Image Processing and Computer Vision

Robert M. Haralick Editor

A Fast Parallel Algorithm for Thinning Digital Patterns

T. Y. ZHANG and C. Y. SUEN

ABSTRACT: A fast parallel thinning algorithm is proposed in this paper. It consists of two subiterations: one aimed at deleting the south-east boundary points and the north-west corner points while the other one is aimed at deleting the north-west boundary points and the south-east corner points. End points and pixel connectivity are preserved. Each pattern is thinned down to a "skeleton" of unitary thickness. Experimental results show that this method is very effective.

1. INTRODUCTION

It is well known that the general problem of pattern recognition lies in the effectiveness and efficiency of extracting the distinctive features from the patterns. The stroke analysis method is a powerful approach to recognizing certain types of digital patterns such as alphanumeric characters and ideographs. It should be noted that the strokes thinned by hardware or software are accompanied by different kinds of distortion. Different thinning algorithms produce different degrees of distortion [1–5, 7–12].

There is no general agreement in the literature on an exact definition of thinness. Pavlidis [6] describes a thinning algorithm that determines skeletal pixels by local operations. At the same time, the pixels are labeled so that the original image can be reconstructed from its skeleton. The goal of this paper is to find a faster and more efficient parallel thinning algorithm. The distortion should be as little as possible. Experimental results indicate that this method can be used to thin a variety of digital patterns.

2. PARALLEL PICTURE PROCESSING

A binary digitized picture is defined by a matrix IT where each pixel $IT(i, j)$ is either 1 or 0. The pattern consists of those pixels that have value 1. Each stroke in the pattern is more than one element thick. Iterative transformations are applied to matrix IT point by point according to the values of a small set of neighboring points. It is assumed that the neighbors of the point (i, j) are $(i-1, j)$, $(i-1, j+1)$, $(i, j+1)$, $(i+1, j+1)$, $(i+1, j)$, $(i+1, j-1)$, $(i, j-1)$, and $(i-1, j-1)$, as is shown in Figure 1. In parallel picture processing, the new value given to a point at the nth iteration depends on its own value as well as those of its eight neighbors at the $(n-1)$th iteration, so that all picture points can

P_9 $(i-1, j-1)$	P_2 $(i-1, j)$	P_3 $(i-1, j+1)$
P_8 $(i, j-1)$	P_1 (i, j)	P_4 $(i, j+1)$
P_7 $(i+1, j-1)$	P_6 $(i+1, j)$	P_5 $(i+1, j+1)$

FIGURE 1. Designations of the nine pixels in a 3 × 3 window.

be processed simultaneously. It is assumed that a 3 × 3 window is used, and that each element is connected with its eight neighboring elements. The algorithm presented in this paper requires only simple computations.

3. THINNING ALGORITHM

Our method for extracting the skeleton of a picture consists of removing all the contour points of the picture except those points that belong to the skeleton. In order to preserve the connectivity of the skeleton, we divide each iteration into two subiterations.

In the first subiteration, the contour point P_1 is deleted from the digital pattern if it satisfies the following conditions:

(a) $2 \leq B(P_1) \leq 6$

(b) $A(P_1) = 1$

(c) $P_2 * P_4 * P_6 = 0$

(d) $P_4 * P_6 * P_8 = 0$

where $A(P_1)$ is the number of 01 patterns in the ordered set $P_2, P_3, P_4, \cdots P_8, P_9$ that are the eight neighbors of P_1 (Figure 1), and

$B(P_1)$ is the number of nonzero neighbors of P_1,

that is,

$$B(P_1) = P_2 + P_3 + P_4 + \cdots + P_8 + P_9.$$

If any condition is not satisfied, e.g., the values of $P_2, P_3, P_4, \cdots P_9$ as shown in Figure 2, then

$$A(P_1) = 2$$

Therefore, P_1 is not deleted from the picture.

In the second subiteration, only conditions (c) and (d) are changed (Figure 3) as follows:

(c′) $P_2 * P_4 * P_8 = 0$

(d′) $P_2 * P_6 * P_8 = 0$

and the rest remain the same.

By conditions (c) and (d) of the first subiteration, it will be shown that the first subiteration removes only the south-east boundary points and the north-west corner points which do not belong to an ideal skeleton.

The proof for the first subiteration is given, that is, the points to be deleted satisfy conditions:

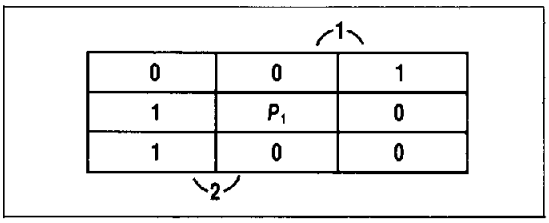

FIGURE 2. Counting the 01 patterns in the ordered set $P_2, P_3, P_4, \ldots, P_8, P_9$.

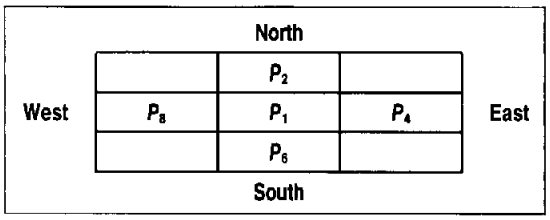

FIGURE 3. Points under consideration and their locations.

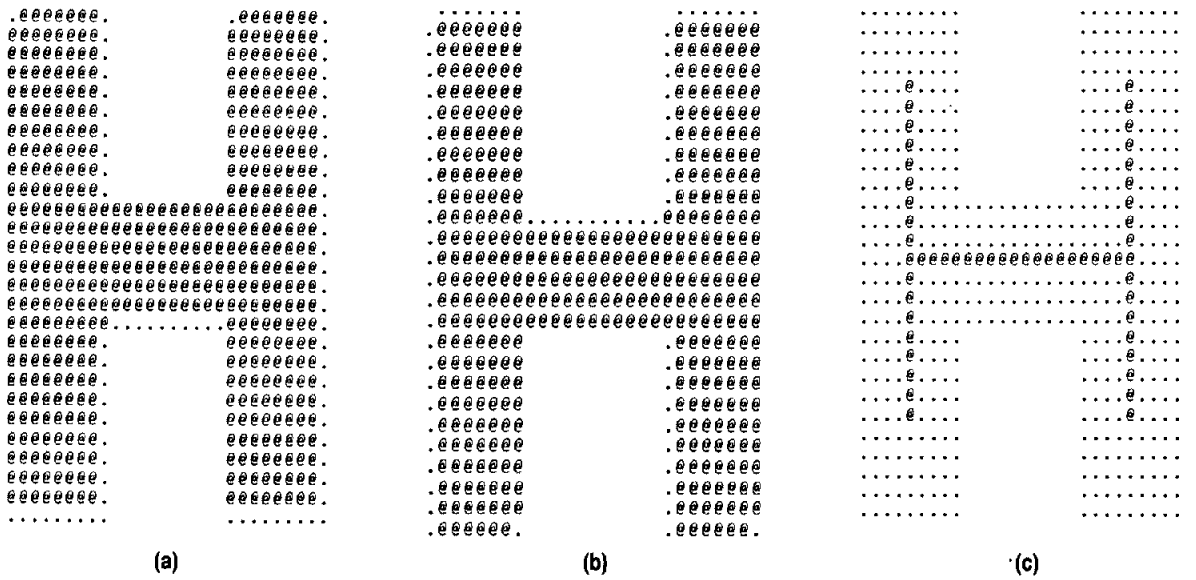

FIGURE 4. Results of thinning the character 'H' by the subiterations.

(c) $$P_2 * P_4 * P_6 = 0 \quad (1)$$

(d) $$P_4 * P_6 * P_8 = 0 \quad (2)$$

The solutions to the set of equations (1) and (2) are $P_4 = 0$ or $P_6 = 0$ or ($P_2 = 0$ and $P_8 = 0$). So the point P_1, which has been removed, might be an east or south boundary point or a north–west corner point. Similarly, it can be proved that the point P_1 deleted in the second subiteration might be a north–west boundary point or a south–east corner point. For example, the results of processing the character "H" by both subiterations are shown in Figure 4(a) and 4(b). The points marked by "·" have been removed. The final result is shown in Figure 4(c).

By condition (*a*), the endpoints of a skeleton line are preserved. Also, condition (*b*) prevents the deletion of those points that lie between the endpoints of a skeleton line, as shown in Figure 5. The iterations continue until no more points can be removed.

A flowchart of the proposed thinning algorithm is shown in Figure 6. Initially, the original picture is stored in matrix *IT* and a counter *C* is set to 0. The result of the processed picture is stored in matrix *IT*. To save memory space, only two matrices, *IT* and *M*, are used in our computation.

Figures 7(a)–7(c) show the results obtained by our algorithm for a Chinese character "體," a letter "B," and a digital "moving body," respectively. Skeleton points are marked by "*," or "@," and all those points that have been deleted in the thinning process are marked by "·."

The above algorithm yields very good results with respect to both connectivity and contour noise immunity. Furthermore, the conditions for searching those points that should be deleted from the pattern are very simple. In order to assess the performance of our algorithm, we have chosen one given by Stefanelli and Rosenfeld [11] for comparison. Both algorithms were written in FORTRAN, run on the same CDC Cyber 172 computer, and tested with the same digitized patterns. The result shows that the execution time of our algorithm is only 50 percent of the one given [11]. As can be predicted, the execution time depends on the complexity of the pattern and the thickness of the strokes: 0.505 CPU seconds for the Chinese character "體" 0.454 CPU seconds for the letter "B" and 1.163 seconds for the moving body.

TABLE I. Comparison of CPU Time (in seconds) Consumed by Different Parallel Thinning Algorithms

Pattern	Method		
	Four-Step	Two-Step	Our Algorithm
B	0.765	0.678	0.454
體	1.031	0.882	0.505
Moving body	2.713	2.221	1.163

FIGURE 5. Preventing the deletion of endpoints.

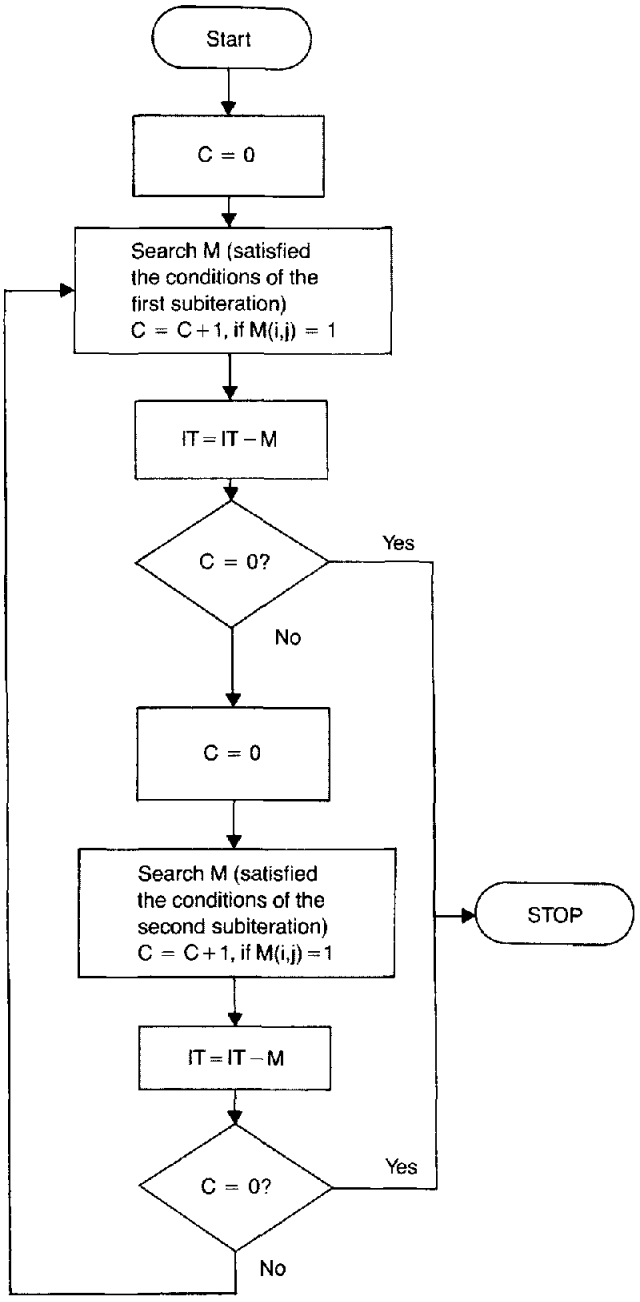

FIGURE 6. Flowchart of the thinning algorithm.

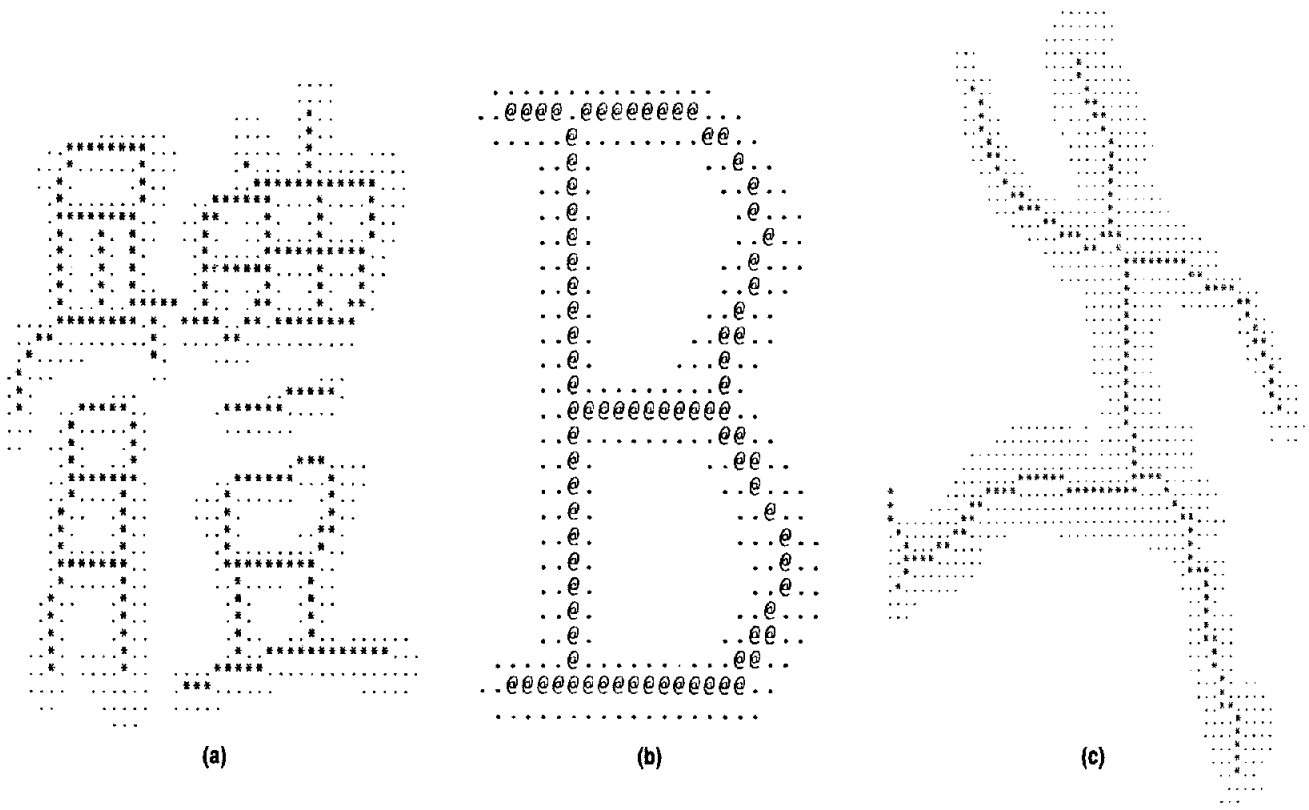

FIGURE 7. Thinning of different digital patterns.

4. SUMMARY

A parallel algorithm for thinning different types of digital patterns is presented in this paper. Each iteration is divided into two subiterations that remove the boundary and corner points of the digital patterns. After several iterations, only a skeleton of the pattern remains.

The proposed algorithm appears to be very efficient in the thinning of digital patterns and it compares favorably with those described in [11]. The results in Table I indicate that our method is 1.5 to 2.3 times faster than the four-step and two-step methods described in [11] while the resulting skeletons look very much the same.

REFERENCES

1. Arcelli, C., Cordella, L.P. and Levialdi, S. From local maxima to connected skeletons. *IEEE Trans. Pattern Analysis and Machine Intell. PAMI-3* (March 1981), 134–143.
2. Arcelli, C. A condition for digital points removal. *Signal Processing 1*, 4 (1979), 283–285.
3. Deutsch, E.S. Thinning algorithms on rectangular, hexagonal, and triangular arrays. *Commun. ACM 15*, 9 (Sept. 1972), 827–837.
4. Hilditch, C.J. Linear skeletons from square cupboards. In: *Machine Intelligence IV*, B. Mertzer and D. Michie, Eds., University Press, Edinburgh, 1969, pp. 403–420.
5. Ogawa, H. and Taniguchi, K. Thinning and stroke segmentation for handwritten Chinese character recognition. *Pattern Recognition 15*, 4 (1982), 299–308.
6. Pavlidis, T. A Flexible Parallel Thinning Algorithm. *Proc. IEEE Comput. Soc. Conf. on Pattern Recognition and Image Processing*. Aug. 1981, pp. 162–167.
7. Rosenfeld, A. Connectivity in digital picture. *J. ACM 17*, 1 (Jan. 1971), 146–160.
8. Rosenfeld, A. A characterization of parallel thinning algorithm. *Info. Control 29* (Nov. 1975), 286–291.
9. Rosenfeld, A. and Davis, L.S. A note on thinning. *IEEE Trans. Syst. Man Cybern. SMC-6*, 3 (March 1976), 226–228.
10. Rutovitz, D. Pattern Recognition. *Proc. Royal Statists. Soc. 129*, Series A (1966), 504–530.
11. Stefanelli, S. and Rosenfeld, A. Some parallel thinning algorithms for digital picture. *J. ACM 18* (April 1971), 255–264.
12. Tamura, H. A comparison of line thinning algorithms from digital geometry viewpoint. *Proc. 4th Int. Conf. Pattern Recognition*, 1978, pp. 715–719.

CR Categories and Subject Descriptors: I.5.2 [**Pattern Recognition**]: Design Methodology—*pattern analysis*; I.5.4 [**Pattern Recognition**]: Applications—*computer vision*

General Terms: Algorithms, Theory, Applications

Additional Key Words and Phrases: parallel algorithm, thinning of digital patterns, skeletonization

Received 3/83; accepted 9/83

Authors' Present Addresses: T. Y. Zhang, Department of Computer Science and Automatic Control, Chongqing University, Chongqing, China; C. Y. Suen, Department of Computer Science, Concordia University, 1455 de Maisonneuve West, Montreal, Quebec H3G 1M8, Canada