An Efficient Parallel Binary Image Thinning Algorithm

Moumita Sarkar
Lecturer, Dept. of Computer Science & Technology, Kingston Polytechnic College
Barasat, (West Bengal) India.

ABSTRACT
Thinning plays very vital role in image processing applications. Thinning basically reduce a thick digital object into a thin skeleton. In this paper we propose a new parallel image thinning algorithm for binary image thinning. The proposed algorithm is efficient enough to produce one pixel width skeleton for all kind of image structures with on loss of connectivity and no extra pixels.

***Keywords*:** Binary image, Thinning, Skeleton, Connectivity, Junction point, Medial line approximation

I INTRODUCTION

Thinning is the process of alteration of digital images into a simplified version of topologically equivalent image. It is very much useful in solving different image processing related problems such as matching, recognition in different types of image analysis. Thinning is shown in Fig 1.

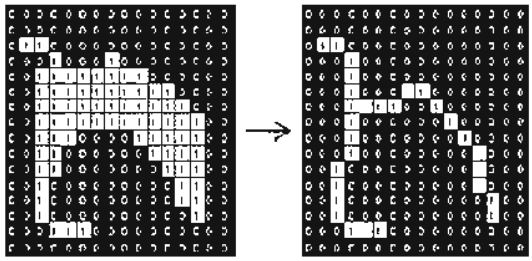

Fig 1: Image Thinning

Image thinning algorithms can be classified as shown in Fig 2.

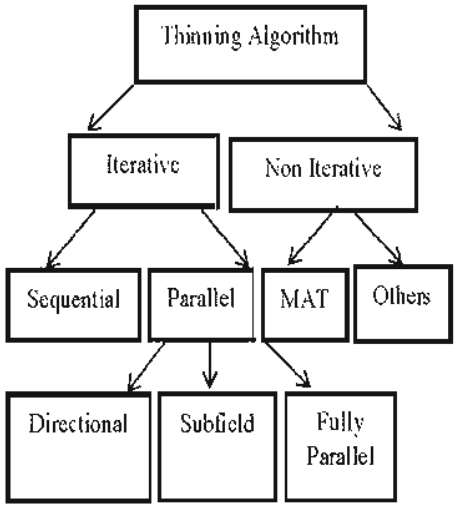

Fig 2: Classification of image thinning algorithms

Iterative thinning algorithms remove pixels iteratively with the analogous conditions for every iteration until pertinent skeleton is found.

Sequential algorithms examine the pixels in a fixed sequence in every iteration for deletion, and the deletion of a point in the $(n+1)^{th}$ iteration depends on all the operations that have been performed previously, i.e., on the outcome of the n^{th} iteration as well as on the pixels processed in the current iteration.

For parallel algorithm omission of pixels in the $(n+1)^{th}$ iteration would depend only on the result that leftovers after n^{th} iteration.

In directional approach iterations are divided into sub iterations which be governed by a single or combination of directions.

Subfield approach breaks down the picture into subfields based on some predefined conditions and pixels of same grounds are removed in parallel.

The same thinning operators (deletion criteria) are used for every iteration in the fully parallel approach.

Non iterative thinning algorithms do not examine discrete pixels one after another, but, they yield a median line or some centerline of the shape and then make a judgment whether to remove that exact boundary pixel or not. Skeleton is obtained in step by step computation.

The medial axis (MA) of a figure can be obtained as the locus of centers of all the maximal disks, inside the figure but not contained in any further disk.

A worthy thinning algorithm should own many properties. First, the algorithm should preserve the connectivity. Second, it should produce one pixel thinned skeleton. Third, it should preserve the topology of the object. Forth, the thinned should approximate the medial line. Fifth, excessive erosion must be disallowed and finally, the algorithm should be fast to produce the output image. These are the most basic requirements from any thinning algorithm.

II PRELIMINARY NOTATIONS

Let, I denotes a binary image which is represented by a matrix M, of size R×C, where R denotes the row and C denotes the column. M (i, j) represents the binary value of the pixel (i, j). The pixel at position (i, j) is black if M (i, j) = 1 and it is white if M(i,j) = 0. The white pixels form the background of I, and the black pixels form the foreground.

In a 3x3 mask, the pivot pixel and its eight neighbors with corresponding coordinate position are shown in Fig 3 where, the center pixel Pic_1 is the pixel of interest and also called Pivot pixel.

Pic_9 (i-1, j-1)	**Pic_2** (i-1, j)	**Pic_3** (i-1, j+1)
Pic_8 (i, j-1)	**Pic_1** (i, j)	**Pic_4** (i, j+1)
Pic_7 (i+1, j-1)	**Pic_6** (i+1, j)	**Pic_5** (i+1, j+1)

Fig 3: 3x3 mask

In general any foreground pixel can fall into any four of the following category.

(a) Junction pixel: a non-zero pixel for which the deletion would break the connectivity of the original pattern.
(b) Edge pixel: also called boundary, or border pixel which is a non-zero pixel that has at least one zero 4-neighbor pixel (top, bottom, left and right pixels of the pivot pixel).
(c) End pixel: this is a non-zero pixel that cannot have more than one non-zero neighbor pixel.
(d) Simple pixel: This is an edge pixel whose removal from the object does not change the topology. We say that an algorithm preserves the connectivity [8] of any binary image if and only if:
(e) No object in I can be completely deleted by the algorithm in any iteration.
(f) It does not connect any actually dis-joint objects in I, nor does it disconnect an existing hole or create any new hole in S.
(g) It does not disconnect any object in I. And it does not connect any hole of I to another distinct hole or the background of I where a hole may be defined as a background region surrounded by a connected border of foreground pixels.

III RELATED WORK ON NON TEMPLATE BASED PERALLEL THINNING ALGORITHM

In the past several decades several image thinning algorithm has been proposed. Some of them used some template for deletion. This section excludes those algorithms. Zhang and Suen in the year 1984 proposed an extremely popular thinning algorithm [1]. It works in two sub iterations. But, it is not able to yield correct thinned output for the two pixel width slant lines (in both left to right and right to left direction). And two pixel width squares are completely eliminated. Lu and Wang in the year 1985 projected another algorithm [2] which is the single point alteration of ZS algorithm. This algorithm keeps the two pixel width slant lines intact which was rub out by the ZS algorithm. Performance is same as ZS algorithm on other images. Holt, Stewart, Clint, and Perrott introduced a new thinning method [3] in 1987. There was only one iteration with no sub iteration. Their algorithm yields skeletons more or less same as that of ZS or LW algorithm but does not disturb connectivity. The negatives of this algorithm are diagonal lines are deleted and not able to create one pixel thin horizontal and vertical lines. Y.Y. Zhang and P.S.P. Whang offered a single pass thinning algorithm [4] in the year 1988 which remove completely the patterns which are removed by the ZS algorithm. Moreover two pixel width horizontal and vertical lines are also erased. Zichan Guo and Richard W.Hall in 1989 proposed a two sub-iteration parallel thinning algorithm [5]. This algorithm is able to thin the two pixel width inclined lines (both left-to-right and right-to-left) into one pixel width inclined line, which was a major shortcoming of the earlier algorithms. It yields medial axis points for horizontal, vertical and diagonal lines. It also yields the results in smaller number of iterations but does not preserve the junction points and generates some extra points. Hilditch's algorithm [6] performs multiple passes on the pattern. This algorithm has the identical problem as ZS algorithm. A. Jagna and V. Kamakshiprasad, describes a parallel thinning algorithm for binary images [7] in 2010. The problems of ZS algorithm are still there in this algorithm. Lynda Ben Boudaoud, Abderrahmane Sider and Abdelkamel Tari proposed a Hybrid Thinning algorithm [8], a combination of directional and sub field approach. This algorithm keeps the topology, connectivity and end points of the original image in the modified thinned image and can yield one pixel thin image. But it has staircase effects in the output image and produces some extra pixels also.

IV PROPOSED THINNING ALGORITHM

In this section, we describe our proposed thinning algorithm. This is a parallel thinning algorithm which uses the 3x3mask shown in Fig 3 and also an extended mask shown in Fig 4.

Pic_9	Pic_2	Pic_3	Pic_{43}
Pic_8	Pic_1	Pic_4	Pic_{44}
Pic_7	Pic_6	Pic_5	Pic_{54}
Pic_{67}	Pic_{66}	Pic_{56}	Pic_{55}

Fig 4: Extended 4x4 mask used in proposed algorithm

The proposed thinning algorithm works on two phases. The first phase has two sub iterations. Phase1 will continue iteratively until no more changes with respect to last iteration are determined. Phase1 will produce either one pixel thin images with some redundant points in some cases or keep the input image intact. The aim of phase2 is to delete the redundant points produced by phase1 and make one pixel thin output when the phase1 kept the input image intact.

The proposed algorithm requires the computation of the following parameters.

(a) A = number of 0 to 1 transition in the ordered set (Pic_2, Pic_3, Pic_4, Pic_5, Pic_6, Pic_7, Pic_8, Pic_9, Pic_1).
(b) B = number of non-zero neighbor in the 8 neighbors of pivot pixel p_1.
(c) $M_1 = (Pic_2 * Pic_4 * Pic_6)$ for first sub iteration and $(Pic_2 * Pic_4 * Pic_8)$ for second sub iteration.
(d) $M_2 = (Pic_4 * Pic_6 * Pic_8)$ for first sub iteration and $(Pic_2 * Pic_6 * Pic_8)$ for second sub iteration.
(e) $C = (\sim Pic_2 * (Pic_3 \mid Pic_4)) + (\sim Pic_4 * (Pic_5 \mid Pic_6)) + (\sim Pic_6 * (Pic_7 \mid Pic_8)) + (\sim Pic_8 * (Pic_9 \mid Pic_2))$.
(f) $N_1 = (Pic_9 \mid Pic_2) + (Pic_3 \mid Pic_4) + (Pic_5 \mid Pic_6) + (Pic_7 \mid Pic_8)$
(g) $N_2 = (Pic_2 \mid Pic_3) + (Pic_4 \mid Pic_5) + (Pic_6 \mid Pic_7) + (Pic_8 \mid Pic_9)$
(h) $N = \min(N_1, N_2)$
(i) $M_3 = ((Pic_2 \mid Pic_3 \mid \sim Pic_5) * Pic_4)$

Here, "~" denotes NOT, "|" denotes bitwise OR and "*" denotes multiplication.

Algorithm:

Phase 1:

Scan each pixel from left to right and top to bottom

- **if** (Pic_1 * Pic_4 * Pic_5 * Pic_6 = 1) **AND** (Pic_2 | Pic_3 | Pic_7 | Pic_8 | Pic_9 = 0) **then**
- **if** (Pic_{43} | Pic_{44} | Pic_{54} | Pic_{55} | Pic_{56} | Pic_{66} | Pic_{67} = 0) **then**
- Mark Pic_4, Pic_5 and Pic_6 deletable.
- **end if**
- **end if**
- **if** a). (A = 1) **AND** b). (B ≥ 3 **AND** B ≤ 6) **AND** c). (M_1 = 0)**AND** d). (M_2 = 0) **then**
- Mark Pic1 deletable.
- **end if**
- Delete all pixels marked as deletable.

Phase 2:

Scan each pixel from left to right and top to bottom

- **if** (C = 1) **AND** (N ≥ 2 **AND** N ≤ 3) **AND** (M_3 = 0) **then**
- Mark Pic1 deletable.
- **end if**
- **if** (Pic_4 | Pic_5 | Pic_6 = 0) **then**
- **if** (Pic_1 * Pic_2 * Pic_3 * Pic_8 = 1) **OR** (Pic_1 * Pic_8 * ~Pic_3 = 1)
- Mark Pic1 deletable.
- **end if**
- **end if**
- **if** (Pic_1 * Pic_4 = 1) **AND** (Pic_2 | Pic_3 | Pic_5 | Pic_6 | Pic_7 | Pic_8 | Pic_9 = 0) **then**
- **if** (Pic_{43} | Pic_{54} = 1) **then**
- Mark Pic1 deletable.
- **end if**
- **end if**

V EXPERIMENTAL RESULT AND DISCUSSION

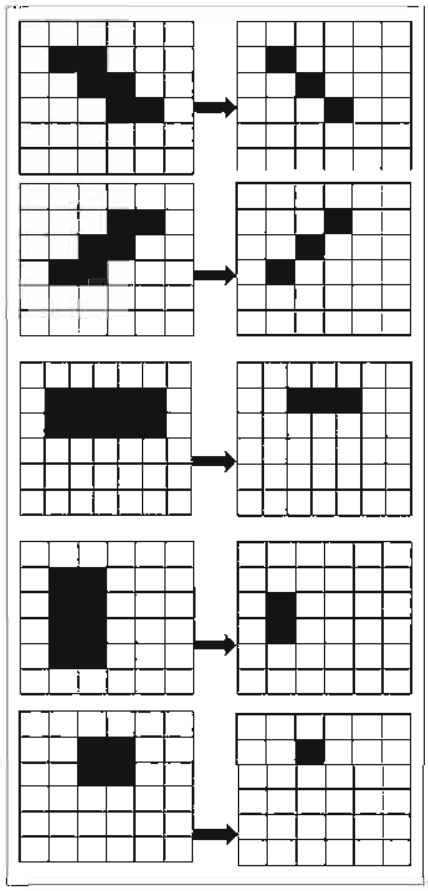

Fig 5: Thinning result of two pixel width images using the proposed algorithm

The programs are written in C++ using OpenCV 2.9 in Ubuntu 14.10. The proposed algorithm is tested on different kinds of images and we find our approach is robust and good enough to produce one pixel width skeleton. The proposed algorithm is tested on many different kinds of images but in this paper only a few of them are shown to demonstrate the performance of this algorithm. Fig 5 shows the input image and corresponding output image produced by this algorithm for some of the two pixel width images where other algorithms have problems. Fig 6 shows the input image and corresponding output image produced by this algorithm for some other varieties of images.

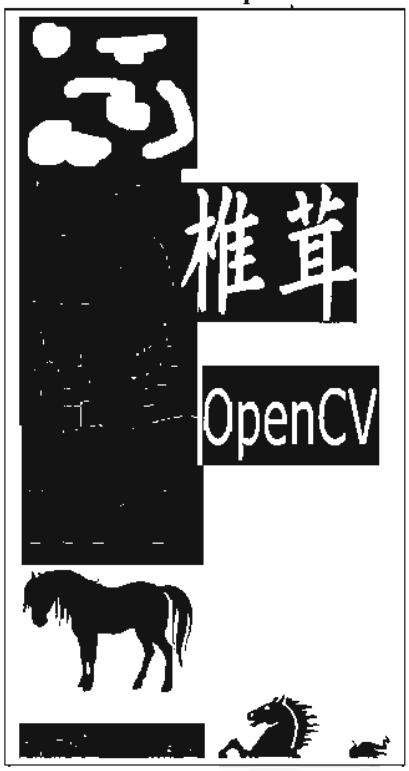

Fig 6: Thinning result of different images using the proposed algorithm

(a) Comparison result with other parallel algorithms

Our proposed algorithm is compared with other parallel algorithms: ZS algorithm [1], LW algorithm [2], HSCP algorithm [3], GH algorithm [5], Hilditch's algorithm [6], Jagna and Kamakshiprasad algorithm [7], Ben, Sider and Tari algorithm [8]. All of these algorithms are implemented using same platform using same language. Table 1 shows the comparison results. The parameters to measure the performance of the algorithm are represented by the number i to xi.

(i) Connectivity
(ii) Single pixel
(iii) Medial line approximation
(iv) Junction point elimination
(v) Two pixel width slant line problem
(vi) Two pixel width square problem
(vii) Two pixel horizontal/ vertical line problem
(viii) Unnecessary branches
(ix) Coordinate position dependency
(x) Redundant point
(xi) Staircase effects

The algorithms are referred by their reference number in table1. The proposed algorithm is referred as MS algorithm. "√" denotes yes.

Table1
Comparison of other thinning algorithm with MS algorithm

	[1]	[2]	[3]	[5]	[6]	[7]	[8]	MS
i.	√	√				√	√	√
ii.				√			√	√
iii.	√	√		√	√	√	√	√
iv.				√	√		√	√
v.	√	√	√		√	√		
vi.	√	√			√	√		
vii.			√					
viii.							√	
ix.							√	
x.				√	√		√	
xi.				√	√		√	

VI CONCLUSION

This paper improves the performance of the existing iterative image thinning algorithms. The proposed algorithm performs very well in the extraction of skeleton with no diagonal line or square or horizontal /vertical line problem compared to previous thinning algorithms. The proposed thinning approach has a good potential to be used in different image processing applications to improve their performance. In future, we shall use this method as an important preprocessing step for thinning 2D image.

REFERENCES

[1] T. Y. Zhang and C. Y. Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns", ACM, VOL.27, NO. 3 March 1984.

[2] H.E. Lu and P.S.P. Wang, "An improved fast parallel algorithm for thinning digital patterns", IEEE Conf. on computer vision and pattern recognition .pp. 364-367, 1985.

[3] Christopher M. Holt, Alan Stewart, Maurice Clint, and Ronald H. Perrott, "An Improved Parallel Thinning", Communications of the ACM, Volume 30 Number 2, February 1987.

[4] Y.Y. Zhang and P.S.P. Whang, "A Modified Parallel Thinning Algorithm", CH2614-6/88/0000/1023, IEEE 1988.

[5] Z. Guo and R. Hall, "Parallel thinning with two-sub iteration algorithms", Communications of the ACM, vol. 32, pp. 359–373, Mar 1989.

[6] http://cgm.cs.mcgill.ca/~godfried/teaching/projects97/azar/skeleton.html [Accessed: 10-October-2015]

[7] A. Jagna and V. Kamakshiprasad, "New parallel binary image thinning algorithm", ARPN Journal of Engineering and Applied Sciences. VOL.5, NO.4, April 2010.

[8] Lynda Ben Boudaoud, Abderrahmane Sider , Abdelkamel Tari, "A New Thinning Algorithm for Binary Images".