Advances in Intelligent Systems Research, volume 147
International Conference on Network, Communication, Computer Engineering (NCCE 2018)

Research on the Improvement of EPTA Parallel Thinning Algorithm

Rui Li [a)] and Xiaoyu Zhang [b)]

School of Computer and Communication, LanZhou University of Technology, Lanzhou 730050, China.

[a)] lirui@nwnu.edu.cn
[b)] Corresponding author: 707217733@qq.com

Abstract. Aiming at the problem of non-smooth contours (contour noises, scratches and jitters), it is easy to generate redundant branches. Based on EPTA algorithm, an improved parallel thinning algorithm is proposed. Through an iteration that eliminates the restrictions and proportional iterations, the pixels in the no-branching phenomenon are first regularly refined; then, the global smoothing refinement is performed to solve the branching problem caused by partial pixels. Experimental results show that the improved algorithm can solve the problem of the existing thinning algorithm, such as poor noise immunity, redundant branches and thin lines, and guarantee the robustness of the thinning results. At the same time, the erosion problem of the original algorithm for 4x4 square is also solved.

Key words: Contours Noises; Redundant Branches; EPTA Algorithm; Parallel Thinning Algorithm.

INTRODUCTION

Image thinning algorithm is a method to extract the image skeleton, which is widely used in character recognition [1], biological engineering [2], fingerprint identification [3] and so on. The purpose of refinement is to eliminate a large number of unwanted points to extract the refined skeleton, preserve the geometric features of the pattern so that the computer can perform image-related tasks efficiently [4]. Image thinning algorithms can be roughly divided into two categories: non-iterative thinning algorithms and iterative thinning algorithms. Non-iterative thinning algorithm directly extracts the image skeleton after a round of calculation [6], [7]. The iterative thinning algorithm can be divided into serial thinning algorithm and parallel thinning algorithm. For the serial thinning algorithm, the deletion of pixels depends on all operations previously performed, and the pixels are deleted as soon as the deletion condition is satisfied. For the parallel thinning algorithm, the deletion of pixels depends on the result after the last iteration. All the pixels satisfying the deletion condition will be marked and deleted after completing an iteration.

A good thinning algorithm usually has the following features [8]:

After the image with the connection structure is refined, it should remain connected (connection retention).

- Refined close to the original position of the finish line (no excessive erosion).
- The thinning result should be at least 8 connections (single pixel width).
- After thinning, it should be as close as possible to the central axis of the original image (approximation of the center line).
- The impact of external noise should be minimized (boundary noise immunity).

At present, there are many thinning algorithms [9], [10], [11], [12]. However, many parallel thinning algorithms are applied in image processing. There are mainly ZS (ZHANG and SUEN) thinning algorithm [13], LW thinning algorithm [14], [15] and EPTA (Enhanced Parallel Thinning Algorithm) thinning algorithm [16].

CLASSICAL PARALLEL THINNING ALGORITHM

ZS Parallel Thinning Algorithm

The ZS parallel thinning algorithm [13] adopts the 8-neighborhood method to delete and judge each pixel, delete the pixel that meets certain conditions, and iteratively delete it until it is completed. Assuming that the current pixel is p, its 8-neighborhood form looks like Fig.1:

p_9	p_2	p_3
p_8	p	p_4
p_7	p_6	p_5

FIGURE 1. The 8-neighborhood of a pixel p.

When the pixel value of p is 1, it means that the point belongs to the former scenic point; when the pixel value of p is 0, it means that the point belongs to the background point.

The ZS parallel thinning algorithm consists of two sub-iteration processes. First remove the pixels in the southeast and northwest corners, then eliminate the pixels in the northwest and southeast corners. During the first iteration, the pixel p that meets the following conditions is deleted:

$$\text{(a)}\quad 2 \le N(\mathrm{p}) \le 6$$
$$\text{(b)}\quad B(\mathrm{p}) = 1$$
$$\text{(c)}\quad \mathrm{p}_2 \times \mathrm{p}_4 \times \mathrm{p}_6 = 0$$
$$\text{(d)}\quad \mathrm{p}_4 \times \mathrm{p}_6 \times \mathrm{p}_8 = 0$$

During the second iteration, if the pixel p meets the following conditions, it will be deleted:

$$\text{(a)}\quad 2 \le N(\mathrm{p}) \le 6$$
$$\text{(b)}\quad B(\mathrm{p}) = 1$$
$$\text{(e)}\quad \mathrm{p}_2 \times \mathrm{p}_4 \times \mathrm{p}_8 = 0$$
$$\text{(f)}\quad \mathrm{p}_2 \times \mathrm{p}_6 \times \mathrm{p}_8 = 0$$

Where $N(\mathrm{p})=\sum_{i=2}^{9} \mathrm{p}_i$, it represents the number of non-zero pixels in the 8-neighborhoods of the current pixel p. $B(\mathrm{p})$ represents the number of changes from 0 to 1 in the neighborhood pixel of the current pixel p.

The ZS algorithm is simple and efficient. After the image is refined, it is close to the central axis and has good connectivity. Most of the pixels are single pixels. It still has some drawbacks, such as pixel redundancy, two-pixel wide slashes to refine transition erosion, redundant branches, and so on. Boudaoud et al. [17] used the parity and extended deletion conditions of pixel coordinates to make changes to the two sub-iteration processes of the ZS algorithm and solved the 2×2 square refinement erosion problem. However, the 4×4 square erosion problem cannot be solved, and more uneven branches are easily generated for the unsmooth contour image.

LW Thinning Algorithm

The LW thinning algorithm [14], [15] changes the condition (a) in sub-iteration 1 to $3 \le N(\mathrm{p}) \le 6$. It solves the defect of the ZS algorithm's two-pixel wide oblique line refinement distortion. At the same time, it also causes the increase of redundant pixels and the serious phenomenon of branching after refinement.

EPTA Thinning Algorithm

The EPTA thinning algorithm [16] is divided into two phases. On the basis of the ZS algorithm, the first phase retains the set of deletion points that satisfy $N(p)=2$; in the second phase, two sub-conditions are used to remove redundant pixels:

$$(g)\ \left(p_6 \times p_8 = 1\right) \cap \left(p_3 = 0\right)$$

$$(h)\ \left(p_4 \times p_6 = 1\right) \cap \left(p_9 = 0\right)$$

The EPTA algorithm largely solves the problem of the ZS algorithm slashes the information erosion and refinement of the redundant pixels. The refinement efficiency is relatively high, but the algorithm is still lacking in anti-noise. For the unsmooth outline image, the fine After the production of more branches. The IEPTA algorithm proposed in [19] is based on the EPTA algorithm and uses elimination template refinement in four different sub-iterations respectively, and adds two decision conditions based on the EPTA algorithm:

$$(i)\ \left(p_2 \times p_4 = 1\right) \cap \left(p_7 = 0\right)$$

$$(j)\ \left(p_2 \times p_8 = 1\right) \cap \left(p_5 = 0\right)$$

IEPTA mainly solves the problem of pixel redundancy. The improvement of the branching phenomenon is affected by the order of iterations and the number of iterations in the four directions. The algorithm design does not aim at pixel points that cause branching and does not fundamentally solve the branching problem.

THINNING ALGORITHM BASED ON EPTA IMPROVEMENT

Based on the EPTA algorithm, this paper proposes an improved EPTA algorithm for the fine branching problem caused by the unsmooth contours of the image (contour noise, scratches, and jitter, etc.) by increasing the restrictions on the deletion conditions and the iterative scaling mechanism. Change the order of pixels to be deleted and refined and achieve no branch refinement for various contours of unsmoothed images.

The branching phenomenon caused by the unsmooth contour of the image is usually caused by uneven edges of the image. In the refinement, the points where the depressions meet the conditions are also refined at the same time, causing the depressions to slowly spread to both sides, and finally forming a refinement effect of the “V” character. This algorithm lags behind the smooth contour by the point in the concave part that satisfies the refinement condition, first refines and deletes some pixels of the non-depressed contour, and then performs global refinement to make the pixel in the depression a smooth contour pixel. At the same time, in order to ensure the refinement efficiency and the robustness of the algorithm, a proportional mechanism is used to limit the number of iterations it increases.

Figure 2 is a partially enlarged view of the end of the stroke at the fourth position of the word "Yong", and its edge contour is not smooth due to the depression. Through the analysis of the EPTA algorithm and the contour non-smoothing branch, it is found that the edge foreground pixel bumps are severely caused by contour noise, scratches, and jitter. The sag of these edge outline points causes some pixel points (relative to the smooth outline) to be iteratively deleted in advance, resulting in a refinement of the branching phenomenon. Therefore, the key to achieving no branch refinement is how to avoid these pixel points being deleted iteratively in advance.

FIGURE 2. Partial enlargement of the word "Yong"

The two-pixel points A and B shown in FIG. 3 below are due to the absence of their right-side pixel points so that they can be deleted iteratively in advance. When these two pixels are deleted in advance, they will affect the adjacent A' and B' pixels so that they can be deleted in advance. Such loop iterations, due to the effect of the sag, will run through the thinning algorithm at all times and will eventually lead to refinement of the branching phenomenon. FIG. 3 shows the situation caused by the right-side depression and the same reason there are the left side, the upside, and the underside. This type of point is usually released early and iteratively deleted due to anomalies in the sag. In the ideal case its deletion is lagging behind one iteration.

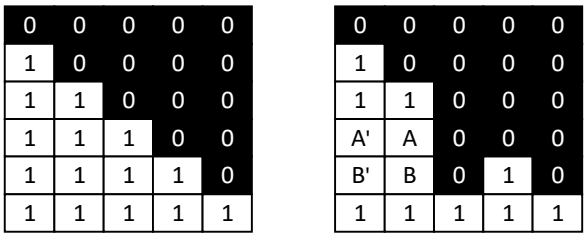


FIGURE 3. Contour analysis: (a) smooth outline, (b) unsmooth outline

The Improved Thinning Algorithm

In this section, on the basis of EPTA algorithm, two sub-iterations are added to delay the deleting of smooth contour points. In this way, the algorithm turns from the original two sub-iterations into four sub-iterations and defines the $Cp=p_2+p_4+p_6+p_8$ as the deletion restriction of the first two sub-iterations. In the first two iterations, we delete the deleted pixels that satisfy $Cp<3$, which are used to delay the deletion of the pixels that cause branching phenomena in the concave, and the global iteration is deleted in the latter two iterations.

At the same time, in order to ensure the efficiency of the iteration, an iterative scaling mechanism is added to allocate the execution ratio of the first two sub-iterations and the last two sub-iterations: Let $M = 4k(k=1,2,3,...)$ be an integer multiple of 4. Use the set $S=\left\{ r \mid r \in \sum_{n=0}^{M-1} n \bmod M \right\}$ to represent all the remainder of the set M. Then define four sets of A, B, C and D to describe the distribution of the remainder:

$$A = \{a \mid a \in S \cap a = 0+4i, i=0,1,2...\}$$

$$B = \{b \mid b \in S \cap b = 1+4i, i=0,1,2...\}$$

$$C = \{c \mid c \in S \cap c = 2+4i, i=0,1,2...\}$$

$$D = \{d \mid d \in S \cap d = 3+4i, i=0,1,2...\}$$

For a given k value, all the remainder sets of M are determined, and are evenly allocated to the four sets of A, B, C and D, and the execution order and times of the four sub-iterations are determined respectively.

The Process of the Algorithm

The algorithm is divided into two parts: In the first part, by using the delete constraint condition, the pixels that do not produce branching are deleted in the first two sub-iterations; in the second part, global deletion is performed in the last two iterations, and no longer considered. Delete conditional restrictions. The first part and the second part are executed according to the assigned ratio until the refinement is completed. The steps of the algorithm are as follows:

Step1: Determine the deleting ratio (value of k) and the iterable element values in the four sets of A, B, C and D, and initialize the number of judgments $N=0$.

Step2: Judge the value of $N \bmod M$. If it belongs to the iterable element value in the set A, then sub-iteration 1 is performed: a point marker that satisfies the four conditions (a), (b), (c), (d) and removes the constraint condition $Cp<3$ and waits once Delete after scanning. Then $N=N+1$.

Step3: Judge the value of $N \bmod M$. If it belongs to the iterable element value in the set B, then sub-iteration 2 is performed: a point marker that satisfies the four conditions (a), (b), (e), (f) and removes the constraint condition Cp<3 and waits once Delete after scanning. Then $N=N+1$.

Step4: Judge the value of $N \bmod M$. If it belongs to the iterable element value in the set C, then sub-iteration 3: the point marker that satisfies the four conditions (a), (b), (c), (d) at the same time is to be deleted after one scan is completed. Determine if there are any marker deletions. If yes, then $N=N+1$, execute (5); otherwise, execute (6).

Step5: Judge the value of $N \bmod M$. If it belongs to the iterable element value in the set D, then sub-iteration 4: the point marker that satisfies the four conditions (a), (b), (e), (f) at the same time is to be deleted after one scan is completed. Determine if there are any marker deletions. If yes, then $N=N+1$, execute (2); otherwise, execute (6).

Step6: Pixels that satisfy conditions (g), (h), (i), and (j) are deleted.

Analysis of Algorithms

The improved algorithm lags the deletion order of some pixels that satisfy the deletion conditions and can easily cause branching, and it effectively solves the branching problem caused by unsmooth contours. Moreover, due to the iterative ratio mechanism, refinement can solve the branch problem and ensure the efficiency of the algorithm. At the same time, the phenomenon of transition erosion of the EPTA algorithm in the 4×4 square is also solved, as shown in FIG. 4:

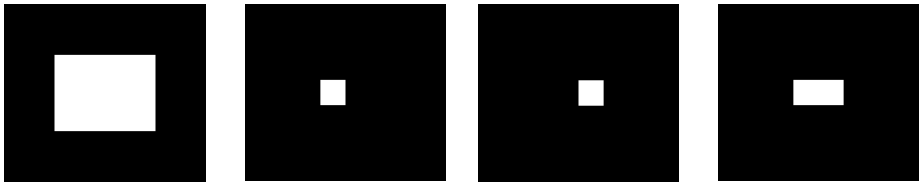


FIGURE 4. Thinning result of a 4 × 4 square: (a) original image, (b) result of EPTA algorithm, (c) result of IEPTA algorithm, (d) result of improved algorithm

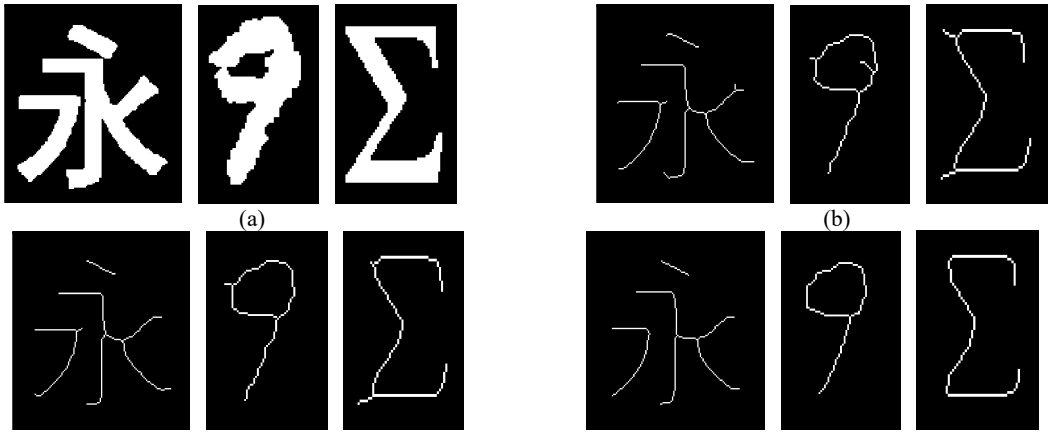
IMPLEMENTATION, TESTS AND RESULTS

In order to verify the effect of the improved algorithm, select a large number of characters (Chinese characters, letters and numbers), symbols (mathematical symbols, punctuation and special characters) and fingerprint images, respectively, the EPTA algorithm and the IEPTA algorithm were compared experimentally, and some experimental results were compared.

Figure 5 shows the thinning results of the three algorithms for non-smooth edges, Chinese characters, numbers, and symbols. In the third stroke of the Chinese character “Yong”, the EPTA algorithm has a right-side depression at the stroke transition and a branch after thinning; at the end of the fourth stroke, a “V” word phenomenon occurs after thinning and verification. Although the IEPTA algorithm has improved the branching phenomenon, it has not completely overcome it. The algorithm of this paper is directly aimed at those pixels that produce branching phenomena. The refinement effect of Chinese characters with unsmooth contours has been significantly improved, and the refinement result is a single pixel width.

For the thinning of the number "9", it can be seen that both the EPTA algorithm and the IEPTA algorithm show a disadvantage in terms of noise resistance, and they are subdivided on digital images with conspicuous bumps, and all have branches. This algorithm can implement robust thinning of the image with severe scratches, approaching the central axis, and ensure connectivity.

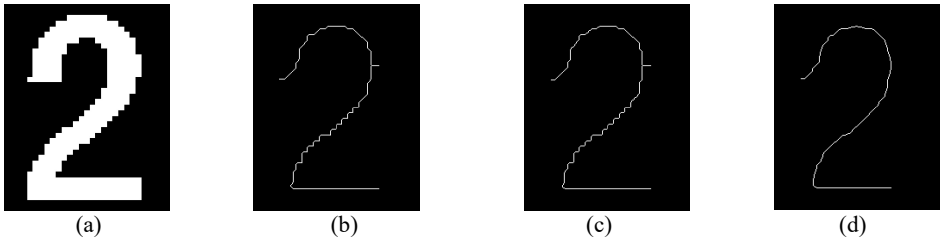
From the thinning result of the mathematical summation symbol, it can be seen that the EPTA algorithm and the IEPTA algorithm for the outer and inner left and left lower corners of the concave and convex profile, because there is no symmetry with the upper boundary, the lower boundary of the smooth contour, resulting in thinning results There are branching phenomena; while in the middle part, the convex and concave contours on the left and right sides form symmetry, so the thinning result is ideal. This algorithm can solve the negative effects brought by this unsmooth contour, while retaining the advantages of the original algorithm.



(a) (b) (c) (d)

FIGURE 5. Thinning result of Chinese characters, numbers and symbols: (a) original image, (b) result of EPTA algorithm, (c) result of IEPTA algorithm, (d) result of improved algorithm

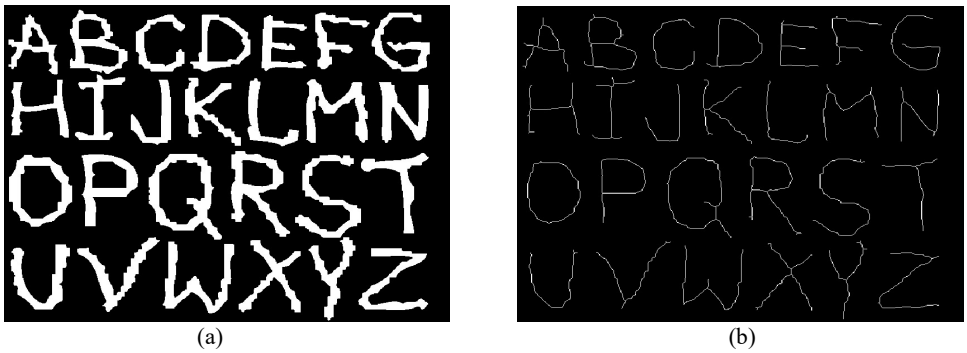
Figure 6 shows the thinning of the number "2". Due to the unevenness of the image contours, the EPTA algorithm and the IEPTA algorithm have branches at the right curve when the refinement is performed, and the middle curve part is also affected by the contour unevenness. After the refinement, the curve appears wavy and is not smooth. The improved algorithm solves this problem very well and obtains a smooth, branchless refinement.



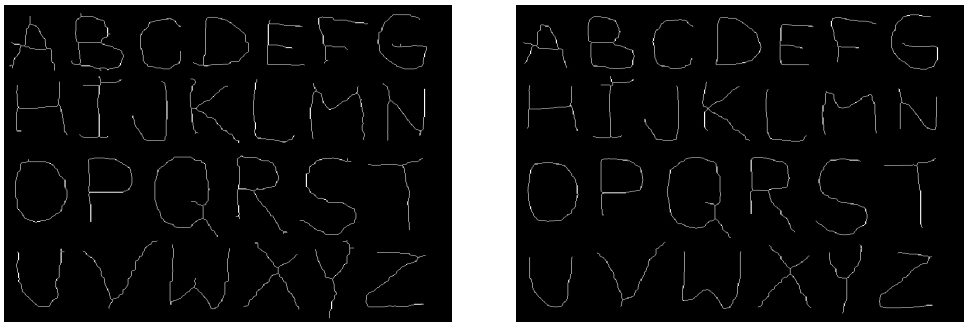
(a) (b) (c) (d)

FIGURE 6. Thinning result of the number "2": (a) original image, (b) result of EPTA algorithm, (c) result of IEPTA algorithm, (d) result of improved algorithm

Figure 7 shows the effect of the three algorithms on uppercase letters. The edges of the uppercase letters shown in the original image are not smooth. Experimental results show that the improved algorithm can handle this image better. For example, 'A', 'B', 'H', 'S', 'U', 'W', 'Y' and other letters, the EPTA algorithm and the IEPTA algorithm are more serious after the thinning. Moreover, the thinning effect of the IEPTA algorithm is affected by the four iteration sequences, such as 'B' and 'G'. When the number of iterations is high, the thinning effect is counterproductive. The improved algorithm has better stability and is not affected by the number of iterations. After thinning, the number of branches is greatly reduced, and the thin lines are also smoother.



(a) (b)



(c) (d)

FIGURE 7. Thinning result of the capital letters: (a) original image, (b) result of EPTA algorithm, (c) result of IEPTA algorithm, (d) result of improved algorithm

Table 1 shows the experimental result data of the EPTA algorithm, the IEPTA algorithm, and the improved algorithm, including the object pixel number (*OP*), deleted points (*DP*), the thinning rate (*TR*), and the thinning speed *TS* (in units of p/s). These parameters are given by the formulas proposed in [17] and [20]:

$$TR = 1 - \frac{TM1}{TM2} \tag{1}$$

$$TM1 = \sum_{i=0}^{n} \sum_{j=0}^{m} TC(P[i][j]) \tag{2}$$

$$TM2 = 4 \times \left[\max(m,n) - 1\right]^2 \tag{3}$$

Where *TC* is a function used to calculate the number of neighboring triangles that can be created around a white pixel:

$$\begin{aligned} TC(P[i][j]) = {} & P[i][j] \times P[i][j-1] \times P[i-1][j-1] \\ & + P[i][j] \times P[i-1][j-1] \times P[i-1][j] \\ & + P[i][j] \times P[i-1][j] \times P[i-1][j+1] \\ & + P[i][j] \times P[i-1][j+1] \times P[i][j+1] \end{aligned} \tag{4}$$

The target pixel number *OP* refers to the number of pixel points of the object to be refined, that is, the number of white pixel points. The deleted points number *DP* is the number of changes in the pixel before and after the refinement. Thinning speed $TS = \frac{DP}{ET}$, where *ET* denotes the thinning algorithm execution time. In formula (3), *m* and *n* represent the image dimensions. When *TR* =1, the image is perfectly refined; when *TR* =1, it is not refined.

TABLE 1. Experimental results of EPTA, IEPTA, and improved algorithms.

Images	OP	EPTA Algorithm			IEPTA Algorithm			Improved Algorithm		
		DP1	TR1	TS1(p/s)	DP2	TR2	TS2(p/s)	DP3	TR3	TS3(p/s)
The Word "Yong"	4733	4403	0.999963	2633	4415	0.999974	2745	4437	0.999987	2643
The Number "9"	2095	1945	0.999952	2645	1952	1.000000	2736	1961	1.000000	2630
Summation Notation	1001	859	1.000000	2953	863	0.999950	3126	876	1.000000	2988
The Number "2"	14061	13598	0.999967	1399	13603	0.999992	1402	13692	0.999996	1407
Alphabet	91666	84952	0.999959	1993	85005	0.999988	2022	85358	0.999990	2047

From the results in Table 1, we can see that the improved algorithm can achieve better refinement under the premise of satisfying basic refinement conditions. The increase in it indicates that the number of refined pixels is large, further illustrating the improvement of the branching phenomenon. At the same time, *TR* is closer to 1 than EPTA algorithm and IEPTA algorithm, indicating that the improved algorithm has a better granularity for redundant pixels.

Although the algorithm adds 2 sub-iterations, due to the limitations of the iterative scaling mechanism, the sub-iteration only limits the number of pixels that produce branching, allowing them to wait until the outer contour is refined to smooth, and then refine them together instead of the refinement of the refinement is performed at the same time as the external contour and the refinement is completed in advance. In contrast, the improvement of the branching phenomenon is much better than this. When the *OP* is small, the *TS* is slightly higher than the EPTA algorithm, slightly lower than the IEPTA algorithm, which ensures the refined efficiency; when the *OP* is more, the *TS* is improved compared to the EPTA algorithm and the IEPTA algorithm, indicating that the algorithm has a larger processing resolution the performance of the image is better.

CONCLUSION

This paper analyzes the reason why the unsmooth contour is easily refined after branching and proposes an improved EPTA algorithm based on the EPTA algorithm. The improved algorithm eliminates the limitation and iterative scaling mechanism and solves the effect of the branching phenomenon on the pixel points where the off-smooth contour generates branching phenomenon and ensures the refinement efficiency. The experimental results show that the improved algorithm effectively solves the problems of poor anti-noise performance, redundant branches and thin lines in the existing thinning algorithm, guarantees the refinement efficiency, and has good robustness and practical application value. At the same time, it also solves the problem of the original algorithm for the 4×4 square transition erosion.

REFERENCES

1. K. C. Santosh and K. Wendling, Front. Comput. Sci. 9, 678–690 (2015).
2. J. M. Park, S. Y. Park and H. Kim, Phys. Med. Biol. 60, 7101–7125 (2015).
3. A. Salazargonzalez, D. Kaba, Y. M. Li and X. H. Liu, IEEE J. Biomed. Health Inform. 18, 1874–1886 (2014).
4. M. Fons, F. Fons and E. Canto, IEEE Trans. Circuits Syst. II Express Briefs 57 991–995 (2010).
5. Y. S. Chen and M. T. Chao, J. Imaging 3, 29 (2017).
6. J. T. Bi, J. Comput. 8, 3012–3019 (2013).
7. G. Bertrand and M. Couprie, Graph. Models 80, 1–15 (2015).
8. J. S. Bwon, Int. J. Multimed. Appl. 5, 1–14 (2013).
9. Y. S. Chen and M. T. Chao, Int. J. Patt. Recogn. Artif. Intell. 30, 1654001 (2016).
10. P. K. Saha, G. Borgefors and G. S. di Baja, Pattern Recogn. Lett. 76, 3–12 (2016).
11. G. Németh, P. Kardos and K. Palágyi, Acta Cybernet. 20, 125–144 (2011).
12. K. Palágyi, Int. J. Patt. Recogn. Artif. Intell. 28, 1460009 (2014).
13. T. Y. Zhang and C. Y. Suen, Commun. ACM 27, 236–239 (1984).
14. H. E. Lu and P. S. P. Wang, “An improved fast parallel thinning algorithm for digital patterns,” in Computer Vision and Pattern Recognition, IEEE Conference Proceedings (IEEE, San Francisco, CA, 1985), pp. 364–367.
15. H. E. Lu and P. S. P. Wang, Commun. ACM 29, 239–242 (1986).
16. J. J. Bao and J. Fan, Comput. Aided Eng. 15, 43-46 (2006).
17. L. B. Boudaoud, A. Sider and A. Tari, “A new thinning algorithm for binary images,” in Control, Engineering & Information Technology (CEIT), IEEE 3rd International Conference (IEEE, Tlemcen, Algeria, 2015), pp. 1–6.
18. J. W. Dong, W. H. Lin and C. Huang, “An improved parallel thinning algorithm,” in Wavelet Analysis and Pattern Recognition (ICWAPR), IEEE 7th International Conference (IEEE, Jeju, South Korea, 2016), pp. 162–167.
19. D. D. Zhao, H. B. Wang, L. Tao and J. Zhou, Comput. Eng. Appl. 52, 196–201 (2016).
20. R. W. Zhou, C. Quek and G. S. Ng, Pattern Recogn. Lett. 16, 1267–1275 (1995).