ELSEVIER

Pattern Recognition Letters 17 (1996) 523-527

Pattern Recognition Letters

Character image enhancement by selective region-growing

Zhixin Shi, Venu Govindaraju *

Center of Excellence for Document Analysis and Recognition, State University of New York at Buffalo, Buffalo, NY 14260, USA

Received 8 June 1995

Abstract

Preprocessing of character images is an important step in recognition. We describe a method of enhancing binary character images to assist the subsequent recognition process in both handwritten and machine-printed documents. The method performs selective and adaptive stroke "filling" with a neighborhood operator which emphasizes stroke connectivity. An improvement of 7% points was realized in recognition of address fields.

1. Introduction

A character image is usually a bilevel image produced by an imperfect binarization process which results in fragmentation. It is non-trivial to choose a threshold (see Pal and Rosenfeld, 1988) so that the connectivity of character strokes is preserved. Traditional ways of image enhancement include filtering methods such as *low-pass filters* and *high-pass filters* using Fourier transforms. They are suitable for general enhancement such as smoothing and edge detection. However for the purpose of character image enhancement and subsequent recognition these methods are not adequate.

Neighborhood operators have been widely used for image enhancement of general scenes as well. Although their implementation on general purpose computers is slow (Van Vliet and Verwer, 1988), their simplicity renders them suitable for customized hardware. While they have been found to be suitable for enhancing character images, typically they have failed to reconstruct broken strokes.

* Corresponding auhtor. Email: govind@cedar.buffalo.edu

In this paper, we present a new approach to character image enhancement using a neighborhood operator. The method applies to both handwritten and machine-printed binary images efficiently. It emphasizes stroke connectivity while at the same time it conservatively checks aggressive "over-filling". It is implemented using a binary tree structure that ensures an efficient single-pass algorithm.

2. Algorithm

Reconstruction of character strokes requires "filling" in gaps in the broken strokes caused by imperfect binarization. It is only local connectivity information that is available for the reconstructing procedure. The task can be explained in terms of the task of making a map of a forest based on a series of local views. Each local view is centered upon a tree and few of its immediate neighbors. One can go around a tree and mark the neighboring trees. The procedure is repeated at the boundary of the marked neighborhood to reach unmarked trees until all trees in the forest are marked. While preparing the map, small gaps between trees are also marked.

SSDI 0167-8655(95)00126-3

2.1. Selective region-growing

Following is the algorithm for the region growing procedure.

Region_Growing()

1. *In_img* ← input image.
2. *Out_img* ← initialized to blank image for holding the enhanced image.
3. Find the left most black pixel *P_init* (with pixel value 0). If none found goto Step 6.
4. Call **Block_Connect**(*P_init*).
5. Go to Step 3.
6. Output *Out_img*.
7. Return.

Locate N_{bd} operator

The algorithm is developed based on the concept of preparing a map of a character. We start with a black pixel (P_{init}) and a small neighborhood (N_{bd}: 5×5) which contains the black pixel. The center of the neighborhood of interest is found by drifting to the border of the black pixel region. A neighborhood starting with P_{init} is found as follows.

1. Go to the left of P_{init} (at most four pixels away) to find a run (lc) of black pixels.
2. If $lc \leqslant 5$ (including P_{init}) go to the right of P_{init} for at most $5 - lc$ pixels to find a run of $5 - lc$ black pixels.
3. If the sum total of the black pixel run found is cn ($cn \leqslant 5$), then a neighborhood is marked centered around the pixel in the middle of the run ($\frac{1}{2}cn$th from the leftmost black pixel).

Block_Connect()

Subsequent to choosing a neighborhood(N_{bd}), filling in the gaps between pixels is done as follows.

1. Store the input image in a buffer, *In_img* and the output image (initially empty) in *Out_img*.
2. Rows in N_{bd} are numbered $1, \ldots, 5$. (Rows with no black pixels are "trivial".)
3. Identify the "uppermost" and "lowermost" non-trivial rows.
4. Identify the "leftmost" and "rightmost" black pixels in N_{bd} for each row between the uppermost and lowermost rows including themselves. If any of these rows with row number *row* is "trivial" (blank in N_{bd}), then let leftmost(*row*) = rightmost(*row*) = center of the row.
5. Fill the uppermost and lowermost rows.
   - If the row has row number 3, change the run of black pixels (found before) to 1 in *In_img* and set the corresponding pixels to be black in *Out_img*.
   - Else, fill the row by changing the white pixels between black pixels to be black pixels with values 1 in *In_img* and then set the pixels in the corresponding positions of the same row in *Out_img* to black.
6. Fill the rows in between the uppermost and lowermost rows of N_{bd}.
   - Renew the leftmost and rightmost positions for all rows in between uppermost and lowermost rows as follows. If *row* is such a row, then let leftmost(row) = $\min\{$leftmost(row), (leftmost($row - 1$) + leftmost($row + 1$))/2$\}$ and rightmost(row) = $\max\{$rightmost(row), (rightmost($row - 1$) + rightmost($row + 1$) + 1)/2$\}$.
   - Change the corresponding pixels from leftmost to rightmost in each row to 1 in *In_img* and to 0 in *Out_img*.

Subsequent to filling gaps in one neighborhood, new candidate neighborhoods are located for further filling. Each neighborhood yields two new neighborhoods. While filling gaps as described above, the upper and lower row black pixels (0) are unchanged in *In_img* as long as they are not the central row of the N_{bd}. For the new neighborhoods, take the leftmost black pixel with value 0 in the upper and lower rows as the new starting point P_{init}.

Fig. 1 shows the region growing procedure on a character stroke with a randomly chosen starting pixel.

3. Experimental observations

Fig. 2 illustrates the processing on a character image extracted from an address block. The strokes from adjacent characters are in close proximity which makes it difficult to reconstruct broken strokes using traditional methods (such as Fourier based methods). The method described in this paper is selective in its choice of neighborhoods that undergo reconstruction. This is

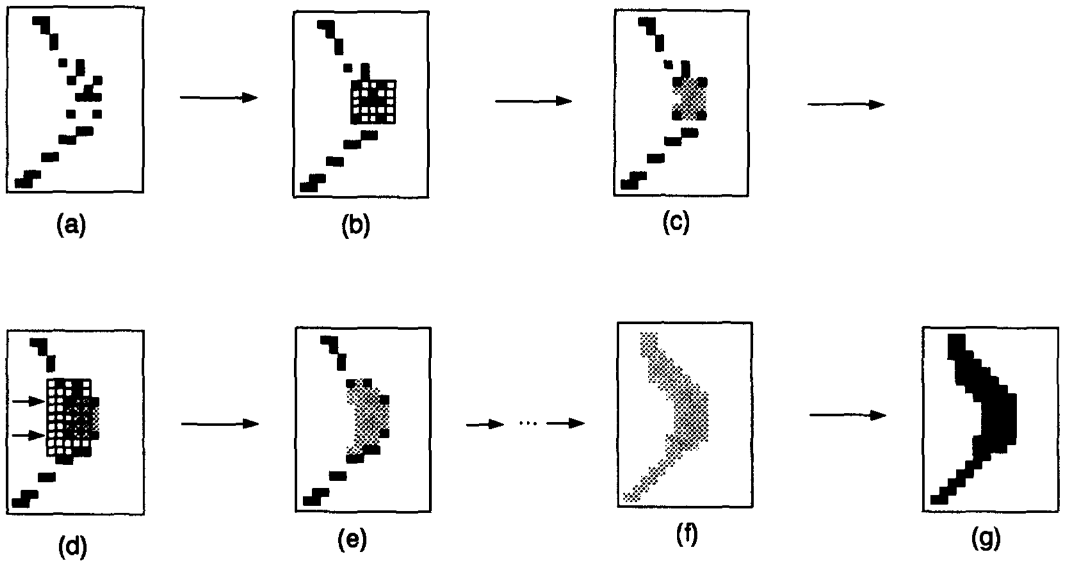

Fig. 1. Enhancement procedure.

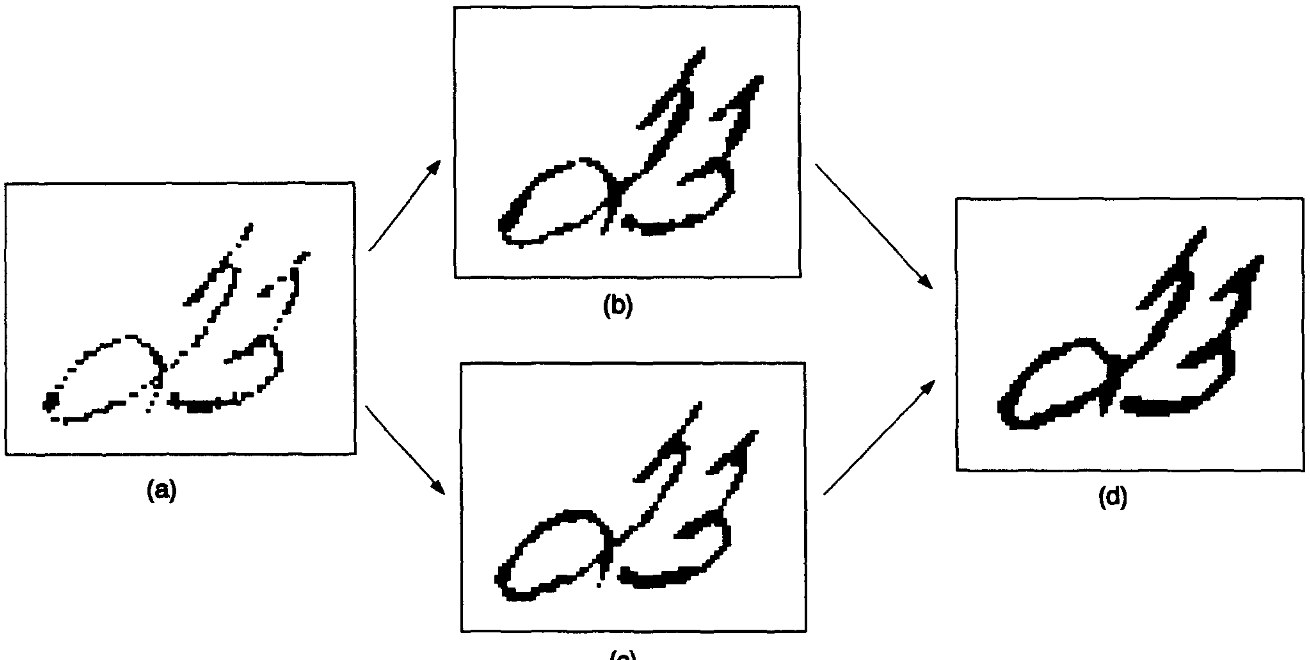

Fig. 2. (a) Original image. (b) Row major processing with single run. (c) Column major processing with single run. (d) Combining row major and column major processing.

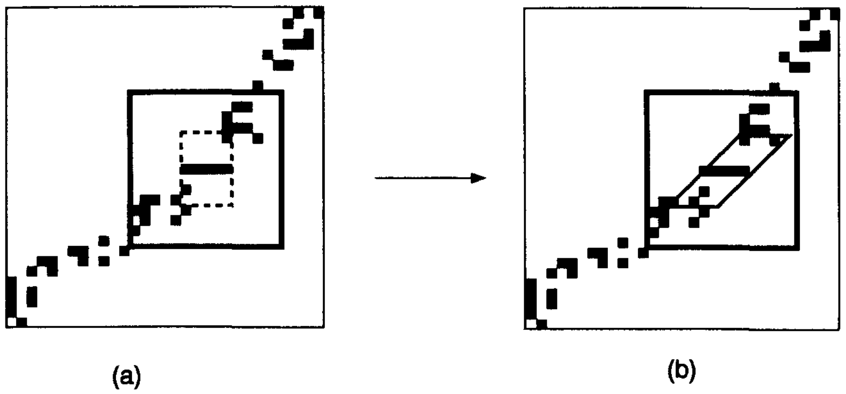

Fig. 3. Neighborhood operator with dynamic slant correction. (a) The double box neighborhood operator is located at the center of a run of black pixels. (b) The inner box is dynamically selected based on the slant of the stroke.

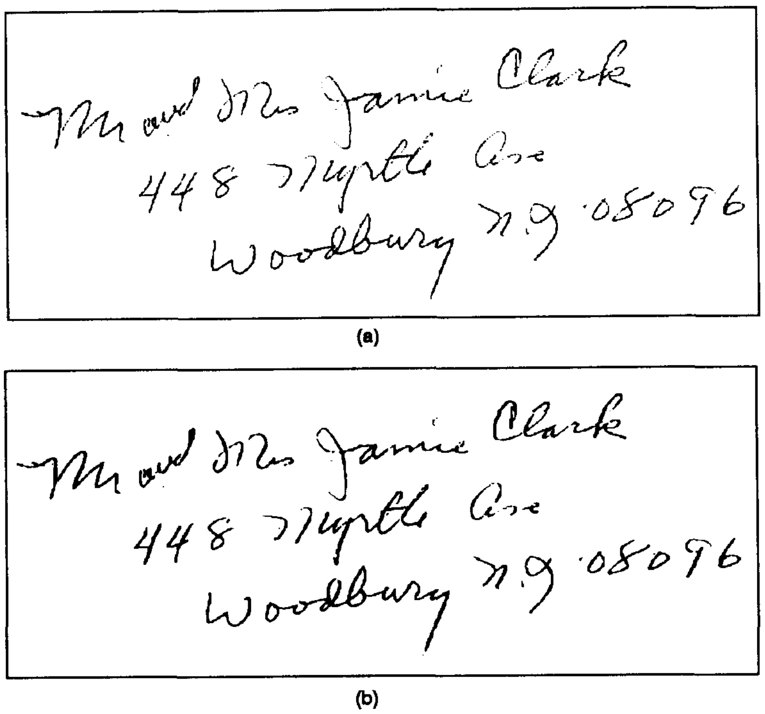

Fig. 4. Enhancement result on an entire address block image. (a) Original image. (b) Enhanced image using the row major processing.

achieved by centering the neighborhood on black pixel areas. As is indicated in Fig. 2 the width of character strokes is unchanged in the enhanced image. The selective mechanism has the added advantage of being fast as enhancement is invoked only in selected regions of the image.

3.1. Row major and column major processing

As can be noted from the description of the algorithm, it is essentially a row major algorithm. A side effect of this is that vertical strokes are reconstructed better than horizontal strokes. The problem can be circumvented by taking a column major approach (Fig. 2(c)). The two approaches can be combined to obtain the best results (Fig. 2(d)).

3.2. Dynamic selection of neighborhood size

Given that typically the gaps in character strokes are only a few pixels, the selective region growing algorithm described with a neighborhood window size of 5 × 5 effectively reconstructs character strokes. However, if the gaps are large a larger neighborhood window is warranted. Fig. 2(b) illustrates that the top of the loop in numeral "2" remains unfilled in spite of the enhancement procedure. The gap can be readily filled by increasing the size of the neighborhood window at the risk of merging adjacent character strokes.

We propose to modify the algorithm described so that the size and shape of the neighborhood operator can be dynamically adjusted based on the properties of the neighborhood. Specifically, we use a double neighborhood with a 5 × 7 box B in a bigger 15 × 15 box A. The center of the boxes are the same. First, the neighborhood is located as before. The predominant slant direction of the stroke under examination in box A is determined as follows.

Slant detection

For each 16 × 16 section of the input binary bitmap do the following:

1. Shift the 16 × 16 section to the right by one pixel and perform a logical NAND between the original image and the shifted copy. This erases all pixels where the two images overlap. Count of the remaining pixels is stored.
2. Repeat Step 1, this time shifting the 16 × 16 section to the right by two pixels.
3. Steps 1 and 2 are repeated for angles 45°, 90°, and 135°, starting each time with the original image.
4. The minimum of the pixel counts stored in Step 1 corresponds to the slant angle of Step 3.

Table 1

	Original images	Enhanced images
ZIP code recognition	56.9%	66.4%
Encode rate	23.2%	30.6%

The slant direction is used to orient the inner box B. Typically, this renders box B into a diamond shaped box. Now the inner box B contains more pixels in the direction of the stroke than in any other direction (Fig. 3). Filling proceeds with the inner box as before.

The sensitivity of the method to row/column major processing can be addressed in this scheme as well. When the slant is determined to be greater than 45° then proceed with row major processing.

Since the inner box is elongated in the direction of the slant, it allows filling of larger gaps along the direction of the stroke. This turns out to be a beneficial feature as gaps along the direction of the stroke are very likely to be parts of the stroke and need to be filled in. Conversely, the filling is conservative in direction orthogonal to the stroke slant.

3.3. Results

Table 1 illustrates the impact of the enhancement procedure described on reading of address block images. On a set of 1500 address block images, the performance on recognition of ZIP codes and encodes (recognition of ZIP code and the delivery line) went up by about 7 percentage points. Fig. 4 illustrates enhancement results on entire address blocks.

References

Pal, S.K. and A. Rosenfeld (1988). Image enhancement and thresholding by optimization of fuzzy compactness. *Pattern Recognition Lett.* 7, 77-86.

Van Vliet, L.J. and B.J.H. Verwer (1988). A contour processing method for fast binary neighborhood operations. *Pattern Recognition Lett.* 7, 27-36.

O'Gorman, L. (unpublished). Binarization and multi-thresholding of document images using Connectivity. Personal reference.