RESEARCH ON OPTIMIZATION OF IMAGE USING SKELETONIZATION TECHNIQUE WITH ADVANCED ALGORITHM

Sarita Jain
Department of CSE
Geeta Engineering College, Panipat, India

Sumit Rana
Department of CSE
Geeta Engineering College, Panipat, India

ABSTRACT
A number of image processing and pattern recognition application demand that are raw digitized binary pattern array be normalized, so that the constituents components of that array are of uniform thickness. The skeletonization process reduces such components to a thickness of one pixel or sometimes to a few pixels. The skeletonization of binary images has been studied extensively since the early sixties. During these years, many skeletonization algorithms for data compression by skeletonization have been devised and applied to a great variety of patterns for different purpose. For example, in the biomedical field, this technique was found to be useful in the early 1960’s, when a “shrink” algorithm was applied to count and size the constituent parts of white blood cells in order to identify abnormal cells. Since that time, applications in this area have included analysis of white blood cells and chromosomes, automatic X-ray image analysis and pattern recognition. This wide range of application shows the usefulness of reducing the patterns to thin line representations, which can be attributed to the need to process a reduced amount of data as well as to the fact that shape analysis can be more easily made on line like patterns. In this paper we are making advanced parallel skeletonization algorithm for reducing width of pixels.

Keywords: Database, C language.

I. INTRODUCTION
Skeletonization is a morphological operation that is used to remove selected foreground pixels from binary images[4]. It can be used for several applications, but normally only applied to binary images, and produces another binary image as output. The skeletonization operation is related to the hit-and-miss transform, and so it is helpful to have an understanding of that operator before reading on. The term ‘skeleton’ has been used in general to denote a representation of a pattern by a collection of thin arcs and curves. Other nomenclatures have been used in different context. For example the term ‘medial axis’ is being used to denote the locus of centers of maximal blocks.

Some authors also refer to a ‘thinned image’ as a line drawing representation of pattern [5]. In recent years, it appears that the term ‘skeleton’ is used to refer to the result, regardless the shape of the original pattern or the method employed [6]. Thus, skeletonization is defined as process of reducing the width of pattern to just a single pixel. This concept is shown in Fig 1.0

II. WORKING OF SKELETONIZATION
Like other morphological operators, the behavior of the skeletonization operation is determined by a structuring element. The binary structuring elements used for skeletonization are of the extended type described under the hit-and-miss transform (i.e. they can contain both ones and zeros). The skeletonization operation is related to the hit-and-miss transform and can be expressed quite simply in terms of it. The skeletonization of an image I by a structuring element J is:

\[ \text{thin}(I, J) = I - \text{hit-and-miss}(I, J) \]

In everyday terms, the skeletonization operation is calculated by translating the origin of the structuring element to each possible pixel position in the image, and at each such position comparing it with the underlying image structuring element exactly match foreground and background pixels in the image, then the image pixel underneath the origin of the structuring element is set to background (zero). Otherwise it is left unchanged. Note that the structuring element must always have a one or a blank at its origin. The choice of structuring element determines under what situations a foreground pixel will be set to background, and hence it determines the application for the skeletonization operation. For example, consider the structuring elements as shown in Fig.
At each iteration, the image is first skeletonized by the left hand structuring element, and then by the right hand one, and then with the remaining six $90^\circ$ rotations of the two elements. The process is repeated in cyclic fashion until none of the skeleton produces any further change. As usual, the origin of the structuring element is at the center. Fig. shows the result of this skeletonization operation on a simple binary image.

We have described the effects of a single pass of a skeletonization operation over the image. In fact, the operator is normally applied repeatedly until it causes no further changes to the image (i.e. until convergence). Alternatively, in some applications, e.g. pruning, the operations may only be applied for a limited number of iterations.

A. Requirements

The skeletonization process has following requirements

1. Geometrical: The skeleton must be in the middle of the original object and must be invariant to translation, rotation, and scale change.
2. Topological: The skeleton must retain the topology of the original object.

III. PROBLEM FORMULATION

The reduction of image can eliminate some counter distortions while maintaining significant topological and geometric properties. In more practical terms, thin-line representations of elongated patterns would be more suitable for extraction of critical features such as end-points, junction-points, and connection among the components. The vectorization algorithms often used in pattern recognition tasks also require one-pixel-wide lines as input.

Therefore, in real world, we have to thin the various images like BMP images to minimize the data to be handled, therefore there is need to have a software which can thin the images. There is need to study the various aspects of skeletonization concepts. Keeping in mind that our goal is to come up with a set of strokes representing our original image, it would be nice to have a way to preserve the orientation of our shapes while removing information that doesn’t interest us. One possibility is to consider the skeleton of an object. We approximate the skeleton with a skeletonization filter which gradually removes pixels from the borders of objects until something that looks a lot like the skeleton remains [7]. This skeletonization algorithm has two desirable features; the first is that it will not remove any pixel that would cause one object to become two disconnected objects. It is also sensitive enough to properly thin segments which are two pixels thick.

Any good skeletonization algorithm must favor one direction over another in order to avoid removing every pixel in search of the one-true-single-pixel skeleton. The problem of connectivity has been solved in this algorithm which was present in the previous algorithms. Optical character recognition (OCR) is the process of converting scanned images of machine printed or handwritten text into a computer processable format. Handwritten symbol recognition has received and continues to receive much attention by many researchers.

IV. PAST WORK

T. Y. ZHANG et. al [1] in their research paper “A fast parallel algorithm for thinning digital patterns” presented a parallel algorithm for skeletonizing different types of digital patterns. The algorithm is divided into two sub iterations that remove the boundary and corner points of the digital patterns. The first sub iteration aims at deleting the south-east boundary points and the north – west corner points, the second sub iteration is aimed at deleting the north-west boundary points and the south-east corner points. After several iterations only a skeleton of the pattern remains.

Ben K. Jang et al.[2] in their paper “One pass parallel thinning : analysis, properties and quantitative evaluation” defined skeletonization as a procedure to
transform a digital pattern, say, a connected component, to a connected skeleton of unit width. The one-pass parallel algorithm proposed requires only a single pass per iteration and uses a set of 5x5 templates. However, because of the limitation of the square-grid representation, resulting skeleton do not always closely approximate their corresponding medial axes. A number of other important issues, such as medial axis presentation, noise sensitivity, and over shrinking have been addressed. A set of measures to evaluate the proposed skeletonization algorithm are also discussed. The skeletonization algorithm is further extended to the derived grid for an isotropic medial axis representation.

M.V.Nagendraprasad et al. [3] in their paper “An improved algorithm for thinning binary digital patterns” presented a parallel algorithm which yield good results with respect to speed and connectivity. However during implementation, the algorithm involves a number of time consuming steps.

V. PROPOSED WORK

We propose the following plan for carrying out the present work:

1. To create a database of regional language numerals from different users.
   We create a database of hand written regional language numerals by considering the data created by different persons so that results can be observed for large image database.

2. To visualize the outputs of the two parallel skeletonization algorithms.
   We implement the two parallel skeletonization algorithms for BMP images using C language. The outputs of these two algorithms are visualized to examine the connectivity of pixels and generation of spurious branches.

3. To give an alternative parallel skeletonization algorithm
   We discuss an alternative parallel skeletonization algorithm and then implement the same using C language.

4. To compare the performance features of alternative parallel skeletonization algorithm.
   We visualize the results of the alternative skeletonization algorithm and compare with the results of the previous two algorithms.

**Fast Parallel Skeletonization Algorithm.**

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 transformation are applied to matrix IT point by point according to values of a small set of neighboring points as shown in Figure below.

<table>
<thead>
<tr>
<th>$P_9$</th>
<th>$P_3$</th>
<th>$P_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(i-1,j-1)</td>
<td>(i-1,j)</td>
<td>(i-1,j+1)</td>
</tr>
<tr>
<td>$P_8$</td>
<td>$P_7$</td>
<td>$P_5$</td>
</tr>
<tr>
<td>(i,j-1)</td>
<td>(i+1,j-1)</td>
<td>(i+1,j+1)</td>
</tr>
<tr>
<td>$P_6$</td>
<td>$P_4$</td>
<td>$P_2$</td>
</tr>
<tr>
<td>(i,j)</td>
<td>(i,j+1)</td>
<td>(i,j)</td>
</tr>
</tbody>
</table>

**Fig: Designation of 9 pixels In 3x3 windows:**

In Parallel picture processing the new value given to a point at nth iteration depends on its own value as well as those of its 8 neighbors at the (n-1)th iteration, so that all picture points can be processed simultaneously. It is assumed that a 3x3 window is used, and that each element is connected with its 8-neighbouring elements. This algorithm requires only simple calculations.

**Algorithm**

The 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 skeleton, each iteration is divided into two sub-iterations.

**First sub iteration**

In the first sub-iteration, the contour points $P_1$ is deleted from the digital pattern. If it satisfies following conditions:

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

b. $A(P_1) = 1$

c. $P_2 * P_4 * P_6 = 0$

d. $P_2 * P_5 * P_6 = 0$

Where $A(p_i)$ is the number of 01 patterns in the ordered of $P_2,P_3,P_4,\ldots,P_5,P_9$ that are eight neighbors of $p_i$ (fig above), and $B(p_i)$ is the non-zero neighbors of $P_1$, that is $B(P_1) = P_2+P_3+\ldots+P_9$

If any condition is not satisfied then $P_1$ is not deleted from the picture.

**Second sub iteration**

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

(c’) $P_2 * P_4 * P_6 = 0$

(d) $P_2 * P_5 * P_6 = 0$

and the rest remain the same.

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

By condition (a), the end-points of a skeleton line are preserved. Also, condition (b), prevents the deletion of those points that lie between the end-points of skeleton line. The iterations continue until no more points can be removed.
Preprocessing Skeletonization Algorithm

This algorithm is required to reduce the hand-written character in to the unitary thin form, each element in the picture. Each element is assigned the value ‘1’ if it is covered by part of the character, and the value ‘0’ otherwise. Depending whether a point is thin or not depend on its 8-neighbors. Thus, a window of 3*3 pixels used. Firstly, the thin of the character is extracted. The output thin is not unitary as it involves distortion at some points. This distortion will cause difficulties in the detection of edge and tree points, to overcome this drawback a procedure is developed which is capable of producing noise free unitary thin. This procedure involves two iterations as follows:

ALGORITHM

The algorithm is divided into two sub iterations:

Iteration 1

The skeleton is scanned horizontally by the 3*4 pixels window shown in Figure below.

<table>
<thead>
<tr>
<th>P9</th>
<th>P2</th>
<th>P3</th>
<th>P10</th>
</tr>
</thead>
<tbody>
<tr>
<td>P8</td>
<td>P1</td>
<td>P4</td>
<td>P11</td>
</tr>
<tr>
<td>P7</td>
<td>P6</td>
<td>P5</td>
<td>P12</td>
</tr>
</tbody>
</table>

Fig. : A 3*4 pixels window

Any two points which are horizontally adjacent to each other and horizontally isolated from other points, are detected. With P1 and P4 representing these two points, apply the following tests to locate the redundant point.

P1 is deleted if one of the following conditions are true:
1. SP1 and P6 = 1;
2. SP2 and P2 = 1;
3. [(P2 and P3) or (P3 and P2 and P6)] and [(P5 and P8) or (P3 and P6 and P7)]

Where SP1 = P5 or P6 or P9, SP2 = P6 or P3 or P5 and ( ) ‘and’, ‘or’ are complement, logical ‘AND’ and logical ‘OR’ respectively.

If P1 is not redundant them P2 must be deleted if the following condition is not true:
(P1 and P10) or (P3 and P12)

Iteration 2

In this iteration the thin is scanned vertically by the 4*3 pixel window shown in Figure below.

<table>
<thead>
<tr>
<th>P9</th>
<th>P2</th>
<th>P3</th>
</tr>
</thead>
<tbody>
<tr>
<td>P8</td>
<td>P1</td>
<td>P4</td>
</tr>
<tr>
<td>P7</td>
<td>P6</td>
<td>P5</td>
</tr>
<tr>
<td>P12</td>
<td>P11</td>
<td>P10</td>
</tr>
</tbody>
</table>

Fig: A 4*3 pixel window

Any two points which are vertically adjacent to each other and vertically isolated from other points are detected. With P1 and P6 representing these points, apply the following tests to locate the redundant point.

P1 is deleted if one of the following conditions are true:
1. SP1 and P6 = 1;
2. SP2 and P2 = 1;
3. [(P8 and P7) or (P7 and P8 and P9)] and [(P4 and P5) or (P5 and P4 and P3)]

Where SP1 = P5 or P8 or P7, SP2 = P1 or P4 or P3, and ( ) ‘and’, ‘or’ are complement, logical ‘AND’ and logical ‘OR’ respectively.

If P1 is not redundant then P6 must be deleted if the following condition is not true:
(P7 and P12) or (P5 and P10)

Proposed robust Skeletonization Algorithm

Describing the shape of objects is often necessary in image processing. While edges are used to represent the boundary of an object, the general shape of the object is represented by “stick figures”. The former technique for shape representation of objects is boundary-based. and the latter one is region-based. Both the techniques have attracted researchers in the field of shape analysis and recognition of digital images. Here we propose a shape – based technique for the description of a digital binary object- that is, the process of obtaining the “stick-figures”, the so-called skeletonization process from a binary object. The algorithm belongs to class of multi-pass iterative boundary removal skeletonization algorithms. Iterative boundary removal algorithms delete pixels on the boundary of a pattern repeatedly until only unit pixel-width thinned image remains. When a contour pixel is examined, it is usually deleted or retained according to the configuration of N(p) shown in figure. To prevent sequentially eliminating an entire branch in one iteration, a sequentially algorithm usually marks (or flags) all the pixels to be deleted, and all the marked pixels area then removed at the end of an iteration. This generally ensures that only one layer of pixels would be removed in each cycle. Let As (p) be the 8-
adjacency set of a pixel p and the elements X_i, of this adjacency are numbered in a way as shown in Figure

<table>
<thead>
<tr>
<th>X_4</th>
<th>X_3</th>
<th>X_2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X_5</td>
<td>P</td>
<td>X_1</td>
</tr>
<tr>
<td>X_6</td>
<td>X_7</td>
<td>X_8</td>
</tr>
</tbody>
</table>

Fig: Adjacency set of a pixel

The 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 skeleton, each iteration is divided into two sub-iterations. The pattern is scanned left to right and from top to bottom, and pixels are marked for deletion under four additional conditions:

**Proposed Algorithm**

A pixel is 8-(4-) deletable if its removal does not change the 8-(4-) connectivity of p. The pixels considered for deletion are contour pixels i.e. the pixels that have at least one white neighbor. The 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 Skeleton, each iteration is divided into two sub-iterations.

The pattern is scanned from left to right and from top to bottom, and pixels are marked for deletion under four additional conditions:

H_1: At least one neighbor of p must be unmarked.
H_2: X_h (p) =1 at the beginning of the iteration.
H_3: If X_3 is marked, setting X_3=0 does not change X_h (p).
H_4: If X_5 is marked, setting X_5=0 does not change X_h (p).

The above mentioned conditions are elaborated as given below:

H_1: The unmarked neighbors of a pixel P_i denoted by B (P_i) is the non-zero neighbors of P_i, that is

B (P_i) = P_2+P_3+…+P_9

The condition that we implement is that B (P_i) ≥1. Condition H1 was designed to prevent excessive erosion of small “circular” subsets.

H_2: In this step we calculate the crossing number, connectivity number or Hilditch number X_h (p) as described below.

It is defined as the number of times one crosses over from one white pixel to a black when points in the N(p) are traversed in order, cutting the corners between 8-adjacent black 4-neighbours where X_9 =X_1 therefore

\[ X_h (p) = \sum_{i=1}^{4} C_i \]

Where:

\[ C_i = \begin{cases} 1; & \text{if } X_{2i-1}=0 \text{ and } (X_{2i}=1 \text{ or } X_{2i+1}=1) \\ 0; & \text{otherwise} \end{cases} \]

The crossing number is equivalent to the number of 8-components of 1s in As(p) if at least one 4-neighbor is a 0.

X_h (p) is equal to the number of black components in N(p) except when p has all 4-neighbors, in which case X_h (p)=0. Obviously, crossing number for a pixel having all black 8-neighbors would be 0 as would an isolated pixel. If X_h (p) =1, deletion of p would not change the 8-connectedness of the pattern.

The deletion of p would not affect 4-connectivity if X_h (p) =2 since the black pixels in N (p) are 4-connected in these cases.

The condition H_2 is used to maintain connectivity. The conditions H_3 and H_4 reflect that if X_1 or X_3 are marked then setting them to background does not change the connectivity number. These conditions are used to preserve two-pixel wide lines. As a result of skeletonization process, a two-pixel-width or even pixel width in horizontal or vertical direction may be deleted, which would cause the loss of connectivity of object pattern. For example, a two-pixel-width rectangular pattern will disappear completely. In order to keep up the connectivity, these pixels should not be deleted. On the other hand, if all two–pixels-width are retained, the skeleton would not be one pixel width.

**B. Research Objective**

Following are the primary objectives of the research.

- To reduce the amount of data required to be processed.
- To reduce the time required to be processed.
- Extraction of critical features such as endpoints, junction-points, and connection among the components.
- The vectorization algorithms often used in pattern recognition tasks also require one-pixel-wide lines as input.
- Shape analysis can be more easily made on line like patterns.

**C. Objectives**

We aim to achieve the following objectives:

- To give an alternative parallel skeletonization algorithm.
- To visualize and compare the performance of given alternative algorithm in terms of connectivity for regional language numerals.
VI. CONCLUSION
The different parallel skeletonization algorithms give different results in terms of maintaining the connectivity and generating the spurious branches. When we implement the discussed alternative parallel skeletonization algorithm we observe that it provides better connectivity of pixels in the thinned image for almost all the test images. In the proposed algorithm we apply the single template in each pass and the output of each pass is passed onto the next pass, the connectivity and one-pixel width is guaranteed.

VII. FUTURE SCOPE
The discussion of various aspects such as convergence to unit width, connectivity and spurious branches can be taken together to compare these algorithms for regional language numerals for the purpose of skeletonization. The results obtained by applying the above discussed parallel skeletonization algorithms are shown in Figure. The differences in the outputs are very much clear from the visual inspection.

![Figure](a) Original image (b) Fast parallel algo (c) preprocessing algo (d) Robust algorithm.

REFERENCES