M.Tech(computer science) Dissertation Series

STUDY OF THINNING ALGORITHMS

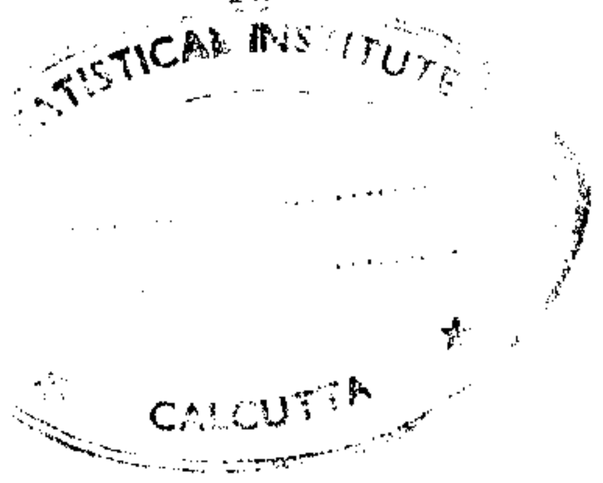
Dissertation submitted in partial fulfilment of the requirements for the award of the Degree of Master of Technology in computer science

by

Anupam Ray

Under supervision of

Dr. Srimanta Pal



INDIAN STATISTICAL INSTITUTE
203 B.T Road,Calcutta, 700035

CERTIFICATE

This is to certify that the work described in the dissertation entitled "STUDY OF THINNING ALGORITHMS", has been undertaken by Anupam Ray under my guidence and supervision. The dissertation is found worthy of acceptance for the award of the Degree of Master of Technology in Computer Science.

Srimanta Pal

Dated, 12/07/93.
Calcutta - 35.

(Dr. Srimanta Pal)
Indian Statistical Institute.
203, B. T. Road, Calcutta - 700 035.

CONTENTS

ABSTRACT

Thinning is one of the important low level segmentation procedure. There are number of thinning algorithms, but no one is good. Existing thinning algorithms (chapter 1) are not generalised in the sense that a particular method may be suitable to a particular class of images but not to all kind of images. There doesn't exist any generalised model for measuring time complexity of thinning algorithms. As a result their performance can be compared in order to select with minimum time complexity with better skeletone. Most of the thinning algorithms are iterative approximation method, when in most of the cases shape of the skeletone are not preserved by the approximation method. As a result inaccurate results are submitted to the next stage thus error propagation has occurred. One model of analysis of the most widely used template matching thinning algorithms, based on markov process method (chapter 3) for conducting average case analysis of thinning algorithms in order to measure their performance, have been proposed. Also we have designed a model for the generation of random binary image (chapter 2) which are either normally distributed or 4/8 connected uniformlly distributed. We have used these binary images to study the emperical average performance of some template matching thinning algorithms.

SOME TEMPLATE MATCHING THINNING ALGORITHMS

1.1. INTRODUCTION

Thinning is one of the most important operation being performed during low level segmentation. Thinning is perfomed on the edge images which are obtained by edge detection procedure. An edge image is consisting of a set of edge points lying on the boundaries of the object present in the original image. Usually the boundary of the object are elongated due to the side effect of the edge detection process. Such elongated boundary of the object are thinned to get skeletone of the object. In general thinning can be defined as the process of unusing points from elongated boundaries of the object until the boundaries are reduced to one-pixel wide boundaries called **skeletone** of the object. Thinning plays an important role in the preprocessing stage of Image Processing. It deals with extracting the distinctive features known as skeletons from the images.

1.2. ALGORITHMS:

1.2.1. Zhang and Suen's thinning algorithm [15]

It is a 2-pass parallel template matching thinning algorithm. They have considered window of size 3x3 with centre element as the candidate pixel. A typical window is given below,

P_8 $(i-1, j-1)$	P_1 $(i-1, j)$	P_2 $(i-1, j+1)$
P_7 $(i, j-1)$	P (i, j)	P_3 $(i, j+1)$
P_6 $(i+1, j-1)$	P_5 $(i+1, j)$	P_4 $(i+1, j+1)$

Fig(1.1) : A 3x3 window

algorithm : Input - a binary image matrix I.
output - thinned version of the input image.

```
PROCEDURE  PASS_1(P);
BEGIN
   IF (p=1) AND ( 2≤ B(p)≤ 6 ) AND ( A(p)=1 ) THEN
(*     B(p)= total # 1's in the neighbour of p.                    *)
(*     A(p)= # 01 sequences in p1,p2, .. ,p8                       *)
       BEGIN
          IF ( p1*p3*p5=0 )  AND ( p3*p5*p7=0 ) THEN
               delete (i.e., make 0) the candidate pixel p
          ELSE       remains unchanged;
       END;
END;

PROCEDURE  PASS_2(p);
BEGIN
   IF (p=1) AND ( 2≤ B(p)≤ 6 ) AND ( A(p)=1 ) THEN
   BEGIN
      IF ( p1*p5*p7=0 )  AND ( p1*p3*p7=0 ) THEN
           delete (i.e., make 0) the candidate pixel p
      ELSE     remains unchanged;
```

```
        END;
END;

BEGIN  (*  main body  *)
   REPEAT
       FOR each pixel p of I, call PASS_1(p);
       get a new modified image matrix I';
       FOR each pixel p of I', call PASS_2(p);
       get a new modified image matrix I";
     (* two passes make one iteration  *)
       I <— I" ;
   UNTIL ( # deletions = 0 );
END.
```

1.2.2. Holt et al thinning algorithm [4]

It's a modified version of zhang & suen's algorithm. They have used boolean representation of '0' and '1'. It is an 1-pass parallel template matching thinning algorithm with 4x4 window.A typical window is given below,

P_8	P_1	P_2	P_{12}
P_7	P	P_3	P_{13}
P_6	P_5	P_4	P_{14}
P_9	P_{10}	P_{11}	P_{15}

Fig(1.2) : A 4x4 window

algorithm : Input - a binary image matrix I.
output - thinned version of the input image.

```
FUNCTION edge(p) : boolean;
BEGIN
   IF (p=1) AND ( 2≤ B(p)≤ 6 ) AND ( A(p)=1 ) THEN
(*  B(p) and A(p) are as in ZHANG & SUEN algorithm      *)
```

```
(*        with '1' as 'true' and '0' as 'false'                    *)
       edge <— true
   ELSE   edge <— false;
END;

PROCEDURE   PASS(p : boolean);
BEGIN
  IF ( p AND (    NOT edge(p)
                  OR ( edge(p3) AND p1  AND p5 )
                  OR ( edge(p5) AND p3  AND p7 )
                  OR (  edge(p3)  AND edge(p4)  AND edge(p5)  )
  THEN      delete the candidate  pixel
(*  i.e.,  if either (1) p will be on the edge.                      *)
(*     or (2) n='1',s='1' and e neighbour is on  the edge.           *)
(*     or (3) e='1',w='1' and s neighbour is on the edge.            *)
(*     or (4) e,se,s neighbours will  be  on  edge.                  *)
(*      then  delete;                                                *)
(*     where, e = east, w = west, s = south, n = north,              *)
(*      se = south-east                                             *)
  ELSE        remains unchanged;
END;

BEGIN (*   main body   *)
   REPEAT
       FOR  each pixel p of I, call PASS(p);
       get a new modified  image  I';
       I <— I'
   UNTIL (# deletions = 0 );
END.
```

1.2.3. Chin et al thinning algorithm [16]

It is an one pass parallel template matching thinning algorithm. They have used windows of different sizes which are given below . [fig 1.3 : (1)-(8)] are used for detecting boundary pixels. [fig 1.3 :(9)-(10)] are used for disabling

the deletion of pixels if certain conditions are satisfied and the rest templates are for trimmining.

0	0	0
1	1	1
x	1	x

(1)

0	1	x
0	1	1
0	1	x

(2)

x	1	x
1	1	1
0	0	0

(3)

x	1	0
1	1	0
x	1	0

(4)

x	0	0
1	1	0
x	1	x

(5)

0	0	x
0	1	1
x	1	x

(6)

x	1	x
0	1	1
0	0	x

(7)

x	1	x
1	1	0
x	0	0

(8)

x	x	x	x
0	1	1	0
x	x	x	x

(9)

x	0	x
x	1	x
x	1	x
x	0	x

(10)

0	0	0
1	1	1
p	1	q

(11)

0	1	p
0	1	1
0	1	q

(12)

p	1	q
1	1	1
0	0	0

(13)

p	1	0
1	1	0
q	1	0

(14)

where, p * q = 1

fig 1.3 : templates used in chin's algorithm.

where (1)-(8) for thinning , (9)-(10) for restoring and (11)-(14) for trimming.

Algorithm

Step 1 : If an window matches with any of (9) to (10) templates then the candidate pixel can not be deleted and move to the next position and go to step 1 else goto step 2.

Step 2: If an image window matches with any of (1) to (8) templates then the candidate pixel will be deleted for the next iteration else move to the next position. If any pixel is deleted then go to step 1 else goto step 3

Step 3: If an image window matches with any of (11) to (14) templates then the candidate pixel will be deleted.

Step 4 : STOP.

1.2.4. Pal and Bhattacharyya's thinning algorithm [18]

In this parallel thinning algorithm the binary pattern consists of those pixels those are 1's. In this algorithm a 5x5 window is sliding from left to right and from top to bottom fashion over the image.

P_{23}	P_{24}	P_{9}	P_{10}	P_{11}
P_{22}	P_{8}	P_{1}	P_{2}	P_{12}
P_{21}	P_{7}	P	P_{3}	P_{13}
P_{20}	P_{6}	P_{5}	P_{4}	P_{14}
P_{19}	P_{18}	P_{17}	P_{16}	P_{15}

fig(1.4) : A 5x5 window

A vertical stroke of width 2 is guarded by peeping of its edges. So a point on west edge can be preserved only if it is not on a corner and its east neighbour is on an edge. The of horizontal and vertical straight lines can be preserved by if one of its four templates in fig 1.6 matches.

x	x	x	x	x
1	1	1	0	x
1	1	1	1	x
x	1	1	1	x
1	x	1	1	x

(a)

1	x	1	1	x
x	1	1	1	x
1	1	1	1	x
1	1	1	0	x
x	x	x	x	x

(b)

x	1	1	x	1
x	1	1	1	x
x	1	1	1	1
x	0	1	1	1
x	x	x	x	x

(c)

x	x	x	x	x
x	0	1	1	1
x	1	1	1	1
x	1	1	1	x
x	1	1	x	1

(d)

fig 1.5 : horizontal and vertical line preserving templates.

Algorithm

Step 1: All the basic steps of Holt et al thinning algorithm except loops.

Step 2: If the 5x5 window matches with any of the 4 [fig 1.5 (a) to (d)] templates then the candidate pixel will be deleted,otherwise goto step 1 after sliding the window by one position.

Step 3: If any pixel is deleted in the above steps then go to step 1 otherwise STOP.

1.2.5. Guo and Hall's thinning algorithm [17]

In this algorithm they had assumed a 3x3 window (fig 1.6) is sliding over the binary image. It is an two pass parallel template matching thinning algorithm.

P_8	P_1	P_2
P_7	P	P_3
P_6	P_5	P_4

fig(1.6) : A 3×3 window

Definition 1.

C(P) is defined as the number of distinct 8-connected components of 1's in P's eight neighbourhood. C(P) = 1 implies P is 8-simple when P is a boundary pixel.

Definition 2

N(P) is used to detect endpoint and which can be helpful to achive thinner results where

$$N_1(P) = (P \ V \ P_1) + (P_2 \ V \ P_3) + (P_4 \ V \ P_5) + (P_6 \ V \ P_7)$$
$$N_2(P) = (P_1 \ V \ P_2) + (P_3 \ V \ P_4) + (P_5 \ V \ P_6) + (P_7 \ V \ P) .$$

$N_1(P)$ and $N_2(P)$ each break the ordered set of P's neighboring pixels into four pair of adjoining pixels and count the number of pairs which contains one or two 1's.

Algorithm

PASS I

Step 1: An element is to be removed when it is a boundary point. This often identified by $2 \leq N(P) \leq 3$ and $C(P) = 1$ if a pixel is on boundary point and $(P_2 \ V \ P_3 \ V \ \bar{P}_5) \ V \ P_4 = 0$ is there then the candidate pixel is to be deleted for the next PASS 2.

Step 2 : Scan the window one position at a time or scan all the positions at a time and apply step 1.

Step 3: when all the pixels of the given image have been processed and at least one candidate pixel has become 0 then go to PASS 2 else exit.

PASS 2:

Step 1 : An element is to be removed when it is a boudary point. This often identified by $2 \leq N(P) \leq 3$ and $C(P) = 1$ if a pixel is on boundary point and $(P_6 \vee P_7 \vee \bar{P}_1) \vee P_8 = 0$ is there then the candidate pixel is to be deleted for the next PASS 1.

Step 2 : Scan the window one position at a time or scan all the positions at a time and apply step 1.

Step 3: when all the pixels of the given image have been processed and at least one candidate pixel has become 0 then go to PASS 1 else exit.

1.3. RESULT AND CONCLUSION

We have studied and implemented five parallel template matching thinning algorithms of which two algorithms,viz., Zhang & Suen's algorithm [15] and Guo et al's algorithms [17] are two pass algorithms others are one pass. In the one pass algorithms,Chin's algorithm [16] explicitely describe the templates used in algorithm, and others describe the property of the templates used in the thinning process. A set of output (Appendix A) of different algorithm for a particular set of input shows the quality of thinning algorithms. Hardware implementations are very simple. But no one is a generalized thinning algorithm. So new thinning algorithm development scope is always open in the field of Image Processing/Computer Vision.

2

RANDOM BINARY IMAGE GENERATION

2.1.INTRODUCTION

In computer vision, image is a matrix of integer values which is indicating gray value at a particular position of an image. Binarization method converts the gray level image into a binary image. There are a number of image processing operations [11] such as coding, contour following, skeletonization performed on a binary images (i.e., binary patterns or bilevel images/patterns). Classification of binary pattern recognition [10,13], biological or medical image processing [12], engineering drawing [9], map processing [6], and other machine manipulations of imaginary data often incorporate binary image processing at some stage in their application. In 1992, Haralick [2] encourages to the community of image analysis/machine vision to design a model for the determation of the performance of image analysis algorithms. This is an awful state of affairs for the engineers whose job it is to design and build image analysis or machine vision systems. Study of performance analysis of thinning algorithms have started from last few years. The empirical study of thinning algorithms have been studed by Chen and Hsu [1] and Heydorn and Weidner [3]. Jang and Chin [5] studed formally the behavior of thinning algorithms using mathematical morphology and also Pal and Bhattacharyya [7,8] studed the average behavior of template mathcing thinning algorithms using a probabilistic urn model. In the emperical study of the average performance of various image processing and classification algorithms on binary images, it is necessary to generate a statistically distributed binary image and also necessary to analyze their tolerance and sensitivity to noise [14]. In this paper we have presented a statistically distributed binary image generation method.

2.2.DEFINITIONS AND TERMINOLOGIES

Definition 1 : A binary image $B = [b_{ij}]$ be represented by

$$b_{ij} = \begin{cases} 1 \text{ , Object} \\ 0 \text{ , background} \end{cases}$$

where $i = 1,2,\ldots,n$; $j = 1,2,\ldots,m$ and n x m is the size of the binary image matrix B (Figure 1).

```
. . . . . .
. 1 1 1 . . 1 1 1 .
. 1 1 1 . . 1 1 1 .
. 1 1 1 1 1 1 1 1 .
. 1 1 1 1 1 1 1 1 .
. 1 1 1 . . 1 1 1 .
. 1 1 1 . . 1 1 1 .
. . . . . .
```

Figure 1 : A binary image.

The real world image contains some natural patterns. Mostly they are connected. The connectivity of an image is defined by the following definitions.

P_9	P_2	P_3
P_8	P_1	P_4
P_7	P_6	P_5

Figure 2 : The 4-neighbour and 8-neighbour of the point P_1.

P_1 = Central element, P_2 = North, P_4 = East, P_6 = South, P_8 = West, P_3 = North-East, P_5 = South-East, P_7 = South-West, P_9 = North-West.

Definition 2 : In a binary image four elements, namely, top (P_2), bottom (p_6), left (P_8) and right (P_4) are 4-neighbours of the elements P_1 (or 4-adjacent to P_1) (Figure 2). Two subsets U and V of S (say) are said to be 4-adjacent (Figure 3) if there exist at least one element of U which is 4-adjacent to atleast one element of V.

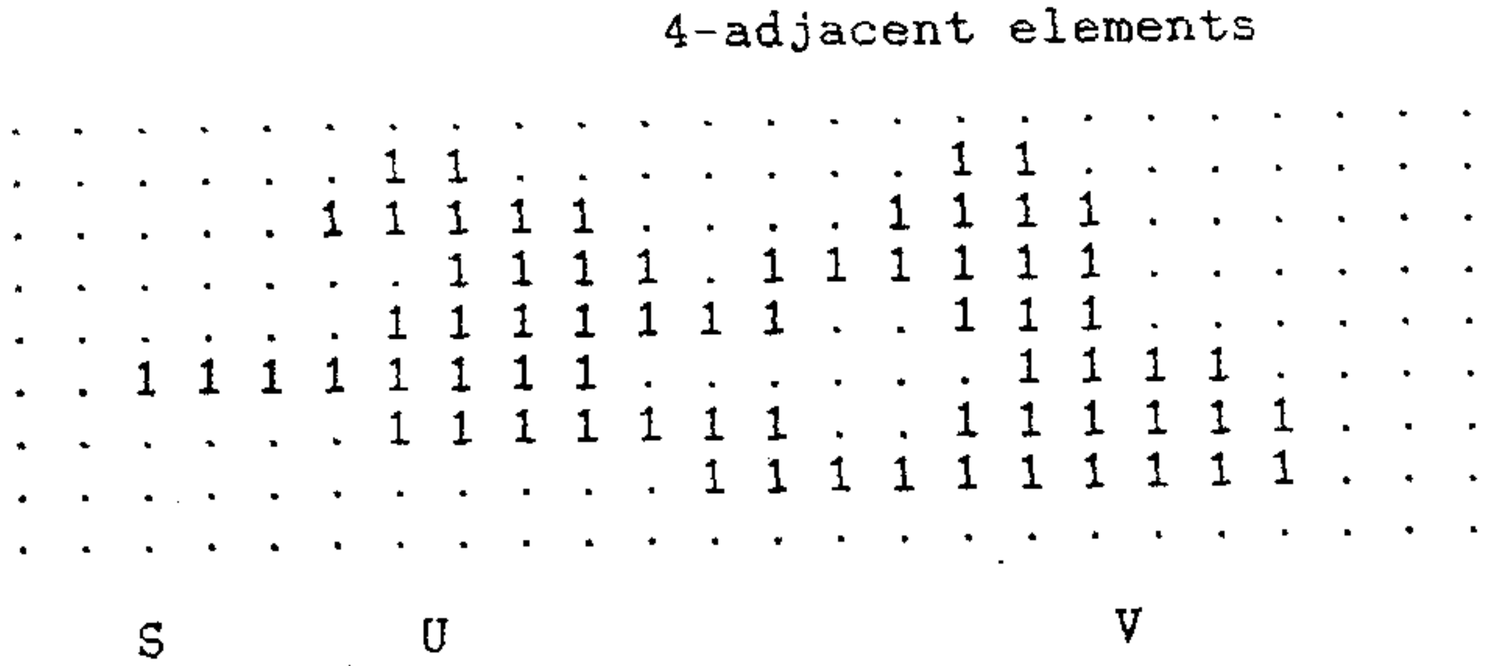


Figure 3 : U and V are 4-adjacent to each other

Definition 3 : In a binary image four diagonal elements of P_1 (Figure 2) are said to be 8-neighbours of P_1 (or 8-adjacent to P_1). Two subsets U and V of S (say) are said to be 8-adjacent if there exists at least one element of U which is 8-adjacent to at least one element of V (Figure 4).

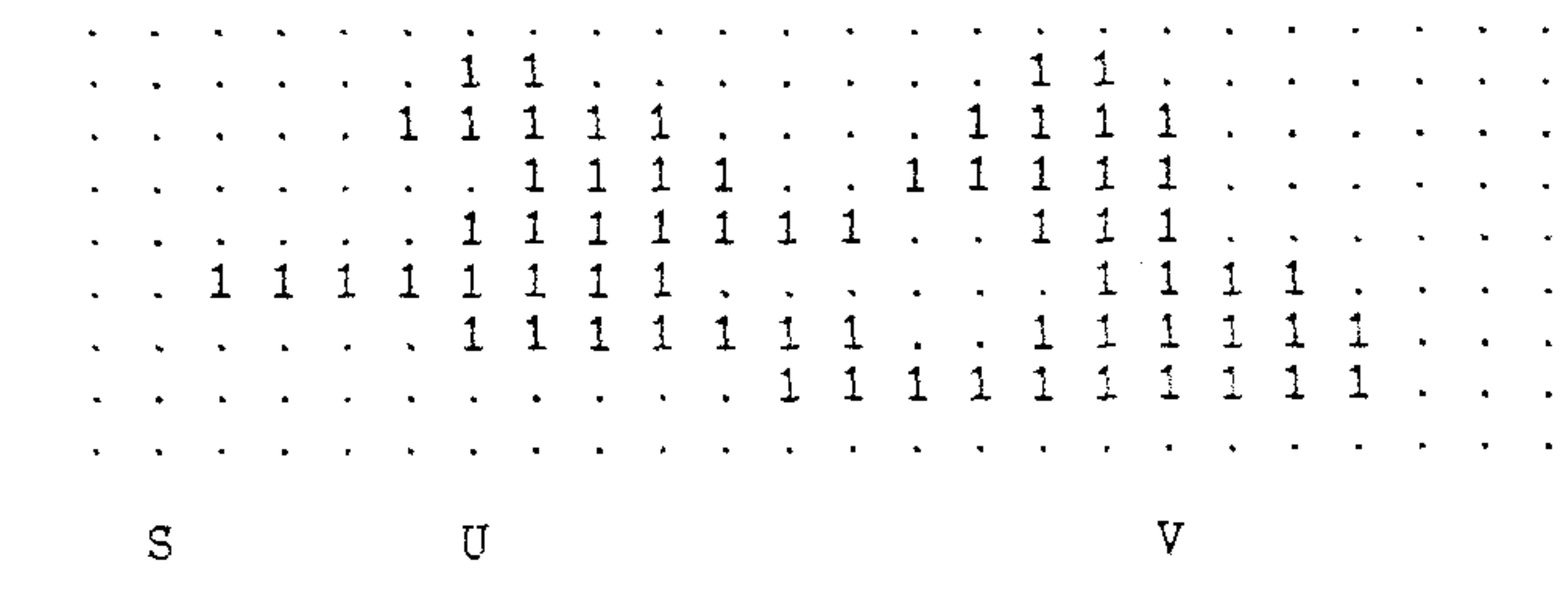
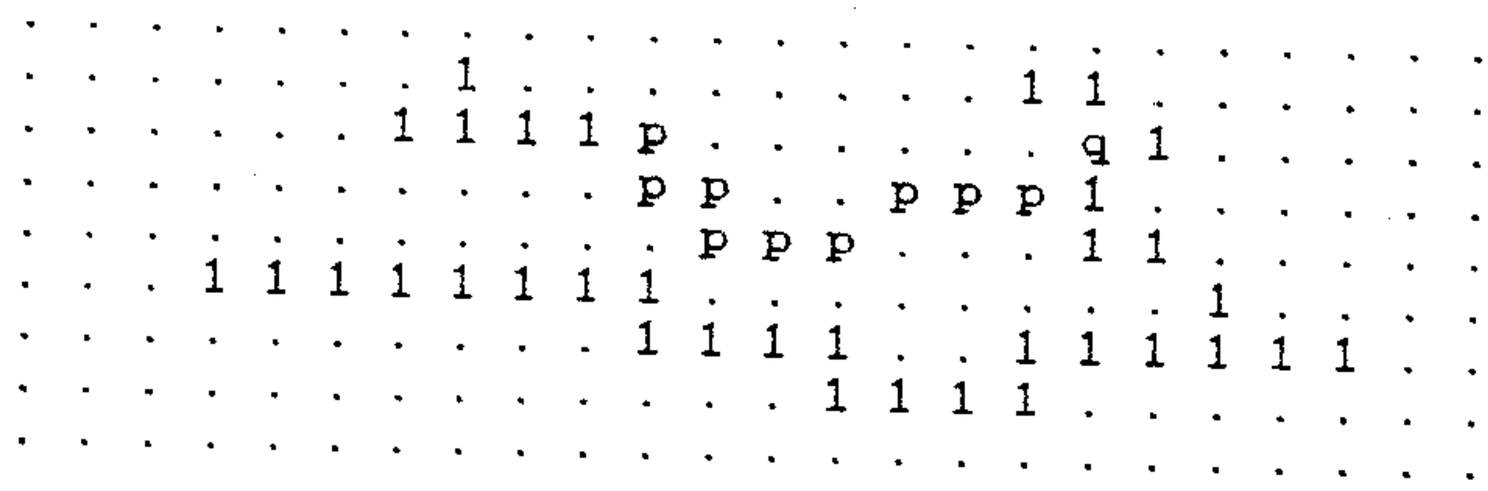


Figure 4 : U and V are 8-adjacent to each other

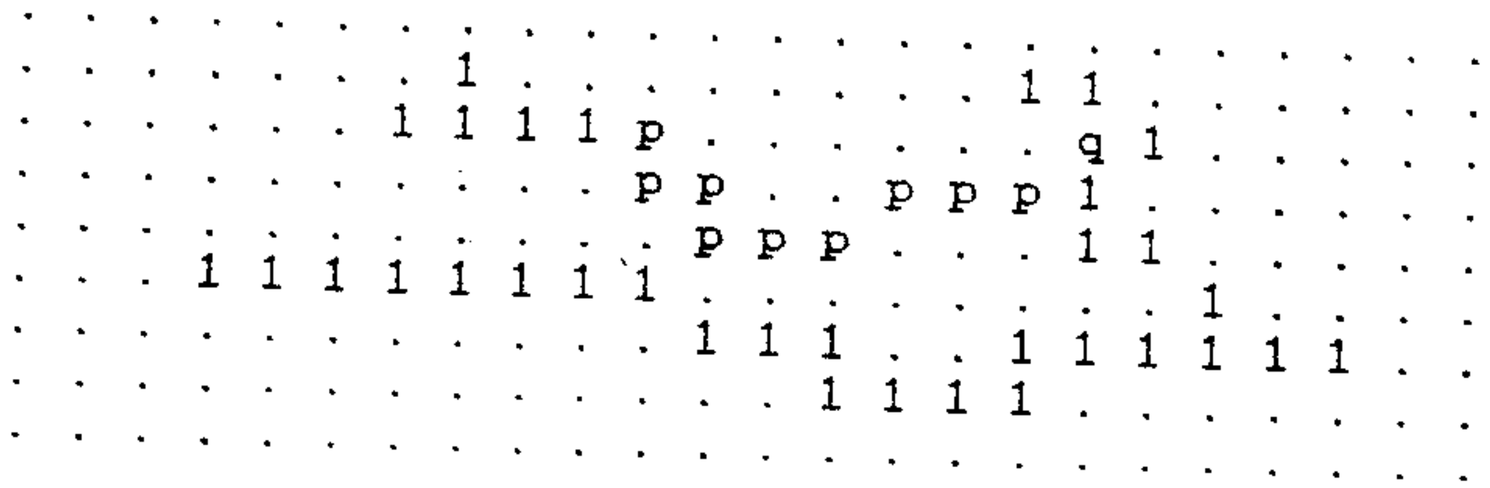
Definition 4 : An 8-path (4-path) $\prod_n(p,q)$ (Figure 5) of length n from p to q is a sequence of points (elements) $\langle p = p_0, p_1,\ldots,p_n = q\rangle$ such that p_i is an 8-neighbour (4-neighbour) of p_{i-1} and $p_i \in B$ where B is a binary image and $1 \leq i \leq n$.



B

Figure 5 : $\Pi_9(p,q)$ shown in the image B

Definition 5 : Two different points p and q of B, where B is a binary image, the point p is said to be 8-path connected (4-path connected) or simply 8-connected (4-connected) to q if there exists an 8-path (4-path), $\prod_n(p,q)$ in B (Figure 6).



B

Figure 6: 8-path connected image.

2.3. UNIFORMLY DISTRIBUTED BINARY IMAGE

Definition 6 : A binary image is said to be uniformly distributed if the binary image generated from an uniformly distributed binary 0/1 generator.

An uniformly distributed binary image generated in different ways.

(1) Each elements of a binary image is generate using an uniformly distributed binary 0/1 generator. This image does not guarantee the conectivity of the pattern.

(2) A 4-connected (8-conected) random binary pattern can be generated as follows. In this method a binary uniformly distributed random number (0 or 1) generator generates 0's and 1's and placing them on its 4-adjacents (8-adjacents) according

to its incoming sequence then move to the upper level and find next 1 and apply the above method recursively and when it does not find any 1 of its adjacent then come down to a step. This ultimate pattern will be bounded by 0's only.

To generate 4-connected binary image from uniform random number,we proceed as follows:

algorithm :- Input : Uniform random number.
Output : a 4-connected binary image.

```
PROCEDURE FILL_4( x,y );
BEGIN
   IF ( NORTH is valid, i.e., NORTH of the candidate pixel is
within the image          matrix and it is NOT yet filled by '0' or
'1' )      THEN  put a random no. to NORTH and set a flag, say,
NFLAG;
(*          random no. is generated from uniform(0,1). It's      *)
(*           a fraction between '0' and '1'. So to get '0'       *)
(*                        or '1', use modulo 2                   *)
   IF ( EAST is valid, i.e., EAST of the candidate pixel is
within the image          matrix and it is NOT yet filled by '0' or
'1' )      THEN  put a random no. to EAST and set a flag, say,
EFLAG;
   IF ( SOUTH is valid, i.e., SOUTH of the candidate pixel is
within the image          matrix and it is NOT yet filled by '0' or
'1' )      THEN  put a random no. to SOUTH and set a flag, say,
SFLAG;
   IF ( WEST is valid, i.e., WEST of the candidate pixel is
within the image          matrix and it is NOT yet filled by '0' or
'1' )      THEN  put a random no. to WEST and set a flag, say,
WFLAG;
   IF ( WFLAG ) then PUSH( x,y-1);
   IF ( SFLAG ) then  PUSH( x+1,y);
   IF ( EFLAG ) then  PUSH( x,y+1);
   IF ( NFLAG ) then  PUSH( x-1,y);
END;
```

```
PROCEDURE  4_CONNECTED_IMAGE( x,y );
BEGIN
    call    FILL_4( x,y );
    REPEAT
        POP an element from the stack which contains the
        co-ordinate of the pixel to be expanded. Let the
        co-ordinate be (p,q);
        call FILL_4(p,q);
    UNTIL ( stack is empty );
END.
```

Similarly, we can generate 8-connected random binary image using above algorithm, only we have to use FILL_8 procedure which is obtained from FILL_4 considering also the diagonal pixels together with east, south, west and north pixels.

2.4. NORMALLY DISTRIBUTED BINARY IMAGE

A normally distributed binary image can be generated by using a normally distributed random number generator. Since the image is binary image so we have to select the positions on the image plane such that it will produce an edge line binary image. We can define a normally distributed binary image as follows.

Definition 7 : A gray level image G is said to be normally distributed or a mixture of normally distributions if its histogram of gray values of the image is approximated by a smooth curve and the histogram looks like a normal distribution curve or a mixture of normal distribution curves (i.e., multi-peak) then the image is said to be normally distributed.

Definition 8 : A binary image contains (0,1) with natural patterns of lines and points. Natural pattern formation by using (0,1) is positional dependent. The position of 1's are selected by a normally distributed random number generator on a zero image matrix B. B is said to be a normally distributed

binary image.

Now we shall generate the positions of 1's in B (initially 0). Let us assume that B is a pixel matrix so that the distance between two pixels is fixed in a particular direction. So there is no deviation on distance, only deviation may exists on the value of orientation θ (say). So we shall generate a normally distributed angle θ with respect to a random starting position.

The parameters of a normally distributed random direction generator are standard deviation (σ_θ), mean (μ_θ) and the initial value of $\theta = \theta_0$ (say). Mathematically we can write

$$\theta = N(\theta_0, \mu_\theta, \sigma_\theta).$$

where θ is a real number, in degree or radian. So we shall make this θ in degree to its nearest integral value and take a modulus over 360^0. If θ is negative then it convert to a positive value by $360^0+\theta$. Then choose the θ nearest of an element of the set

$\{0^o, 30^o, 45^o, 60^o, 90^o, 120^o, 135^o, 150^o, 180^o, 210^o,$

$225^o, 240^o, 270^o, 300^o, 315^o, 330^o, 360^o\}$.

This angle helps us to select the position of 1 using Table 1 and Figure 7. A formal algorithm of this method is given below.

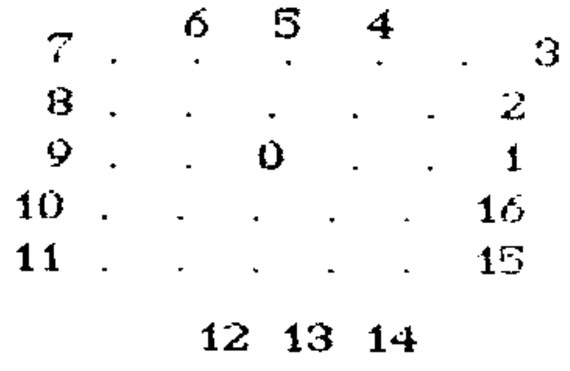


Figure 7 : pixels with corresponding positions;
0 having the centre pixel.

Table 1: *Angle and the corresponding position of pixel.*

1	0^o	5	90^o	9	180^o	13	270^o
2	30^o	6	120^o	10	210^o	14	300^o
3	45^o	7	135^o	11	225^o	15	315^o
4	60^o	8	150^o	12	240^o	16	330^o

Formal Algorithm

1> The binary image matrix $B = [b_{ij}]$

$\forall$ i,j $\in$ [0,..m;0,..n] where $b_{ij} = 0 \ \forall$ i,j

2> We shall generate w << n^2 elements of 1

3> Initialize value of $\theta = \theta_0$, $\mu = \mu_\theta$ and $\sigma = \sigma_\theta$

4> Generate a normally distributed random number

$\theta = N(\theta,\mu,\sigma)$ for a seed of uniformlly distributed random number.

5> $\theta = \sigma * NORMAL(\mu,\sigma) + \theta_0$

6> θ = Integer(θ), assume θ is in degree,

if not convert it to its corresponding values of degree

7> $\theta = \theta$ mod 360

8> If θ is negative then $\theta = \theta + 360$

9> Select θ to its nearest value of the set

$\{0^\circ, 30^\circ, 45^\circ, 60^\circ, 90^\circ, 120^\circ, 135^\circ, 150^\circ, 180^\circ, 210^\circ,$

$225^\circ, 240^\circ, 270^\circ, 300^\circ, 315^\circ, 330^\circ, 360^\circ\}$.

Also choose its corresponding position as shown in table 1

10> Fill that position by 1 and its 8-neighbour in B.

11> $\theta_0 = \theta$

12> Repeate the steps 1 to 11 until it generates w points.

2.5. EXPERIMENTAL RESULTS AND CONCLUSION

We have implemented uniformly (both 4-connected and 8-connected) and normally distributed binary image generator using pascal (in Micro-VAX and IBM-PC). We have performed a number of experiments to generate different binary images. Few results of the experiment are shown in Figure 8-10 In the study of the average performance of template matching thinning algorithms [7,8] used uniformly distributed binary image. The

same study can be performed by using normally distributed binary image. Also we can make this binary image more realistic by adding some noise [14]. This idea can be expanded for the random natural line segment (i.e. map line segment) generation using normally distributed random number generation for the two parameter in polar co-ordinate system (r,θ) where r is the length of the line segment and θ is the orientation.

3

ANALYSIS OF TEMPLATE MATCHING THINNING ALGORITHMS USING MARKOV PROCESS

3.1. INTRODUCTION

In this chapter we have analysed the template matching thinning algorithms for measuring their time complexity. The objective is to measure the total amount of time required by the algorithm and the average number of iterations required to converge the thinning process. Here we have proposed a probabilistic model using **Markov process** for measuring average time complexity of the template matching thinning algorithms. Using the proposed model it is also possible to compute a bound on the number of iterations requried for thinning process applied on a normally distributed binary image.

3.2. SOME DEFINITIONS

Stochastic process can be defined [22] as a collection of rando varibles X_t ,$t \in T$. These random varibles are defined on a common probability space and $T \subset (-\infty, \infty)$ is thought of as a time parameter set. The process is called a continuous parameter process if T is an interval having positive length and a discrete parameter process if T is a subset of the integers. If the random variables X_t all take on values from the fixed set S, then S is called the state space of the process.

Many stochastic processes possess the property that, given the present state of the process, the past history does not affect conditional probabilities of events defined in terms of the furute. Such processes are called **Markov processes**.

The property that given the present state, the past states have no influence on the future, is known as **Markov property** and the

systems having this property are called **Markov chains**. Equivalently, we can say a process $\{X_n\}$ having state space S is said to be Markov process if for every choice of non-negative integer n and the numbers $x_0, x_1, \ldots, x_{n+1}$ each in S,

$$P(X_k = x_k \mid X_0 = x_0, \ldots, X_{k-1} = x_{k-1}) = P(X_k = x_k \mid X_{k-1} = x_{k-1})$$

Also the conditional probabilities $P(X_t = y \mid X_0 = x)$ are called the Transition probabilities of the chain. A Markov process $\{X_t\}$, $t \in T$, is said to be **jump process** if

$$X_t = \begin{cases} x_0, & \text{for } 0 \le t < \tau_1 \\ x_1, & \text{for } \tau_1 \le t < \tau_2 \\ x_2, & \text{for } \tau_2 \le t < \tau_3 \\ \quad \ldots & \\ \quad \ldots & \\ x_{k-1}, & \text{for } \tau_{k-1} \le t < \tau_k \\ x_{k-1}, & \text{for } \tau_k \le t < \infty \end{cases}$$

where, $T = [0, \infty)$.

3.3. PURE DEATH PROCESS

Let Y_t be the random variable (r.v.) representing a number of deaths occured at time interval of length t. So Y_t takes values 0,1,2,.. etc . The process $\{Y_t, t \ge 0\}$ is known as death process [21] provided the following assumptions hold,

(a1) The conditional probability that during an interval (t,t+h), where h (>0) is small, a death occurs, given that at the beginning of the interval the system was in state i, is approximately equal to μ_{ih}.

(a2) The conditional probability of more than one death during (t,t+h) is negligible.

where, μ_i is death rate in state i. In pure death process there is absolutely no birth. Thus, the pure death process can be treated as jump process.

Define the transition probabilities as

$p_{ij}(t) = P(Y_{t+h} = j \mid Y_h = i)$; $t,h \geq 0$, which gives the probability that at time (t+h) the system is in state j (i.e. number of alives at time t+h is j) given that at time h it was instate i. So, we have $p_{ij}(t) = 0$, for $j > i$, since it is pure death process.

So, by (a1) $p_{i,i-1}(h) = P(Y_{t+h} = i-1 \mid Y_t = i) = \mu_{ih}$, and by

(a2) $p_{j+k,j}(h) = P(Y_{t+h} = j \mid Y_t = j+k)$, is negligible,

for $k > 1$.

Suppose that at time $t = 0$ the system is at state i. In order to have state j at time t+h (h is small positive), we consider the following three possibilities,

(a) at time t, it is in state j+1 and one death occurred during (t,t+h).

(b) at time t, it is in state j and no death occurred during (t,t+h).

(c) more than one death occurred during (t,t+h).

Thus we have,

$$p_{ij}(t+h) = p_{i,j+1}(t)\, p_{j+1,j}(h) + p_{i,j}(t)\, p_{j,j}(h) + \sum_{k=j+2}^{i} p_{i,k}(t)\, p_{k,j}(h)$$

$$= p_{i,j+1}(t)\, \mu_{j+1}\, h + p_{i,j}(t) \left[1 - \mu_j\, h \right] + \text{negligible terms}$$

[by a1,a2 and the fact, $\sum_{k=0}^{i} p_{j,j-k}(h) = 1$]

which gives us the following,

$$p'_{i,k}(t) = p_{i,j+1}(t)\, \mu_{j+1} - p_{i,j}(t)\, \mu_j$$

[dividing both sides by h and taking limit on h]

.....(1)

Initial conditions are $p_{i,j}(0) = 0, \quad \forall \quad i \neq j.$

and $p_{i,i}(0) = 1.$

Now, $p'_{i,i}(t) = p_{i,i+1}(t)\, \mu_{i+1} - p_{i,i}(t)\, \mu_i$ [from (1)]

$$= - p_{i,i}(t)\, \mu_i .$$

$$\Rightarrow \quad p_{i,i}(t) = e^{-\mu_i t}, \ \forall\ t \geq 0 \quad \ldots\ldots\ldots(2)$$

[using initial conditions]

put $j = i - 1$ in (1), so

$$p_{i,i-1}'(t) = p_{i,i+1}(t)\,\mu_i - p_{i,i-1}(t)\,\mu_{i-1}$$

$$= \mu_i e^{-\mu_i t} - p_{i,i-1}(t)\,\mu_{i-1}, \text{[using (2)]}$$

$$\ldots\ldots\ldots\ldots(3)$$

LEMMA 1 :

If $f'(t) = -\alpha f(t) + g(t)$, for $t \geq 0$, then

$$f(t) = e^{-\alpha t} f(0) + \int_0^t e^{-\alpha(t-s)} g(s)\, ds .$$

one can easily prove it by multiplying both sides by $e^{-\alpha t}$ of the given condition and integrating over s from 0 to t.

So using LEMMA 1, we get from (3)

$$p_{i,i-1}(t) = e^{-\mu_{i-1} t}\, p_{i,i-1}(0) + \mu_i \int_0^t e^{-\mu_{i-1}(t-s)}\, e^{-\mu_i s}\, ds$$

$$= \frac{\mu_i}{\mu_i - \mu_{i-1}} \left(e^{-\mu_{i-1} t} - e^{-\mu_i t} \right) \quad \ldots\ldots(4)$$

Now putting $j=i-2$ in (1), we get

$$p_{i,i-2}'(t) = \mu_{i-1}\, p_{i,i-1}(t) - \mu_{i-2}\, p_{i,i-2}(t)$$

$$\Rightarrow p_{i,i-2}(t) = \frac{\mu_i\, \mu_{i-1}}{\mu_i - \mu_{i-1}}\, e^{-\mu_{i-2} t} \left[\frac{1 - e^{-(\mu_{i-1} - \mu_{i-2})t}}{\mu_{i-1} - \mu_{i-2}} - \frac{1 - e^{-(\mu_i - \mu_{i-2})t}}{\mu_i - \mu_{i-2}} \right]$$

[using LEMMA 1]

It is reasonable to assume that $\mu_i \propto i$, $\forall$ i. So $\mu_i = \mu i$ where μ is the proportionality constant.

Now,

$$p_{i,i-1}(t) = i\, e^{-\mu t(i-1)} \left(1 - e^{-\mu t} \right)$$

and $$p_{i,i-2}(t) = \binom{i}{2} e^{-\mu t(i-2)} \left[1 - e^{-\mu t} \right]^2$$

Thus for any $j \leq i$, we have

$$p_{i,j}(t) = \binom{i}{j} e^{-\mu t j} \left[1 - e^{-\mu t} \right]^{i-j} \quad \ldots\ldots\ldots(5)$$

3.4. MODELLING OF THINNING PROBLEM

Suppose we have a thinning algorithm A and we want to thin an image, say, I_0. Now applying A on I_0 we get different modified images, say, $I_1, I_2, \ldots, I_k$ at different iterations and the algorithm will stop after k-th. iterations, if the number of 1's (considering binary image) in I_{k-1} is same as that in I_k. We see that the intermediate thinned image I_{r+1} depends only on the immediate past thinned image I_r, not on the images $I_0, I_1, \ldots, I_{r-1}$. So if X_r is a r.v. denoting the number of 1's in I_r, r =0,1,..k, then we can say that

$$P(X_k = x_k \mid X_0 = x_0, \ldots, X_{k-1} = x_{k-1}) = P(X_k = x_k \mid X_{k-1} = x_{k-1})$$

where, $x_0, x_1, \ldots, x_k$ are non-negative integers.Thus we can conclude, thinning is a Markov process.Thinning can also be treated as Pure Jump process, as

$$X_t = \begin{cases} x_0, & \text{for } 0 \leq t < \tau_1 \\ x_1, & \text{for } \tau_1 \leq t < \tau_2 \\ x_2, & \text{for } \tau_2 \leq t < \tau_3 \\ \cdots & \\ \cdots & \\ x_{k-1}, & \text{for } \tau_{k-1} \leq t < \tau_k \\ x_{k-1}, & \text{for } \tau_k \leq t < \infty \end{cases}$$

gives number of 1's at different time point t. Also, we note that in thinning process of binary image, 0 can never be changed to 1. But 1 may change to 0 or remains unchanged. So it is a Pure Death Process.

Now, $P(Y_0 = x_0, Y_{\tau_1} = x_1, \ldots, Y_{\tau_{k-1}} = x_{k-1}, Y_{\tau_k} = x_{k-1})$

$= P(Y_0 = x_0)\, P(Y_{\tau_1} = x_1 \mid Y_0 = x_0) \ldots$

$P(Y_{\tau_{k-1}} = x_{k-1} \mid Y_0 = x_0, \ldots, Y_{\tau_{k-2}} = x_{k-2})$

$P(Y_{\tau_k} = x_{k-1} \mid Y_0 = x_0, \ldots, Y_{\tau_{k-1}} = x_{k-1})$.

$= P(Y_0 = x_0)\, P(Y_{\tau_1} = x_1 \mid Y_0 = x_0) \ldots$

$P(Y_{\tau_{k-1}} = x_{k-1} \mid Y_{\tau_{k-2}} = x_{k-2})$

$P(Y_{\tau_k} = x_{k-1} \mid Y_{\tau_{k-1}} = x_{k-1})$.

[by Markov Property]

$$= \pi(0) \binom{x_0}{x_1} e^{-\mu x_1 \tau_1} \left[1 - e^{-\mu \tau_1} \right]^{x_0 - x_1} \binom{x_1}{x_2} e^{-\mu x_2 (\tau_2 - \tau_1)} \ldots \ldots$$

$$\binom{x_{k-2}}{x_{k-1}} e^{-\mu x_{k-1}(\tau_{k-1} - \tau_{k-2})} \left[1 - e^{-\mu(\tau_{k-1} - \tau_{k-2})} \right]^{x_{k-2} - x_{k-1}} e^{-\mu x_{k-1}(\tau_k - \tau_{k-1})}$$

where, $\pi(0) = P(Y_0 = x_0)$ and using (5)

$$= \pi(0) \prod_{i=1}^{k-1} \binom{x_{i-1}}{x_i} e^{-\mu \left[\sum_{i=1}^{k-1} x_i(\tau_i - \tau_{i-1}) + x_{k-1}(\tau_k - \tau_{k-1}) \right]} \prod_{i=1}^{k-1} \left[1 - e^{-\mu(\tau_i - \tau_{i-1})} \right]^{x_{i-1} - x_i}$$

[with $\tau_0 = 0$]

$$= \pi(0) \prod_{i=1}^{k-1} \binom{x_{i-1}}{x_i} e^{-\mu d \left[\sum_{i=1}^{k-1} x_i + x_{k-1} \right]} \left(1 - e^{-\mu d} \right)^{x_0 - x_{k-1}} \quad \ldots\ldots\ldots\ldots (6)$$

[assuming that $\tau_i - \tau_{i-1} = d \;.\; \forall\, i$]

Let w = u*v be the # elements in the templates used for thinning. So 2^w is # different patterns of the window (all possible templates) produced.

Let m be the # templates used in the algorithm for thinning. So

P(0 is changed to 1) = $p_{01} = 0$.

P(0 is changed to 0) = $p_{00} = 1$.

P(1 is changed to 0) = $p_{10} = \frac{\text{total \# possible windows which are matched}}{\text{all possible windows}}$

$$= \frac{m}{2^w}$$

P(1 is changed to 1) = $p_{11} = 1 - p_{10}$

Initially, there was x_0 1's in the input image . After one iteration # 1's will be $x_1 = x_0 * p_{11}$. Similarly $x_2 = x_1 * p_{11}$

Hence $x_i = x_0 * p_{11}^i$, $\forall$ i = 1,2, . . . , k-1.

So, (6) becomes

$$P(Y_0 = x_0, Y_{\tau_1} = x_1, \ldots, Y_{\tau_{k-1}} = x_{k-1}, Y_{\tau_k} = x_{k-1})$$

$$= \pi(0) \prod_{i=1}^{k-1} \binom{x_{i-1}}{x_i} e^{-\mu d x_0 \left[p_{11} \frac{1 - p_{11}^{k-1}}{1 - p_{11}} + p_{11}^{k-1} \right]} \left[1 - e^{-\mu d} \right]^{x_0 (1 - p_{11}^{k-1})}$$

$\leq$ 1 , [as it is a probability] .....(7)

LEMMA 2 :

If $N = n_1 + n_2 + \ldots + n_k$ for any non-negative integer N, where, n's are also non-negative, then we have

$$N! \geq n_1! \, n_2! \ldots n_k!$$

Now, $\prod_{i=1}^{k-1} \binom{x_{i-1}}{x_i} = \frac{x_0!}{(x_0 - x_1)!(x_1 - x_2)! \ldots (x_{k-2} - x_{k-1})! x_{k-1}!}$

$\geq$ 1 , [by LEMMA 2] .....(8)

Hence combining (7) & (8) we get,

$$\pi(0) \quad e^{-\mu d x_0 \left[p_{11} \frac{1 - p_{11}^{k-1}}{1 - p_{11}} + p_{11}^{k-1} \right]} \left[1 - e^{-\mu d} \right]^{x_0 (1 - p_{11}^{k-1})} \leq 1$$

$$\Rightarrow \log \pi(0) - \mu d x_0 \frac{p_{11}(1-p_{11}^{k-1})}{1 - p_{11}} - \mu d x_0 p_{11}^{k-1} + x_0 (1-p_{11}^{k-1}) \log(1-e^{-\mu d}) \leq 0$$

[taking logarithm on both sides]

$$\Rightarrow x_0 p_{11}^{k-1} \left[\mu d \frac{2p_{11}-1}{1-p_{11}} - \log(1-e^{-\mu d}) \right] \leq \mu d x_0 \frac{p_{11}}{1-p_{11}} - x_0 \log(1-e^{-\mu d}) - \log \pi(0) \quad \ldots\ldots\ldots\ldots (*)$$

[on simplification]

Here, μd is average # deletion during time d. So, $\mu d > 0$.

Thus $\log(1-e^{-\mu d}) < 0$

case 1 : $0.5 < p_{11} \leq 1$

so, $1-p_{11} > 0$ and $2p_{11}-1 > 0$.

Thus, $\frac{2p_{11}-1}{1-p_{11}} > 0$.

Hence , $\mu d \frac{2p_{11}-1}{1-p_{11}} - \log(1-e^{-\mu d}) > 0$, provided $p_{11} > 0.5$

So, from (*)

$$p_{11}^{k-1} \leq \frac{\mu d x_0 \frac{p_{11}}{1-p_{11}} - x_0 \log(1-e^{-\mu d}) - \log \pi(0)}{x_0 \left[\mu d \frac{2p_{11}-1}{1-p_{11}} - \log(1-e^{-\mu d}) \right]}$$

$$= 1 + \frac{\mu d x_0 - \log \pi(0)}{x_0 \left[\mu d \frac{2p_{11}-1}{1-p_{11}} - \log(1-e^{-\mu d}) \right]} = C, \text{ say}$$

[on simplification]

As $0 < \pi(0) < 1$, So, $\mu d x_0 - \log \pi(0) > 0$

Thus, $C > 1$

So, $(k-1) \log p_{11} < \log C$

$$\Rightarrow k > 1 + \frac{\log C}{\log p_{11}} \Rightarrow k \geq 1,$$

[as $p_{11} < 1$ and so $\log p_{11} < 0$]

case 2 : $0 \leq p_{11} \leq 0.5$

Let us assume that,

$$\mu d \frac{2p_{11}-1}{1-p_{11}} - \log(1-e^{-\mu d}) < 0,$$

$$\Leftarrow -2 + \frac{1}{1-p_{11}} - \frac{\log(1-e^{-\mu d})}{\mu d} < 0, \quad [\text{ as } \mu d > 0]$$

$$\Leftarrow p_{11} < 1 - \frac{1}{2 + \frac{\log(1-e^{-\mu d})}{\mu d}}$$

[on simplification]

Again, $0 < 2 + \frac{\log(1-e^{-\mu d})}{\mu d} < 2$, as $\log(1-e^{-\mu d}) < 0$

So, $\frac{1}{2 + \frac{\log(1-e^{-\mu d})}{\mu d}} > 0.5$

$$\Rightarrow 1 - \frac{1}{2 + \frac{\log(1-e^{-\mu d})}{\mu d}} < 0.5$$

Thus,

$$\blacktriangleright 0 < p_{11} < 1 - \frac{1}{2 + \frac{\log(1-e^{-\mu d})}{\mu d}} < 0.5, \text{ which is true.}$$

So, $\mu d \dfrac{2p_{11}-1}{1-p_{11}} - \log(1-e^{-\mu d}) < 0,$

Thus, $k < 1 + \dfrac{\log C}{\log p_{11}}$, provided $p_{11} < 0.5$ (proceeding as case 1)

But in practice, $p_{11} \geq 0.5$. So, this relation provides lower bound of the # iteration required. We are now trying to get an upper bound of k.

Suppose, $P(Y_t = y \mid Y_0 = x) = e^{-\lambda} \dfrac{\lambda^{x-y}}{(x-y)!}$, which is **Poisson Distribution** with λ as the parameter denoting average deletion of 1's in each iteration.

Now,

$$1 \geq P(Y_0 = x_0, Y_{\tau_1} = x_1, \ldots, Y_{\tau_{k-1}} = x_{k-1}, Y_{\tau_k} = x_{k-1})$$

$$= P(Y_0 = x_0)\, P(Y_{\tau_1} = x_1 \mid Y_0 = x_0) \ldots$$
$$P(Y_{\tau_{k-1}} = x_{k-1} \mid Y_0 = x_0, \ldots, Y_{\tau_{k-2}} = x_{k-2})$$
$$P(Y_{\tau_k} = x_{k-1} \mid Y_0 = x_0, \ldots, Y_{\tau_{k-1}} = x_{k-1}).$$

$$= P(Y_0 = x_0)\, P(Y_{\tau_1} = x_1 \mid Y_0 = x_0) \ldots$$
$$P(Y_{\tau_{k-1}} = x_{k-1} \mid Y_{\tau_{k-2}} = x_{k-2})$$
$$P(Y_{\tau_k} = x_{k-1} \mid Y_{\tau_{k-1}} = x_{k-1}).$$

[by Markov property]

$$= \pi(0)\, e^{-\lambda} \frac{\lambda^{x_0-x_1}}{(x_0-x_1)!}\, e^{-\lambda} \frac{\lambda^{x_1-x_2}}{(x_1-x_2)!} \ldots$$
$$e^{-\lambda} \frac{\lambda^{x_0-x_1}}{(x_0-x_1)!}\, e^{-\lambda}$$

$$= \pi(0)\, e^{-k\lambda} \frac{\lambda^{x_0-x_{k-1}}}{(x_0-x_1)! \ldots (x_{k-2}-x_{k-1})!}$$

$$\geq \pi(0) \quad e^{-k\lambda} \frac{\lambda^{x_0 - x_{k-1}}}{(x_0 - x_{k-1})!} \quad [\text{ by LEMMA 2 }]$$

$$\geq \pi(0) \quad e^{-k\lambda} \frac{\lambda^{x_0 - x_{k-1}}}{(x_0 - x_{k-1})^{x_0 - x_{k-1}}} \quad [\text{ as } \frac{1}{i!} \geq \frac{1}{i^i} \text{ }] \qquad \ldots\ldots\ldots\ldots(9)$$

Now, $x_0 > x_{k-1}$ [trivial]

$$\Rightarrow \quad 0 < x_0 - x_{k-1} < x_0$$

$$\Rightarrow \quad \frac{1}{(x_0 - x_{k-1})^{x_0 - x_{k-1}}} > \frac{1}{(x_0)^{x_0}} \qquad \ldots\ldots\ldots(10)$$

combining (9) & (10), we get

$$\pi(0) \quad e^{-k\lambda} \frac{\lambda^{x_0 - x_{k-1}}}{(x_0)^{x_0}} < 1 \qquad \ldots\ldots\ldots\ldots(11)$$

Here, λ is estimated unbiasedly by the mean of the r.v. denoting # deletion in each iteration.

$$\hat{\lambda} = \frac{(x_0 - x_1) + (x_1 - x_2) + \ldots + (x_{k-2} - x_{k-1})}{k} = \frac{(x_0 - x_{k-1})}{k}$$

$$\Rightarrow \quad x_0 - x_{k-1} = k\hat{\lambda} \qquad \ldots\ldots\ldots\ldots(12)$$

combining (11) & (12)

$$\pi(0) \quad e^{-k\hat{\lambda}} \frac{(\hat{\lambda})^{k\hat{\lambda}}}{(x_0)^{x_0}} < 1$$

$$\Rightarrow \frac{\pi(0)}{(x_0)^{x_0}} \left(\frac{\hat{\lambda}}{e}\right)^{k\hat{\lambda}} < 1$$

$$\Rightarrow \left(\frac{\hat{\lambda}}{e}\right)^{-k\hat{\lambda}} > \frac{\pi(0)}{(x_0)^{x_0}}$$

$$\Rightarrow k\ \hat{\lambda}\ \log\left(\frac{e}{\hat{\lambda}}\right) \quad > \quad \log\ \pi(0) - x_0\ \log(\ x_0\ )$$

$$\Rightarrow k \quad < \quad \frac{1}{\hat{\lambda}} \left[\frac{\log\ \pi(0) - x_0\ \log(\ x_0\ )}{1 - \log(\ \hat{\lambda}\ )} \right] = B\ ,\ \text{say} \qquad \ldots\ldots\ldots\ldots(13)$$

[assuming $\hat{\lambda} > e$]

Since $0 < \pi(0) < 1$, So, $\log\ \pi(0) < 0$, and thus

$$\log\ \pi(0) - x_0\ \log(\ x_0\ ) \ < \ 0 \quad \text{as } x_0 > 1$$

Also we have assumed $\hat{\lambda} > e$. Thus $(\ 1 - \log\ \hat{\lambda}\ ) < 0$.
Hence RHS of (13) is positive. So, (13) gives us an upper bound of k.

3.5. COMPLEXITY ANALYSIS

Usually performance of an algorithm is measured by time and space required for that algorithm. Here, in fact, these things are function of size of the input image. We are considering average case complexity to measure the performance of template matching thinning algorithm. Measuring time complexity of images of different sizes, we took average of them. There are so many such algorithms, but nobody has described their performance except Pal & Bhattacharyya [7].

We have proposed a stochastic model regarding this. The following tables give the actual number of iterations required for 10 different normally distributed binary images [described in chapter 2] of different sizes, their estimated bounds and time required for five different algorithms [15,16,17,4,18], where, θ, σ, n are the parameter of the algorithm for normally distributed binary image, N is the no. of iterations required. B is the bound of no. of iteration obtained experimentally and $\hat{\lambda}$ is the average no. of deletions in each iteration. Table 1 to table 6 are corresponding to image size 32x32 and rest are corresponding to 64x64.

Table 1 :

Alorithm	θ	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	0	0.356	456	610	5	37.6	39.542	2260
Chin					11	42.72	34.272	3430
Hall					7	66.86	18.242	1790
Holt					8	56.37	22.85	5820
Pal & Bhatt.					9	57.78	22.116	9470

Table 2 :

Alorithm	θ	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	36	0.467	456	616	4	55.75	23.456	2010
Chin					10	48.5	28.266	2990
Hall					6	80.33	14.522	1570
Holt					7	67.00	18.398	5200
Pal & Bhatt.					10	58.30	22.103	8590

Table 3 :

Alorithm	θ	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	47	0.567	456	694	6	41.83	39.646	2960
Chin					13	43.69	37.365	4000
Hall					9	62.89	22.95	2300
Holt					10	55.4	27.148	7570
Pal & Bhatt.					8	80.0	16.757	8850

Table 4 :

Alorithm	e	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	9	0.245	357	440	4	31.0	35.419	1190
Chin					6	51.67	17.562	1630
Hall					3	105.00	6.965	770
Holt					4	74.75	10.786	2540
Pal & Bhatt.					4	86.0	8.995	3240

Table 5 :

Alorithm	e	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	47	0.677	478	742	7	39.57	46.213	3600
Chin					16	37.75	49.311	5380
Hall					10	61.4	25.587	2760
Holt					11	53.64	30.618	9290
Pal & Bhatt.					9	75.78	19.421	10540

Table 6 :

Alorithm	e	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	28	0.367	389	605	5	38.0	38.599	1950
Chin					17	26.59	63.805	5050
Hall					7	65.86	18.43	1770
Holt					7	62.28	19.833	5430
Pal & Bhatt.					8	66.12	18.332	7920

Table 7 :

Alorithm	θ	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	0	0.478	698	1717	6	83.67	44.579	6330
Chin					9	136.78	23.848	8520
Hall					6	209.67	14.028	4620
Holt					7	172.00	17.917	14590
Pal & Bhatt.					7	177.86	17.188	16900

Table 8 :

Alorithm	θ	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	15	0.699	798	1636	9	68.44	54.795	9140
Chin					17	78.65	45.718	13980
Hall					11	122.91	25.827	7590
Holt					12	109.25	29.983	21940
Pal & Bhatt.					11	120.27	26.544	23560

Table 9 :

Alorithm	θ	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	30	0.955	1276	2930	11	96.10	68.249	18040
Chin					31	78.387	88.731	36090
Hall					18	134.5	44.557	16510
Holt					18	130.89	46.107	52140
Pal & Bhatt.					19	133.21	45.099	63150

Table 10 :

Alorithm	θ	σ	n	x_0	N	$\hat{\lambda}$	B	time (μ.sec)
Zhang & Suen	45	0.966	910	2144	11	75.36	65.65	13420
Chin					26	68.46	74.421	25280
Hall					12	149.33	27.476	10010
Holt					13	133.46	31.631	31630
Pal & Bhatt.					15	116.87	37.398	39190

REFERENCES

1. Y. S. Chen and W. H. Hsu "A comparison of some one-pass parallel thinnings. Pattern Recog. Lett.. Vol. 11, 1990, 35-41.

2. R. M. Haralick "Performance characterization in image analysis: thinning, a case in point. Pattern Recog. Lett. Vol 13, 1992, 5-12.

3. S. Heydorn and P. Weidner "Optimization and performance analysis of thinning algorithms on parallel computers". Parallel Comput. Vol. 17, 1991, 17-27.

4. C. M. Holt and A. Stewart "A parallel thinning algorithm with fine grain subtasking". Parallel Comput. Vol. 10, 1989, 329-334.

5. B. K. Jang and R. T. Chin "Analysis of thinning algorithms using mathematical morphology". IEEE-PAMI, Vol. 12, 1990, 541-551.

6. M. T. Musavi, M. V. Shirvaikar, E. Ramanathan and A. R. Nekovei "A Vision Based Method To Automatic Map Processing". Pattern Recognition, Vol. 21, No. 4, 1988, 319-326.

7. S. Pal and P. Bhattacharyya "Analysis of Template Matching Thinning Algorithms". Pattern Recognition, Vol. 25, No. 5, 1992 497-503.

8. S. Pal "Some Low Level Image Segmentation Methods, Algorithms and Their Analysis." Ph.D. Thesis, Department of Computer Science & Engineering, Indian Institute of Technology, Kharagpur, 1991.

9. W. Pferd and G. Stocker "Optical Fibers for Scanning Digitizers", Bell Sys. Tech. J. 60 , 1981, 523-534.

10. L. G. Shapiro, "Inser Matching of Line Drawings in a

Syntactic Pattern Recognition System", Pattern Recognition, 10, 1978, 313-321.

11. J. C. Stoffel(Ed) Graphical and Binary Image Processing and Applications. Artech House, Dedham, M A, 1982.

12. H. J. Trussell, Processing of X-ray Images, Proc. IEEE 69, 1981, 615-627.

13. J. R. Ullman,"Binarization Using Associate Addressing", Pattern Recognition, 6, 1974, 127-135.

14. X. Zhou and R. Gordon "Generation of Noise in Binary Images". C V G I P: Graphical Mdels and and Image Processing. Vol. 53, no. 5, Sept. 1991, 476-478.

15. T. Y. Zhang and C. Y. Suen "A fast parallel algorithm for thinning digital patterns." CACM, Vol. 27, No. 3, March 1984, 236-239.

16. Chin R. T, Wan H K , Stover D L , Iverson R D. " A one pass thinning algorithm and its parallel implementation".Computer Vision,Graphics and Image Processing,vol 40 , no 1, pp-30-40, oct, 1987.

17. Guo Z , Hall R W. "Parallel Thinning with Two Subiteration Algorithm",CACM,Vol 32, No 3, March 1989, pp 359- 373.

18. Pal S and Bhattacharyya P. "A Shape Preserving One Pass Parallel Thinning Algorithm". Indian Institute of Management, Calcutta, working paper series No 123(89), 1989.

19. Queueing systems,vol 1: Theory by Leonard Kleinrock,John Wiley & Sons.1975.

20. Bratley, Fox & Schrage, "A giude to Simulation", Springer-Verlag, New York1987, 160-164.

21. R. Syski," Random Process ", Marcel Dekker Inc., New York and Basel, vol 98 second edition,revised expanded,

283-294.

22 P. G. Hoel, S. C. Port and C. J. Stone, Introduction to Stochastics Processes" Houghton Mifflin Company, U.S.A.

23. S. Pal, R. Bhattacharyya, S. Bhattacharyya, A. Ray, "A Statistically Distributed Random Binary Image Generator", IJPAM, 1993. (communicated)

Figure 8: 4-connected random binary image of size 128x128 with seed (a) 903284, (b) 25, (c) 2300 and (d) 65.

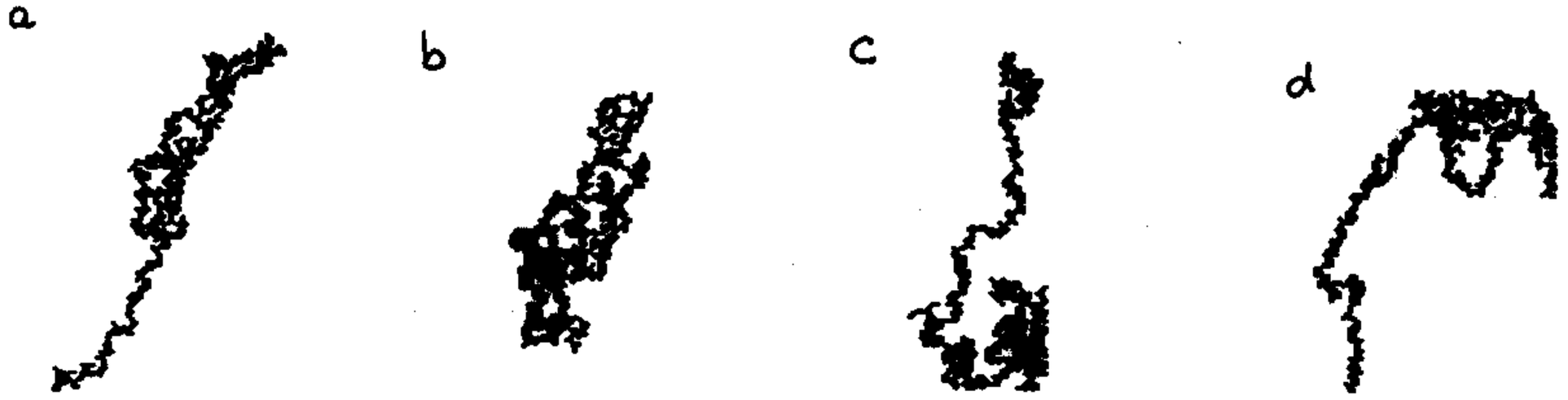


Figure 9: 8-connected random binary image of size 128x128 with seed and number of 1 generated in (a) 2387, 500, (b) 8792, 600, (c) 9345, 550 and (d) 3784,550.

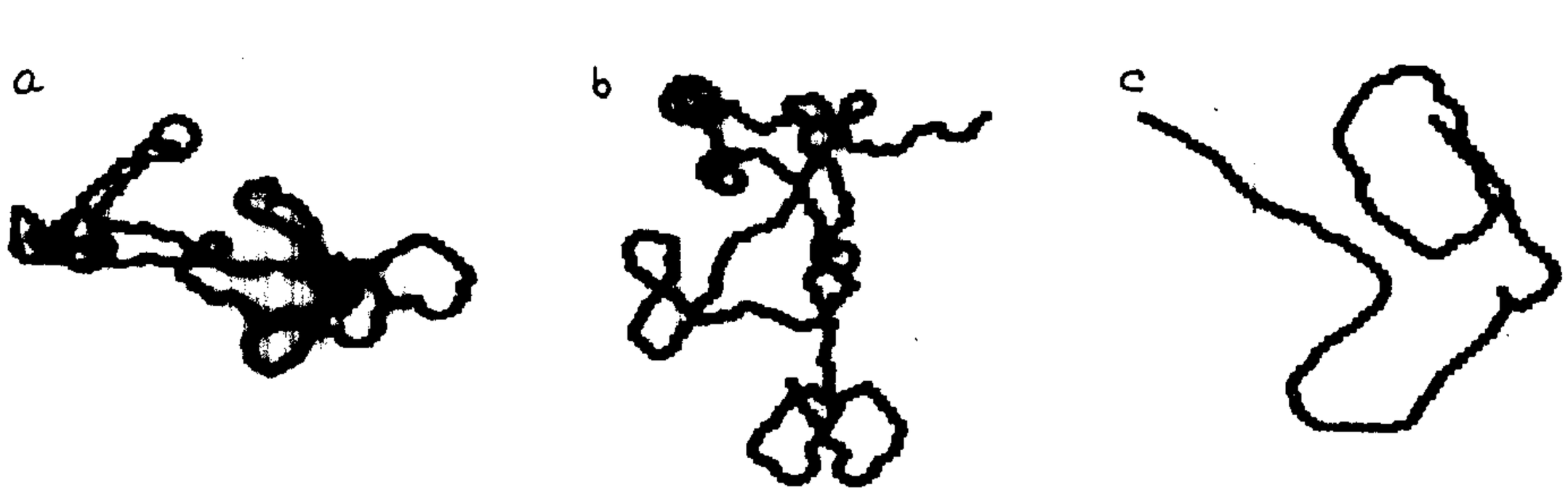


Figure 10: Normally distributed random binary image of size 200x200 with seed 4376, and mean & s.d. of the normal distribution and the number of point generated in (a) N(0,0.55), 400, (b) N(0,0.6), 400, (c) N(0,0.4), 200 and (d) N(1.0,0.2), 250.

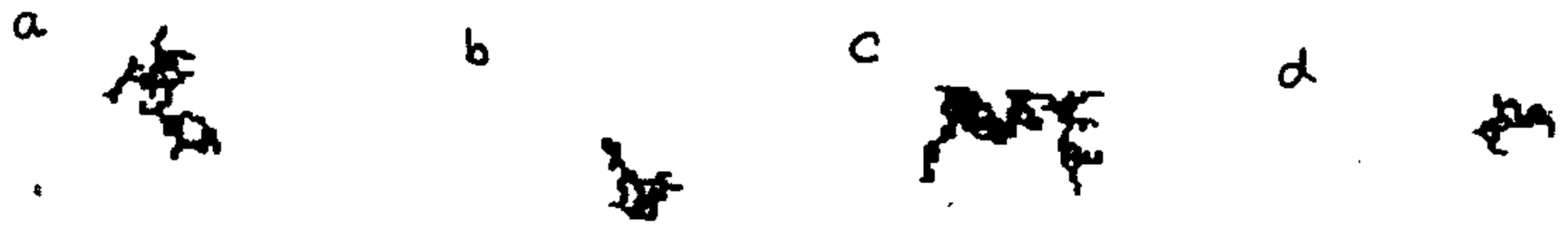


Figure 11: Thinned image of Figure 8 by using Zhang and Suen's thinning algorithm [15].

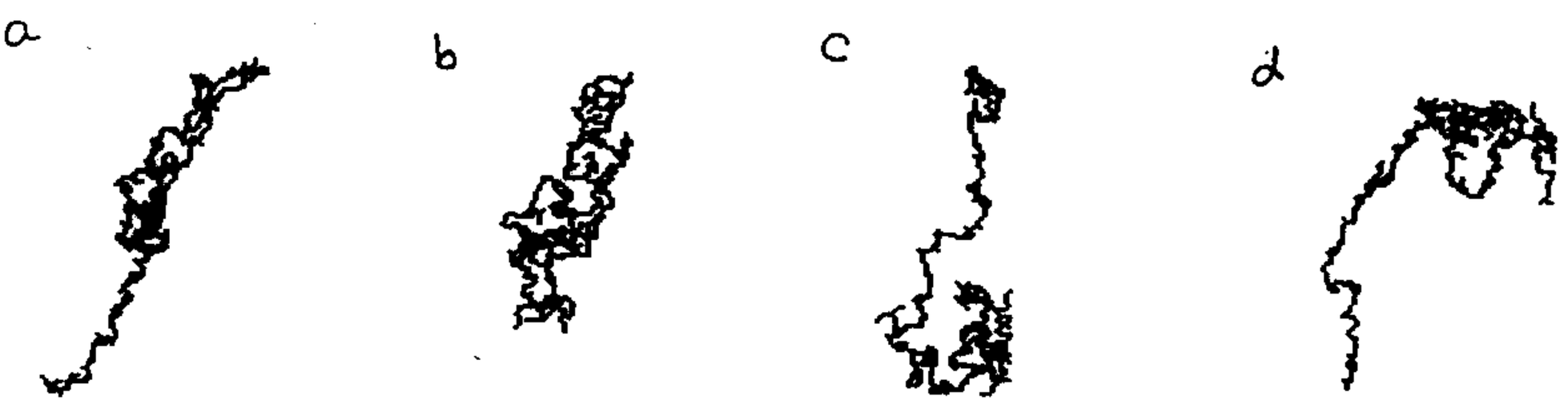


Figure 12: Thinned image of Figure 9 by using Zhang and Suen's thinning algorithm [15].

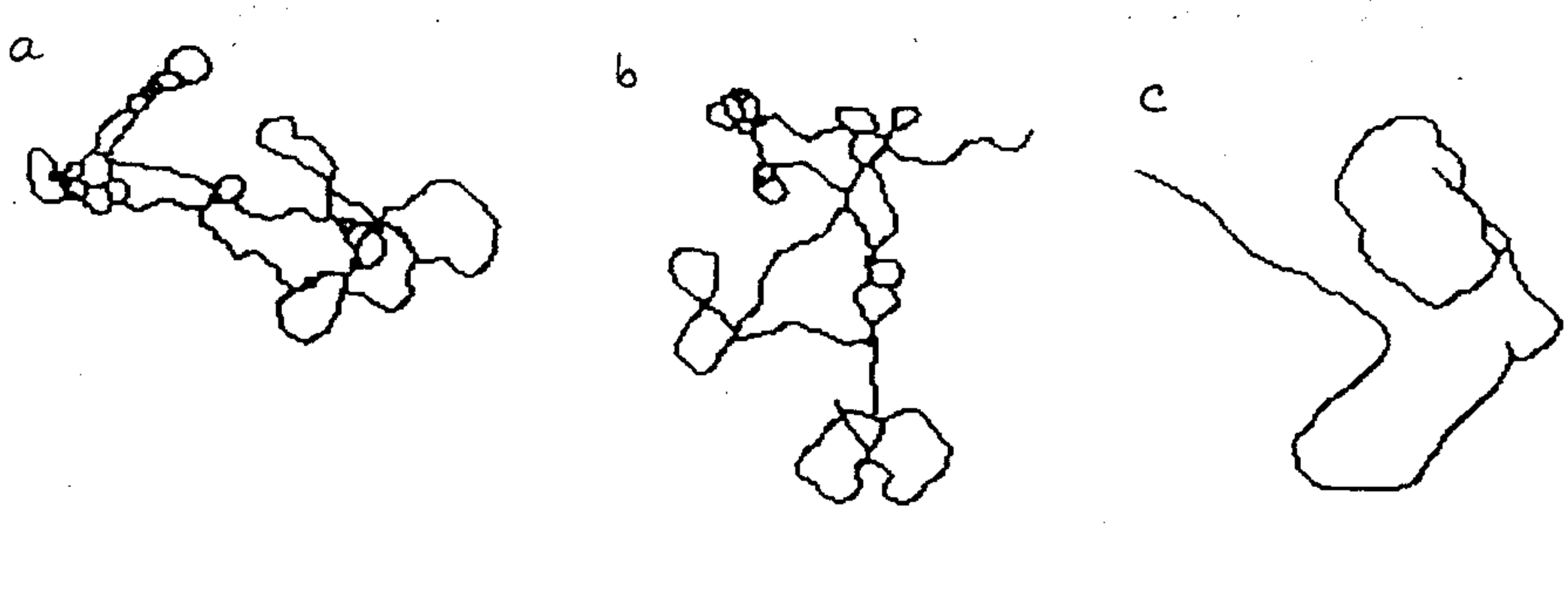


Figure 13: Thinned image of Figure 10 by using Zhang and Suen's thinning algorithm [15].

ORIGINAL IMAGE

The following images are the thinned version of the given image by (a) Zhang & Suen (b) chin et al (c) Guo & Hall (d) Holt et al and (e) Pal & Bhattacharyya 's algorithm

(a)

(b)

(C)

(d)

(e)

ORIGINAL IMAGE

(a)

(b)

(c)

(d)

(e)

ORIGINAL IMAGE

(a)

(b)

(c)

(d)

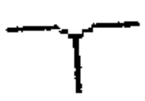
(e)

ORIGINAL IMAGE

(a)

(b)

(c)



(d)

(e)

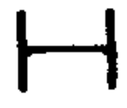
ORIGINAL IMAGE

(a)

(b)

(c)

(d)



(e)

ORIGINAL IMAGE

(a)

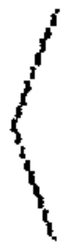
(b)

(c)

(d)

(e)

ORIGINAL IMAGE



(a)

(b)

(c)

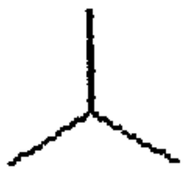
(d)

(e)

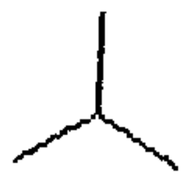
ORIGINAL IMAGE

(a)

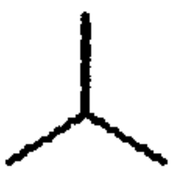
(b)



(c)



(d)



(e)

ORIGINAL IMAGE

(a)

(b)

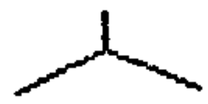
(c)

(d)

(e)

ORIGINAL IMAGE

(a)



(b)

(c)

(d)

(e)

ORIGINAL IMAGE

(a)

(b)

(c)

(d)

(e)

ORIGINAL IMAGE

(a)

(b)

(c)

ORIGINAL IMAGE

(a)

(b)

(c)

(d)

(e)