Design & Implementation of Thinning Algorithm for Object Recognition

Abhishek Singhal[1] and A.K.Sinha[2]

Abstract:

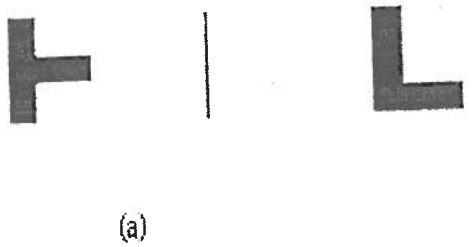
Thinning is an important preprocessing step for optical character recognition, finger print recognition, document preprocessing etc.. The main objective of this paper is to design, implement the thinning algorithm and to compare the results with those of the other existing algorithms. The proposed thinning algorithm removes the erosion hair and distortion problem, which occurred in other algorithms. It has been applied on the characters to thin them to figures which depict the characters but are just 1-pixel wide. During the initial phase, the motive is to make the user enter the character on its own by actually drawing the character and then apply thinning algorithm. Finally, after covering all the characters in English vocabulary, attempt has been made to improve the result more by adding images. In the beginning thinning was done directly by converting the gray scale image to binary and then thinning it. Afterwards, it was implemented for the colored images by first detecting the edges and then turning them into grayscale then converting it into a binary image and then finally thinning it. The user can see all the steps the image goes through to reach the final thinned image.

Key Words: Thinning Algorithm, Object Recognition, Artificial intelligence

1. INTRODUCTION:

Thinning is an important preprocessing step often used in pattern recognition techniques, which are an essential part of artificial intelligence. Thinning has been used in a wide variety of other applications as well including: medical imaging analysis, bubble-chamber image analysis (a device for viewing microscopic particles), text and handwriting recognition and analysis, metallography (materials analysis), printed circuit board design, robot vision, and for many image analysis operations such as optical character recognition, fingerprint recognition and document processing. Thinning involves removal of points or layers of an outline from a pattern until all lines or curves are all single pixel wide. The resulting set of lines or curves is called the skeleton of the object.

Thinning is a simple task for human beings. We humans can thin patterns with a wide a variety of shapes without any relative difficulty. It has been proposed that a human first catches a global view of the shapes, and then applies different methods to thin different shapes of different parts of the same patterns. As a result, the skeletons produced by humans are usually considered as reference skeletons, which are always superior to those obtained from any thinning algorithm. This suggests that such knowledge can be very helpful to the thinning process. As can be seen, the original image is reduced to a skeleton after application of a thinning algorithm but the output is deformed. The output as perceived by a human is also shown in the figure.



(a) (b)

Figure1: Original Shape, Desired Shape, and Deformed Shape

Most thinning algorithms are iterative in nature. In iteration, the edge pixels are examined against a set of criteria to decide whether the edge pixels should be removed or not. There exist several thinning algorithms both for sequential as well as parallel computers. A sequential algorithm uses results from the previous iteration and the result obtained so far in the current iteration to process the current pixel. Thus at any point in an iteration a number of pixels have already been processed. These results can be used immediately to process the next pixel. On the other hand, parallel algorithms use only the results from the previous iteration in the decision for removal of the next pixel. This makes parallel algorithms more suitable for implementation on parallel hardware such as array processor. Most algorithms use one of these two methods to thin various shapes. One algorithm may generate good skeletons for some shapes but can produce poor skeletons with other. It is very difficult to develop a generalized thinning algorithm, which can produce satisfactory results with all varieties of pattern shapes.

Characteristics of A Skeleton

We now introduce the general characteristics of the skeleton and some terminology that is frequently associated with skeletons. Consider a binary image described by a 2D array of pixels (Figure 2). The object which forms the foreground Q of this image is represented by a set of dark points while the background Q' corresponds to a set of white points.

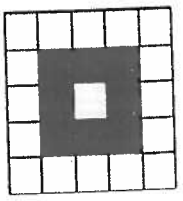


Figure 2: A Binary Image

For a given pixel p there are eight neighbors n_0, n_1, …, n_7, with the subscripts denoting the direction of the neighbors from p, with respect to the x-axis (Figure 3). Thus for n_i the direction is i * 45^0. The neighbors with even subscripts are known as D-neighbors, which are accessible from p by moving in a horizontal or vertical direction.

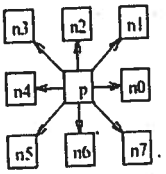


Figure 3 : The 8 Neighbors Of The Pixel P

The other neighbors are called I-neighbors, which are accessible from p by moving along any of the 45^0 lines. If p is a dark point and one of its eight neighbors n_i is also dark, p and n_i is said to be 8-connected. On the other hand, if p is dark and one of its four D-neighbors n_{2i} is also dark, p and n_{2i} is said to be 4-connected.

If p_0 and p_m are two dark points that belong to the same object, there exists a path, which can be described as a chain of dark points p_0, p_1, p_m, with each consecutive pair of pixels, p_i and p_{i+1} being neighbors of each other. If all eight neighbors are considered, p_0 and p_m are said to be 8-connected. If only D-neighbors are considered then p_0 and p_m are said to be 4-connected.

An object is 8-connected if all pairs of points in the object are 8-connected and 4-connected if all pairs of points in the objects are 4-connected. The essential characteristics of a skeleton are as follows:

- Connectivity should be preserved. If the object is connected the resulting skeleton should also be connected. In general 8-connectivity should be preserved for the foreground, while 4-connectivity should be preserved for the background.
- Excessive erosion should be prevented. The endpoints of a skeleton detected as soon as possible so that the length of a line or curve that represents a true feature of the object is not shortened excessively.
- The skeleton should be immune to noises. Noise, or small convexities, which do not belong to a skeleton, will very often result in a tail after thinning. The length of these tails should be minimized.

The preservation of 8-connectivity for the foreground is an important feature of skeleton. If 4-connectivity was preserved it would result in skeletons two pixel wide and that would go against the properties of an ideal skeleton (single pixel wide). But the preservation of 8-connectivity has one side effect. The skeletons generated at cross and corner points are distorted especially when the object is very thick.

Classification of thinning Algorithms

All thinning algorithms can be classified as one of two broad categories:

1. Iterative thinning algorithms or
2. Non-iterative thinning algorithms.

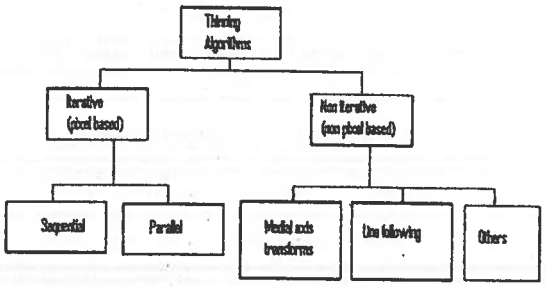


Figure 4: Classification of Common Thinning Algorithms

Figure 4 shows the major classifications of thinning algorithms. In general, iterative thinning algorithms perform pixel-by-pixel operations until a suitable skeleton is obtained. Non-iterative thinning methods use methods other than sequential pixel-scan of the image. We first examine iterative thinning methods.

Iterative thinning algorithms delete successive layers on the edges of a pattern until a skeleton remains. Usually the pixels of an image are considered consecutively and a choice is made to either delete or keep the pixel. The criterion for this choice is a set of "rules" based on the pixels in the neighborhood around the current pixel. Usually the neighborhood is a 3 x 3 area around the pixel. Iterative algorithms may be classified as either sequential or parallel.

Sequential thinning algorithms examine contour points of an object in a predetermined order. Either raster scan or contour following accomplishes this. Raster scanning is essentially scanning an image until a contour point is found and then applying a set of rules. Contour following algorithms pre-compute the border pixels of connected objects and can provide a speed increase since every pixel of the image does not have to be examined.

The second class of sequential thinning algorithms is parallel. In parallel thinning algorithms the decision for individual pixel deletion is based on the results of the previous

iteration. Like sequential algorithms, parallel thinning usually consider a 3 x 3 neighborhood around the current pixel. A set of rules for deletion is applied based on pixels in the neighborhood. Fully parallel algorithms have trouble-maintaining connectedness, so they are often broken into sub-iterations where only a subset of the pixels is considered for deletion.

Non-iterative thinning methods are not based on examining individual pixels. Some popular non-pixel based methods include medial axis transforms, distance transforms, and determination of centerlines by line following.

In line following methods, midpoints of black spaces in the image are determined and then joined to form a skeleton. This is fast to compute but tends to produce noisy skeletons. It has been conjectured that human beings naturally perform thinning in a manner similar to this. Another method of centerline determination is by following contours of objects. By simultaneously following contours on either side of the object a continual centerline can be computed. The skeleton of the image is formed from these connected centerlines.

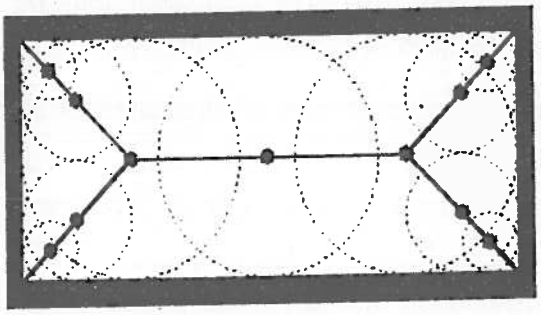


Figure 5 Pixel Intensity in Medial Axis Transforms

Medial axis transforms often use gray-level images where pixel intensity represents distance to the boundary of the object. The pixel intensities are calculated using distance transforms. In Figure 5 the maximum pixel intensity would increase toward the dark lines at the centers of the circles. Figure 6 shows a pixel level view of a distance-transformed block. Note that there are other methods of computing medial axis transforms.

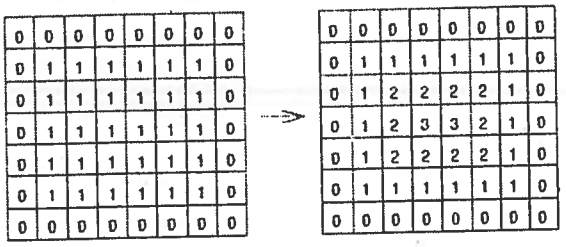


Figure 6: Distance Transformation

DEVELOPMENT OF ALGORITHMS

We discuss the complete step by step process of the execution of the system by giving all preprocessing steps and then proposed algorithm.

The system consists of three distinct modules, depicting the logical division of the problem statement with the point of view of development and implementation of these modules. The first module consists of detecting the edges in the input image. The output of this is a edge map of the input image which is then scan converted to a gray scale image. The second module is a simple step of converting the gray scaled image into a binary image. This is essential because all the thinning algorithms evaluated in the present work have a limitation that they can only work with only binary images. The final module implements the different thinning algorithms and compare with our proposed algorithm as discussed in chapter 2 and chapter 3 of this report and showing the comparision of the different output skeletons.

Pre-Processing Algorithms

Pre processing deals with preparation of relevant information. Generally, in any image processing application, preprocessing is required to eliminate noise, distortion etc. Depending on the features needed in the next

stage, the preprocessing operations can perform more specific operations on the input image. In the context of the present work, we cannot apply Thinning algorithms directly on the input image. The following steps are involved in preprocessing of the present work: -

1. Detection of Edges, .

2. Conversion to Gray Scale

3. Binarization of the Image

Detection of Edges

In this present work we have implemented the detection of edges of any input image file by using the Canny method. The Canny method uses the Sobel operators and employs two 3 x 3 filters Gx and Gy which act together to detect the horizontal and vertical edges present in the input image.

Conversion to Gray Scale

The term Gray Level refers to a scalar measure of intensity that ranges from black to gray, and finally black. All colors are seen as variable combinations of the three primary colors, red, blue and green. If a color image is converted into a gray scale image, each pixel of the image is assigned a particular value in terms of its RGB components. The formula for conversion is:

Gray = sqrt (((Red * Red)+ (Green * Green)+ (Blue * Blue))/3)

Binarization of the Image

A binary image is one in which there are only two colors i.e. either black or white. Since all thinning algorithms analyzed have a limitation that they work only with binary images we now need to convert our gray scaled edge map into a binary image. For this purpose we need to determine a threshold value of the gray scale. All gray scale values which lie below the threshold value are converted to black color (gray level 0) and others are converted to white color (gray level 1). For better performance we select the threshold value dynamically by scanning the complete gray scaled image pixel by pixel and taking an average of the gray levels. This average can be chosen as the threshold value or we can select the threshold value randomly.

PROPOSED ALGORITHM

The new thinning algorithm developed is based on serial approach. It raster scans the complete binary image and stores the pixel values in a two dimensional array.

It makes use of the following conventions

- $p_i = 1$ if pi is a dark pixel
- $p_i = 0$ if pi is a light pixel
- $N(p_i) = p_0 + p_1 + p_2 + p_3 + p_4 + p_5 + p_6 + p_7$
- S(pi) = Number of 0 to 1 and 1 to 0 transactions in the sequence p0, p1, p2, p3, p4, p5, p6, p7, p0 skipping the zero values if they occur at the odd surrounding pixels(i.e. p1, p3, p5, p7 or corners) divided by 2.

Here S(p) = 4 / 2 = 2

The array values are read for each pixel and the following conditions are applied. The pixel is marked to be deleted if either of the conditions C1 or C2 is satisfied

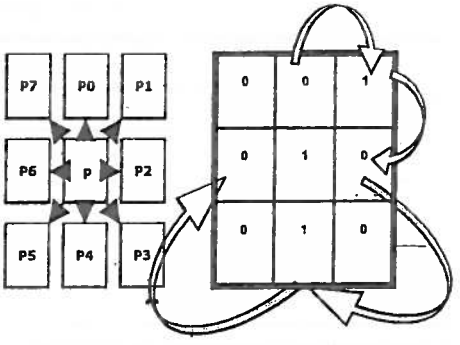


Figure 7

Condition C1

(i) $p_i = 1$

(ii) $3 <= N(p_i) <= 6$

(iii) $S(p_i) = 1$

Condition C2

(i) $p_i = 1$

(ii) $N(p_i) = 7$

(iii) $N(p_0)$ AND $N(p_1)$ AND $N(p_2)$ AND $N(p_3)$ AND $N(p_4)$ AND $N(p_5)$ AND $N(p_6)$ AND $N(p_7)$ is not equal to 7

The pixels to be deleted are marked in another two-dimensional array. Once the scan of the complete array is finished the marked pixels are deleted. The updated array is scanned again to mark the pixels to be deleted. These iterations are continued till no further pixels are marked to be deleted.

Experimental Results And Discussion

We discuss the results of the implementation of the algorithms developed in the previous section. We discuss various test cases and also document the performance of the thinning algorithms implemented in the last section. We first discuss the various metrics that are used for the evaluation of the performance of the thinning algorithms.

METRICS

In this section We summarize the metrics used for comparison of the algorithms. These metrics are applied independently to each of the algorithms and the results summarized in this section.

- **Conformity** : We define conformity as the degree to which the skeletons adhere to the original input image. For measuring the conformity We count the number of visible and identifiable characters in the input image and compare it with the similar count in the output image. Conformity is expressed as a percentage value and is the most important metric.

- **Execution Time** : The time required to process the input image in order to produce the skeletons is computed and used as a metric to determine the efficiency of the algorithms under study.

- **Erosion Hair** : Thinning algorithms produce a skeleton of the object. This skeleton usually is not an exact replica of the original object owing to the performance of the thinning algorithm in question. Usually small hair like structures are introduced especially at contour points of the object and are termed as erosion hairs in literature. These erosion hairs are undesirable and must be minimized. I choose erosion hair as a performance metric to ascertain the goodness of an algorithm.

EXPERIMENTAL SETUP

In order to measure the various metrics defined in the previous Section of this report the experimental setup is described in this section.

Test Cases for characters

FIGURE 7: SAMPLE IMAGE

Test Cases for Hindi Colored Characters:

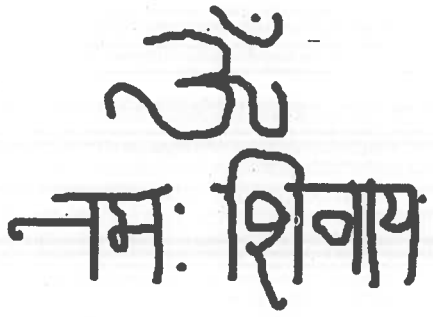


FIGURE 8: SAMPLE IMAGE

Test Cases for Images:

The sample image chosen for result verification and analysis were original degraded manuscripts and a number of experiments were performed with a large number of sample images of size 2" x 3" and fairly consistent results were obtained.

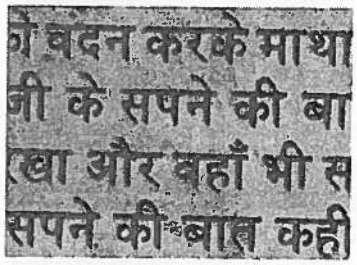


Figure 9: Sample Image

In order to present the results and thereafter the analysis we shall chose sample image which is a mythological document called *Devi Chalisa* and was degraded because of oil droppings and inappropriate storage.

We first show the performance different thinning algorithms on the sample image along with the results of proposed thinning algorithms.

EXPERIMENTAL RESULTS

The Following page shows the results of the thinning algorithms. First for characters and then results for images.

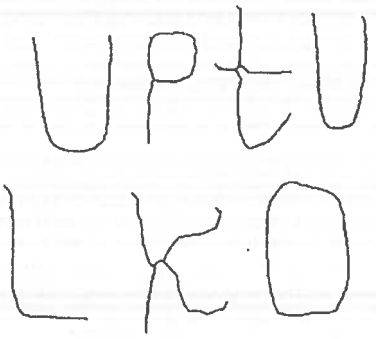


FIGURE 10: THINNED IMAGE FOR CHARACTERS

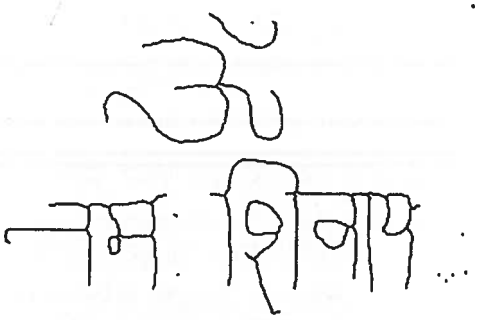


FIGURE 11: THINNED IMAGE FOR HINDI CHARACTERS

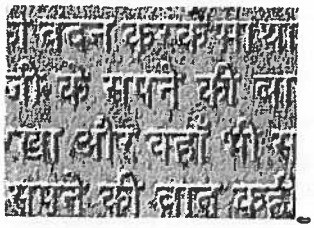


Figure 12: Output Image After Edge Detection

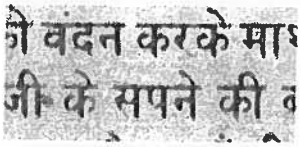


Figure 13: Output Image After Gray Scale Conversion

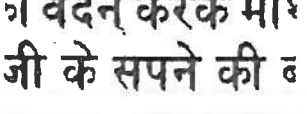


Figure 14: Output Image After Binarization

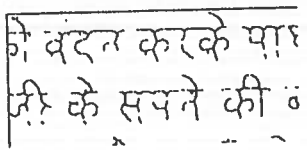


Figure 15: Output Image After Thinning (Zhang and Suen)

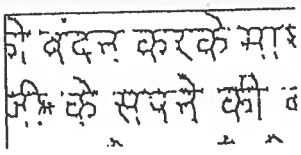


Figure 16: Output Image After Thinning (Liu-Wang)

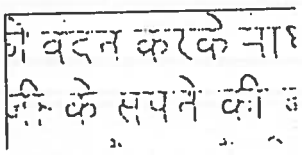


Figure 17: Output Image After Thinning (Richard W. Hall)

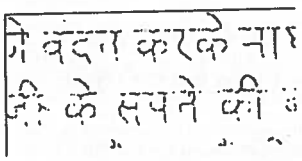


Figure 18: Output Image After Thinning (Holt)

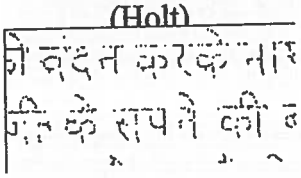


Figure 19: Output Image After Thinning (Paul-C-Kwok)

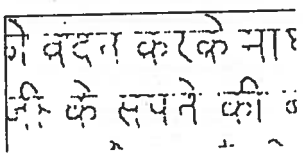


Figure 20: Output Image After Thinning (Proposed Algorithm)

Experimental Analysis & Discussion on the Results

As discussed, we shall use three metrics for evaluation of the performance of the algorithms under study, namely the Conformity, Execution Time and the Number of Erosion Hairs generated. While the first metric is to be maximized we need to minimize the other two.

Conformity

The following graph shows the degree of conformity of the thinned images to the original images after the application of the different algorithms.

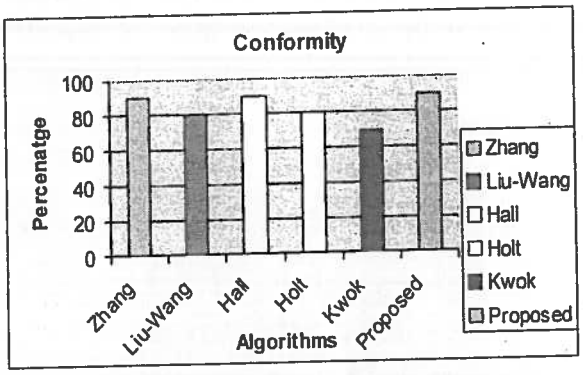


FIGURE 21: COMPARISON OF CONFORMITY

Execution Time

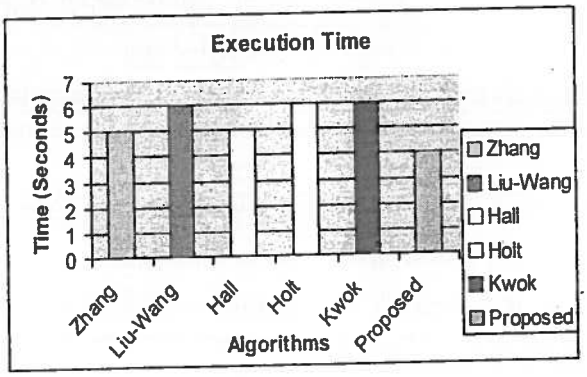


FIG22: COMPARISON OF EXECUTION TIME

Erosion Hair

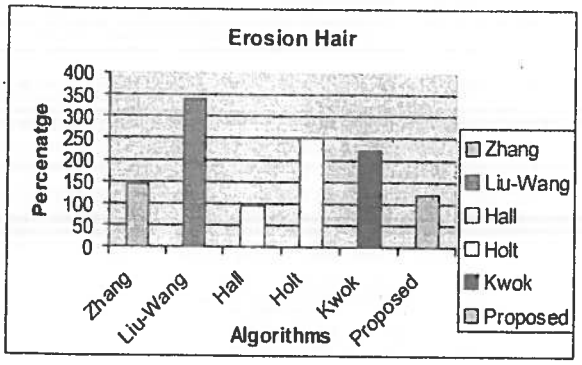


FIGURE 23: COMPARISON OF EROSION HAIR

As seen from the performance analysis and the inspection of sample image we find that the results are fairly consistent. The thinning algorithm with some types of images gives consistently good results. Also the execution time of the algorithm is fairly less and therefore does not increase the overheads in our work. However the performance of the proposed algorithm in terms of Execution Time and Conformity is quite good. although in terms of erosion hair algorithm proposed by Richard Hall is having least erosion hair for the given image, however the results for erosion hair of given image for proposed algorithm is quite good, So in all respects, Our proposed algorithm is quite good.

CONCLUSION

A new thinning algorithm has been designed and implemented. This algorithm is a new dimension in thinning and produces a good output compared to the other thinning algorithms available. The skeletons produced are superior to those produced by other contour generation methods. It produces seamless results for cross and corner regions in images and characters of varying thicknesses. The present work is limited for few applications of thinning but the present work can be extended in future for more applications.

REFERENCES

[1] Christian Neusius & Jan Olszewski : A non iterative thinning algorithm, Mar 1994, ACM Transactions Volume 20, No.1

[2] T. Y. Zhang & C. Y. Suen : A fast parallel algorithm for thinning Mar 1984, ACM Transactions Volume 27, No. 3

[3] Gonzalez, R.C. and Woods, R.E. [2002] Digital Image Processing, 2nd Edition, Prentice Hall, New Delhi

[4] E. S. Deutsh, University of Maryland: Thinning algorithm on Rectangular, hexagonal and triangular arrays, Sep 1972, ACM Transactions Volume 15, No. 9

[5] Christopher M. Holt, Allan Stewart, Maurice Clint and Ronald H. Perrott An improved parallel thinning algorithm, Feb 1987, ACM Transactions, Volume 30, No. 2

[6] Richard W. Hall : Fast Parallel Thinning Algorithm: Parallel Speed and Connectivity presentation, Jan 1989, ACM Transactions, Volume 32, No. 1

[7] PAUL C. K. KWOK: A Thinning Algorithm by Contour Generation, Nov 1988, ACM Transactions, Volume 31, No. 11

[1]ABES Engineering College, Ghaziabad, U.P. India

[2]Bharati Vidyapeeth's College of Engineering, New Delhi, India

abhi_skd@rediffmail.com, aksinha_1@yahoo.com