Journal of Electronic Imaging 7(1), 79–85 (January 1998).

Length estimation of digit strings using a neural network with structure-based features

Zhongkang Lu
Zheru Chi
Wan-chi Siu
Hong Kong Polytechnic University
Department of Electronic Engineering
Hung Hom, Kowloon, Hong Kong
E-mail: enzklu@en.polyu.edu.hk

Abstract. *Accurate length estimation is very helpful for the successful segmentation and recognition of connected digit strings, in particular, for an off-line recognition system. However, little work has been done in this area due to the difficulties involved. A length estimation approach is presented as a part of our automatic off-line digit recognition system. The kernel of our approach is a neural network estimator with a set of structure-based features as the inputs. The system outputs are a set of fuzzy membership grades reflecting the degrees of an input digit string of having different lengths. Experimental results on National Institute of Standards and Technology (NIST) Special Database 3 and other derived digit strings shows that our approach can achieve an about 99.4% correct estimation if the best two estimations are considered.* © *1998 SPIE and IS&T.* [S1017-9909(98)00901-5]

1 Introduction

Recently, many algorithms have been developed for off-line unconstrained connected characters recognition. According to the targeted problems, these algorithms can be categorized into two groups, those for recognizing connected character strings with a fixed number of characters[1–4] and those for recognizing character strings without knowing the length of each string (unfixed), where the segmentation and recognition are performed interactively.[5–12]

For techniques in the first group, character strings are first separated into isolated characters, which are then identified by a recognizer. Cheriet *et al.* presented a region-based background analysis algorithm to find a married pair of background valleys to separate a connected digit string.[1] On the other hand, a context-directed hierarchical algorithm was proposed by Shridhar and Badreldin to separate connected two-digit strings.[2] In their approach, the segmentation was done based on the analysis of horizontal border-to-background transitions. Yu and Yan recently proposed a recognition-based segmentation algorithm[3] in which a set of structure-based models together with several criteria were adopted to select the most promising one-touching point or pair of two-touching points at which a string was separated. Another contour-analysis-based algorithm was proposed in Ref. 4 in which contour information was used to find pairs of cutting points. Nine credits on contour features were then assigned to each pair of cutting points and the weighted sum of nine credits was used to prioritize cut links where the weights were trained using a genetic algorithm. Most of these techniques worked well on character strings with a fixed number of characters.

In the second group of techniques for recognizing character strings with variable lengths, the hidden Markov model (HMM), a segmentation-free approach, is used in most applications.[5–8,11] An HMM is a doubly stochastic process with an underlying stochastic process that is not observable but can be observed through another set of stochastic processes that produce the sequence of symbols.[13] It does not model the whole pattern or shape as a single feature vector. Instead, the HMM explores the relationships (transition probabilities) between consecutive segments of a pattern to be classified. For its applications, a word is first separated into a sequence of segments that are properly encoded. Then, a dynamic programming technique, such as the Viterbi algorithm, is used to find the path of the maximal probability from the sequence of segments. A word, which corresponds to the path, is finally extracted. A lexicon is often used in postprocessing to obtain a valid word from a set of candidates (paths) of high probabilities.[7,14] Another segmentation-free algorithm, a subgraph matching algorithm, was presented in Refs. 9 and 10. The algorithm extracts meaningful subgraphs (characters or digits) from the feature graph (word) by template matching. For these algorithms, the number of characters in a string can be determined only after they have been recognized. Obviously, the segmentation and recognition performance on character strings with variable lengths can be improved significantly if the number of the characters in each string can be satisfactorily estimated beforehand. After the length is determined, the segmentation and recognition problem of character strings of variable lengths can be simplified to the one of a fixed length, where a better performance can be expected. Unfortunately, little work has been reported on estimating the length of a character string.

Paper CIP-09 received Feb. 22, 1997; revised manuscript received June 23, 1997; accepted for publication Aug. 4, 1997.
1017-9909/98/$10.00 © 1998 SPIE and IS&T.

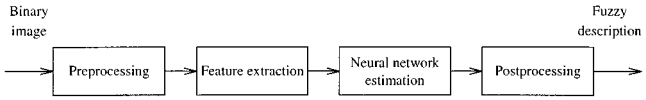


Fig. 1 Block diagram of the length estimation system for digit strings.

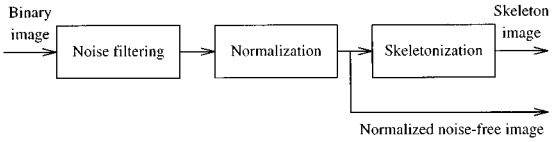


Fig. 2 Block diagram of the preprocessing step.

To solve the recognition problem of strings of variable lengths, an iterative segmentation-based algorithm was proposed by Fenrich.[12] In his algorithm, vertical projection and upper and lower contours were used to find splitting positions. In each iteration, a character was recognized and removed from the image. This process continued until all characters were recognized. In his paper, a length estimation method was presented, which was based on constrained linear regression (CLR) by using the image density as the independent variable and the number of recognized characters as the dependent variable. Since characters were removed as they were recognized, their effective contribution to the image density could be measured. The measured values were then used to establish a least-squares linear model, which could be used to estimate the number of characters remaining in the image.

In this paper, we present a neural network solution to the length estimation of digit strings, which is an important part of our automatic off-line digit recognition system. The outputs of the neural network estimator are further processed so that a set of fuzzy membership grades are obtained to reflect the degrees of an input digit string of having different length values. These membership grades will be very helpful for the digit segmentation and recognition at a later stage.

As shown in Fig. 1, our length estimation approach mainly includes four steps, preprocessing, feature extraction, neural network estimation, and postprocessing. Preprocessing plays an important role in digit separation as well as in digit recognition. In this step, morphological operators together with if-then rules are applied to remove noise in binary digit string images. The detail of the preprocessing is presented in Sec. 2. A set of structure-based features from the normalized noise-free image and its skeleton image are extracted in the step of feature extraction, which is discussed in Sec. 3. A multilayer feedforward neural network is then trained in step 3 to perform the length estimation based on the extracted features. Section 4 discusses data collection and the training of the neural network estimator. In Sec. 5, we explain how to derive fuzzy membership grades from the outputs of the neural network estimator. Experimental results on National Institute of Standards and Technology (NIST) Special Database 3 and other derived digit strings are reported in Sec. 6. Finally, conclusions are drawn in Sec. 7.

2 Preprocessing

Preprocessing includes three substeps: noise filtering, normalization, and skeletonization, as shown in Fig. 2. Two images are output from each input image, the normalized noise-free image and the skeleton image. Both images are used to extract structure-based features for the length estimation of digit strings.

2.1 *Noise Filtering*

Mathematical morphology, which can be used to handle geometrical features in an image, has been studied for about three decades. Using *a priori* geometrical information, features in an image can be extracted, suppressed, or preserved by applying morphological operators.[15,16] Mathematical morphology consists of set transformations that transform a set into another set. These transformations are carried out via the use of a structure element that contains the desired geometrical structure. Various interactions of the original set with the structuring element form the basis of all morphological operations.[17]

There are four basic morphological operators: erosion, dilation, opening, and closing. In our paper, the opening and closing operations are used to extract noise from an original image.

Let I be an original image and let N be the noise. We have

$$N = I - (I \circ B) \cdot B, \tag{1}$$

where $\circ$ denotes the opening operation, and $\cdot$ denotes the closing operation. The selection of the structure element B is a key problem in mathematical morphology analysis, because there exist a huge number of potential structure elements for the vastness of the universe of all possible object shapes, and our experience shows that it is difficult to choose a suitable single structure element for filtering a variety of noise. The selection is very much dependent of the property of the original data and the task that we want to carry out. Preliminary experiments suggested that choosing a 3×3 lattice for B could achieve good performance in removing the possible noise in digit string images.

Figure 3(a) shows a few examples of digit string images. Potential noise (dark regions) extracted by morphological operations are shown in Fig. 3(b). The potential noise regions include:

1. burrs at the border of foreground due to image scanning and binarization
2. small holes from the morphological operations
3. large artifacts from writing boxes
4. narrow strokes due to poor writing, such as the narrow part at the top of the 2 in the fourth digit string in Fig. 3(b).

Burrs and large artifacts should be erased, small holes should be ignored, and narrow strokes should be kept to avoid broken strokes. Production rules are used to distinguish a noise region from other foreground regions according to its size, shape, and the number of connected foreground regions.

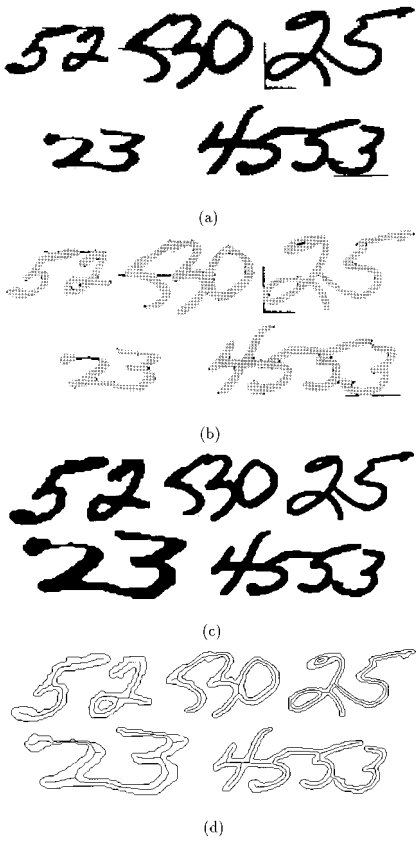


Fig. 3 Preprocessing of binary digit string images: (a) original binary images, (b) the potential noise (dark regions) extracted from the original binary images, (c) the normalized digit strings after removing noise, and (d) the skeleton images.

Let the size of a potential noise region f_n be given by its total pixel number, let the shape of the region f_s be defined as the ratio of the number of boundary pixels to the number of inner pixels, and let the number of connected foreground regions be denoted by f_p. A potential noise region with $f_n<5$ is removed immediately without any further processing. However, three rules are used to distinguish larger noise regions from the other foreground ones. Assume that T_n and T_s are thresholds. The three rules used are:

1. If $f_n<T_n$, then it is a noise region.
2. If $f_s>T_s$, then it is a noise region.
3. If $f_p\leqslant 1$, it is a noise region; otherwise, it is a foreground region.

We associate a fuzzy membership function with each of the first two rules. For rule 1, we adopt a membership function as shown in Fig. 4(a) where T_n is set to 10:

$$m(f_n)=\begin{cases} 1.0: & f_n\leqslant T_n(1-\alpha_n/2) \\ 0.0: & f_n\geqslant T_n(1+\alpha_n/2) \\ \dfrac{T_n(1+\alpha_n/2)-f_n}{T_n\alpha_n}: & \\ \quad T_n(1-\alpha_n/2)<f_n<T_n(1+\alpha_n/2), & \end{cases} \tag{2}$$

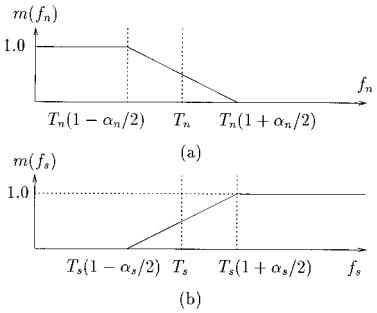


Fig. 4 Two types of membership functions used for the rules for removing noise.

where α_n is an extension factor that is set to 1.0. For rule 2, we use a membership function as shown in Fig. 4(b) where T_s is set to 1.3:

$$m(f_s)=\begin{cases} 1.0: & f_s\geqslant T_s(1+\alpha_s/2) \\ 0.0: & f_n\leqslant T_s(1-\alpha_s/2) \\ \dfrac{f_s-T_s(1-\alpha_s/2)}{T_s\alpha_s}: & \\ \quad T_s(1+\alpha_s/2)<f_s<T_s(1+\alpha_s/2). & \end{cases} \tag{3}$$

Again, α_s is an extension factor that is set to 0.15. For rule 3, $m(f_p)=1$ if $f_p\leqslant 1$; otherwise $m(f_p)=0$.

Let P_n represent the degree that a region belongs to noise, and P_n is given by

$$P_n=\alpha m(f_n)+\beta m(f_s)+\gamma m(f_p), \tag{4}$$

where weights $\alpha=\beta=0.1$ and $\gamma=0.8$. If $P_n\geqslant 0.5$, the region is considered to be noise and can be removed.

2.2 *Normalization*

The second part of the preprocessing is normalization, which is necessary because the original images are usually of different sizes, as shown in Fig. 3(a). One normalization scheme is to scale a foreground image to touch all four sides of the predefined box and the other is to touch top-bottom sides or left-right sides only. The first normalization scheme is rarely used because it may introduce deformation. In our paper, the length of a digit string is unknown, so the second normalization scheme with a predefined height will do a better job. Figure 3(c) shows the normalized noise-free images where all the foreground images are scaled to the same height.

2.3 *Skeletonization*

The last part in the preprocessing is skeletonization. The task here is to find a thinned representation of a foreground image. The skeleton image together with the normalized noise-free image are used to extract features for estimating the length of the digit string. The skeletonization algorithm is adapted from Hilditch's thinning algorithm,[18] which depends on the ''hit-or-miss'' transform.[18,19] The shortcoming of this algorithm is that it produces artifacts (buds) or extra branches during the processing. The extra branches are usually short in comparison with the other strokes.

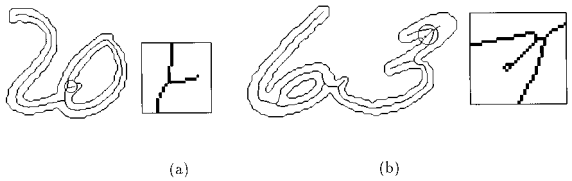


Fig. 5 Artifacts generated from the Hilditch's thinning processing: (a) an artifact due to poor writing and (b) an artifact due to the hit-or-miss transform.

There are two types of artifacts, one from the poor writing in the form of a short line, as shown in Fig. 5(a), and the other generated by the ''hit-or-miss'' transform in the form of a straight line with a small circle at one end and acute angle(s) to the neighboring stroke(s), as shown in Fig. 5(b).

Figure 3(d) shows the final skeleton images. We can see that not all artifacts should be removed. Some long artifacts are kept because they may indicate the sharp corner points of the strokes, which is useful for the later processing. The decision of whether or not to remove an artifact is made based on its length and shape and its angles to the neighboring branches.

3 Feature Extraction

Feature extraction is an important step for a good pattern recognition system. The choice of features should match the requirements of the chosen classifier for the task to be carried out. The extracted features should be invariant to the expected distortions and variances. Moreover, with a limited training set, the number of features should be kept reasonably small if a statistical classifier is to be used.[20]

In this algorithm, 17 structure-based features were extracted to train a multilayer feedforward neural network to perform the length estimation of digit strings. Among them, 10 features are horizontal transitions extracted from the normalized noise-free image, and 6 features come from feature points extracted from the skeleton image, and the aspect ratio of the image.

3.1 *Horizontal Transitions*

Horizontal transitions were proposed for the segmentation of handwritten digit strings by Shridlar and Badreldin.[2] As shown in Fig. 6, the approach is to scan the image horizontally and to count the number of foreground-background and background-foreground transitions. The whole image can be divided into 10 bands from top to bottom, and the average number of horizontal transitions in each band is used as a feature (10 features in total).

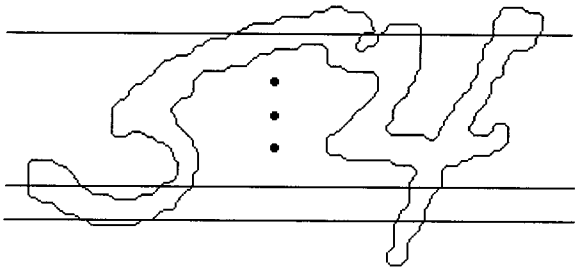


Fig. 6 Horizontal transitions of a digit string image.

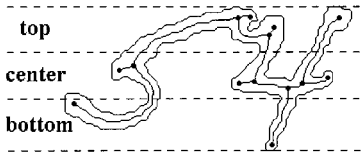


Fig. 7 Feature points in a skeleton image.

3.2 *Feature Points*

Fork points and terminal points in the skeleton image are important feature points. As shown in Fig. 7, a fork point is a skeleton point that has more than two connected branches and a terminal point is a skeleton point that has only one connected branch, that is, at the end of a stroke. We can see that some artifacts are deliberately kept to indicate sharp turning corners. The whole image is partitioned into three bands: top, center, and bottom bands. Two features, the number of the fork points and the number of terminal points, are extracted from each band. Therefore, six features in total are obtained.

4 Data Collection and Training of a Neutral Network Estimator

The data we used for experiments were extracted from NIST Special Database 3. In total, 4,555 connected two-digit strings, 355 connected three-digit strings, and 48 connected four-digit strings are available in the database. In addition, we also extracted 20,852 isolated digits from the database. Obviously, the numbers of the connected three- and four-digit strings are too small as compared with those of the other two types. To solve the problem, we derived some three- and four-digit strings by merging the existing shorter strings, as shown in Fig. 8, where a two-digit string 53 is merged with an isolated digit 3 to derive a connected three-digit string 533. The degree of overlapping O is set randomly within a prespecified range.

Finally, we chose 2000 samples of isolated digits, 1200 connected two-digit strings, 1200 connected three-digit strings, and 1200 connected four-digit strings as the training set. For the testing, we have an independent set of 2000 isolated digits, 3355 connected two-digit strings, 3355 connected three-digit strings, and 1200 connected four-digit strings. The strings of five or more connected digits are rare in real-world applications, so they are not included in our experiments.

An 18-60-4 three-layer feedforward neural network was trained to estimate the string length based on 17 features discussed in the previous section. Note that an extra input node with fixed activity is used to produce bias terms to the hidden nodes through the weight connections. Each of the four outputs of the neural network is assigned to each of four classes: isolated digit, two-digit string, three-digit

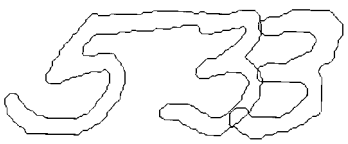


Fig. 8 Artificial three-digit string from merging a two-digit string with an isolated digit.

Table 1 The experimental results on the test set (CE: correct estimation).

	Isolated	Two-digit	Three-digit	Four-digit	Total	CE (%)
Isolated	1973	27	0	0	2000	98.7
Two-digit	19	3268	68	0	3355	97.4
Three-digit	0	55	3085	215	3355	92.0
Four-digit	0	0	70	1130	1200	94.2

string, and four-digit string. The neural network was trained using the backpropagation algorithm with the conjugate-gradient optimization technique.

5 Postprocessing

The outputs of the neural network are a set of continuous values in [0,1]. It will be more helpful for the later segmentation and recognition if the system can also provide a confidence level at which a string belongs to each of four categories. Assume that the output vector of the neural network for the i'th input digit string is $\mathbf{o}_i$. We first compute the center of the output vectors of the digit strings of the same length, that is,

$$\mathbf{c}_j = \frac{\Sigma_{k=1}^{N_j} \mathbf{o}_{i_k}}{N_j} \quad j = 1,2,...,L, \tag{5}$$

where L is the number of classes (four in our application), and N_j is the total number of digit strings in class j. A set of fuzzy membership grades is then derived for each digit string according to its Euclidean distances to the class centers:

$$d_{ji} = [(\mathbf{o}_i - \mathbf{c}_j)^T \times (\mathbf{o}_i - \mathbf{c}_j)]^{1/2}, \tag{6}$$

where d_{ji} is the distance from output vector $\mathbf{o}_i$ to center $\mathbf{c}_j$. Finally, the membership grade of digit string i belonging to class j, m_{ji}, is defined as

$$m_{ji} = \begin{cases} \dfrac{1/d_{ji}}{\Sigma_{l=1}^{L}(1/d_{li})}: & d_{ji} \neq 0 \\ 1: & d_{ji} = 0 \ . \end{cases} \tag{7}$$

6 Experimental Results and Discussion

Table 1 lists the results of the neural network estimation on the test set. The overall correct classification rate is 95.3%. We can see from the table that the misclassification occurs among the neighboring classes only, for example, a connected three-digit string might be misclassified into a two-digit string or a four-digit string.

Based on fuzzy membership grades, we introduce a measure λ, which is defined as the difference between the two maximum grades, that is,

$$\lambda = \max(m_{0i}, m_{1i}, m_{2i}, m_{3i}) - \text{second} \ \max(m_{0i}, m_{1i}, m_{2i}, m_{3i}). \tag{8}$$

Table 2 summarizes the classification of digit strings in terms of different λ values. Among the 9456 correctly classified digit strings, 70% of them were well classified ($\lambda \geqslant 0.5$), Concerning the 454 wrongly classified digit strings, about 88% were not completely misclassified, that is, the values of λ were smaller than 0.5, which means that an alternative choice could also be favorable. The length estimation is a part of our automatic digit recognition system and the results from this estimation serve only as a guideline for the later processing. If the value of λ is not large

Table 2 The analysis of the system outputs on the test set based on membership grades.

	Correct		Wrong		All	
λ	Number	%	Number	%	Number	%
0 to 0.1	234	2.47	167	36.8	401	4.05
0.1 to 0.2	352	3.72	100	22.0	452	4.56
0.2 to 0.3	411	4.35	70	15.4	481	4.85
0.3 to 0.4	648	6.85	37	8.15	685	6.91
0.4 to 0.5	973	10.3	25	5.51	998	10.1
0.5 to 0.6	1837	19.4	17	3.74	1854	18.7
0.6 to 0.7	1828	19.3	20	4.41	1848	18.6
0.7 to 0.8	2048	21.7	10	2.20	2058	20.8
0.8 to 0.9	862	9.12	5	1.10	867	8.75
0.9 to 1.0	263	2.78	3	0.66	266	2.68
Total	9456	100.0	454	100.0	9910	100.0

enough, say smaller than 0.5, both of the two highest outputs should be considered. As a result, among all these 9910 testing samples, 6838 (60%) samples were correctly classified with a high confidence level of $\lambda \geqslant 0.5$, 3017 (30.4%) digit strings had the correct estimation if the two best estimations were considered, and only 55 (0.6%) samples were poorly classified. The experimental results suggests that our approach of length estimation of digit strings is reliable and will be very helpful for the separation and recognition of digit strings at the later stage.

7 Conclusion

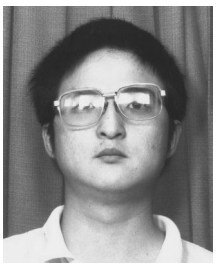
In this paper, the length estimation of connected digit strings using a two-layer feedforward neural network with structure-based features is presented. The approach consists of four steps, preprocessing, feature extraction, neural network estimation, and postprocessing. In the preprocessing step, morphological operations are first applied to remove noise and artifacts in digit string images, which are followed by the normalization and skeletonization. Two images, the normalized noise-free image and the skeleton image, are produced from the preprocessing step. In the feature extraction step, 17 structure-based features are extracted from preprocessed images. A multilayer feedforward neural network is then trained to learn the mapping between the features and the target lengths of digit strings. In the postprocessing step, the results from the neural network estimator are used to compute the class centers, which are then used to determine the fuzzy membership grades of a digit string belonging to different classes. Experimental results on NIST Special Database 3 and other derived digit strings show that only 55 (0.6%) of a total number of 9910 digit strings are poorly classified, which means that a high correct estimation rate has been achieved.

Acknowledgment

The authors would like to thank the Hong Kong Polytechnic University for financial support.

References

1. M. Cheriet, Y. S. Huang, and C. Y. Suen, ''Background region-based algorithm for the segmentation of connect digits,'' in *Proc. 11th Int. Conf. on Pattern Recognition*, pp. 619–622 (Sep. 1992).
2. M. Shridhar and A. Badreldin, ''Context-directed segmentation algorithm for handwritten numeral strings,'' *Image Vis. Comput.* **5**(1), 3–9 (1987).
3. D. Yu and H. Yan, ''Separation of touching handwritten numeral strings based on structural analysis,'' in *Proc. of Real World Computing Symp. '97*, pp. 238–245 (1997).
4. N. W. Strathy, C. Y. Suen, and A. Krzyzak, ''Segmentation of handwritten digits using contour features,'' in *Proc. ICDAR'93*, pp. 577–580 (1993).
5. M. Mohamed and P. Gader, ''Handwritten word recognition using segmentation free hidden Markov modeling and segmentation-based dynamic programming technique,'' *IEEE Trans. Pattern Anal. Mach. Intell.* **18**(5), 548–554 (1996).
6. K. Aas and L. Eikvil, ''Text page recognition using grey-level features and hidden Markov models,'' *Pattern Recog.* **29**(6), 977–985 (1996).
7. W. Cho, S. W. Lee, and J. H. Kim, ''Modeling and recognition of cursive words with hidden Markov models,'' *Pattern Recog.* **28**(12), 1941–1953 (1995).
8. A. Kaltenmeier, T. Caesar, J. M. Gloger, and E. Mandler, ''Sophisticated topology of hidden Markov models for cursive script,'' in *Proc. ICDAR'93*, pp. 139–142 (1993).
9. J. Rocha and T. Pavlidis, ''A solution to the problem of touching and broken characters,'' in *Proc. ICDAR'93*, pp. 602–605 (1993).
10. J. Rocha and T. Pavlidis, ''Character recognition without segmentation,'' *IEEE Trans. Pattern Anal. Mach. Intell.* **17**(9), 903–909 (1995).
11. M. Y. Chen, A. Kundu, and J. Zhou, ''Off-line handwritten word recognition using a hidden Markov model type stochastic network,'' *IEEE Trans. Pattern Anal. Mach. Intell.* **16**(5), 481–497 (1994).
12. R. Fenrich, ''Segmentation of automatically located handwritten numeric strings,'' in *From Pixels to Features III: Frontiers in Handwriting Recognition*, pp. 47–59, Elsevier Science (1992).
13. A. Kundu and P. Bahl, ''Recognition of handwritten script: a hidden Markov model based approach,'' in *Proc. ICASSP'88*, pp. 928–931 (1988).
14. R. Hoch and T. Kieninger, ''On virtual partitioning of large dictionaries for contextual post-processing to improve character recognition,'' *Int. J. Pattern Recog. Artifi. Intell.* **10**(4), 273–289 (1996).
15. J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, New York (1983).
16. R. M. Haralick, S. R. Stenberg, and X. Zhuang, ''Image analysis using mathematical morphology,'' *IEEE Trans. Pattern Anal. Mach. Intell.* **9**(4), pp. 532–550 (1987).
17. J. Song and E. J. Deep, ''The analysis of morphological filters with multiple structuring elements,'' *Comput. Vis. Graph. Image Process.* **50**, 308–328 (1990).
18. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, Boston (1992).
19. W. K. Pratt, *Digital Image Processing*, John Wiley & Sons, New York (1991).
20. Q. D. Trier, A. K. Jain, and T. Taxt, ''Feature extraction methods for character recognition—a survey,'' *Pattern Recog.* **29**(4), 641–662 (1996).



Zhongkang Lu received his BEng degree in biomedical engineering from Southeast University, Nanjing, China, in 1993, and his MEng degree in electrical engineering from Shanghai Jiaotong University, Shanghai, China, in 1996. Currently he is a PhD candidate in the Department of Electronic Engineering at the Hong Kong Polytechnic University. His research interests include pattern recognition and artificial intelligence.

Zheru Chi received his BEng and MEng degrees from Zhejiang University in 1982 and 1985, respectively, and PhD degree from the University of Sydney in March 1994, all in electrical engineering. Between 1985 and 1989, he was a faculty member with Zhejiang University. From 1989 to 1993, he was a PhD candidate working on artificial neural networks for ICEG classification at the University of Sydney. In April 1994 he joined the Laboratory for Imaging Science and Engineering at the same university, first as a senior research assistant and then as a research fellow, working on fuzzy models for image processing and pattern recognition. Since February 1995, he has been an assistant professor at the Hong Kong Polytechnic University. His current research interests include image processing, pattern recognition, neural networks, and fuzzy systems. He has coauthored a book and a book chapter and published more than 30 technical papers. He is a member of IEEE.

Wan-Chi Siu received the Associateship in electronic engineering from the Hong Kong Polytechnic University (formerly called Hong Kong Polytechnic), the MPhil degree in electronics from the Chinese University of Hong Kong, and the PhD degree in signal processing from the Imperial College of Science, Technology and Medicine, London, in 1975, 1977, and 1984, respectively. Between 1975 and 1980 he was with the Chinese University of Hong Kong as an electronic engineer. He then joined the Hong Kong Polytechnic University in 1980 as a Lecturer, and became a Senior Lec-

turer in 1985, a Principal Lecturer in 1987, and a Reader in 1989. He subsequently became Chair Professor and Associate Dean of Engineering Faculty in 1992, and has been Chair Professor and Head of the Department of Electronic Engineering of the same university since 1994. He has published more than 160 research papers. His research interests include digital signal processing, fast computational algorithms, transforms, high-performance signal processors, video coding, computational aspects of image processing and pattern recognition, and neural networks. He is a Chartered Engineer, a Fellow of the IEE, and a Senior Member of the IEEE.