An Efficient and Rotation Invariant 4-Step thinning Algorithm for Binary Document Images

Manjunath Aradhya V N*, 1Hemantha Kumar G, 2Shivakumara P
1Department of Studies in Computer Science, University of Mysore, Mysore, Karnataka, India
2Department of Computer Science, School of Computing, National university of Singapore (NUS), 3 Science, Drive 2, Singapore 117543

*Email: mukesh.mysore@rediffmail.com hudempsk@yahoo.com

Abstract- Skeletonization algorithms have played an important role in the preprocessing phase of OCR systems. In this paper, we present a novel technique for thinning a given image. The proposed algorithm has got unique feature that distinguishes the thinning system is that it thins symbols to their central lines. This means that the shape of the symbol is preserved. It also means that the method is rotation invariant. The algorithm has 4 steps which are applied to each object pixel in the image. This is generalized algorithm which can be used to thin symbols, digits, characters, irrespective of the script written, and the like. Several experimental results revealed that the proposed method shows good results compared to (Maher and Ward, 2002 and Gonzalez and Woods, 2002) in terms of accuracy and computations.

1 INTRODUCTION

Thinning is an important step in character recognition, since thinning reduces the computational burden as well as it improves character recognition rate. Previously developed thinning algorithms do not address the problem of rotation invariance property. In this paper, we present an algorithm that thins symbol without disturbing its topology or shape. Thus, it can be applied to any symbol or character written in any language. It has also the advantage of producing the same thinned symbols regardless of rotation of the original symbols. Our algorithm also does not produce extraneous pixels (distortions).

A comprehensive survey of thinning algorithms is described by Lam et al. [4]. Templates of (3 x 3) and (3 x 4), and (4 x 3) windows are usually used to perform thinning and trimming of extraneous pixels [5-8]. The algorithm described in [6] has the advantage that its rules are applied in one sequential pass only. In [8], an iterative parallel thinning algorithm that uses four passes per iteration is proposed. A modified algorithm with sub iterations is proposed in [9]. The later method, however, may produce extraneous branches. A two-step algorithm that improves thinning results of Chinese characters is described in [10]. In [11], an artificial neural network, based on adaptive resonance theory, is proposed for thinning Arabic characters. The ZS algorithm described in [12] has the advantage of speed. It also preserves the topology of the image in the majority of cases. However, all the pixels in the case of 2 x 2 image pixels will be deleted. In addition, a 45°, 135° long or short diagonal line of 2 pixels width is reduced to a dot of 1 or 2 pixels. A solution for this case is suggested in [13]. The algorithm proposed in [14] produces thinner results (fewer pixels) than the above ZS algorithm, maintains high speed, and also generates one pixel wide skeletons. The HSCP algorithm described in [15] uses the first two conditions as in the ZS method to determine which pixel can be deleted. The HSCP algorithm is compared to the ZS algorithm [16] and suggests a solution to solve the diagonal line problems in the ZS and HSCP algorithms.

Many thinning algorithms, their performance and evaluation have been studied in [17]. None of the previous methods address rotation invariant thinning. Many are specific to digits, characters, or letters, written in English, Chinese, Arabic, or any other scripts. However, to solve above problems, the rotation invariant rule-based thinning algorithm for character recognition is proposed by [21]. This is generalized algorithm, which is used to thin the symbols irrespective of their scripts. This method is rotation invariant. This system preserves the topology and does not produce any discontinuity since the resultant thinned symbols will be the central lines of the original pattern. However, this algorithm deletes end pixels of symbols. In addition the method is computationally expensive since it uses 20 rules.

From the above literature, it is realized that, there is a scope for developing a fast and efficient thinning algorithm. This is the motivation to propose new modified thinning 4 steps based algorithm for character recognition.

In this paper, we present a new improved version of 2-step method [19] based thinning algorithm for character recognition. The algorithm preserves topology of the rotated symbols, does not produce any extraneous pixels, and does not remove end pixels of characters. It has language independent capability. It works for all types of characters. It takes less time compared to [21].

The rest of the paper is organized as follows. In section 2, we describe new thinning methodology. Experimentation and comparative study is given in section 3. Finally, conclusion is given in section 4.
II. PROPOSED METHODOLOGY

Thinning is the process of reducing thickness of each line of pattern to just a single pixel. Thinning is usually used as the first step in applications such as optical character recognition to improve the recognition rate. Here a new improved 4-step thinning algorithm is proposed. Proposed algorithm is iterative, at each iteration proposed algorithm deletes every point that lies on the outer boundaries of the symbol, as long as the width of the symbol is more than one pixel wide. Region points are assumed as 1 and background points to have value 0. The proposed method is improved version of 2-step method [19]. Fig 3 shows the 8-neighborhood of the pixel. In 2-step method [19] 8-neighborhood of a pixel is used to thin a character image. The proposed method uses 4-step to thin a given image. First, the number of nonzero neighbors of p0 is calculated as \( N(p_0) = p_1 + p_2 + p_3 + p_4 + p_5 + p_6 + p_7 + p_8 \). Transition say, \( T(p_0) \) is calculated as the number of 0-1 transition in the order sequence \( p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8 \) respectively. Finding number of nonzero neighbors and 0-1 transition using 8-neighborhood, the proposed method uses 4 steps to complete iteration. In every iteration the proposed method deletes on all 4-direction i.e., \( p_2, p_4, p_6, p_8 \). The iterations are repeated until no further changes occur. The 4-step of the proposed method are shown in the following algorithm.

Algorithm for thinning

**Input:** Any Symbols  
**Output:** Thinned Image

**Method:**

Step 1:  
If \( 2 \leq N(p_0) \leq 8 \)  
If \( T(p_0) = 1 \)  
If \( p_2 = 0 \)  
If \( p_3 = 0 \)  
If the above condition is satisfied, make \( p(0) = 0 \).

Step 2:  
If \( 2 \leq N(p_0) \leq 8 \)  
If \( T(p_0) = 1 \)  
If \( p_4 = 0 \)  
If \( p_5 = 0 \)  
If the above condition is satisfied, make \( p(0) = 0 \).

Step 3:  
If \( 2 \leq N(p_0) \leq 8 \)  
If \( T(p_0) = 1 \)  
If \( p_6 = 0 \)  
If \( p_7 = 0 \)  
If the above condition is satisfied, make \( p(0) = 0 \).

Step 4:  
If \( 2 \leq N(p_0) \leq 8 \)  
If \( T(p_0) = 1 \)  
If \( p_8 = 0 \)  
If \( p_1 = 0 \)  
If the above condition is satisfied, make \( p(0) = 0 \).

Step 5: Repeat the above procedure for all the region points.

Step 6: Stop.

III. EXPERIMENTAL RESULTS AND COMPARATIVE STUDY

In this section we show some experimental results that carried out by the proposed method. Thinning results for different images are shown in Fig 2(a), Fig 2(b), Fig 2(c) and Fig 2(d) respectively. Here the thinning results are compared with well known existing methods [19] [21]. From the following Figures we can clearly see that the proposed method preserves topology of the symbols. And it is also clear from the figures that, it preserves topology even after rotation of the symbols and it is independent of languages. To compare the proposed one with existing system, we considered parameters such as Time taken to execute the system and Computations involved during the process. Results obtained from the Table 1, Table 2, Table3, and Table 4 is from the image carried out from the Figure shown in 2(a), 2(b), 2(c), 2(d) respectively. From the Table 1, Table 2, Table 3, and Table 4 it is clear that the proposed method takes less time and computations compare to method [21]. Proposed method takes quite more time and computations compared to [19]. This is because the method described in [19] has 2-steps where as in proposed method 4-steps are needed to thin a given image. Method described in [21] takes more time and computations because of more rules have been adopted to thin a given character image. From this we can say that proposed system preserves topology of the symbols and letters, irrespective of scripts they are written. One of the most advantages being compared to 2-steps method [19] is that the proposed method preserves topology of the symbols. And even after it is rotation invariant to any type of symbols. Compared to method [21] proposed method takes fewer time and computations. Experimental results showed that the proposed thinning algorithm is better compared [19] [21] methods, in terms of time, computations, and preserving topology.
(a)

Original Image

2-steps method

Maher and Ward

Proposed Method

(b)

Original Image

2-steps method

Maher and Ward

Proposed Method
Fig. 2 showing the resultant thinned image for different document images.
### TABLE 1. showing results of time and computations obtained for figure 2(a)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Time</th>
<th>Computations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-steps</td>
<td>1.14</td>
<td>28610</td>
</tr>
<tr>
<td>Maher and ward</td>
<td>5.98</td>
<td>1430515</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>1.39</td>
<td>49621</td>
</tr>
</tbody>
</table>

### TABLE 2. showing results of time and computations obtained from figure 2(b)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Time</th>
<th>Computations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-steps</td>
<td>1.11</td>
<td>8979</td>
</tr>
<tr>
<td>Maher and ward</td>
<td>4.69</td>
<td>448986</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>1.28</td>
<td>14166</td>
</tr>
</tbody>
</table>

### TABLE 3. showing results of time and computations obtained for figure 2(c)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Time</th>
<th>Computations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-steps</td>
<td>1.09</td>
<td>5920</td>
</tr>
<tr>
<td>Maher and ward</td>
<td>4.52</td>
<td>502016</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>1.20</td>
<td>8123</td>
</tr>
</tbody>
</table>

### TABLE 4. showing results of time and computations obtained from figure 2(d)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Time</th>
<th>Computations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-steps</td>
<td>1.09</td>
<td>8519</td>
</tr>
<tr>
<td>Maher and ward</td>
<td>4.77</td>
<td>425990</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>1.30</td>
<td>15001</td>
</tr>
</tbody>
</table>
IV. CONCLUSION

Good thinning algorithms should preserve the topology (shape) of symbols. Preserving shape is accomplished by representing symbols by their central lines. The proposed system thins the symbols to their central lines. Our method has the advantage of being rotation invariant: if the original image is rotated, the resultant thinned image is also rotated by the same angle. The proposed thinning has greater advantage of being compared to 2-steps method [19]. Compared to methods [19] [21] proposed method is better with respect to time, computations, and preserving topology. Experimental results are presented on symbols, characters, and letters written in different languages, and on rotated. The results show that the developed method is accurate, effective, fast, and can thin any symbols in any language, irrespective of the direction of rotation or flipping. However, the presence of noisy images remains the subject of further research.

REFERENCES