Context-directed segmentation algorithm for handwritten numeral strings

M Shridhar and A Badreldin*

A context-directed algorithm is proposed for segmenting connected numeral strings into their components. The algorithm is hierarchical (tree-like structure) in the sense that it tests various hypotheses ranging from the case where the numerals are completely isolated to that where the numerals may be connected, touching and/or existing in overlapping fields. Test results indicate that the algorithm is very effective in providing an accurate segmentation in a form suitable for further processing by a recognition algorithm.

Keywords: machine recognition, segmentation, handwritten numerals

Machine recognition of handwritten numerals has been the subject of extensive investigation for nearly two decades. Many algorithms utilizing topological features, templates, features derived from transformations (Fourier, moments, Walsh, Karhunen–Loeve transforms, etc.)[1] have been proposed for numeral recognition systems. High accuracies, exceeding 98%, have been reported for these systems.

However, most of these algorithms assume that the numerals in a given field are completely isolated and possibly unbroken as well. An analysis of handwritten numeral specimens (such as those in postal codes, social security numbers, etc.) reveals that the numerals in a field are often connected, overlapping and/or touching. In addition, the individual numerals in a string could be broken.

An extensive literature survey revealed some work in the field of cursive script analysis and recognition; however, there has been a singular lack of significant work in the area of connected numeral string recognition.

A major difficulty in the case of connected numerals is the derivation (extraction) of the features of the individual numerals in the string. In view of the many possible ways in which the numerals may be connected (touching, overlapping, etc.), it is preferable to segment the string into its individual numeral components. If successful segmentation has been achieved, then it is relatively easy to recognize the isolated numerals. It must be noted that further processing may be needed to filter out the pseudofeatures generated by the connecting tail.

In this paper, a context-directed algorithm is proposed for segmenting connected strings of numerals into their components. The algorithm is hierarchical(of a tree-like structure) in the sense that it tests various hypotheses ranging from the case where the numerals are completely isolated to that where the numerals may be simply connected, touching and/or occurring in overlapping fields. The algorithm is derived on the basis of the following assumptions:

- The number of numerals in a string is known *a priori*.
- The heights of the numerals constituting a string are nearly equal.

DATA COLLECTION AND PREPROCESSING

Twenty volunteers varying in age from 7 to 50 years and coming from different countries were approached for the collection of numeral specimens. The specimens consisted of about 10 000 samples of two- and three-digit strings. The collected data was then studied carefully and grouped together on the basis of the similarity in shape of the various strings. These groups were then reproduced on a white sheet of paper with a felt pen.

Department of Electrical and Computer Engineering, University of Michigan-Dearborn, Dearborn, MI 48128–1491, USA
*Instrumentation Department, General Motors Research Laboratories, Warren, MI 48090-9055, USA
This work was done while the authors were with the Department of Electrical Engineering, University of Windsor, Windsor, Canada N9B 3P4

Figure 1 displays a sample of the data used. The sheets containing the numeral strings were imaged by a Hamamatsu Vidicon camera and digitized into images of 128 × 128 pixel resolution with each pixel represented by one of 256 grey levels (8-bit words) ranging from dark (0) to bright (255). The digitized images were stored on a Nova 840 minicomputer system. The images were then processed by a fast border-following algorithm[2] that consisted of the following steps:

- Thresholding to obtain a binary image. Since the numerals were reproduced on white sheets of paper and using a felt pen, a threshold value of $T = 110$ was adequate for all the images to produce binary strings without broken edges.
- Border following to derive all the contours of the numerals in the string. In the case of several contours, the first detected border is assigned a label value 4, while other borders are labelled as 6.
- Enclosing the numeral string with a rectangular frame of height H and width W.

SEGMENTATION ALGORITHM

The algorithm is developed by testing the validity of certain assumptions about the numerals in a given string. It is assumed, without loss of generality, that there are only two numerals in a string. The following are the various tests, presented in an ascending order of complexity.

Test for isolated and well separated numerals

Two numerals are considered to be isolated if a vertical line separating the two numerals can be found.

The assumption that the two numerals are isolated is tested first by initiating a vertical scan starting in the middle of the frame, and counting the number of transitions, VT, from bright (boundary) to dark (character).

If VT = 0, then the two numerals are isolated and hence are already in a segmented form.

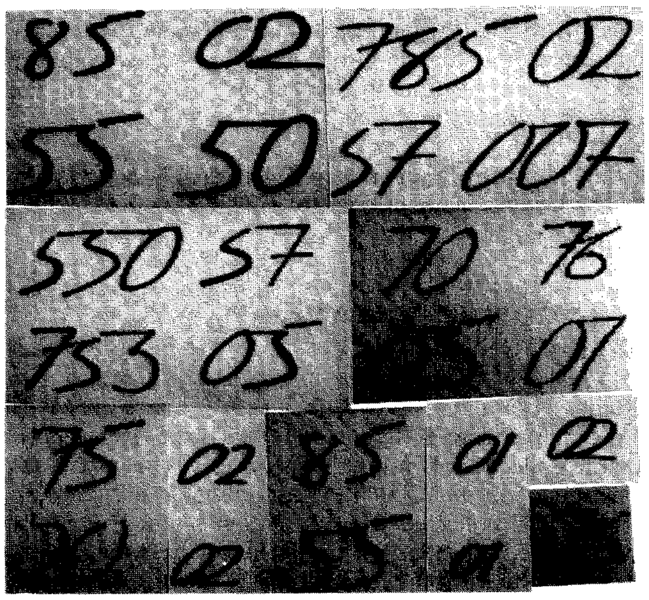


Figure 1. Samples of numeral strings

In the case VT ⩾ 1, initiate a vertical scan at a point one pixel to the left of the centre of the frame and count the number of transitions. If VT is still greater than or equal to 1, start the vertical scan at a point one pixel to the right of the centre of the frame. This procedure is repeated by moving further to the left or to the right and initiating a new scan until VT = 0. Figure 2 displays a string with two isolated numerals and the corresponding decision boundary.

If the procedure fails, that is, VT is still greater than or equal to 1, after a prescribed number of vertical scans, ε_v (e.g. ε_v = 3 pixels), then a test for nonisolated numerals is implemented.

Test for nonisolated numerals

If the numerals in a string are connected or touching or if they exist in overlapping fields, the above test will fail. In such cases, segmentation may still be achieved if at least one point on the decision boundary is either known *a priori* or determined from topological analysis. In this procedure, the scan is initiated at a point P inside the frame with coordinates (L, K), as shown in Figure 3, and proceeds up by a path that is midway between the left and right numerals to the top of the frame, and down by following the border of the left numeral to the end of the frame. In the case of touching borders (as in the left-hand portion of Figure 3), the path follows the touching border.

The success of this scheme is critically dependent on

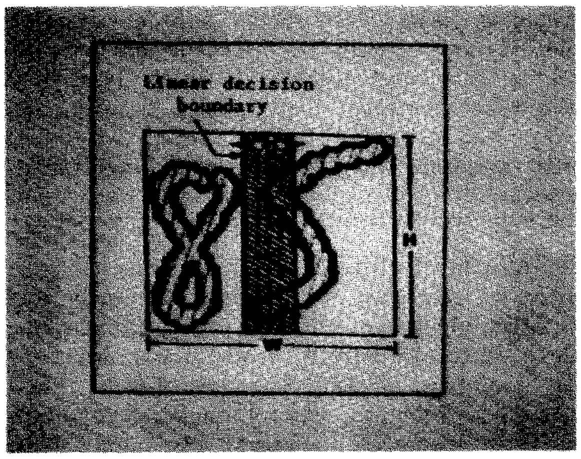


Figure 2. Linear decision boundary

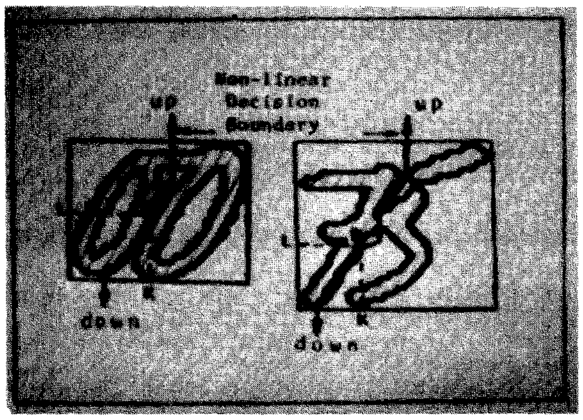


Figure 3. Nonlinear decision boundary

the knowledge or the evaluation of the initial point P on the decision boundary. The choice of this initial point is discussed next.

Determination of the initial scanning point

The determination of the initial scanning point, a key aspect of the segmentation scheme, is in general quite complicated. This difficulty is caused by the diversity and the manner in which nonisolated numerals occur in the real world. The procedure for choosing the initial scanning point utilizes the known topological features of the individual numerals. The topological features used in this procedure are:

- The number of horizontal border-to-background transitions (HT) at any location in the frame. This number is computed over the connected numerals in the string. The above features provide clues about the numerals in a string. Thus, if the number of transitions is only two, then the two numerals in a string are any two of 1, 2, 3, 5 and 7 (e.g. the strings 37, 21, 52, etc.). If the number of transitions at a location is four, then the numerals are 0, 4, 6, 8 or 9 (e.g. the strings 00, 89, 60, 88, etc.).
- The smoothness of the left and right profiles of the string. The smoothness measure gives important information about the type of numeral in the left- or right-most positions. Thus numerals such as 0, 1, 6 and 8 have relatively smooth left profiles, while numerals such as 0, 1, 3, 7, 8 and 9 have relatively smooth right profiles.

The following steps utilize the above features to determine the initial scanning point.

Step 1 (Number of horizontal transitions HT = 2 or 4)

Starting at the middle of the frame, evaluate the number of horizontal transitions HT. This number is the total number evaluated over the whole string. If HT is not equal to two or four, then initiate a horizontal scan at a point one pixel up from the middle of the frame and count the number of transitions HT. If HT is still not equal to two or four, then start the horizontal scan one pixel down from the middle of the frame. Repeat this procedure by moving further up or down until HT is equal two or four.

If HT = 2, then L corresponds to the row where the number of horizontal transitions is equal to two, and K is chosen after the first transition as shown in Figures 4a and 4b.

If HT = 4, then L corresponds to the row where the number of horizontal transitions is equal to four, and K is chosen after the second transition as shown in Figure 4c.

Once the initial point is determined, then the segmentation procedure described earlier is applied to determine the decision boundary. If the procedure fails after a prescribed number of horizontal scans ε_H (e.g. ε_H = 3 pixels) as shown in Figure 4d, then step 2 of the algorithm is implemented.

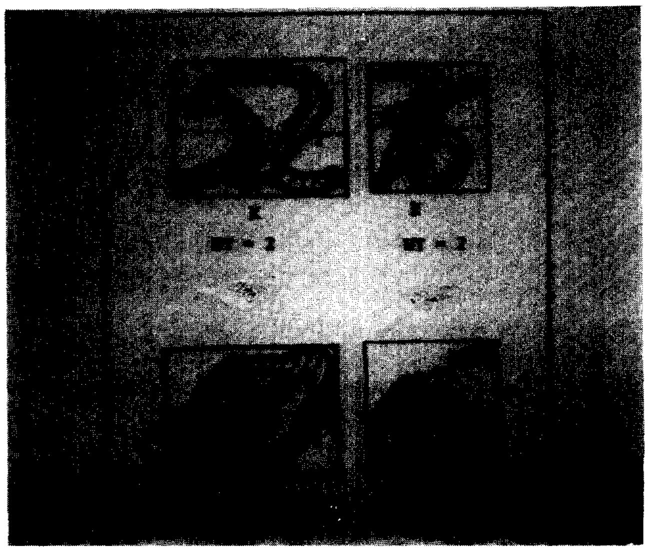


Figure 4. Horizontal transitions

Step 2 (Number of horizontal transitions HT = 3)

Search for three horizontal transitions as described in step 1. If there are not three transitions then the string is rejected. When the number of horizontal transitions equals three, the following possibilities are implied: one of the two numerals is a 0; the two numerals are not connected but occur in overlapping fields; or the two numerals are connected, touching and/or occur in overlapping fields.

In all cases, L corresponds to the horizontal line where HT = 3 as shown in Figure 4d, while K is determined as follows.

Scan horizontally at L from left to right to generate the string $S_j^k(i)$, $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, M$; where N is the total number of symbols in the string, and M is the total number of states, and k represents the string under consideration. The string $S_j^k(i)$ is derived over the alphabet

$$Z = \{0, 1, 4, 6\}$$

where 0 denotes the label for the character, 1 denotes the label for the background, 4 denotes the label of the first detected border, and 6 denotes the label of other borders. (Note that the above labels 0, 1, 4 and 6 are just notation; variables such as a, b, c and d can be used instead.)

As an illustration, consider the two numeral strings shown in Figures 5a and 5b. The string $S_j^1(i)$ derived at L for the first numeral string is

$$S_j^1(i) = \underline{1\,4\,0\,6\,1\,6\,0\,4\,1\,6\,0\,6\,1}$$

while $S_j^2(i)$ derived at L for the second numeral string is

$$S_j^2(i) = \underline{1\,4\,0\,6\,1\,6\,0\,4\,1\,4\,0\,4\,1}$$

where $i = 1, 2, \ldots, N$, and N is the total number of pixels in the string $S_j^k(i)$; $j = 1, 2, \ldots, M$, where M

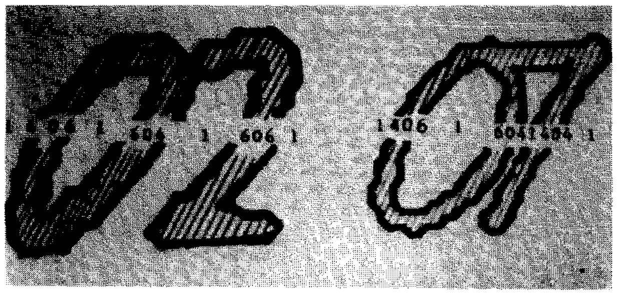


Figure 5. Numeral string with three horizontal transitions

is the number of states and is equal to 13 in both strings $S_j^1(i)$ and $S_j^2(i)$; and a state is defined as $\underline{A}$ where

$$\underline{A} = \underline{0} \text{ or } \underline{1} \text{ or } \underline{4} \text{ or } \underline{6}$$

and $\underline{A}$ is a string of pixels having the label A, where A can take the value of 0, 1, 4 or 6.

It should be mentioned that $S_1^k(i)$ and $S_M^k(i)$ will always be equal to 1 since the scan is started from the background and the last scan will also be a background.

The string $S_j^k(i)$ is then examined and two different situations may arise.

Case 1

If

$$[(S_2^k(i) = 4) \wedge (S_{M-1}^k(i) = 6)] \vee [(S_2^k(i) = 6) \wedge (S_{M-1}^k(i) = 4)]$$

then numerals are overlapping and not connected, i.e. if the prefix is $\underline{14}$ and the suffix is $\underline{61}$ or if the prefix is $\underline{16}$ and the suffix is $\underline{41}$, then the two numerals are overlapping and not connected, as shown in Figure 5a.

Case 2

If

$$[(S_2^k(i) = 4) \wedge (S_{M-1}^k(i) = 4)] \vee [(S_2^k(i) = 6) \wedge (S_{M-1}^k(i) = 6)]$$

then numerals are connected, i.e. if the prefix is $\underline{14}$ (or $\underline{16}$) and the suffix is $\underline{41}$ (or $\underline{61}$), then the two numerals are connected as shown in Figure 5b.

In cases 1 and 2 we count the number of transitions, i.e. the number of 4 to 1 transitions, $(41)_p$, the number of 6 to 1 transitions, $(61)_p$, the number of 4 to 6 transitions, $(46)_p$, and the number of 6 to 4 transitions, $(64)_p$, where p represents the number of transitions encountered. These two cases are discussed in detail below.

Case 1: Overlapping but not connected numerals

In this case, $S_2^k(i) \neq S_{M-1}^k(i)$. There are then two cases to be considered, as follows.

In the first, the suffix is $\underline{14}$ and the prefix is $\underline{61}$, i.e.

$$(S_2^k(i) = 4) \wedge (S_{M-1}^k(i) = 6)$$

In this case, the two numerals are not connected and the border of the left numeral is detected first.

In the second case, the prefix is $\underline{16}$ and the suffix is $\underline{41}$, i.e.

$$(S_2^k(i) = 6) \wedge (S_{M-1}^k(i) = 4)$$

The two numerals are also not connected but the border of the right numeral is detected first.

Figure 5a illustrates an example of the first case, where the left numeral is a 0; the string $S_j^k(j)$ generated at L is

$$S_j^k(i) = \underline{1}\,\underline{4}\,\underline{0}\,\underline{6}\,\underline{1}\,\underline{6}\,\underline{0}\,\underline{4}\,\underline{1}\,\underline{6}\,\underline{0}\,\underline{6}\,\underline{1}$$

In this case, there is one 4 to 1 transition and two 6 to 1 transitions, i.e.

$$(41)_1 \wedge (61)_2$$

and K is the coordinate of the pixel corresponding to 4 of the first 4 to 1 transition, i.e.

$$K = \{i | S_j^k(i) = 4 \,\varepsilon\, (41)_1\}$$

Case 2: Connected numerals

In this case, $S_2^k(i) = S_{M-1}^k(i)$, i.e. the prefix is $\underline{14}$ and the suffix is $\underline{41}$ as in Figure 6a, or the prefix is $\underline{16}$ and the suffix is $\underline{61}$ as in Figure 6b.

When the two numerals are connected, two different numeral strings can generate the same string $S_j^k(i)$. As an example, the two strings shown in Figures 7a and 7b generate the same string $S_j^k(i)$, which is

$$S_j^k(i) = \underline{1}\,\underline{4}\,\underline{0}\,\underline{6}\,\underline{1}\,\underline{6}\,\underline{0}\,\underline{6}\,\underline{1}\,\underline{6}\,\underline{0}\,\underline{4}\,\underline{1}$$

This creates ambiguities since for the first string

$$K = \{i | S_j^1(i) = 6 \,\varepsilon\, (61)_2\}$$

while for the second string

$$K = \{i | S_j^2(i) = 6 \,\varepsilon\, (61)_1\}$$

In order to resolve this ambiguity, the decision will be based on global contextual information. The profile of the two connected numerals is computed over the whole frame (for more details about profile evaluation see Reference 3). The smoothness of the left and right profiles is evaluated as the measure of discontinuity of that profile, and is given by LPEAK and RPEAK, where LPEAK and RPEAK are the peak-to-peak value of the first difference of the left and right profiles respectively. The left

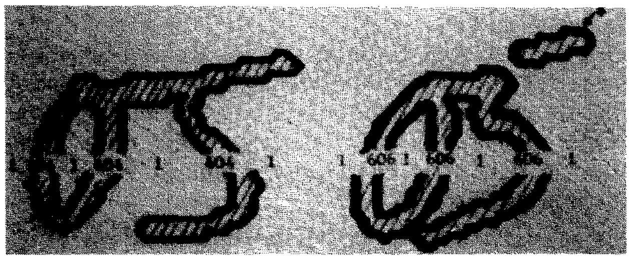


Figure 6. Connected numerals

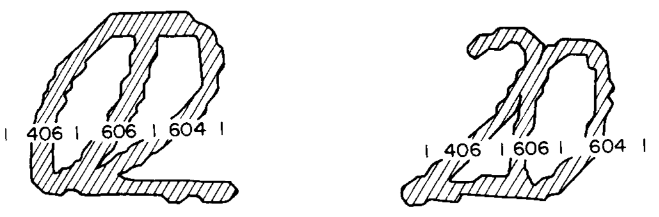


Figure 7. Two numeral strings generating the same string $S_j^k(i)$

(or right) profile is considered to be smooth if LPEAK (or RPEAK) is less than $W/5$, where W is the width of the frame. If both profiles are not smooth (i.e. (LPEAK and RPEAK) $> W/5$), then divide the frame vertically into two equal regions and evaluate the left profile from the left side of the frame to its middle, and the right profile from the right side of the frame to its middle. This is illustrated in Figures 8 and 9 respectively.

Consider as an example the numeral string of Figure 7a. The string $S_j^k(i)$ evaluated at L is

$$S_j^k(i) = \underline{1\,4\,0\,6\,1\,6\,0\,6\,1\,6\,0\,4\,1}$$

In this case, the left profile is smooth while the right profile is not, and there are one 4 to 1 transition and two 6 to 1 transitions, i.e.

$$(\text{LPEAK} < W/5) \wedge (\text{RPEAK} > W/5) \wedge (41)_1 \wedge (61)_2$$

K is the coordinate of the pixel having the value 6 of the second 6 to 1 transition, i.e.

$$K = \{i | S_j^k(i) = 6 \in (61)_2\}$$

Once the initial scanning point has been obtained, then the boundary separating the two numerals can be determined by the tests described at the beginning of this section.

TEST RESULTS

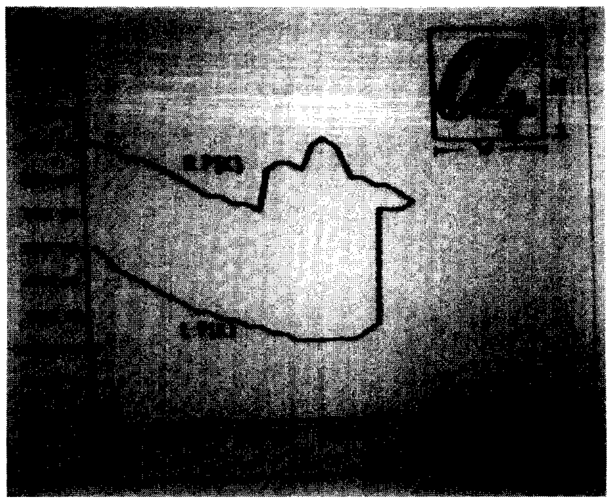
Two hundred numeral strings, each consisting of two connected numerals, were used to evaluate the validity of the algorithm. Figures 10a–10c illustrate some examples of segmented numeral strings.

The segmented numerals (400 of them) were then used as test data in a syntactic numeral recognition algorithm[4]. An accuracy of 92% was obtained. The recognition errors were mainly due to the pseudofeatures generated by the connecting tail that was still present after segmentation. It is felt that these errors can be reduced by modifying the recognition algorithm to account for the pseudofeatures or to give the recognition stage information from the segmenter on where the numerals were disconnected, thus allowing it to predict where the features of the numeral might be computed.

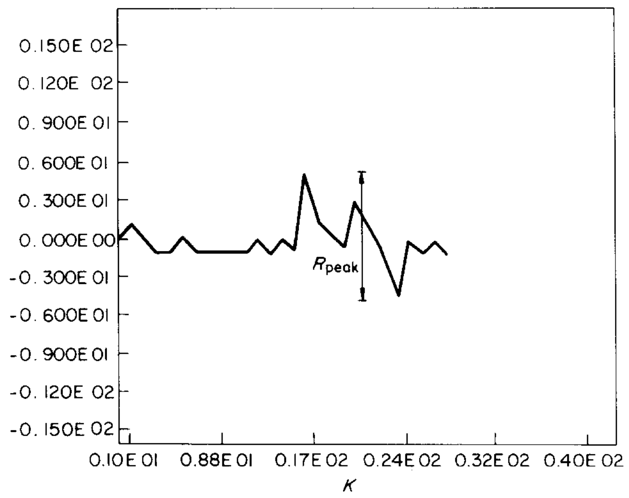
DISCUSSION AND CONCLUSIONS

A new algorithm for segmenting connected numeral strings has been proposed. The algorithm is hierarchical in the sense that it tests various hypotheses about the nature of the numeral strings. The algorithm yields satisfactory segmentation in most of the cases and in a form suitable for subsequent recognition. However, the algorithm assumes that the numeral strings are written in a 'normal way' with each numeral in the string having roughly the same height. It is also assumed that

a



b



c

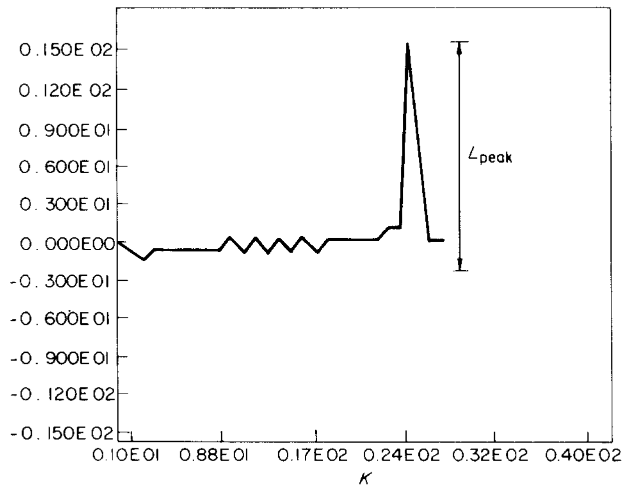
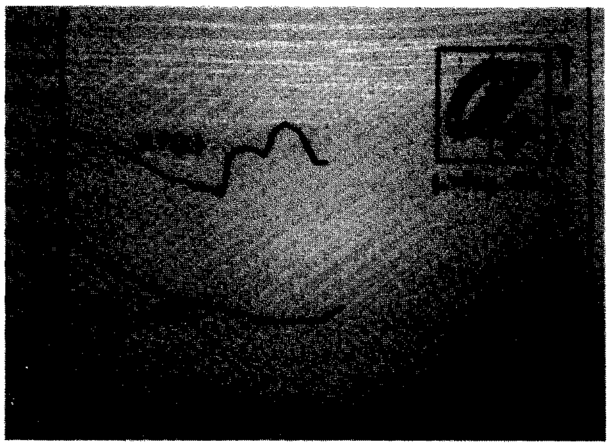
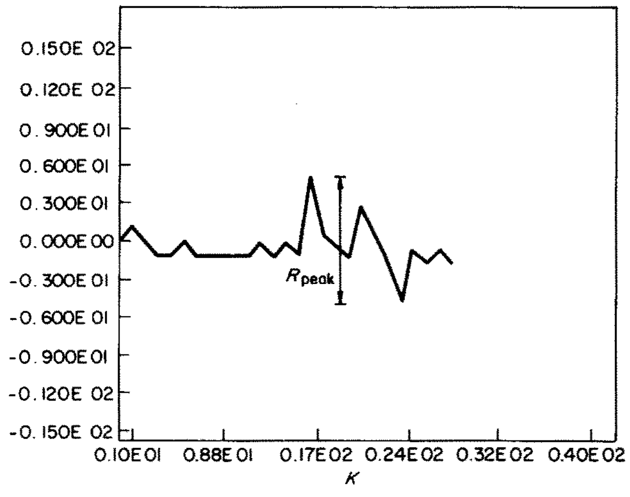
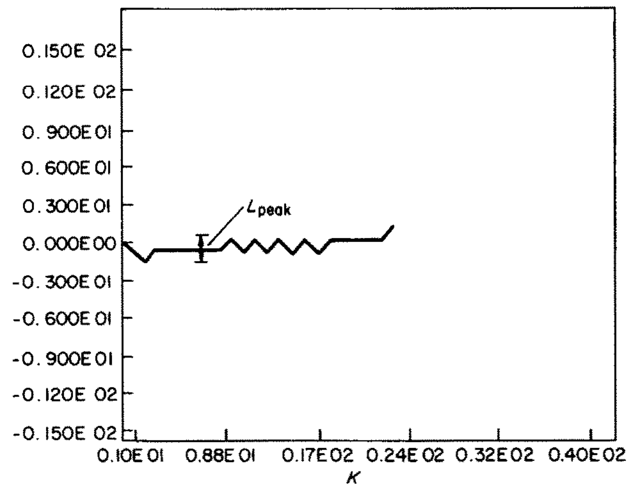


Figure 8. a, Global profiles evaluated over the whole frame; b, first difference of the right profile; c, first difference of the left profile

a




b

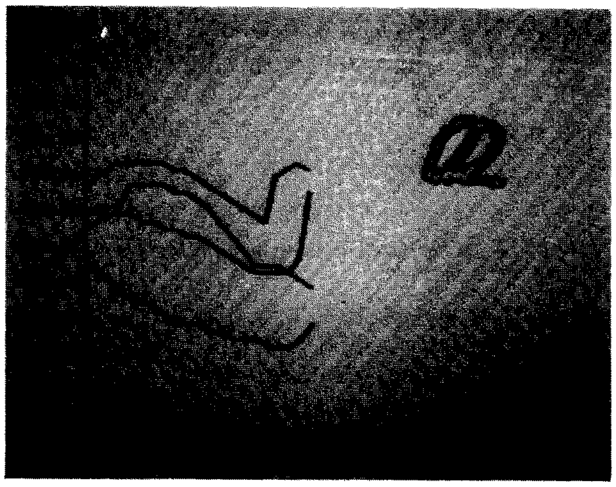


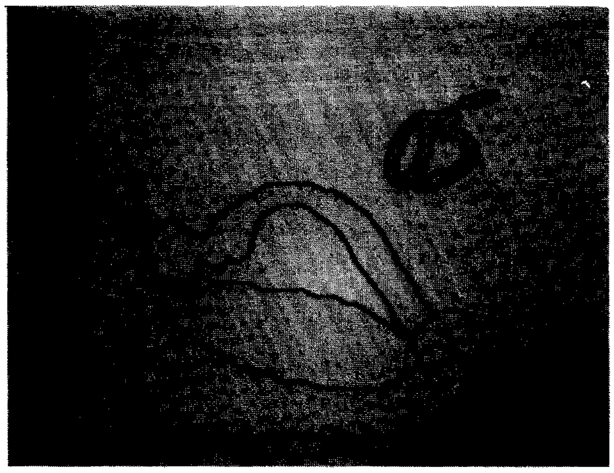
c

Figure 9. a, Global profiles evaluated up to the middle of the frame; b, first difference of the right profile; c, first difference of the left profile

the numerals are written on a specified line with orientations limited to 20° from the vertical. A further restriction requires the number of numerals in the string to be specified *a priori.*

Although the algorithm appears to be elaborate, the execution times are relatively small. This is made possible because of the tree structure and the small number of arithmetic operations required to implement the segmentation. Also in the study, it has been possible to combine the segmentation and feature extraction (for recognition purposes) into one step. Thus, the overall efficiency has been high. In practical studies, the segmentation and recognition of a two-numeral string was achieved in a fraction of a second (on a Nova 840 Data General minicomputer).


a


b

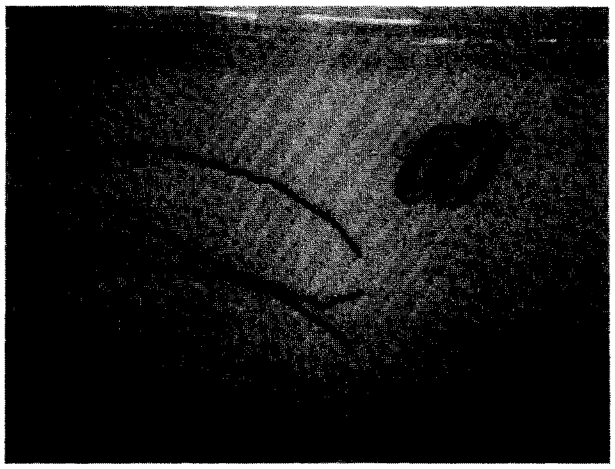

c

Figure 10. Segmented numerals

ACKNOWLEDGEMENT

The authors acknowledge the support extended by the Natural Sciences and Engineering Research Council of Canada for this research.

REFERENCES

1 **Suen, C Y, Berthod, M and Mori, S** 'Automatic recognition of handprinted characters — the state-of-the-art' *Proc. IEEE* Vol 68 No 4 (April 1980)

2 **Chottera, A and Shridhar, M** 'Feature extraction of manufactured parts in the presence of spurious reflections' *Can. Elect. Eng. J.* Vol 17 No 4 (1982) pp 29–34

3 **Badreldin, A** 'Structural recognition of handwritten numeral strings' *PhD dissertation* University of Windsor, Windsor, Canada (1985)

4 **Shridhar, M and Badreldin, A** 'A high accuracy syntactic recognition algorithm for handwritten numerals' *IEEE Trans. Syst., Man., Cybern.* No 1 (1985) pp 152–158