1), (4, 3, 1, 0) are previously calculated and stored in the code-book database. The codewords of two test characters W_{t1} = (6, 0, 0, 2), (4, 1, 2, 0), (3, 2, 1, 0), (1, 2, 0, 0) and W_{t2} = (6, 0, 0, 2), (4, 1, 2, 0), (2, 3, 0, 0), (1, 2, 0, 0) are obtained by the encoding process of feature points. The distances between the test and reference characters can be calculated using equation (3):

$$d(W_{t1}, W_{1561}) = 3,\ d(W_{t1}, W_{2120}) = 15,$$
$$d(W_{t2}, W_{1561}) = 0,\ d(W_{t2}, W_{2120}) = 16.$$

Obviously, test characters C_{t1} and C_{t2} should be identified as character C_{1561}. The selection rule is: if distance $d(W_{t1}, W_r)$ is smaller than a fixed threshold θ, the reference character C_r is selected to be a candidate, i.e., $R = \{C_r \mid d(W_t, W_r) < \theta$, for $r = 1, 2, \ldots, N\}$. Here, R is the candidate set, and N is the number of reference characters.

Rule II: Selecting via adaptive character threshold

In designing this selection rule, the adaptive threshold ξ_r (also called adaptive character threshold) which depends on the reference characters C_r, is adapted by using *learning-by-example* techniques. Consider M training sample images $I_1, I_2, \ldots, I_M$, and a reference image I_r of a specific character C_r. Their generated corresponding codewords are $W_1, W_2, \ldots, W_M$, and W_r, respectively. The adaptive character value ξ_r is chosen to be the maximum values of distances $d(W_1, W_r), d(W_2, W_r), \ldots, d(W_M, W_r)$. The similarity distance between characters C_t and C_r is compared with a specified threshold ξ_r, which is determined by the reference character C_r. The candidate list R is

$$R = \{C_r \mid d(W_t, W_r) < \xi_r, \text{ for all } r = 1, 2, \ldots, N\}. \quad (4)$$

Here, ξ_r is an adaptive value determined by the training samples. Consider the example in Fig. 5, the threshold values for characters C_{1561} and C_{2120} are $\xi_{1561} = 5$ and $\xi_{2120} = 7$, respectively.

In the training phase, a great many training samples are needed to obtain sufficiently reasonable values for ξ_r. Initially, ξ_r can be combined with a pre-fixed value θ to increase the system performance when the training samples are not adequate. This pre-fixed value θ is manually determined by users.

Rule III: Selecting via adaptive cluster threshold

Similar to Rule II, the adaptive threshold ξ can be extended to the cluster level. Four adaptive cluster values $\xi_{r,k}$, $k = 1, 2, 3, 4$, are also obtained from the training samples. In this scheme vector $w_{t,k}$ representing the vector in the k^{th} cluster of test character C_t will be matched with those of reference character C_r. The distances of four clusters $d(W_{t,k}, W_{r,k})$ are computed to evaluate the similarity between the test and reference characters.

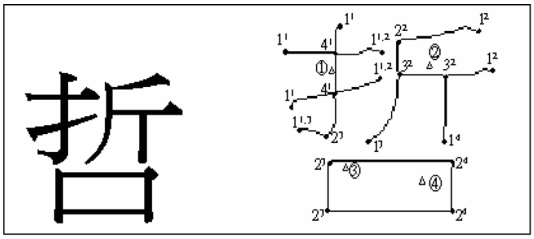


Image size: 120 × 120
Codeword: W_{1561} = (6, 0, 0, 2), (4, 1, 2, 0), (2, 3, 0, 0), (1, 2, 0, 0)

(a) Character C_{1561}.

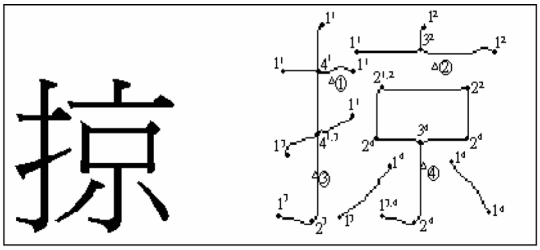


Image size: 120 × 120
Codeword: W_{2120} = (5, 1, 0, 2), (2, 2, 1, 0), (4, 1, 0, 1), (4, 3, 1, 0)

(b) Character C_{2120}.

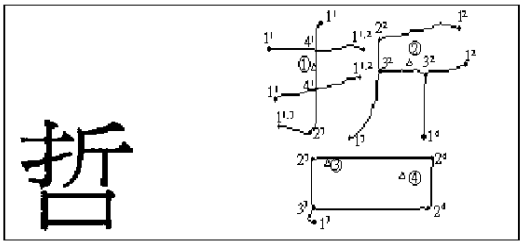


Image size: 90 × 90
Codeword: W_{t1} = (6, 0, 0, 2), (4, 1, 2, 0), (3, 2, 1, 0), (1, 2, 0, 0)

(c) Character C_{t1}.

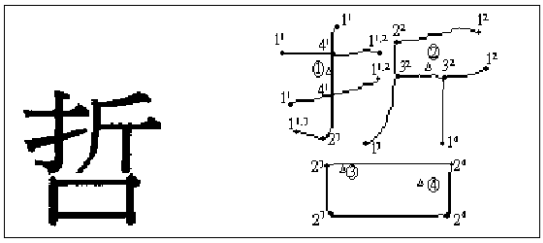


Image size: 120 × 120
Codeword: W_{t2} = (6, 0, 0, 2), (4, 1, 2, 0), (2, 3, 0, 0), (1, 2, 0, 0)

(d) Character C_{t2}.

Fig. 5. An example of the candidate selection process.

Then, voting is applied to decide if the test character is a reference character or not. The selection rule to generate list R is

$$R = \{C_r \mid\mid V_r \mid \geq 3, \text{ for all } r = 1, 2, \ldots, N\}, \quad (5)$$

where $V_r = \{k \mid d(W_{t,k}, W_{r,k}) = \sum_{i=1}^{4} \mid w_{t,k,i} - w_{r,k,i} \mid < \xi_{r,k}, k = 1, 2, 3, 4\}$, $|V_r|$ is the size of set V_r. In Fig. 5 all these values are set to 3, and the cluster threshold values for characters C_{1561} and C_{2120} are set to $\xi_{1561,1} = 0$, $\xi_{1561,2} = 0$, $\xi_{1561,3} = 0$, $\xi_{1561,4} = 0$, $\xi_{2120,1} = 1$, $\xi_{2120,2} = 0$, $\xi_{2120,3} = 1$, and $\xi_{2120,4} = 0$. Based on these threshold values, the test characters C_{t1} and C_{t2} in Fig. 5(c) and (d) are identified as character C_{1561}. Thus, character C_{1561} is selected to be a candidate in list R.

Similarly, four pre-defined values θ_1, θ_2, θ_3, and θ_4 can be added to the adaptive threshold values in each of four clusters to improve the performance of an initial system when the training samples of each character is not adequate.

Rule IV: Selecting the nearest *k* candidates

In this scheme the candidates are selected by choosing the k nearest neighbors. The value of k is independent of the two thresholds θ and ξ. The distances between test character C_t and all reference characters are first calculated and sorted. Then, the characters corresponding to the smallest k distances are selected to form the candidate list R. For example, consider test character C_{t1} as shown in Fig. 5(c). The distance between C_{t1} and C_{1561} is smaller than that between C_{t1} and C_{2120}, i.e., $d(W_{t1}, W_{1561}) < d(W_{t1}, W_{2120})$. Thus, character C_{1561} is selected to be a candidate, but character C_{2120} is not.

In the selection process of Rule IV, the distances of the candidate characters located between ranks k_l and $k_u (k_l < k_u)$ in the output list R may be all the same values when matching with the input character. If the value of k is set between k_l and k_u, i.e., $k_l < k < k_u$, all candidate characters whose rank is smaller than k_u would be selected. Here, the choose of value k is determined by the system requirements. It can be set manually by the users.

Strategies of Rule Selection

As stated in the preceding pages, we have proposed four rules to coarsely choose the candidates from a large character set. They can be adopted to perform a coarse searching operation according to the requirements of the application. First, Rule I is a simple and fast method to find the candidates. The selection results can be improved by using Rule II. However, a large number of training samples have to be collected to obtain the character thresholds. Similarly, extra training samples are needed to determine the cluster thresholds by using Rule III. Though Rule IV provides effective searching results, it needs extra time to rearrange the ranks of characters for each test character. We can set up some slots to remedy the rearranging problem. The characters in each slot indexed by the matched distance are the candidates possessing the same distances. Much memory should be allocated to store the candidates. Moreover, it is more complex than the other three rules in the implementation.

6. EXPERIMENTAL RESULTS

In this section, some experiments are presented to show the effectiveness of our proposed methods. The distances between two codewords $d(W_1, W_2)$ among 5401 characters are tabulated in Table 2. Symbol $D(W_1, W_2)$ denotes the distance between two codewords. The numbers listed at the second, fourth, and sixth rows represent the character numbers of distance $D(W_1, W_2)$. From this tale the threshold used in Rule I can be determined according to the number of surviving characters. These surviving candidates need be re-checked in the following more complex recognition algorithm. Obviously, the match criterion using the distance between two codewords is an effective approach to do coarse classification.

In our experiments, Ming-font character images were generated to evaluate the performance of our proposed method because Ming-font is the most frequently used font in the main text of many documents. For comparison the results of our previous efforts on the general approach to solving the coarse classification problem on multi-font characters has been proposed in [6]. Our previous efforts focused on the separation of *syntactic* structure in Chinese characters, i.e., the left/right, up/down, and single structures. It was evaluated manually by a human. However, in this paper the characters are systematically encoded into vectors of length 16. The reference character database and its corresponding parameters are statically and automatically generated based on the performance requirements from the viewpoint of system construction. In order to obtain better performance, the input characters of a specific font are matched with the reference characters of the same font. Moreover, it is necessary to do the font detection operation off-line and build up many reference character databases for various fonts.

Table 2. The distances among 5401 Chinese characters.

$d(W_1, W_2)$	0	1	2	3	4	5	6	7	8	9
character No.	1	1	1	1	1	2	5	10	18	33
$d(W_1, W_2)$	10	11	12	13	14	15	16	17	18	19
character No.	58	95	149	223	321	444	595	773	977	1203
$d(W_1, W_2)$	20	21	22	23	24	25	26	27	28	29
character No.	1448	1707	1977	2250	2524	2792	3052	3300	2534	2751

The images of 5401 character were directly generated from the computer font and considered to be the reference images. Five sets of 5401×5 test characters scanned from a flat-bed scanner were set up to evaluate the performance of each selection rule as shown in Fig. 6. The feature points in the first three test sets possessing better image quality (i.e., Sets I, II, and III, see Figs. 6(b), (c) and (d)) were extracted as described in the previous sections. However, 2 or 3 feature points were missed while extracting those points from the corrupted character images in Sets IV (Fig. 6(e)) and V (Fig. 6(f)). Furthermore, the ratio of character image (e.g., height to width) in Set V was previously set to 4:3 instead of 1:1. Another example showing the selection results is illustrated in Fig. 7.

In the matching phase, the test character was encoded into a vector of length 16 and

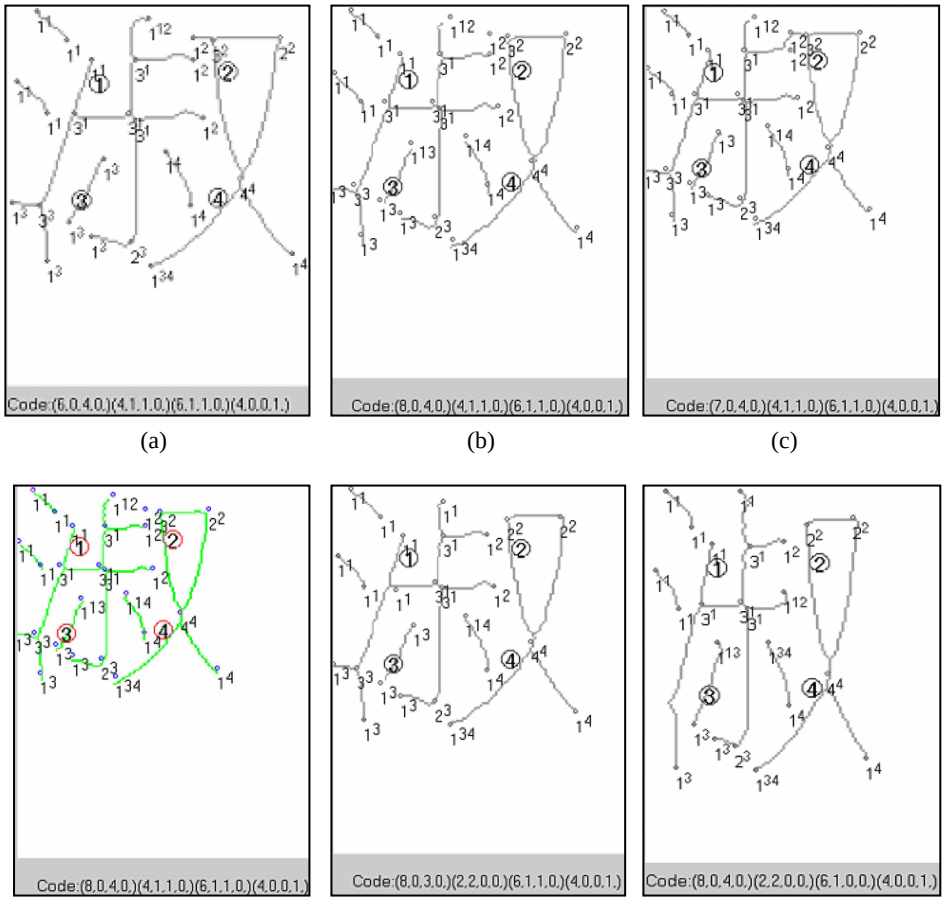
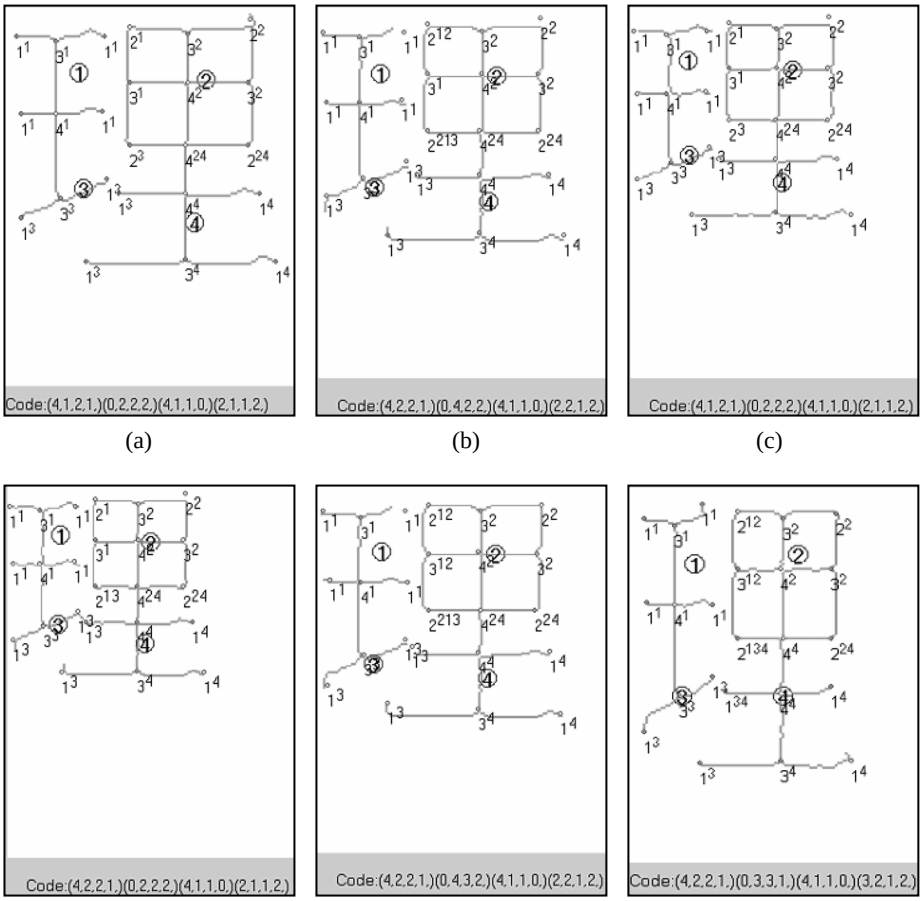


Fig. 6. Five sets of test characters. (a) template, size = 120 × 120, codeword = (6,0,4,0) (4,1,1,0) (6,1,1,0) (4,0,0,1), (b) Set I, size = 110 × 110, codeword = (8,0,4,0) (4,1,1,0) (6,1,1,0) (4,0,0,1), (c) Set II, size = 110 × 110, codeword = (7,0,4,0) (4,1,1,0) (6,1,1,0) (4,0,0,1), (d) Set III, size = 90 × 90, codeword = (8,0,4,0) (4,1,1,0) (6,1,1,0) (4,0,0,1), (e) Set IV, size = 120 × 120, codeword = (8,0,3,0) (2,2,0,0) (6,1,1,0) (4,0,0,1), and (f) Set V, size = 120 × 90, codeword = (8,0,4,0) (2,2,0,0) (6,1,0,0) (4,0,0,1).

matched with the vectors of 5401 reference characters in the database. Using the simple and stable features to do the coarse classification task can eliminate a large number of impossible candidates. Only a few surviving candidates need be re-checked in the following more complex recognition process. Tabulated results are shown in Tables 3 to 6. In these tables, symbol '-' denotes the correct rate being above 99%, symbol '*' represents the rate less than 75%, and symbols '%' and 'No.' denote the correct rate and the average number of chosen candidates in list *R*. First, five sets of test characters are constructed to evaluate the selection results of Rule I as shown in Table 3. Next, the characters in Sets I and III were used to compute the adaptive character thresholds and cluster thresholds in Rules II and III. The pre-defined values θ (for Rule II) and θ_1, θ_2, θ_3, θ_4 (for



(a) (b) (c)

(d) (e) (f)

Fig. 7. Five sets of test characters. (a) template, size = 120 × 120, codeword = (4,1,2,1) (0,2,2,2) (4,1,1,0) (2,1,1,2), (b) Set I, size = 110 × 110, codeword = (4,2,2,1) (0,4,2,2) (4,1,1,0) (2,2,1,2), (c) Set II, size = 110 × 110, codeword = (4,1,2,1) (0,2,2,2) (4,1,1,0) (2,1,1,2), (d) Set III, size = 90 × 90, codeword = (4,2,2,1) (0,2,2,2) (4,1,1,0) (2,1,1,2), (e) Set IV, size = 120 × 120, codeword = (4,2,2,1) (0,4,3,2) (4,1,1,0) (2,2,1,2), and (f) Set V, size = 120 × 90, codeword = (4,2,2,1) (0,3,3,1) (4,1,1,0) (3,2,1,2).

Rule III) are listed in Tables 4 and 5 to show the selection results of test character sets II, IV, and V. On average, the sizes of list R for Sets I, II, and III are between 15 and 40 when the correct rate is larger than 99%. Similarly, the average sizes of list R generated by Rules I, II, and III for character sets IV and V are between 200 and 300, if the correct rate is above 98%. In the selection process of Rule IV, all characters whose rank is smaller than k_u should be selected to be the candidates. On average, 122 and 121 candidate characters were selected for the test character sets IV and V when the value of k was set to 90 in Table 6. The distances of candidate characters between ranks 90 and 122 (or 121) are all the same. Thus, the size of list R generated by Rule IV is 122 when we wish the correct rate to be greater than 98%.

Table 3. Performance with Rule I: selecting via fixed threshold values.

θ		5	6	7	8	9	10	11	12	13	14	15
I	%	66.9%	81.2%	91.1%	95.6%	98.0%	99.2%	–	–	–	–	–
	No.	1	2	3	7	15	31	–	–	–	–	–
II	%	68.3%	82.9%	90.9%	95.8%	98.3%	99.3%	–	–	–	–	–
	No.	1	2	3	7	15	31	–	–	–	–	–
III	%	68.8%	82.2%	90.9%	95.5%	97.8%	99.0%	–	–	–	–	–
	No.	1	2	3	7	16	32	–	–	–	–	–
IV	%	55.4%	62.8%	71.0%	78.4%	85.0%	89.7%	93.4%	95.3%	96.9%	98.1%	99.0%
	No.	1	2	4	8	16	31	58	102	169	265	393
V	%	41.1%	53.8%	64.6%	72.3%	79.7%	85.6%	90.3%	93.3%	95.2%	97.5%	98.5%
	No.	1	2	3	7	15	29	54	95	157	245	367

Table 4. Performance with Rule II: selecting via adaptive character threshold values.

θ		1	2	3	4	5	6	7	9	11	13	15
II	%	42.2%	62.7%	79.8%	90.3%	96.2%	98.8%	99.5%	–	–		–
	No.	1	1	2	4	8	14	26	–	–		–
IV	%	*	*	*	*	*	84.8%	90.1%	95.4%	98.0%	99.2%	99.3%
	No.	*	*	*	*	*	14	26	77	196	408	560
V	%	*	*	*	*	*	78.7%	85.7%	93.6%	97.5%	98.8%	99.1%
	No.	*	*	*	*	*	14	24	72	182	381	525

Table 5. Performance with Rule III: selecting via adaptive cluster threshold values. (a) Set II. (b) Sets IV and V.

(a)

θ_1		1	2	3	3	3	3	2	2	2	2	1	1
θ_2		1	2	3	3	3	2	2	2	1	1	2	1
θ_3		1	2	3	3	2	2	2	1	1	2	1	2
θ_4		1	2	3	2	2	2	1	1	1	1	2	2
II	%	92.3%	99.6%	100%	99.9%	99.9%	99.7%	99.2%	97.9%	95.7%	97.3%	97.3%	96.8%
	No.	8	66	331	226	149	101	39	22	13	23	25	24

(b)

θ_1		1	2	3	4	5	6	3	3	3	4	4	4	4	3	3
θ_2		1	2	3	4	5	6	3	3	2	4	4	3	3	4	3
θ_3		1	2	3	4	5	6	3	2	2	4	3	3	4	3	4
θ_4		1	2	3	4	5	6	2	2	2	3	3	3	3	4	4
IV	%	95.1%	85.0%	98.7%	99.3%	99.6%	100%	97.3%	90.3%	88.3%	99.1%	98.7%	98.7%	99.1%	98.9%	99.2%
	No.	8	65	317	916	1092	2917	218	146	98	732	558	423	560	570	571
V	%	65.0%	85.3%	96.3%	98.9%	99.8%	100%	94.2%	91.6%	89.3%	98.5%	97.8%	96.7%	97.6%	98.1%	97.8%
	No.	8	61	301	905	1853	2887	208	137	92	698	531	402	533	544	545

Table 6. Performance with Rule IV: selecting the *k* nearest candidates.

k		1	5	10	15	20	25	30	50	70	90
I	%	91.3%	98.7%	99.4%	99.6%	99.7%	99.7%	99.8%	–	–	–
	No.	1	7	15	22	29	36	43	–	–	–
II	%	90.6%	98.6%	99.5%	99.7%	99.8%	99.9%	99.9%	–	–	–
	No.	1	7	15	22	29	36	42	–	–	–
III	%	91.6%	98.3%	98.9%	99.3%	99.4%	99.6%	99.6%	–	–	–
	No.	1	7	15	22	28	36	43	–	–	–
IV	%	73.2%	87.7%	92.3%	93.4%	94.3%	95.1%	95.5%	96.4%	97.9%	98.4%
	No.	1	7	15	22	29	36	42	69	96	122
V	%	68.2%	85.6%	90.5%	92.1%	93.1%	94.1%	95.0%	97.1%	97.6%	98.1%
	No.	1	7	15	22	29	36	43	70	96	121

Among these tables, Rule IV provides the highest accuracy and most tolerance power on the scaled images. However, it needs much time (0.131 sec./char.) to select the candidates than Rules I (0.072 sec./char.), II (0.082 sec./char.), and III (0.128 sec./char.).

7. CONCLUSIONS

In this paper we have proposed a coarse classification scheme for machine printed Chinese characters by encoding feature points among strokes. The feature points of degree less than five are extracted from a thinned image and then encoded to obtain a codeword of length 16. This codeword contains four vectors representing the distribution of feature points at four divided portions of a character image. Additionally, four selection rules have been designed to choose fewer candidates to speed up the recognition process. Some experiments have been conducted to show the validity and efficiency of our proposed schemes.

REFERENCES

1. Y. H. Tseng, C. C. Kuo, and H. J. Lee, "Recursive hierarchical radical extraction for handwritten Chinese characters," *Pattern Recognition*, Vol. 30, 1997, pp. 1313-1377.
2. T. W. Pai, G. H. Chang, T. S. Liu, S. S. Huang, K. H. Shyu, G. C. Tai, K. Y. Chang, and B. S. Jeng, "Shape feature analysis based on mathematical morphology theory," in *Proceedings of Third National Workshop on Character Recognition*, 1993, pp. 97-110.
3. J. K. Lin and B. S. J. et al., "Recognition of printed Chinese characters utilizing two-stage classification with mesh and peripheral features," in *Proceedings of Telecommunications Symposium*, 1987, pp. 533-537.
4. R. H. Cheng and Z. Chen, "Coarse classification on handwritten Chinese characters," in *Proceedings of Third National Workshop on Character Recognition*, 1993, pp. 24-30.

5. H. D. Chang and J. F. Wang, "Pre-classification for handwritten Chinese character recognition by a peripheral shape coding method," *Pattern Recognition*, Vol. 26, 1993, pp. 711-719.
6. C. C. Han, Y. L. Tseng, K. C. Fan, and A. B. Wang, "Coarse classification of Chinese characters via stroke clustering method," *Pattern Recognition Letters*, Vol. 16, 1995, pp. 1079-1089.
7. K. C. Fan, M. G. Wen, and C. C. Han, "A new method for Chinese characters classification using pseudo skeleton features," in *Proceedings of 5th Asian Conference on Computer Vision*, 2002.
8. A. B. Wang, K. C. Fan, and W. H. Wu, "Speeding up Chinese character recognition in an automatic document reading system," *Pattern Recognition*, Vol. 31, 1998, pp. 1601-1612.
9. A. B. Wang and K. C. Fan, "Optical recognition of handwritten Chinese characters by hierarchical radical matching method," *Journal of Pattern Recognition*, Vol. 34, 2001, pp. 15-35.
10. T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications ACM*, Vol. 27, 1984, pp. 236-239.
11. W. H. Hsu and F. H. Cheng, "Recognition of handwritten Chinese characters by structure analysis of strokes," *International Journal of Computer Processing of Chinese & Oriental Language*, 1985, pp. 101-112.
12. C. S. Lin, C. F. Wang, and J. S. Huang, "A new Chinese character thinning algorithm based on tracing the boundary," Technical Report, No. TR-88-08, Institute of Information Science, Academia Sinica, Taipei, 1988.
13. C. W. Liao and J. S. Huang, "Stroke segmentation by bernstein-bezier cureve," *Pattern Recognition*, Vol. 23, 1990, PP. 475-487.
14. J. T. Tou and R. C. Gonzalez, *Pattern Recognition Principles*, Addison-Wesley, 1974.

Ming-Gang Wen (溫敏淦) was born in Hsinchu, Taiwan, on 1 February 1965. He received his B.S. degree in Computer Science and Information Engineering from National Chiao-Tung University, Taiwan, in 1989. He started his graduate studies in Computer Science and Electronic Engineering at National Central University in 1991 and received the M.S. degree in 1993. In 1993, he joined the Department of Information Management at National Lien-Ho Institutes of Technology as a lecture. Currently, he is a Ph.D. student in the Institute of Computer Science and Information Engineering at National Central University, Taiwan. His research interests include optical character recognition, image processing, and digital watermarking.

Chin-Chuan Han (韓欽銓) received the B.S. degree in computer engineering from National Chiao-Tung University in 1989, and an M.S. and a Ph.D. degree in computer science and electronic engineering from National Central University in 1991 and 1994, respectively. From 1995 to 1998, he was a Postdoctoral Fellow in the Institute of Information Science, Academia Sinica, Taipei, Taiwan. He was an Assistant Research Fellow in the Applied Research Lab., Telecommunication Laboratories, Chunghwa Telecom Co. in 1999. He is currently an Associate Professor in the Department of Computer Science and Information Engineering, Chung-Hua University. His research interests are in the areas of face recognition, biometrics verification, 2D image analysis, computer vision, and pattern recognition.

Kuo-Chin Fan (范國清) was born in Hsinchu, Taiwan, on 21 June 1959. He received his B.S. degree in Electrical Engineering from National Tsing-Hua University, Taiwan, in 1981. In 1983, he worked for the Electronic Research and Service Organization (ERSO), Taiwan, as a Computer Engineer. He started his graduate studies in Electrical Engineering at the University of Florida in 1984 and received the M.S. and Ph.D. degrees in 1985 and 1989, respectively. From 1984 to 1989 he was a Research Assistant in the Center for Information Research at University of Florida. In 1989, he joined the Institute of Computer Science and Information Engineering at National Central University where he became professor in 1994. From 1994 to 1997 he was chairman of the department. Currently, he is the director of the Computer Center. He is a member of IEEE and SPIE. His current research interests include image analysis, optical character recognition, and document analysis.

Da-Way Tang (湯大偉) started his graduate studies in Computer Science and Electronic Engineering at National Central University in 1994 and received the M.S. degree in 1996. His research interests include optical character recognition, document analysis, and image processing.