From Pixels to Features III: Frontiers in Handwriting Recognition
S. Impedovo and J.C. Simon (eds.)

Combination of segmentation-based and wholistic handwritten word recognition algorithms

J.J.Hull, T.K.Ho, J.Favata, V.Govindaraju and S.N.Srihari

Center of Excellence for Document Analysis and Recognition, Department of Computer Science, State University of New York at Buffalo, Buffalo, New York 14260 USA, e-mail: hull@cs.buffalo.edu

Abstract

An algorithm for the recognition of unconstrained handwritten words is proposed. Based on an analysis of writing styles, it is shown that techniques for isolated character recognition, segmentation, as well as cursive script recognition are needed to achieve a robust solution to handwritten word recognition. A combination of these algorithms is proposed in which the decisions of each method are combined to generate a single consensus ranking of a dictionary of word alternatives. Preliminary results of the implementation of this methodology are given along with future research directions.

1. INTRODUCTION

An algorithm for handwritten word recognition must be able to successfully recognize the image of any word whether it is discretely printed, written cursively, or composed of a mixture of both styles. The writing styles that can be used to form a handwritten word are illustrated in Figure 1. Discrete characters, cursive fragments (groups of characters written with a single continuous motion), and complete cursive words are often used exclusively or in combination.

The algorithmic approach discussed in this paper is directed toward postal addresses. The handwritten words that occur in addresses are completely unconstrained by writer, style, instrument, size of text, placement within an image, and so on. However, one very important constraint is that the words typically come from a fixed vocabulary. For example, the name of a city may be one of over 30,000 possibilities. Also, if some digits of the postal code can be recognized, they can considerably reduce the size of the lexicon. Sometimes it may be possible to limit the choices for a city name to two or three candidates.

(a) (b) (c)

Figure 1. Styles used to form handwritten words: (a) discrete printing; (b) cursive fragment; (c) complete cursive word.

The rest of this paper includes an analysis of writing style on a database of city names. This analysis demonstrates that several methods, each specialized for a particular style of script, could be used simultaneously to recognize handwritten words. The proposed algorithm incorporates three classes of technique: character recognition, segmentation-based, as well as wholistic or whole-word cursive script recognition. The individual algorithms are then described as well as the status of their implementations. The results of various experiments are discussed and directions for future research are presented.

2. ANALYSIS OF HANDWRITING STYLE

The difficulty of the handwritten word recognition problem depends on the variability in style that occurs. The problem would be considerably easier if many words were discretely printed without touching characters than if a large number of words were cursively written with a sloppy writing style. The extent of the problem is also complicated by the unconstrained nature of a domain such as postal addresses wherein any style can potentially occur and the writer can be any member of the population. This differentiates the problem from many applications where writer training and feedback from a recognition algorithm to a writer is feasible.

To determine the extent of the handwritten word recognition problem and the need for alternate recognition strategies, a large database of handwritten examples of the same word was inspected and the variation in style was determined. The identity of the word was held constant to guarantee that any difference in style was not attributable to the truth value. A word of average length ("Buffalo")† was chosen for which many samples were available.

The word images used for this analysis were extracted from a set of handwritten address blocks (known as the *bl* images; for "buffalo-local") that were gathered from live mail at the Buffalo, New York Post Office. The only criterion for scanning an address and placing it in the bl image set was that the city name should be Buffalo. The objective was to obtain a random sample of addresses with the same city name.

† The average city name in the United States contains 6.97 characters.

The first 500 city names in the bl set were visually inspected and assigned to the following categories: discretely printed, fully cursive, broken cursive, and abbreviations. A word was called *discretely printed* if it was formed by printing nearly every character in the word, even if some of those characters touched one another. A *fully cursive* word was one that was formed by a single continuous motion of the writing instrument. A word image was classified as *broken cursive* if it was formed by more than one writing motion and it contained at least one cursively written component spanning more than one character. An *abbreviation* was any group of characters that did not form the complete spelling of "Buffalo".

The samples were further categorized into sub-groups based on their subjective quality and graphological formation. The discretely printed words were classified as "well formed" if most characters were not touching or if it was judged that a reasonably straightforward segmentation algorithm could successfully separate any touching characters. Also, the image should have been free of extraneous noise such as underlines. If any of these conditions were violated, the image was called "poorly formed." A similar procedure was applied to the fully cursive words. In this case, a word needed to be written relatively neatly and had to be free of any imaging defects to be called well formed. The broken cursive words and abbreviations were also assigned similar quality measures. Both these types of words were called "well formed" if all their letters were written so that they did not overlap and the individual letters were completely present. The lack of a quantitative measure of quality and the reliance upon human judgment in assigning quality measures is acknowledged. However, even under these conditions the results should still prove interesting. Figure 2 shows examples of each classification as well as the subjective grading.

The results of the study show that 21 percent of the words were discretely printed (12 percent well-formed, 9 percent poorly formed), 35 percent were fully cursive (10 percent well formed, 25 percent poorly formed), 31 percent were broken cursive (16 percent well formed, 15 percent poorly formed), and 14 percent were abbreviations (9 percent well formed, 5 percent poorly formed). It was perhaps surprising that about 47 percent of the images were judged to have been well formed. This is encouraging since it indicates that this portion of the problem could be solved by strategies that depend on stable input styles.

The formations of the broken cursive words are also interesting. A discretely written "B" followed by a fully cursive representation of "uffalo" was contained in 50 images or ten percent of the whole sample. Overall, 22 percent of the images were one of three different formations of broken cursive script.

These results are important primarily for the design of a handwritten word recognition algorithm. If the results extend to other words besides "Buffalo," then discrete character recognition could be applied in at least 40 percent of the cases and the recognition of cursive fragments to at least 76 percent of the words. It is also important to note that almost all of the abbreviations (13 percent of the sample) are one of only two different spellings. This is especially useful for a lexicon-based word recognition algorithm since abbreviations must be accounted for by such methods. This result indicates that it might be possible to enumerate nearly all of them within a lexicon.

Figure 2. Examples of writing styles used for the word "Buffalo"; a-c illustrate well formed discrete printing, d-f poorly formed discrete printing, g-i well formed cursive, j-l poorly formed cursive, m-r are different broken cursive forms, and s-x are abbreviations.

3. ALGORITHM DESIGN

The algorithm proposed for handwritten word recognition is illustrated in Figure 3. An input word image is first pre-processed to normalize for as many writer-dependent characteristics as possible. The word image is then passed to a global feature analysis stage in which those entries from a lexicon that are visually similar to the input word are located. Features such as the estimated number of characters in the input image are used in this step. The reduced lexicon and the original word image are then passed to three recognition techniques, each of which outputs a ranking of the lexicon. The three independent rankings are then combined to generate a single consensus ranking for the word.

The objective of this procedure is to independently focus each algorithm on the problem and combine their results to maximize recognition accuracy. A similar methodology has had good success for handprinted digit recognition [8] and machine-printed word recognition [7] and it is expected to extend to this domain.

A description of the individual recognition techniques used in this model follows along with examples of their performance. The preprocessing stages are discussed elsewhere [5].

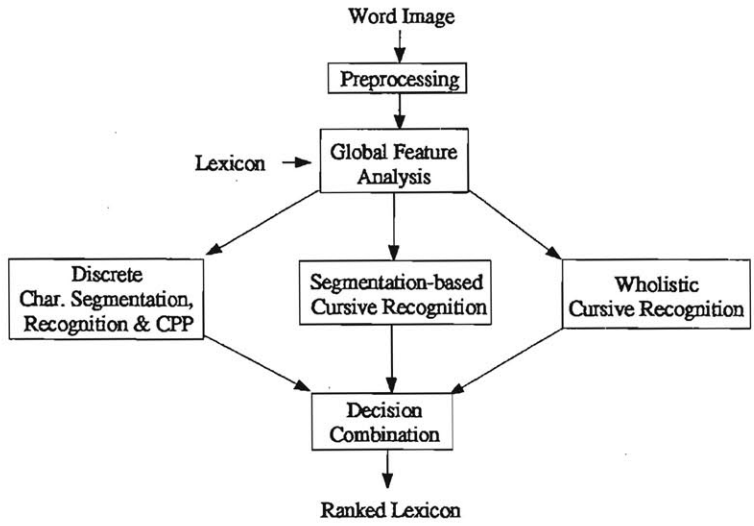

Figure 3. Design of the cursive script recognition algorithm.

4. DISCRETELY PRINTED WORD RECOGNITION

The algorithm specialized for discrete words first segments an image into isolated letters and passes the results to a character recognition algorithm. A number of top choices of the algorithm are given to a postprocessing routine that determines the confidence that each dictionary word matches the input image. The dictionary words are then output in sorted order by confidence value.

The segmentation algorithm is a version of a method used previously to segment postal codes [4]. A word image is recursively divided in halves vertically until it is determined that individual characters have been isolated. If the final recursive division level has been reached and it is determined that a component contains more than one character, specialized splitting routines are invoked. Similarly, if a component is of a significant size but is still too small to be a character, it is grouped with the appropriate nearby character.

The character recognition technique applies multiple algorithms to each image [9]. Four individual classifiers have been tested in the present system: binary template matching to a database of 19,249 prototypes, Bayesian classification of structural features, nearest neighbor matching of moment-based features, and polynomial discriminant recognition using a feature vector of 2083 pairs of pixels. The results of applying these methods to a testing database of 2173 handprinted characters are shown in Table 1. In all cases the training and test data were disjoint. The top choice correct recognition rates are shown as well as the performance in the top two choices.

These figures should only be used to compare the individual methods among themselves. The correct rates are relative to the variation in the test data and these methods will have different performance on other data sets.

algorithm	top choice		top 2 choices	
	N	percent	N	percent
Binary template matching	1979	91%	2083	96%
Structural-Bayes	1945	90%	2063	95%
Nearest neighbor with moments	1999	92%	2092	96%
Polynomial discriminant	1892	87%	2027	93%

Table 1. Performance of character recognition algorithms on a standard test set of 2173 isolated character images.

These results show that the nearest neighbor matching with moment features achieves the best correct recognition rate. The polynomial discriminant routine is three points lower in correct rate at the second choice. However, it has been observed in practice to be much faster.

The contextual postprocessing routine uses a regular expression matching algorithm. This technique uses the top 3 choices for each character to generate a set of regular

expressions that are matched to a list of dictionary words. The regular expressions are designed to use the character decisions to constrain the dictionary, that allows some fuzziness in the positions of the individual characters, and thus can tolerate some segmentation errors. Each constraint is associated with a score, which is assigned to a word when it matches that particular constraint. An example constraint looks like "(.?)[B].....(.?l..)", which says that a B is detected at a position close to the beginning of the word, with zero or one character preceding it, and 5, 6, or 7 characters following it. The words in the dictionary are graded by the scores they accumulate through matching these constraints. A ranking of the words is produced by this grading.

The discrete recognition routine was applied to 60 bl images that had been classified as well-formed and discretely printed. Only the polynomial character recognition was used and the results were matched to the dictionary of 4554 city names with seven characters. Smaller word lists that had been randomly chosen from the original file were also used. This was done to observe the performance of the algorithm as it might be used in a full system where additional information from the postal code or state name might constrain the size of the lexicon. In each case, if the random selection process excluded the correct word, it was inserted. The results of varying the dictionary size are shown in Table 2. It is seen that a correct rate of 73 percent at the top choice was achieved on the full dictionary. This improved to 91.7 percent when the dictionary size was decreased to 100 words.

dictionary size	correct rate					
	top choice		top 10 choices		top 50 choices	
	N	percent	N	percent	N	percent
4554	44	73.3	51	85.0	55	91.7
1000	47	78.3	56	93.3	56	93.3
500	49	81.7	56	93.3	56	93.3
100	55	91.7	56	93.3	57	95.0

Table 2. Performance of discrete recognition on 60 well-formed discretely printed words.

5. SEGMENT AND RECOGNIZE APPROACH

This technique recognizes handwritten words by exploring several alternative segmentations. A handwritten word is first preprocessed and a number of possible segmentation points are hypothesized. A recognition algorithm trained on cursively written characters is then applied to choose the segmentations that provide decisions with high confidence [3]. These decisions are then used to build a tree that gives the most likely interpretations of the word image. If the recognition results are correct, there will exist a path in the tree which contains the correct segmentation of the word and correct recognition of the underlying characters. Note that there may be several plausible segmentations (and the underlying plausibly recognized characters) of an unknown word which generate valid candidate words

in the language. Correct recognition in these cases requires a dictionary of words which allows the rejection of incorrect candidate words. In cases where there are several candidate words even after dictionary matching, additional context can be used to pick the most likely word.

The image preprocessing portion of this system takes a binary image as input and removes noise and smoothes contours. Additionally, word slant correction and baseline slant correction are performed. After smoothing, a number of features are identified and feature tables are built. Some of the features detected are ligatures, horizontal strokes, certain concavities and holes.

The segmentation and feature extraction stages of the approach operate by first estimating a number of alternative segmentation points based on an analysis of structural features. The image between two such points is then normalized to a size of r rows and c columns. Three combinations of r x c are used (24x16, 16x16, or 16x24). The choice is determined by the aspect ratio of the segmented sub-image. Such a sub-image is then compared to a database of templates for about 1000 cursively written characters. Each character is represented by eight individual r x c feature maps. The feature maps represent different features of a cursive character such as convexities pointing in various directions. The match score between a prototype character and a cursive character extracted from an input image is calculated by the following formula:

$$score(proto_i) = \sum_{j=1}^{8} \sum_{k=1}^{r} \sum_{l=1}^{c} weight\_match(\, proto[k][l], input[k][l] \,)$$

The *weight_match* function returns a positive constant if the pixels it is passed are both black, a negative constant if one of the pixels is black and the other white, and zero otherwise. The effect is to bias the response in favor of images that have many pixels in agreement with the feature maps.

After all plausible words are extracted from the image, dictionary matching eliminates unlikely (or invalid) words. The general matching criteria include length of the extracted word compared to the dictionary words as well as the number of correctly matching characters. Other criteria include recognition confidence values. In general, the matching must be flexible to handle incorrect character recognition but correct segmentation. We are extending this work to include spatial relationships between characters and the ability to generate local reference lines. This may be useful for very large dictionaries or poor handwriting styles.

Figure 4 shows some of the steps of the algorithm as it analyzes a word image. Starting at the beginning of the word, two strong responses are generated, as shown in (a): one for character "c" and one for character "a", at two different segmentation points. Also shown is the initial tree for the word after this cycle. The expansion of the node for character "c" from the previous cycle is shown in (b). Figure 4(c) shows the expansion from the character "a", (d) shows the final graph after all expansions have been performed, and (e) shows the possible words that could be extracted from this graph. The correct word could be determined by a dictionary lookup process.

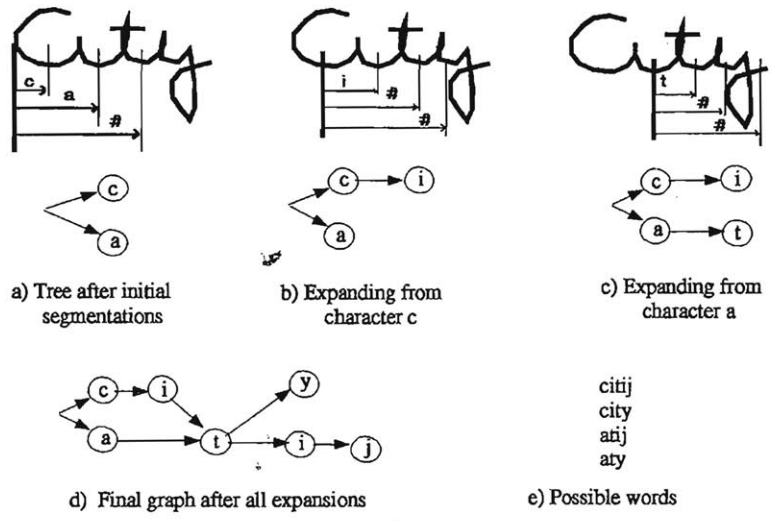

Figure 4. Example of segmentation-based recognition

This system was used in an experiment to recognize 37 handwritten city words extracted from real mail. These words were composed of fully cursive and broken cursive words from the *bs* data set, which contains addresses from a nationwide sample. Using a 32 word lexicon to postprocess the results, a top-choice correct rate of 83 percent was obtained. The performance in the top two choices was 91 percent correct and in the top five choices, performance was 94 percent correct. This demonstrates an initial implementation of this methodology. Future work will be oriented toward increasing the lexicon size and maintaining a high correct recognition rate.

6. WHOLISTIC WORD RECOGNITION

A technique for wholistic word recognition is being investigated that is similar to a method for on-line word recognition that matched the chain code from the initial portion of a word to a probabilistic representation [2]. In our approach, the contour of a word is first traced to generate its chain code. Structural features are then extracted from the chain at points of significant curvature change. Comparisons are then performed between the extracted feature string and positional probability vectors that represent dictionary words. Each comparison results in a confidence value that the input image is recognized as the corresponding dictionary word. The dictionary is output in sorted order by confidence value.

The eight structural features that are extracted from a contour are described in Figure 5 along with an example of each one. These features are similar to those defined in [1] and are called Spur, Stub, Wedge, Curl, Arc, Null, Inlet, and Bay. Each feature is also described by its location (in a 4x4 grid imposed on the word image) as well as its direction (quantized into one of eight values 0..7). The 4x4 grid divides a word so that the top and bottom horizontal regions usually contain the ascenders and descenders.

For the purpose of obtaining a global filter that ranks the words in the dictionary the eight structural features and eight directions are combined into two classes each. Thus, their combination yields four types of features in each cell of the grid. Spur, Stub, Wedge, Curl, and Arc are the convex features and Null, Inlet, and Bay are the concave features. Directions 0 to 3 are upward facing and directions 4 to 7 are downward facing. We construct a feature vector of size 4 x 16 = 64 (each of the 4 types of features can be attached to any one of the 16 grid positions). Thus, each position in the vector v[i], i=0,...,63, is mapped to a particular feature (1 of 4) and a particular position (1 of 16). For example, convex-down at grid position x0y3 is mapped to v[4].

Suppose, there are M words in a dictionary and there are N samples of each one. During training, a positional feature vector v is created for each of the MxN words. v[i] gives the number of times a particular feature type occurs at a particular position, e.g., v[4] gives the number of times convex-down occurs in position x1y1. Given a "test" word image, the conditional probability is computed that the feature vector of the test image matches each class.

A limited experiment was performed to study the feasibility of this approach. A data set containing 35 cursively written city names was used. Their distribution follows: Buffalo (17), Wilmington (9), Washington (4), Portland (3), Honolulu (2). The ability to construct a "filter" for the word Buffalo was tested. Each of the images was compared, in a leave one out fashion, versus only the Buffalo images. It was observed that a simple linear threshold on the distance between an unknown image and the closest prototype in the training data was sufficient to correctly classify 14 of the Buffalo images with one error. The number of correctly classified Buffalo images could be increased to 16 with a cost of 4 errors. A similar experiment was performed with the Wilmington images. Eight out of nine were correctly classified with nine errors. It was also possible to correctly classify five out of nine Wilmington images with three errors.

Both results indicate that the wholistic technique might be suitable as a filtering process. The objective would be to reduce a large dictionary to a small number of candidates that would be further processed by more powerful recognition algorithms.

This experiment was limited in scope because the investigation of the wholistic method is in its preliminary stages. Initial results are encouraging and will lead to continued work on this technique. This method was intended for use mostly with fully cursive words. However, its application to cursive fragments will also be explored. Issues that will need to be addressed include training. An alternative method that uses a structural representation of strokes will also be explored.

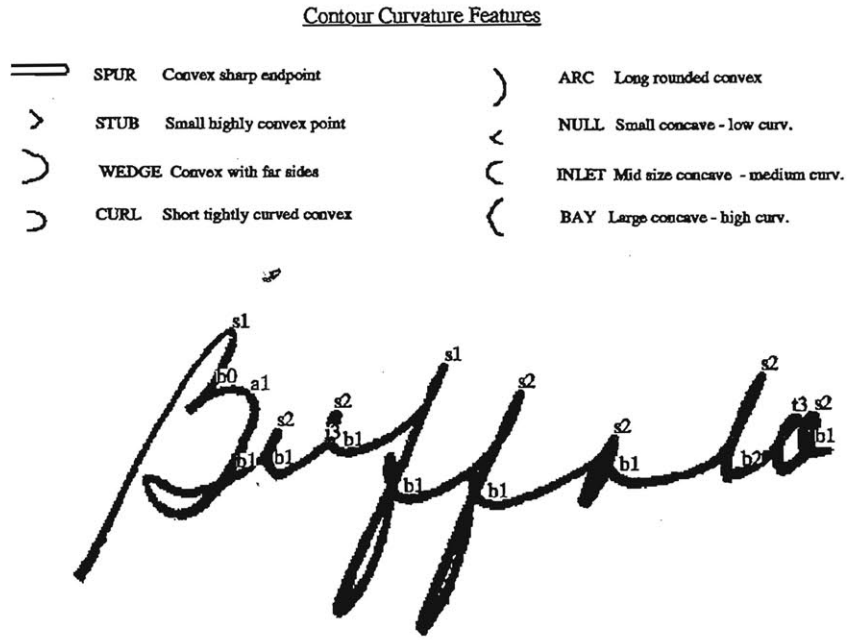

Figure 5. The contour curvature features and an example of their location in a cursive script word. Only the first 20 features that were located are shown.

7. DECISION COMBINATION

The combination of the decisions provided by the classifiers in this system is a topic of ongoing research [6]. The classifiers to be combined are dynamically selected according to the type of input, that is, whether the input word is discretely printed, composed of cursive fragments, or completely cursive. If the type classification cannot be confidently determined, all the classifiers are applied and combined. After feature matching, each classifier outputs a ranking of the dictionary. The combination is a two-step process involving reduction and reordering of the dictionary. A small number of top decisions are extracted from each ranking. A union of the words in these top decisions is computed. A rank combination function is then applied to the union to derive a consensus ranking. One method that has proven very useful is the *Borda Count*. This technique was originally developed to combine the decisions of a committee of experts, each of which provides a rank order of choices. The Borda count for each word is computed as the sum of the distance of that word from the bottom of each ranking. It is a measure of agreement among the classifiers on a single word. The best decision is the word with the maximum value.

8. DISCUSSION AND CONCLUSIONS

A comprehensive approach for handwritten word recognition was outlined. A study of handwriting style demonstrated that the abilities to recognize discretely printed characters, cursive segments, and fully cursive words are needed to successfully recognize any handwritten word in a postal address.

An algorithmic approach was presented that is suitable for such a domain. Three types of algorithm are applied in parallel to each word where each method is a specialist for a particular style of writing. Some early results for each method were presented. Future work will include more complete implementation and testing of the individual techniques as well as experimentation with various methods for combining their outputs. An alternate segmentation-based strategy is also being evaluated.

Acknowledgements

Prof. Amlan Kundu contributed valuable comments and Richard Fenrich and Xin Zhao provided various programs. The authors gratefully acknowledge the support of the Office of Advanced Technology of the United States Postal Service (USPS). Gary Herring and Carl O'Connor of USPS and Dr. John Tan of Arthur D. Little, Inc. provided useful advice.

9. REFERENCES

1 D. D'Amato, L. Pintsov, H. Koay, D. Stone, J. Tan, K. Tuttle and D. Buck, "High speed pattern recognition system for alphanumeric handprinted characters," *Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing*, Las Vegas, Nevada, June 14-17, 1982, 165-170.

2 R. F. H. Farag, "Word-level recognition of cursive script," *IEEE Transactions on Computers C-28*, 2 (February, 1979), 172-175.

3 J. Favata and S. N. Srihari, "Recognition of Cursive Words for Address Reading," *Fourth USPS Advanced Technology Conference*, Washington, D.C., November 5-7, 1990, 191-206.

4 R. Fenrich and S. Krishnamoorthy, "Segmenting diverse quality handwritten digit strings in near real-time," *Proceedings of the Fourth USPS Advanced Technology Conference*, Washington, D.C., November 5-7, 1990, 523-537.

5 V. Govindaraju and S. N. Srihari, "Preprocessing for handwriting recognition," *Second International Workshop on the Frontiers in Handwriting Recognition*, Bonas, France, September, 1991.

6 T. K. Ho, J. J. Hull and S. N. Srihari, "Combination of Structural Classifiers," *IAPR Workshop on Syntactic and Structural Pattern Recognition*, Murray Hill, New Jersey, June 13-15, 1990, 123-136.

7 T. K. Ho, J. J. Hull and S. N. Srihari, "Word recognition with multi-level contextual knowledge," *International Conference on Document Analysis and Recognition*, Saint Malo, France, September 30-October 2, 1991.

8 J. J. Hull, S. N. Srihari, E. Cohen, C. L. Kuan, P. Cullen and P. Palumbo, "A blackboard-based approach to handwritten ZIP Code recognition," *International Conference on Pattern Recognition*, Rome, Italy, November, 1988, 111-113.

9 J. J. Hull, A. Commike and T. K. Ho, "Multiple algorithms for handwritten character recognition," *International Workshop on Frontiers in Handwriting Recognition*, Montreal, Canada, April 2-3, 1990, 117-130.

FROM PIXELS TO FEATURES III

Frontiers in Handwriting Recognition

edited by

S. IMPEDOVO
Dipartimento di Informatica
University of Bari
Italy

J.C. SIMON
Emeritus of University Pierre et Marie Curie – Paris VI
France

1992

NORTH-HOLLAND
AMSTERDAM • LONDON • NEW YORK • TOKYO