# A RECOGNITION ALGORITHM FOR CHINESE CHARACTERS IN DIVERSE FONTS

*Xianli Wu and Min Wu* [*]


## ABSTRACT

This paper proposes an algorithm for recognizing Chinese characters in many diverse fonts including Song, Fang, Kai, Hei, Yuan, Lishu, Weibei, and Xingkai. The algorithm is based on features derived from Peripheral Direction Contributivity and utilizes a set of dictionaries. A 3-level matching is first performed with respect to each dictionary. The distance measures associated with these matchings are then fed into a central discriminator to output the final recognition result. We propose a new multi-dictionary matching algorithm for use in the central discriminator that utilizes estimated information of neighborhood fonts. Experiments have been performed on a practical OCR software system whose recognition kernel is based on the proposed algorithm. Fast and accurate recognition has been accomplished both in title recognition involving all of the 8 fonts and in main-body recognition that usually involves the first 4 most commonly used fonts.


## I. INTRODUCTION

Originating from China, Chinese characters are widely used in China, Japan, Southeast Asia and other Asian communities worldwide. Because of the sophisticated formation of Chinese characters, their computerized input and automated recognition are much more difficult than that for western languages. Furthermore, there have been a large number of variations in fonts during the evolution of Chinese written language. In addition to the four most commonly used fonts of Song, Fang, Kai, and Hei, other popular fonts in Chinese publications include Yuan, Lishu, Weibei, and Xingkai, as shown in Fig.1-1. The latter four fonts are especially common in titles. The recognition of a non-trivial number of diverse fonts is considered as an important yet challenging problem in the R&D of Chinese OCR systems. Many previous works have shown that when compared with systems dedicated to recognizing a small number of commonly used fonts, a system aiming at recognizing many fonts may give significantly lower recognition accuracy on the commonly used fonts as well as lower overall recognition rate.

Extending our work in [1], this paper presents a recognition algorithm for Chinese characters with diverse fonts. The algorithm is able to recognize all of the above-mentioned 8 fonts without compromising the recognition performance on the first 4 commonly used ones. It is based on features derived from Peripheral Direction Contributivity and utilizes a set of dictionaries. A 3-level matching is first performed with respect to each dictionary. The distance measures associated with these dictionaries are then fed into a central discriminator to output the final recognition result. We propose a new multi-dictionary matching algorithm for use in the central discriminator that utilizes estimated information of neighborhood fonts. Experimental results show that the proposed algorithm achieves fast and accurate recognition of characters in diverse fonts.

The paper is organized as follows. In Section II, we introduce Peripheral Direction Contributivity (PDC) features and a 3-level matching algorithm. We propose a new multi-dictionary matching algorithm utilizing neighborhood font information in Section III. The design of an experimental system is presented in Section IV, followed by a discussion on experimental results.


**Fig.1-1 Example of eight common Chinese fonts:**
***Song, Fang, Kai, Hei, Yuan, Lishu, Weibei, & Xingkai***

## II. FEATURE EXTRACTION AND THREE-LEVEL MATCHING ALGORITHM

### II-1. Peripheral Direction Contributivity ( PDC )

Feature extraction is an important element in OCR. Previous works show that the peripherals of character strokes contain a lot of information regarding the shape of characters, therefore are useful features for character recognition. Peripheral Direction Contributivity (PDC) features [2]-[4] are used in this paper.

[*] X. Wu is with Engineering Center of Character Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100080, P.R. China, wxl@hanwang. com.cn . M. Wu is with Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742, U.S.A, minwu@ eng.umd.edu .

*Direction Contributivity (DC)* is an 8-dimension vector $[d_1, d_2, \ldots d_8]$. It is calculated according to

$$d_i = l_i / \sqrt{\sum_{k=1}^{8} l_k^2}$$

where i = 1, …, 8, and $l_i$ is the number of connected black pixels in the i-th direction. As shown in Fig 2-1, DC represents the runs of connected black pixels in 8 directions from a reference point *P*. *Peripheral Direction Contributivities (PDCs)* are the DCs with front edge points of strokes serving as the reference points. Here by *front edge point* we refer to the first black pixel as the pixels encountered along a given scan direction change from white (background) to black (foreground stroke). The front edge points can be visited in layers, as indicated by the arrows of a horizontal scan line in Fig. 2-1. Accordingly, PDCs can be obtained in layers. For example, the 1st layer PDCs are computed using the outermost edge points encountered from 8 scan directions (i.e., from right, left, up, down, and four 45-degree directions). The 2nd and 3rd layer can be obtained similarly.

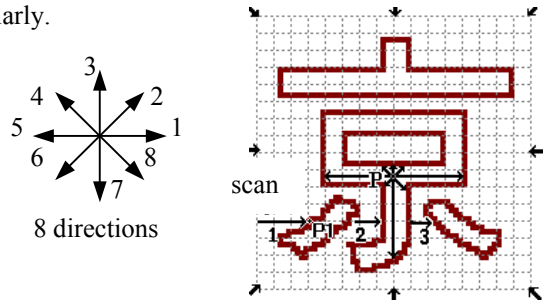


**Fig 2-1 DC and PDC**

In practice, we first normalize the size of each character bitmap to 64x64 pixels. Taking the 3-layer front edge points encountered by horizontal scans from left to right as an example, the number of feature elements is:

64 (rows) × 3 (layers) × 8 (PDC dimensions per edge point) = 1536 dimensions

We can reduce the feature dimension by dividing the 64 rows (or columns) into 8 groups and use the average of the corresponding PDC components in each group as features. Similar grouping can be performed for the diagonal scans, effectively reduce the dimensionality per scan direction to 192, which is 1/8 of the original one. The total number of feature elements in case of 8 scan directions and 3 layers is:

8 (scan directions) × 8 (groups) × 3 (layers) × 8 (PDC dimensions) = 1536 dimensions

Averaging not only reduces feature dimensions, but also enhances robustness against noise and minor disturbance. Another way to reduce feature dimensions is to merge PDC components with opposite directions:

$$d_i = (l_i + l_{i+4}) / \sqrt{\sum_{k=1}^{4} (l_k + l_{k+4})^2}$$

where i = 1,…,4. Thus, the PDC dimensions per edge point are reduced by half, giving a total of 768 features for one character. In general, the total number of feature elements, $n$, is

$n_1$ (scan directions) × $n_2$ (groups) × $n_3$ (layers) × $n_4$ (PDC dimensions) = $n$ dimensions

where $n_1, n_2, n_3, n_4$ are the number of scan directions, of groups, of layers, and PDC dimensions, respectively.

### II-2. Reduction of Feature Redundancy

As we can see from the above discussion, the total dimensionality of PDC features is on the magnitude of $10^3$. Such high dimensionality requires large memory and computation power hence is costly in practical applications. More importantly, it is very difficult to accurately perform classification and recognition in high dimension with high robustness against noise or small variations. The redundancy among feature elements makes it hard to identify what features are most relevant in terms of differentiating one character from others and what are unimportant. It is necessary to perform redundancy reduction and to select relevant features. This step, known as feature compression in some literature, can be done via K-L transform. Specifically, we first compute the covariance matrix of features derived from training samples, then compute and rank its eigen values. The eigen vectors of the first $l$ largest eigen values form a transform matrix $\boldsymbol{T}$, where $l << n$. This transform matrix $\boldsymbol{T}$ is used to transform the original $n$-dimension PDC feature to an $l$-dimension one. In our algorithm, $l$ is chosen to be 128. The new, transformed feature with much lower dimensionality will be used in the subsequent matching/recognition. This eigen analysis also provides information regarding how important a feature is for differentiating one character from others. In particular, considering the eigen vectors that are used to form the transform matrix $\boldsymbol{T}$ and to generate $l$ new feature elements, the larger the associated eigen value is, the more important the corresponding transformed feature element is.

### II-3. 3-level Matching

Like many other OCR systems, our recognition algorithm has two stages, namely, training and testing. A set of character bitmap are used as training samples to (1) generate the K-L transform matrix $\boldsymbol{T}$ for redundancy reduction, (2) create "dictionaries" which are essentially look-up tables that store the mapping between characters (often represented by indexes) and their typical feature values in the transformed domain, (3) other parameters related to recognition. After training, the system is switched to testing stage and performs matching with respect to the dictionaries to determine the most likely character index of a given character bitmap. The matching is based on certain distance measure and may be in a weighted fashion to put emphasis on features that are

more relevant to character shapes. The weights are among the parameters to be determined during training.

We propose a 3-level hierarchical matching from coarse to fine. The first level is a coarse classifier, utilizing a small number of transformed feature components associated with the largest eigen values and yielding a set of candidates. In our algorithm, we use 24 feature components to build the coarse classifier, and keep 120 candidates that have the smallest distance from the 24-dimension feature vector of the character bitmap to be recognized. These 120 candidates are then passed to a fine classifier in the second level. The fine classifier uses more feature components and outputs a smaller number of candidates. In this particular case, 48 feature components are used to narrow down the candidates to 24. The third level will use all of the $l$=128 feature components, selecting the closest character among the 24 candidates as the final recognition result.

The hierarchical structure in the 3-level matching not only reduces computational complexity but also put emphasis on significant feature components. The latter enhances recognition accuracy. The effectiveness of 3-level matching in terms of recognition rate, speed, and memory use has been demonstrated in our earlier work [1].

## III. NEW MULTI-DICTIONARY MATCHING METHOD

To recognize fonts with small difference, such as the fonts of Song, Fang, Kai, and Hei popularly used in main body of Chinese documents (see Fig.1-1), the 3-level matching with one dictionary achieves high recognition rate and speed. However, as more diverse fonts being included, the recognition rate decreases. One way to improve the performance is to divide the fonts into groups, each of which consists of fonts with similar appearance, and create one dictionary for each group. Our experiments show that the recognition rate is not improved much if the minimum of the Euclidean distances across dictionaries is used by the final decision module to combine the matching results from multiple dictionaries. On the other hand, if we could first determine to which font dictionary an input character bitmap belongs and take this into account when combining the matching results from multiple dictionaries, the recognition rate is expected to be improved. This idea motivates us to propose a new matching algorithm with multiple dictionaries utilizing the estimated font information from neighborhood.

Assume that the i-th dictionary is used by a 3-level matching module to generate preliminary recognition result ("$R_i$") with the confidence of such recognition being measured by the matching distance ("$D_i$"). The final recognition result is given by a decision module that combines the preliminary results. To design the combining rule, we assume that the characters in a document are sequentially recognized according to the document layout and that the nearby characters are more likely to be in the same fonts. We introduce $C$ as the number of characters satisfying the condition "D1 < D2" among $m$ most recently recognized characters. $C$ is then compared with two threshold $t1$ and $t2$ where $t1 > t2$, and the final recognition result is determined according to the following formula:

$$\text{Result} = \begin{cases} \text{R1} & \text{if } C \geq t1 \text{ or } (\, t1 > C > t2 \ \ \& \ \text{D1} \leq \text{D2}\,) \\ \text{R2} & \text{if } C \leq t2 \text{ or } (\, t1 > C > t2 \ \ \& \ \text{D1} > \text{D2}\,) \end{cases}$$

That is, if $C \geq t1$, the font of the current character to be recognized is more likely to be in Dictionary-1, hence the recognition result is chosen to be the same as the preliminary result-1; similarly, if $C \leq t2$ , we take the result-2 as the final result because the font of the current character is more likely to be in Dictionary-2. When $C$ is between the two thresholds, it is likely that the fonts of nearby characters are inconsistent. In this case, we use the preliminary result corresponding to the minimum distance measure as the final recognition result.

## IV. EXPERIMENTAL RESULTS

Our experiments for evaluating the proposed recognition algorithm have two phases. The first phase creates recognition dictionaries using training samples, which are produced in two font sizes from laser printer and scanned with two resolution settings and three darkness settings. We then examine the recognition rates on 561 training samples and on 51 testing samples, respectively. Each sample includes a total of 3868 different characters. Among them, 3755 are Chinese characters and 113 are other characters. In the second phase, the proposed algorithm is incorporated as a recognition kernel into a practical OCR system. Using this practical OCR system, the recognition rates of real newspaper and magazine documents containing 4 most commonly used fonts (Song, Fang, Kai, and Hei) are examined. In these experiments, the feature dimensionality before redundancy reduction is 512.

### IV-1. Creation of multiple dictionaries

A few pre-processing steps after scanning are performed to facilitate the creation of multiple dictionaries, including binarization, segmenting individual characters from the document, and size normalization. The dictionary contains the transformed feature vector and its weighting vector for each character. The transformed vector of a character is obtained by applying the transform matrix to the average PDC feature of multiple training samples of the same character. The weighting vector associated with each character is derived from the reciprocal of the variance among the transformed vectors of all training samples of that character. A feature element is considered more important and is given higher weight if it is consistent among many samples.

Two dictionaries are created with different combinations of fonts, as indicated by "D1" and "D2" in Fig.1-1. Dictionary-1 is created from 385 sets of samples with a

font combination of Song, Fang, Hei, and Yuan. Dictionary-2 is created from 176 sets of samples with a font combination of Kai, Lishu, Weibei, and Xingkai.

### IV-2. Recognition experiment

Our recognition experiment follows the procedures discussed earlier in Sec.III. The redundancy among the 512-dimension raw features is reduced by the transform matrix, yielding new features of 128 dimensions. 3-level matching is performed using each of the two dictionaries and the preliminary results are combined by a decision module to output the final result. The matching uses the (squared) weighted Euclidean distance, namely,

$$L_2(F, \overline{F}_j) = \sum_{i=1}^{l} w_{ij}\left(f_i - \bar{f}_{ij}\right)^2$$

where $F$ is the transformed feature vector for the character to be recognized, $\overline{F}_j$ is the transformed feature vector of j-th character in the dictionary, $f_i$ is i-th element of the transformed feature vector for the character to be recognized, $\bar{f}_{ij}$ is i-th element of the transformed feature vector of j-th character in the dictionary, $w_{ij}$ is i-th element of the weighted vector of j-th character in the dictionary, and $l$ is the dimensionality of the transformed feature vector. Smaller distance implies that the two characters are considered more similar.

### IV-3. Experimental results

The recognition rates of the training samples that are used for creating dictionaries and of the testing samples that are not for creating dictionaries are shown in Table 4-1. Each test is performed on two sets of Chinese character samples with a total of 7510 characters. The recognition rates on practical documents by a complete recognition system using the proposed algorithm as its core are shown in Table 4-2. These results show that the average recognition rate of 8 fonts is 99.32% for training samples and is 98.96 % for testing samples, respectively. The lowest recognition rate for an individual font is as high as 98.75% for training samples and 98.15% for testing samples. When using the proposed algorithm as a kernel in a practical OCR system, the average recognition rate is 97.99% for practical documents that were taken from newspapers and magazines and contain 4 most commonly used fonts (Song, Fang, Kai, and Hei). The recognition speed is 57 characters per second on a Pentium-II 233 MHz PC.

## V. CONCLUSION

We have presented an algorithm for recognizing Chinese characters in 8 diverse fonts. K-L transformed Peripheral Direction Contributivity are used as features and the final recognition result is obtained via hierarchical matching with multiple dictionaries. We proposed a new multi-dictionary matching approach that utilizes the estimated font information from neighborhood text region, which helps improve the recognition accuracy without significantly increasing the computational complexity. Experimental results show that the proposed algorithm can recognize characters in as many as 8 different fonts quickly and accurately. In the mean time, the algorithm gives excellent recognition performance on 4 most commonly used fonts when adopted as the core of a practical OCR system.

**References**


[1] X-L. Wu and M. Wu, "An Experimental System of Multi-font Multi-Language Character Recognition", Proc. of National Conf. on Character Recognition and Document Analysis, China, 1996 (in Chinese).

[2] N.Hagita, S.Naito and I.Masuda, "Handprinted Kanji Characters Recognition Based on Pattern Matching Method", Proc. ICTP '83, pp.169-174, 1983.

[3] T.Akiyama and N.Hagita, "Automated entry system for printed documents", *Pattern Recognition*, vol. 23, no. 11, pp. 1141-1154, 1990.

[4] X-Q. Ding and Y-S. Wu, *Chinese Character Recognition — Fundamental, Method, and Realization*, Advanced Education Press, 1992 (in Chinese).


**Table 4-1 Recognition rates of training and testing samples (%)**

| Font | *Song* | *Fang* | *Kai* | *Hei* | *Yuan* | *Lishu* | *Weibei* | *Xingkai* | Average |
|---|---|---|---|---|---|---|---|---|---|
| Training | 99.82 | 99.64 | 99.81 | 99.57 | 98.77 | 98.75 | 99.35 | 98.82 | 99.32 |
| Testing | 99.71 | 99.50 | 99.80 | 99.09 | 98.43 | 98.15 | 98.78 | 98.19 | 98.96 |

**Table 4-2 Recognition rates of practical documents**

| Font | *Song* | *Fang* | *Kai* | *Hei* | Average |
|---|---|---|---|---|---|
| Number of documents | 149 | 252 | 225 | 164 | 197 |
| Total number of characters | 60165 | 98690 | 78264 | 21835 | 64738 |
| Recognition Rate (%) | 99.26 | 97.48 | 97.78 | 97.45 | 97.99 |