UDC 004.932.72'1;004.93'14

A NEW IMPROVED FAST PARALLEL SKELETONIZE ALGORITHM

Ma JUN, V.Yu. TSVIATKOU, V.K. KONOPELKO

Belarusian state university of informatics and radioelectronics, Republic of Belarus

Submitted 8 March 2019

Abstract. The skeletonize of the binary image has crucial application in the field of target recognition. The thinning result of the Zhang's fast parallel thinning algorithm maintains the connection of the original image, has a good structural form and has no burs. However, Zhang's fast parallel thinning algorithm consists two subiterations in where the most of operations are similar, which cause the waste of the resources of the calculate. Besides, it can't ensure the single pixel width, which brings difficulties for post-processing. In this paper, authors make some improvement on the Zhang's fast thinning algorithm. Experimental results show that the use of improved algorithm can not only reduce the total number of the iteration but also ensure a single pixel width.

Keywords: Zhang's fast parallel thinning algorithm, number of iterations, single pixel

Introduction

Skeletonization, also known as the image thinning, is a pre-processing which is widely applied in field of the image processing and pattern recognition. Image thinning refers to finding the skeleton or centerline of the original image as quickly as possible while maintaining the completeness of the topology of the original image, and then replacing the original image with a single-pixel skeleton. The processing of the image thinning can dramatically reduce the superfluous information from the image, which relieve the computation burden of the computer and shorten the time which spend on the process of the recognition [1, 2].

Skeletonization is divided into two main approaches: iterative and non-iterative. In non-iterative approach, the skeleton is extracted directly without examine each pixel individually, however these methods are difficult to implement and slow as well. As for iterative techniques, which can further divided into two methods: the method of parallel processing and the method of processing sequentially. In the parallel way the whole unwanted pixels are erased after identify the whole wanted pixels, whereas in sequential way; the unwanted pixels are removed in the identifying the desired pixels in each iterative [3].

Among the parallel thinning algorithms, Zhang's algorithm [1] has the good performance in continuity and it can relatively precise describe the straight line, inflection point and cross point. However, at the same time, Zhang's algorithm also has some aspects can improved, which will have better performance, this paper will base on Zhang's algorithm to propose a improved ones.

Zhang's Algorithm

Zhang's algorithm consists of two sub-iterations. Iterative transformations are applied to original binary matrix point by point according to the values of a small set of neighboring points. It is assumed that a 3×3 window is used, and that each element connected with its eight neighboring elements which is shown in Fig. 1 [1, 4].

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 $(i+1, j+1)$

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

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

$(a)\ \ 2 \le B(P_1) \ \le 6,$

$(b)\ \ \mathrm{A}(\mathrm{P}_1) = 1,$

$(c)\ \ P_2 \cdot P_4 \cdot P_6 = 0,$

$(d)\ P_4 \cdot P_6 \cdot P_8 = 0,$

where $A(P_1)$ is the number of 01 patterns in the ordered set $P_2, P_3, P_4, ...P_8, P_9$ which are the eight neighbors of P_1.

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

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

In the second sub-iteration, only conditions (c) and (d) are changed as follows:

$(c')\ \ P_2 \cdot P_4 \cdot P_8 = 0,$

$(d')\ P_2 \cdot P_6 \cdot P_8 = 0.$

Zhang's algorithm works by following way. First, finding points which are satisfied the conditions of the first sub-iteration and delete them, and then location these points which are satisfied the conditions of the second sub-iteration and deleting them. The algorithm will not stop and continued to repeat this order forever until there is not any point which can be deleted.

The above algorithm has 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.

However, the main iteration of the Zhang's algorithm consists of two sub-iteration which means it needs two full scans of the image to eliminate the contour points, which cause the costs of the time is relatively high.

Moreover, Zhang's algorithm has not presented an ideal performance in single pixel, which also seen as the main factor to evaluate a thinning algorithm.

So, the proposed method is aimed to tackle the problem which mentioned above. This method will reduce the number of the total iterations and, at the same time, realize the single pixel as far as possible.

Improvement Algorithm

Improvement algorithm include 4 main parts: search module, connectivity check module, single pixel correction module and contour point delete module. The structure of these stages addressed in Flowchart as shown in Fig. 2.

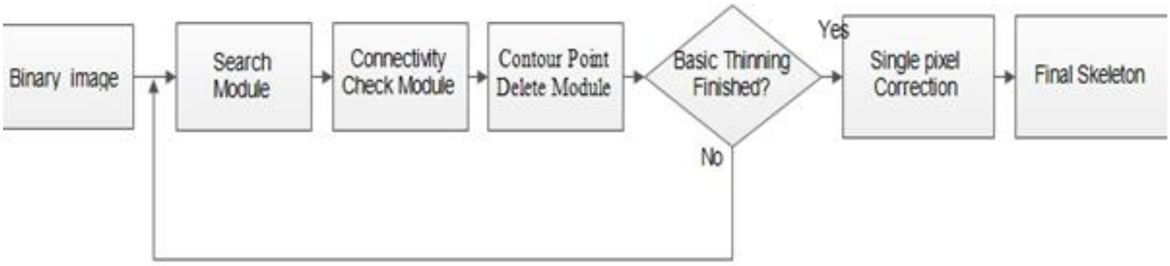


Fig. 2. Flowchart of the purposed method

The binary image is acquisition into the proposed method as black pixels which considered as foreground as well as consider an object pixel for deletion. The pixels having value 0 are considered as background pixels.

Before to introduce the search module, the 3×3 window which we used in the search module need to present, which is a little different from Zhang's algorithm as shown in Fig. 3.

P_5	P_4	P_3
P_6	P_1	P_2
P_7	P_8	P_9

Fig. 3. Designation of the 3×3 window

In the search module, 3×3 window is used to scan the point by point of the original binary image. The center of the window aims to the current pixel. If the current pixel is a foreground point, it's further observed its neighbor is match the corresponding conditions or not. And then if this point satisfied the conditions, we will consider it as a potential contour point and record it as «1» in the matrix *S*. But in order to give the condition, we divided all foreground points into two groups. One is named as general points set, another is named as special points set. The conditions of the general points set are completely the same as the first two condition (condition (*a*) and (*b*)) from Zhang's algorithm. The special points are 4 points which have shown below, they are «0» appear in the upper left corner (Fig. 4, *a*), in the upper right corner (Fig. 4, *b*), in the lower left corner (Fig. 4, *c*), in the lower right corner (Fig. 4, *d*) respectively.

0	1	1
1	1	1
1	1	1

a

1	1	0
1	1	1
1	1	1

b

1	1	1
1	1	1
0	1	1

c

1	1	1
1	1	1
1	1	0

d

Fig. 4. All Special Points

When faced with a special point which is looked like the one of Fig. 4, it's necessary to know these points which are sat around the 0 in this window are potential contour points or not. If they are all potential contour points, the current pixel is also a potential contour point. Otherwise, the current pixel cannot consider as a potential contour point. For example, if we meet a point which is looked like as (*b*) (Fig. 4). We need to confirm the P_4 and P_2 (Fig. 3). We know that P_4 has been checked because the location of P_4 is prior to the current pixel, as a result, we can straight take out the value of $S(x4, y4)$, where $x4$, $y4$ is the coordinate point of the P_4 to see P_4 is a potential contour point or not.

And then we need to check P_2, we didn't know the value of the $S(P_2)$, because the location of P_2 is behind current pixel, so we have to turn our 3×3 window to this pixel and test its neighbor to see it is a potential contour point or not. The judgment of the P_2 is as the same as the general points. When P_2 and P_4 are potential contour point, the original pixel will be returned. It will be considered as a potential contour point.

The function of the connectivity check module is to verify the connectivity under the hypothesis that we have deleted the potential contour points which stored in the matrix *S*. If the deletion of some points will break the connectivity of the image, this potential contour points will be excluded and it will be written as «0» in the matrix *S*. For different type of point set, deploying the different conditions of judgment is reasonable. If the 8-neighbor of a general point appeared as the same as the retention template which has shown in Fig. 5, it's necessary to retain this point.

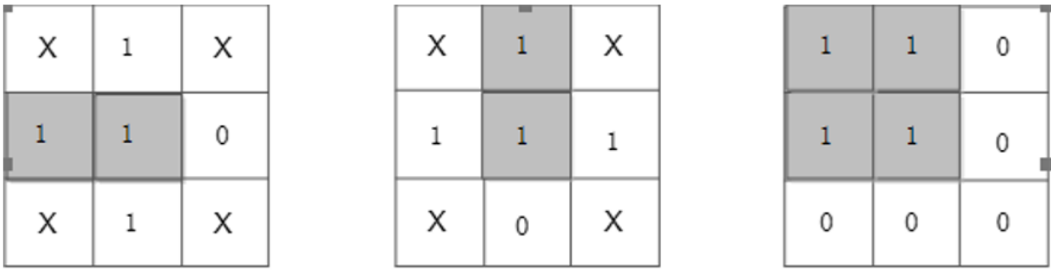


Fig. 5. Retention template for general point

In retention template, the pixel with deep color means that this pixel has been viewed as a potential contour point in the search module.

As for the special point, 4-neighbor points will be observed, if there are more than 3 points have been regarded as potential contour points, the current pixel deserved to keep.

After the connectivity check module, some potential contour points which may break the connectivity of the binary picture are eliminated. And then it's possible to remove contour points according to the matrix *S* from the original binary image with the confidence.

The last step is to conduct a single pixel correction, which is based on the result of the previous image thinning procession. After referred to the [4], a single pixel result is got if these points according to the correction template are deleted (Fig. 5).

0	1	0
1	1	0
0	0	0

0	1	0
0	1	1
0	0	0

0	0	0
0	1	1
0	1	0

0	0	0
1	1	0
0	1	0

0	1	0
0	1	1
0	0	1

0	0	1
0	1	1
0	1	0

0	0	0
0	1	1
1	1	0

1	1	0
0	1	1
0	0	0

0	0	1
0	1	1
1	1	0

1	0	0
1	1	0
0	1	1

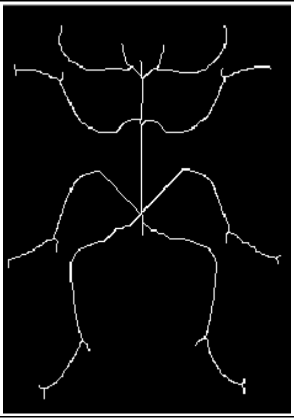
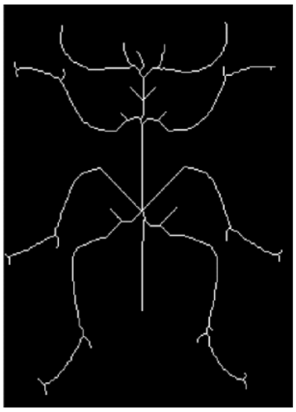
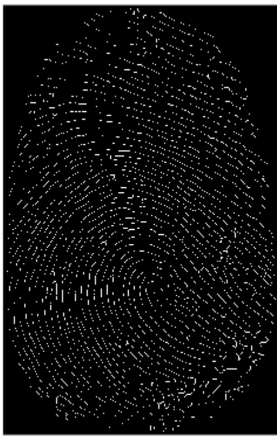
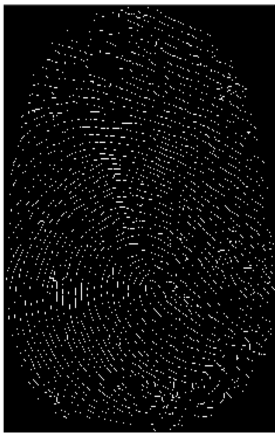
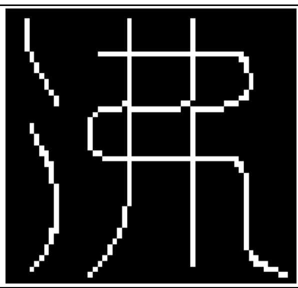
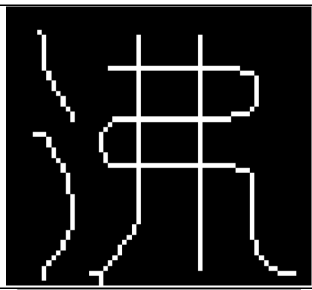
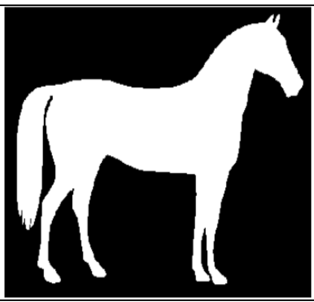
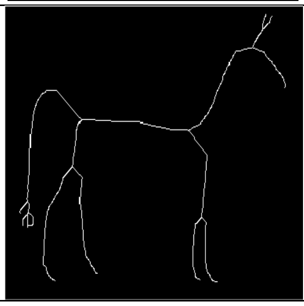
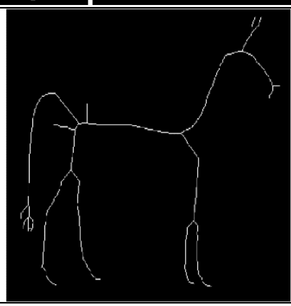
Fig. 6. Single pixel correction template

In order to easy to identify these points, it's possible to translate it into the binary code. The sequence is ordered according the 3×3 window which has shown in Fig. 1. The most significant bit is P_9 and least signification bit is P_2.

Experiments and results

To assess the performance, the improvement algorithm and the Zhang's algorithm were written in Matlab R2018b.This data set has many class shapes. The achieved results are presented in Tables 1 and 2.

Table 1. **The original binary images and the thinning results by different algorithm**

	Original binary image	Zhang’s algorithm	Improved algorithm
a			
b			
c			
d			
e			

According to the Table 1, new method has a better performance in the single pixel.

Table 2. **Comparison of the compute speed between algorithms**

Original binary image	Number of iterations		CPU time consumed(s)	
	Zhang's algorithm	Improved algorithm	Zhang's algorithm	Improved algorithm
a	14	8	0,3074	0,1846
b	82	42	0,6683	0,4427
c	63	32	9,8992	6,0417
d	18	9	0,1137	0,0757
e	89	45	1,9470	1,0489

Thus, improved algorithm has better performance in the single pixel and in the speed, which thanks to the reduction of the number of the iteration.

Theoretically, because the number of the iteration of the improved algorithm is half of the Zhang's algorithm, The CPU time consumed of our algorithm should also be a half when compare with the Zhang's algorithm. However, the result of the experiments reflect that proposed algorithm only cut down about 35 % time which as shown in the Table 2, because the single pixel is added. Correction parts to the algorithm increases the complex of the compute and prolong the execution time.

However, proposed algorithm has some weak point. With the increasement of speed, some burrs appear in the algorithm results. This problem can be solved by refer to the [5].

Conclusion

In this paper, we presented an improved algorithm based on the zhang's algorithm, which has better performance in speed and single pixel. The experiments have proved the effectivity of the new algorithm.

References

1. Zhang T.Y., Suen C.Y. // Communications of ACM. 1984. Vol. 27, No 3. P. 236–239.
2. Бушенко Д.А., Садыхов Р.Х. // Технология построения алгоритмов утоньшения для скелетизации протяженных объектов. 2009. No 7 (45). P. 81–86.
3. Waleed A.A. [et. al.] // Skeletonization Algorithm for Binary Images. Procedia Technology. 2013. P. 704–709.
4. Mu S.M., Du H.Y., Su P., Zhu X.H., Chen G.Y. // Microelectronics & Computer.2013. Vol. 30, No. 1. P. 53–55.
5. Ye F.L. // J. of Xichuang University. .2018. Vol. 32 (3), P. 91–93