Pattern Recognition Pergamon Press 1976 Vol 8 pp 181–192 Printed in Great Britain

FEATURE EXTRACTION FOR FINGERPRINT CLASSIFICATION

T CH MALLESWARA RAO*

Electrical Communication Engineering Department, Indian Institute of Science Bangalore India

(*Received* 16 *January* 1975 *and in revised form* 19 *August* 1975)

Abstract—This paper presents two algorithms for smoothing and feature extraction for fingerprint classification Deutsch's[2] Thinning algorithm (rectangular array) is used for thinning the digitized fingerprint (binary version) A simple algorithm is also suggested for classifying the fingerprints Experimental results obtained using such algorithms are presented

Fingerprint Smoothing Thinning Feature extraction Classification

1 INTRODUCTION

To use fingerprints as the basic tool for identification of criminals in law enforcement (forensic science), for security clearances in the armed services, and for other civilian identification purposes, it is necessary to maintain large files of print records systematically according to some classification scheme For a new print identification, one has to go for file searching With the increase in the quantum of files, manual file searching becomes difficult and consumes much time With automatic identification of fingerprints this problem can be solved to a certain extent In recent years, the use of digital computers has increased in almost all fields In the field of automatic identification of fingerprints digital computers can be used for file maintainance, file searching, etc To store fingerprints in the computer, first of all, the prints have to be digitized by using a scanner After digitizing the print, again one should follow some procedure systematically to process the fingerprint in the computer, and some classification scheme to store the prints This systematic procedure helps in print retrieval and identification from large file at a fast rate Several authors[1,7 9 14] have suggested different systems for automatic identification of fingerprints After going through the above systems, here, a simple system is proposed (Fig 1)

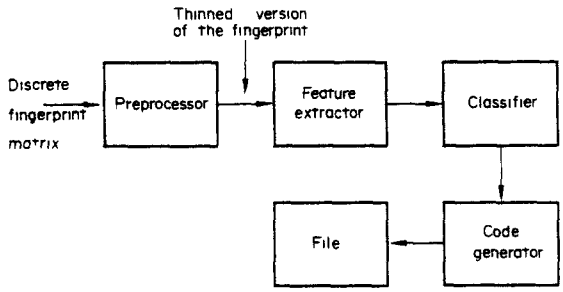

Fig 1

The system consists of five basic subsystems, viz (1) Preprocessor (2) Feature extractor (3) Classifier (4) Code generator (5) File

Input to the preprocessor is the discretized fingerprint in binary form (Fig. 11) Main functions of the preprocessor are given below

1 Filling the isolated holes
2 Removing the noise
3 Bridging the gaps
4 Thinning the picture

Several techniques are available at present for digitizing the print But, finally, for converting the scanned print into the digital version, one has to use the well-known technique of 'thresholding' After using the thresholding technique, it is quite possible to have minor irregularities in the print In fingerprints these irregularities will be more, because of fingerprint imperfections such as ridge gaps, usually caused by skinfolds, and contiguous ridges, which may be caused by spreading of ink due to finger pressure or in the worst cases by excessive inking or by smearing during rolling of the finger First operation of the preprocessor will partially remove minor irregularities In the digitized fingerprints usually noise presents at the ridge edges, because of excessive inking or by smearing during rolling Noise removing operation is performed after filling the isolated holes After this, preprocessor takes up the third operation, i e bridging the gaps in the ridges Finally, preprocessor goes to the thinning operation

1 1 *Preview*

In this paper preprocessing (excluding bridging gaps), feature extraction and classification with the required algorithms are discussed This work was carried on the IBM 360/44 in FORTRAN IV Language and results are given at the end of the paper

* Presently with the National Remote Sensing Agency Secunderabad 500 003 (A P) India

Because of insufficient memory the problem is divided into three parts

1. Smoothing and thinning,
2. Feature extraction, and
3 Classification,

for the first two parts a digitized 72 × 72 matrix picture (Fig 11) is considered. In the third part four figures with different patterns are generated On these figures (Fig 18) the algorithm is applied for classifying them, and finally the same algorithm is used for classifying Figs 19–22

2. SMOOTHING AND THINNING

2 1 *Smoothing algorithm*

Unger[13], and Freyer and Richmond[4] suggested two approaches for smoothing the data. Unger's approach is like a parallel processing algorithm. It changes the value at a point depending upon the value of the Boolean function of its neighbours Freyer and Richmonds' is a numerical method. They use weighting functions for the smoothing filters Here, a simple algorithm is suggested for smoothing digitized fingerprint Fig 13 and 14 show the fingerprints before and after applying the smoothing algorithm

In two-tone digitized fingerprint, ridges are represented by 1s and valleys by 0s Each stroke of the fingerprint is represented by more than one element. This assumption is essential for visual field discretization of the print which is fine enough

2.2. *Algorithm*

(Symbols to be noted

$\wedge$ —Logical AND

$\vee$ —Logical OR)

A convenient submatrix 3 × 3 is considered for operating the algorithm (Fig 2) Counting in counterclockwise direction, let R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8 be the neighbouring points of $X_{(i,j)}$ as shown in Figs 2–5

R_4	R_3	R_2
R_5	$X_{(i,j)}$	R_1
R_6	R_7	R_8

Fig 2

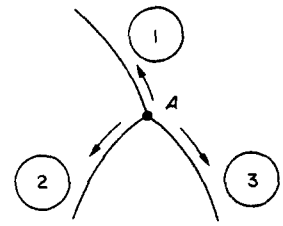

Fig 3

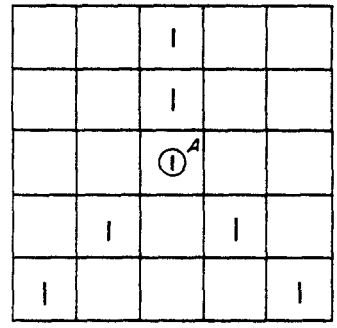

Fig 4

The following 9 conditions and 9 more that follow by the rotation of the lattice through multiples of 90° are used for smoothing.

1 $R_1 \wedge R_2 \wedge R_3 \wedge R_7 \wedge R_8 = 1$
  $R_4 \vee R_5 \vee R_6 = 0$
2. $R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5 = 1$
  $R_6 \vee R_7 \vee R_8 = 0$
3 $R_1 \vee R_5 = 0$
4 $R_3 \wedge R_4 \wedge R_7 = 1$
  $R_2 \vee R_6 \vee R_8 = 0$
5 $R_2 \wedge R_3 \wedge R_6 \wedge R_7 = 1$
  $R_4 \vee R_8 = 0$
6 $R_2 \wedge R_3 \wedge R_8 = 1$
  $R_4 \vee R_6 \vee R_7 = 0$
7 $R_2 \wedge R_3 \wedge R_6 = 1$
  $R_4 \vee R_7 \vee R_8 = 0$
8 $R_3 \wedge R_4 \wedge R_6 = 1$
  $R_2 \vee R_7 \vee R_8 = 0$
9. $R_3 \wedge R_4 \wedge R_8 = 1$
  $R_2 \vee R_6 \vee R_7 = 0$

R_{34}	R_{33}	R_{32}	R_{31}	R_{30}	R_{29}	R_{28}
R_{35}	R_{15}	R_{14}	R_{13}	R_{12}	R_{11}	R_{27}
R_{36}	R_{16}	R_4	R_3	R_2	R_{10}	R_{26}
R_{37}	R_{17}	R_5	$X_{(i,j)}$	R_1	R_9	R_{25}
R_{38}	R_{18}	R_6	R_7	R_8	R_{24}	R_{48}
R_{39}	R_{19}	R_{20}	R_{21}	R_{22}	R_{23}	R_{47}
R_{40}	R_{41}	R_{42}	R_{43}	R_{44}	R_{45}	R_{46}

Fig 5

Point $X_{(i,j)}$ is changed from 0 to 1 if ony one of the first two conditions or their mirror image versions are satisfied

Point R_5 or R_1 is changed from 0 to 1 provided $X_{(i,j)} = 1$, and condition 3 and any one of the conditions from 4 to 9 or their mirror image versions are satisfied

Conditions 3–9 or their mirror image versions are necessary for satisfying the assumption that each stroke of the fingerprint is represented by more than one element First two conditions and their 180° rotation are used for filling the isolated holes Experimental results are shown in Fig 13

Conditions 10–14 and their versions after rotation through multiples of 90° used to change $X_{(i,j)}$ from 1 to 0 These conditions remove the digitized noise present at the edges of the ridge These conditions are applied on Fig 13 and final result is shown in Fig 14

10 $R_1 \wedge R_5 = 0$
$R_3 \wedge R_7 = 0$
11 $R_1 \wedge R_2 \wedge R_8 = 1$
$R_4 \vee R_6 = 1$
12 $R_2 \wedge R_3 \wedge R_4 = 1$
$R_8 \vee R_6 = 1$
13 $R_1 \wedge R_7 \wedge R_8 = 1$
$R_4 = 1$
14 $R_1 \wedge R_2 \wedge R_3 = 1$
$R_6 = 1$

2 3 *Thinning algorithm*

Generally, the thickness of the fingerprint ridges varies, because of bad inking, improper printing, etc The variations in thickness are still worse in the digitized pictures due to digitization noise It is further supposed that the visual discretization is fine enough, so that each stroke of the print is represented by more than one element In classifying the prints and extracting the features from these ridges, it is necessary to reduce the thickness of the ridges to unity (i e obtaining the line diagram) The process of reducing the thickness of the ridges to unity is called thinning or skeletonization of fingerprints

In the digital picture processing literature, several thinning algorithms are available Among them, the algorithms of Saraga and Woollons[10], Stefanelli and Rosenfeld[11], and Deutsch[2] are tested for thinning the fingerprints Saraga and Woollons' algorithm is well suited for thinning alpha-numerics Their algorithm did not give satisfactory results when applied to fingerprints Stefanelli and Rosenfeld, and Deutsch have suggested parallel processing algorithms Their algorithms maintain the connectivity and retain the medial line Deutsch's algorithm removes isolated 1_s and works fast

Here, Deutsch's algorithm[2] is used for thinning fingerprints

Deutsch's algorithm is directly applied on Fig 14 and final results are shown in Fig 15

3 FEATURE EXTRACTION

Input to the feature extractor is a thinned picture which is the output of the preprocessor Functions of the feature extractor are

1 Determination of the exact location of the feature from the main matrix of the figure (i e to determine the coordinates of the feature)

2 Recognition of the type of the feature such as forks (open upward, open downward, open leftward or open rightward), ridge ends, ridge bends etc

3 1 *Algorithm*

In this paper, two methods are discussed for extracting forks Experimental results are shown in Figs 16 and 17

Assume the direction of the line ① as shown in Fig 3, line ① is bifurcated into two more lines ② and ③ at the point A, called 'trisection point' In a thinned picture, trisection point has three neighbours (Fig 4) To determine the coordinates of the trisection point and type of feature 7 × 7 matrix is considered as shown in Fig 5

For $X_{(i,j)}$ to be the trisection point it is required that

1 $$\chi_1 = \sum_{k=1}^{8} |R_{(k+1)} - R_{(k)}| = 6$$

2 $$\chi_2 = \sum_{k=9}^{24} |R_{(k+1)} - R_{(k)}| = 6$$

Condition 1 searches three neighbours of the point A Condition 2 only supports condition 1, and decides whether point A is exactly a trisection point Conditions similar to 1 and 2 on a 3 × 3 submatrix were used earlier by Deutsch[2] for skeletnization of alphanumeric data His conditions adapted to 7 × 7 submatrix are used in the present context

After locating the coordinates of the point *A*, search for the type of feature starts Of the two methods, first method gave 80% correct results, whereas second method gave correct results almost always Experimental results of the two methods are shown in Figs 16 and 17

3.2 *Method*—1

For the ease of explanation, let us consider downward fork (Fig 6)

As shown in Fig 6, 7 × 7 matrix block is divided into 3 small blocks (Fig 6b) Line ① lies in the block

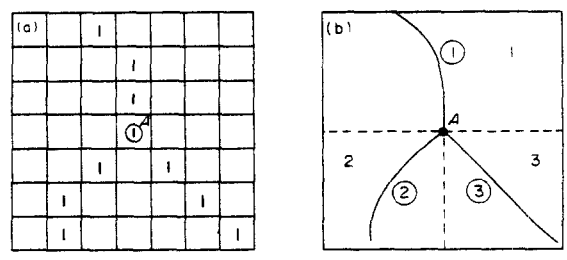

Fig 6

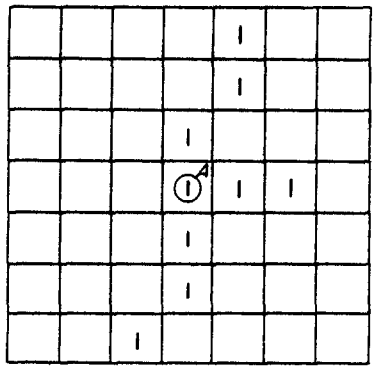

Fig 7

1 and lines ② and ③ lie in blocks 2 and 3 respectively

Let P = sum of the number of 1s in block 1
Q = sum of the number of 1s in block 2
and S = sum of the number of 1s in block 3

Using the technique (10) of putting a thresholding level,

if $P \geq 3$
$Q \geq 3$
and $S \geq 3$,

this feature is recognised as downward fork Similar procedure is used for fork upward, fork open towards left, and fork open towards right

In the feature which was classified as $FD1$ by the above algorithm, line ② has been counted in blocks 2 and 3 (Fig. 7) This is undesirable because the line ② should be included only in one block, and when it is thus included the feature would not have been classified as a downward fork which it, in fact, is not

To avoid this trouble, the possible orientations of the lines are predefined in the three blocks as follows

(FD—Fork downward)

$$FD_1 = R_2 + R_3 + R_4 + R_{11} + R_{12} + R_{13} + R_{14} + R_{15} + R_{28} + R_{29} + R_{30} + R_{31} + R_{32} + R_{33} + R_{34}$$

$$FD_2 = R_5 + R_6 + R_7 + R_{18} + R_{19} + R_{20} + R_{21} + R_{39} + R_{40} + R_{41} + R_{42} + R_{43}$$

$$FD_3 = R_1 + R_7 + R_8 + R_{21} + R_{22} + R_{23} + R_{24} + R_{43} + R_{44} + R_{45} + R_{46} + R_{47}$$

If $FD_1 \geq 3$, $FD_2 \geq 3$ and $FD_3 \geq 3$ are satisfied this feature is recognised as fork downward Conditions FD_1, FD_2, FD_3 take care of not counting the same line in two blocks Conditions for fork open rightward (FHR) are given below

$$FHR_1 = R_4 + R_5 + R_6 + R_{15} + R_{16} + R_{17} + R_{18} + R_{19} + R_{34} + R_{35} + R_{36} + R_{37} + R_{38} + R_{39} + R_{40}.$$

$$FHR_2 = R_1 + R_2 + R_3 + R_9 + R_{10} + R_{11} + R_{12} + R_{25} + R_{26} + R_{27} + R_{28} + R_{29}$$

$$FHR_3 = R_1 + R_7 + R_8 + R_9 + R_{22} + R_{23} + R_{24} + R_{25} + R_{45} + R_{46} + R_{47} + R_{48}$$

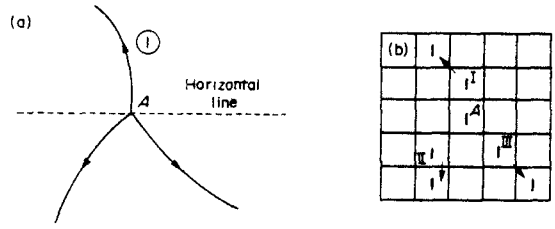

Fig 8

Conditions for fork upward (FU) and fork open leftward (FHL) follow by symmetry

This method, after locating the trisection point, fits the feature in any one of the four classes explained above. Sometimes, this may lead to wrong decisions (for example, FHR_3 shown in Fig. 16) This drawback is rectified in the second method

3 3 *Method—2*

In this method, after locating the trisection point, the algorithm searches the three lines starting from three neighbouring points of $X_{(i,j)}$

Consider, again, fork downward (Fig. 8) line ① lies above the imaginary horizontal line (broken line in

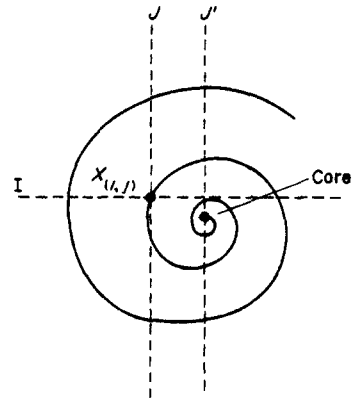

Fig 9

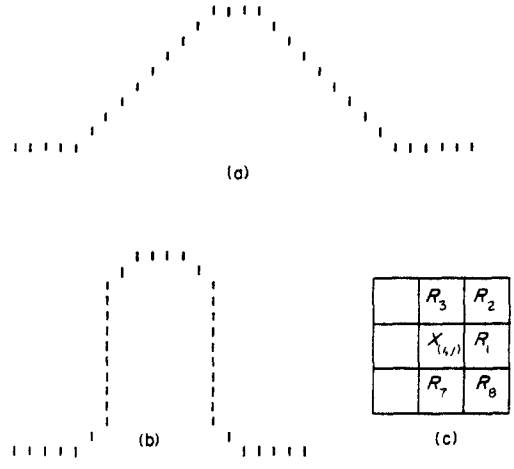

Fig 10

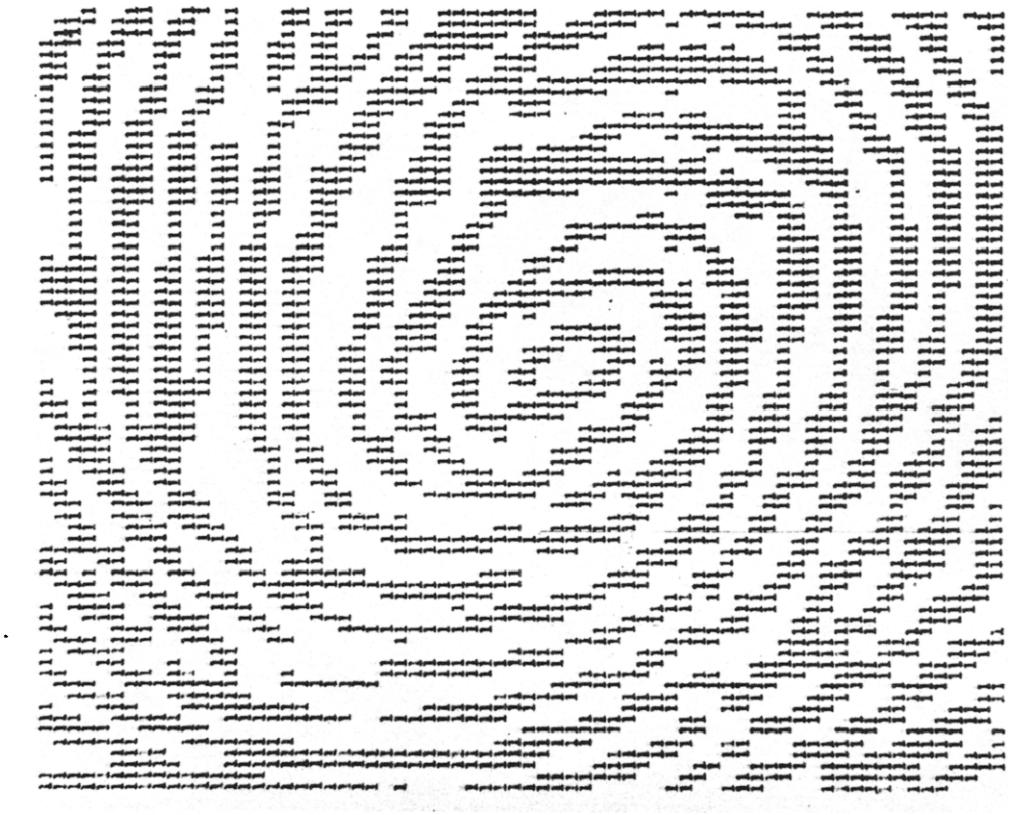

Fig 12

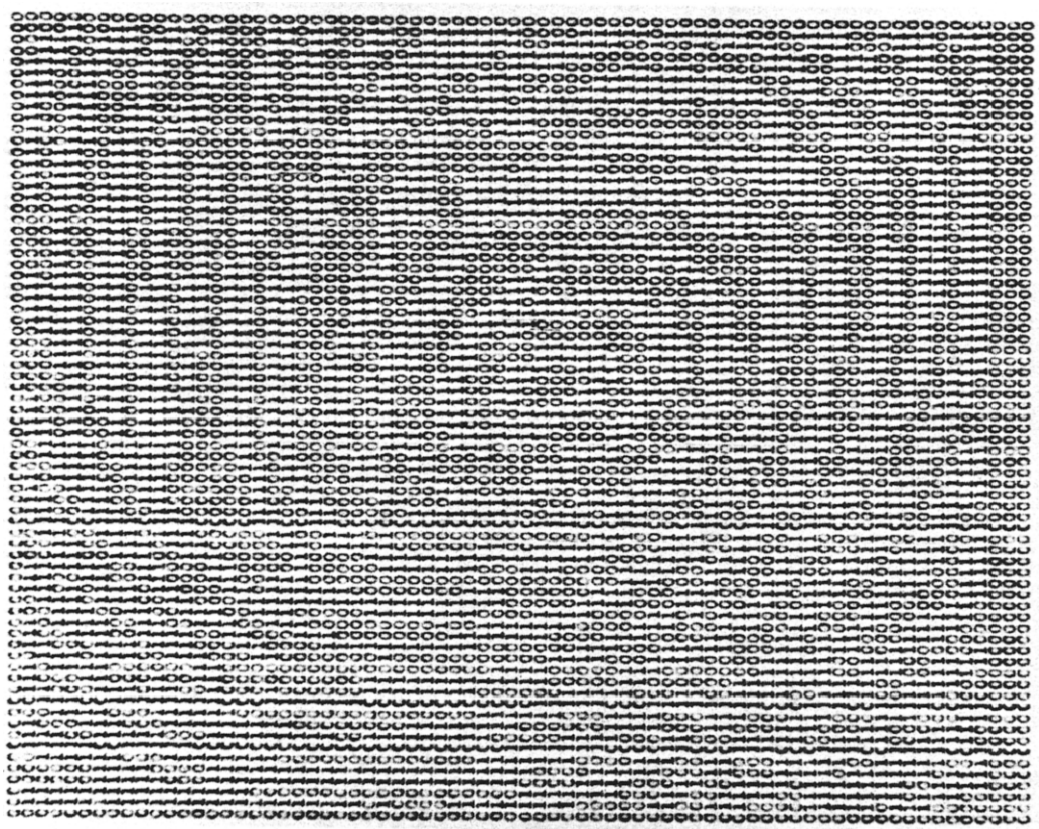

Fig 11

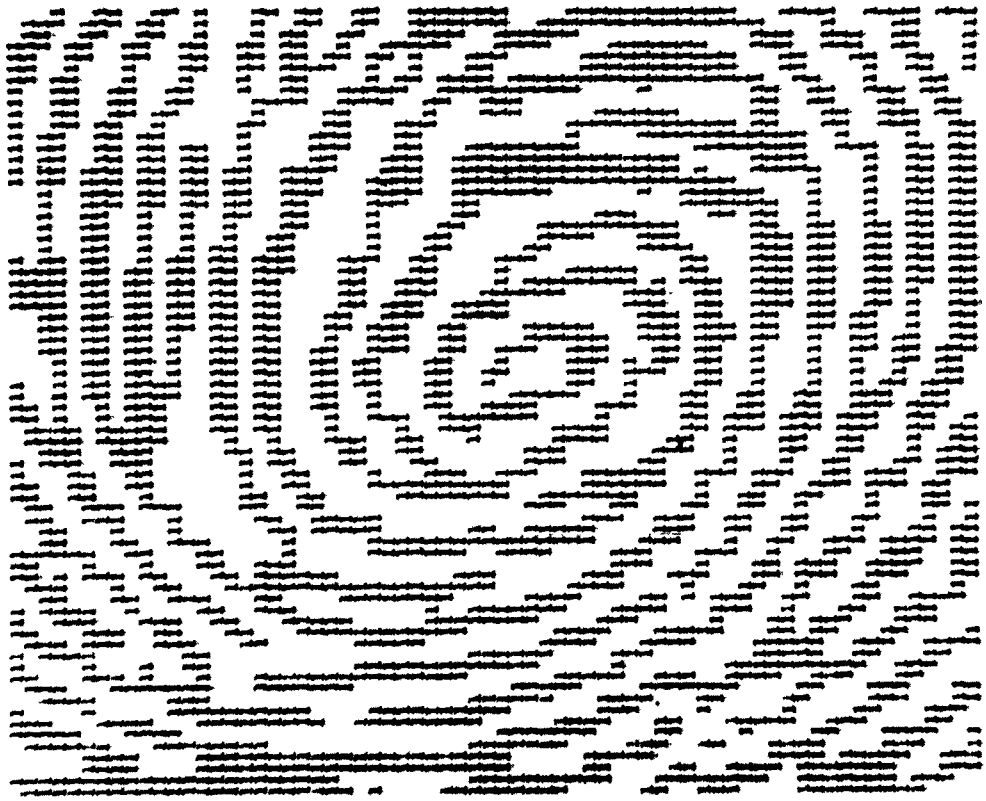

Fig 14

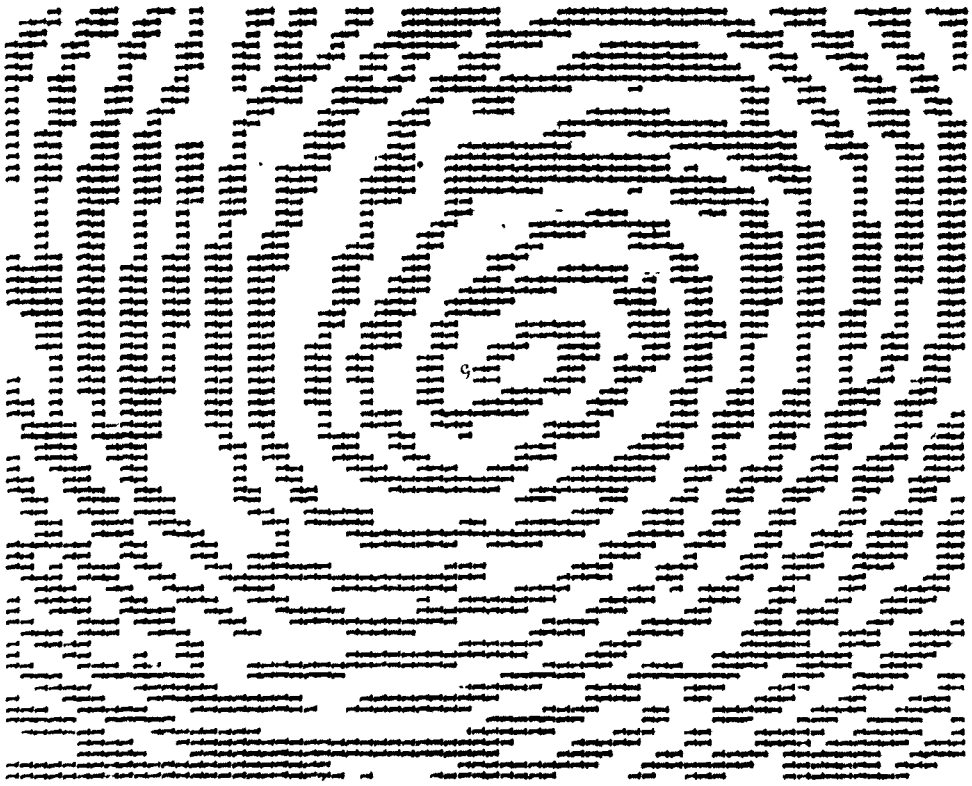

Fig 13

Fig 15

Fig 16

Fig 8a) Lines ② and ③ are below the imaginary horizontal line

For line ①, the search starts from the point I, which is an immediate neighbour of A, and lies on line ① (Fig 8b) The search continues up to 9 elements, not a rigid bound, along the line ①; if all the 9 elements lie above the imaginary horizontal line, algorithm recognizes this as line ① Similarly, for lines ② and ③ search starts from points II and III (Fig. 8b); and if all the 9 elements on each line lie below the imaginary horizontal line, this feature is treated as fork downward The procedures for *FU*, *FHR* and *FHL* follow by symmetry Experimental results of the Method—2 are shown in Fig. 17

Ridge ends can be identified by using the following two conditions (Results not given in this paper)

1 $$\chi = \sum_{k=1}^{8} |R_{(k+1)} - R_{(k)}| = 2,$$

and

2. $$\sum_{k=1}^{8} R_{(k)} = 1$$

4. CLASSIFICATION

Many methods for the automatic classification of fingerprints[1,5,6–8,12,14] are available in the literature Here, a simple algorithm is suggested for basic classification such as

1 Whorl,
2 Loop;
3 Arche,
  a) Arch
  b) Tented Arch (Fig. 18)

4 1 *Algorithm*

In this algorithm, eight neighbours of $X_{(i,j)}$ are divided into four overlapping parts.

Part 0 contains R_3, R_2, R_1
Part 1 contains R_1, R_8, R_7
Part 2 contains R_7, R_6, R_5
Part 3 contains R_5, R_4, R_3

Going round from part 0 to 1, 1 to 2, 2 to 3 and 3 to 0, completes one cycle

Before starting the classification, point $X_{(i,j)}$ is arbitrarily located in the given picture matrix, say $i = 13$ and $j = 7$ If $X_{(i,j)} \neq 1$, then the point $X_{(i,j)}$ moves along the 13th row one step forward That means $X_{(i,j)}$ occupies $X_{(i,j+1)}$th position At this new position, if $X_{(i,j)} = 1$, then the connectivity of $X_{(i,j)}$ with the eight neighbours is tested by the condition

1 $$\sum_{k=1}^{8} R_{(k)} \geq 2$$

As soon as $X_{(i,j)} = 1$ and condition 1 are satisfied, the whole 3×3 submatrix starts first moving upward by considering part 0 In this the algorithm searches for immediate neighbour 1 That means, if 1 presents in any one of the three positions of part 0, $X_{(i,j)}$ occupies that new position If none of the three positions of part 0 contain 1, then part 1 takes the position of part 0 and searches for neighbouring point Again, if part 1 does not contain 1, part 2 will be tested If the same case happens with part 2, part 3 comes into the picture As soon as the next neighbour 1 is found in any one of the four parts, say, nth ($n = 0, 1, 2, 3$), $X_{(i,j)}$ shifts from the present point to this neighbouring point The same procedure is applied from this new location but with part n taking precedence over others If none of the positions of part n contain a 1, the algorithm obviously tests part $n + 1$ mod 4 That is, while tracking a ridge, preference is given to continuation in the same direction.

In case of ridge ends encountered in the tracking, condition 1 fails Then the moving point $X_{(i,j)}$ will jump to its starting point on the ridge that is 13th row Then $X_{(i,j)}$ moves along the 13th row until it finds 1 From this new position, again the above procedure is repeated

Let $X_{I,J}$ be the starting point on the ridge (Fig 9) and $X_{(i,j)}$ be the one moving from point to point while searching neighbouring points To decide the tested print as whorl, following conditions have to be satisfied $X_{(i,j)}$, after going round along the ridge, finally should come to

$$i = I \text{ and } j \gtrless J$$

but final j should be less than J', where J' is the column passing through the core of the Fig. 9

Here, further subclassification of whorl can be achieved as follows

If $j < J$, tested print will be identified as clockwise whorl

If $j > J$, tested print will be identified as anticlockwise whorl

If $X_{(i,j)}$ reaches the $X_{(I,J)}$ after going round along the ridge, the print will be identified as loop

In case of arch and tented arch, the procedure followed is entirely different.

Observing the figures 10(a and b), we can find one simple difference In Fig 10(a) number of 1_s in positions R_2 and R_8 (Fig 10(c), starting from left to right) are more than the total number of 1_s in positions R_3 and R_7 (neglecting the 1_s in position R_1) Similarly, Fig 10(b) contains more 1_s in positions R_3 and R_7 than in positions R_2 and R_8 This point is used in identifying arch and tented arch

For classifying arch and tented arch, again $X_{(I,J)}$ is arbitrarily fixed First of all, the given print will be tested for whorl and loop If it does not belong to any one of them, from the point $X_{(I,J)}$, the point $X_{(i,j)}$ moves to the left side downwards along the curve The movement of $X_{(i,j)}$ will be stopped when

$$\sum_{k=1}^{8} R_{(k)} = 1$$

Fig 17

(a)

(b)

(c i)

(c ii)

Fig 18

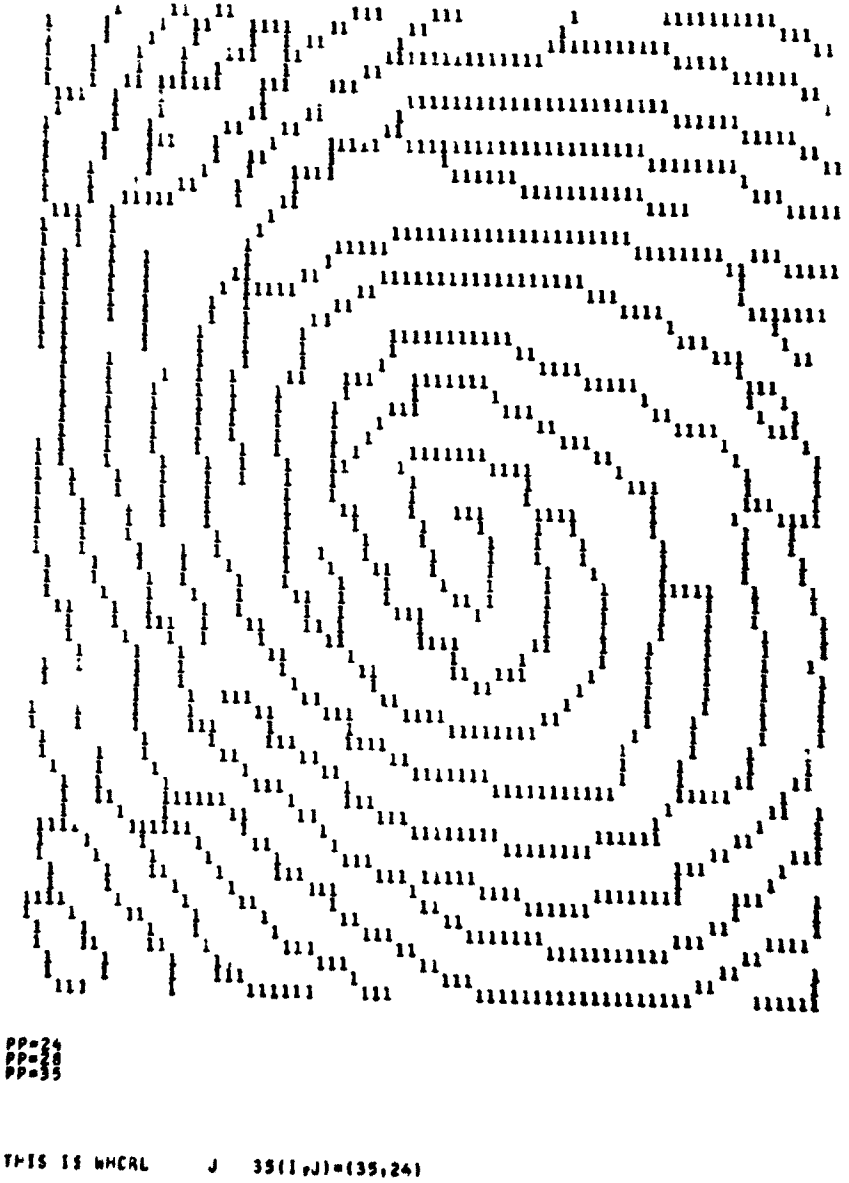

Fig 19

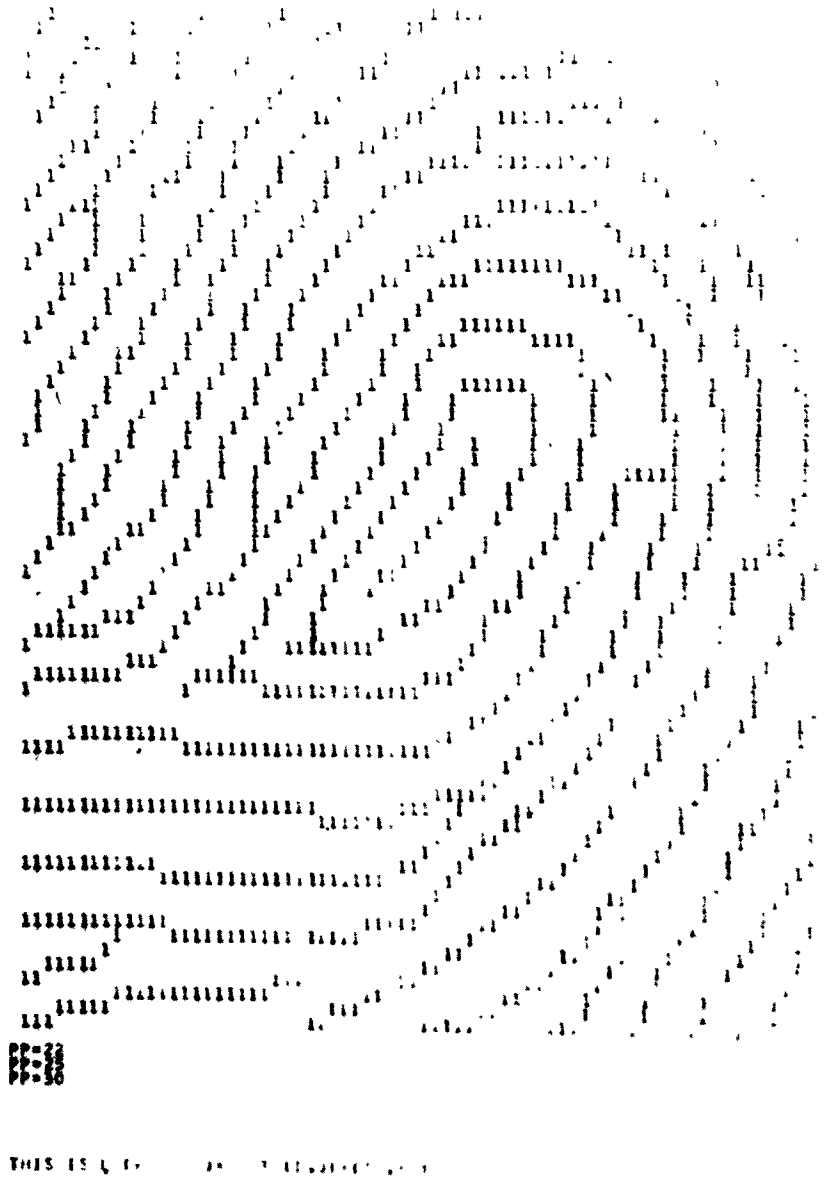

Fig 20

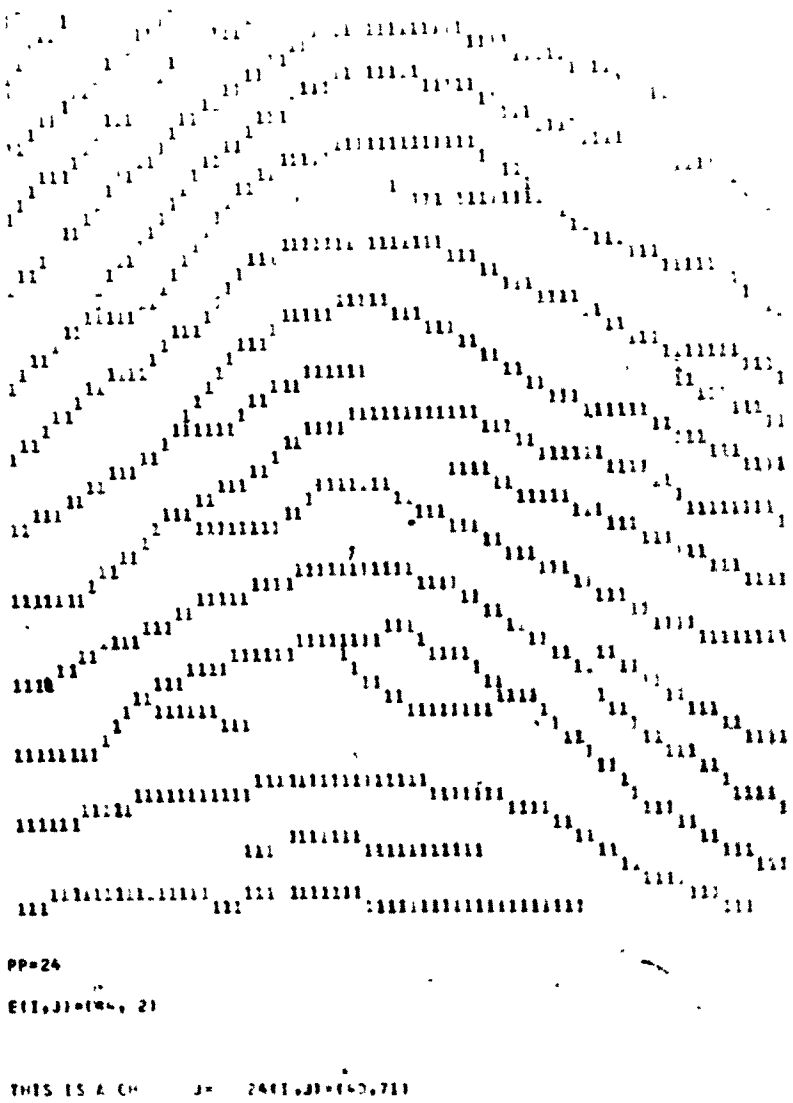

Fig 21

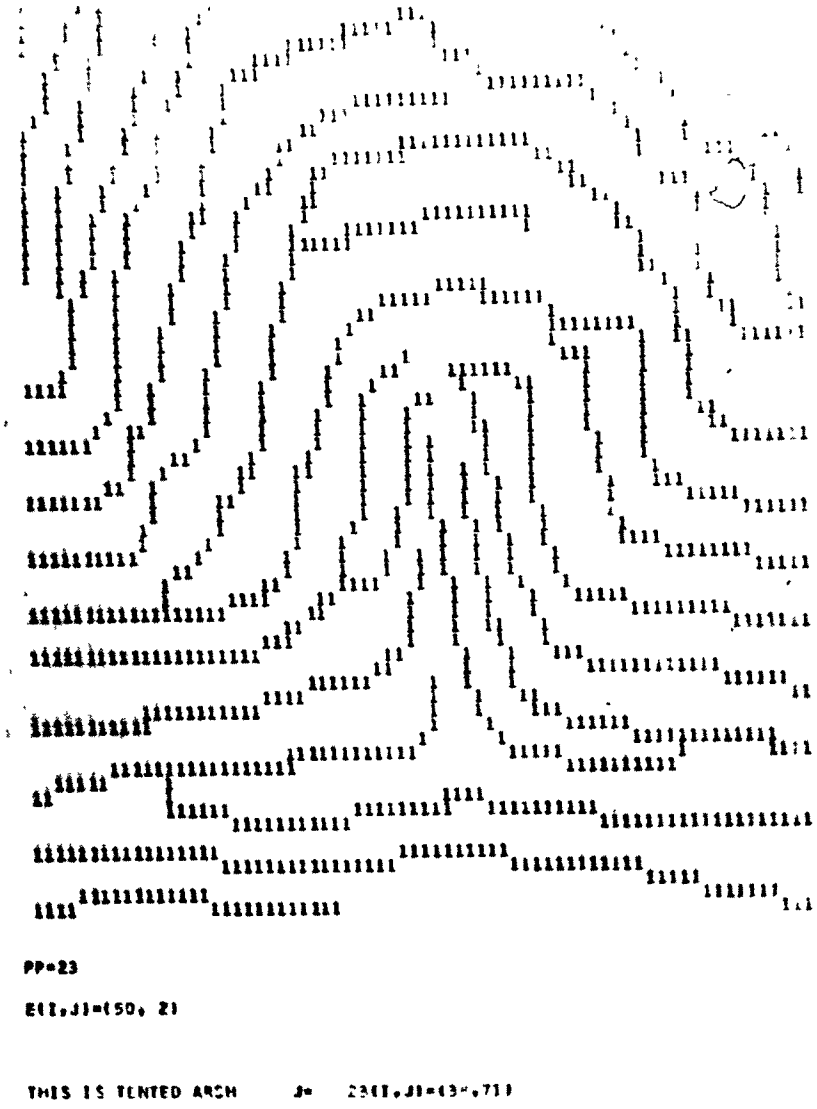

Fig 22

As shown in Fig 18C(a), after satisfying the above condition, $X_{(i,j)}$ started from $X_{(20\ 2)}$ From this point, as $X_{(i,j)}$ proceeds to right along the ridge, the algorithm counts total number of 1_s positioned in R_2 and R_8 and R_3 and R_7 (neglecting 1_s positioned in R_1) Let K be the total number of 1s counted in R_2 and R_8 positions; and L be the total number of 1s counted in positions R_3 and R_7

If $K > L$, tested print will be identified as arch.

If $L > K$, the tested print will be identified as tented arch.

Experimental results of the 4 figures (Fig. 18) are shown at the end. Figures 19–22 are identified as whorl, loop, arch and tented arch respectively with the above algorithm

5. CONCLUSION

The proposed algorithms for preprocessing feature extraction and basic classification have been implemented on IBM 360/44 computer and the results are shown in Figs. 11–22. Figure 18 is generated separately and fed to the computer to test the basic classification. The digitized fingerprint shown in Fig. 11 belongs to the small area 2 × 2 mm around the core of the fingerprint. In the proposed system, this small area around the core is used for basic classification and feature extraction. The coordinates of the core are entered manually in the present system Some of the problems regarding digitized noise and imperfection in the print are solved to some extent However, to improve the proposed system, some more work needs to be done in the problems like automatic identification of the core point, bridging the gaps, and further subclassification to the basic classification of fingerprints.

Acknowledgements—The author wishes to thank P S Moharir for his many helpful suggestions in the course of the preparation of this paper and also wishes to express his thanks to Prof B L Deekshatulu and I S N Murthy for their constant encouragement and suggestions in the organisation of this paper

REFERENCES

1 R Clerici, Fingerprints—A new classification scheme *Nature* **224**, 779 (1969)
2 E S Deutsch, Thinning algorithm on rectangular hexgonal, and triangular arrays *Comm ACM* **15**, (9) 827 (1972)
3 E S Deutsch, Comments on a line thinning scheme, *Compt J* **12**, 412 (1969)
4 W D Freyer & G E Richmond, Two dimensional spatial filtering and computers *Proc Nat Electronics Conf* **18**, p 529 (1962)
5 A Grasselli, On *the Automatic Classification of Fingerprints in Methodologies of Pattern Recognition* (Ed. S Watanabe),-Academic Press, New York (1968)
6 A Grasselli, *Picture Processing, Some Applications in International Conference on Computing Methods in Optimization Problems* (Ed A V Balakrishnan) Academic Press New York (1969)
7 C N Lui & G L Shelton, Computer assisted fingerprints encoding and classification, *IEEE Trans on Man-Machine Systems* **11**, 156–160 (1970)
8 R. Narasimhan & V S N Reddy, A syntax-aided recognition scheme for hand printed English letters Tech Report No 83, Tata Institute of Fundamental Research, Bombay (1970)
9 D Rutovitz, Pattern recognition *J Royal Statist Soc* **129**, 504 (1966)
10 P Saraga & D J Woollons, The Design of operators for pattern processing *IEE, N P L Conf Pattern Recognition* **42**, 106 (1968)
11 R Stefanelli & A Rosenfeld, Some parallel thinning algorithms for digital pictures *JACM* **18**, 2, 255 (1971)
12 J T Tow & W J Hankely, Pictorial pattern recognition automatic interpretation and classification of fingerprints Tech Rept No 4, Dept of Electrical Engng, Ohiostate University, (1970)
13 S Unger, Pattern detection and recognition *Proc IRE* **47**, (1957)
14 J H Wegstein, A semi-automatic single fingerprint identification systems Technical Note 481, National Bureau of Standards, U S Dept of Commerce (1969)

About the Author T CH MALLESWARA RAO received his B E and M Tech degrees in electrical engineering from the Regional Engineering College of Osmania University, Warangal, India in 1969 and 1972 respectively

Mr Malleswara Rao joined the staff of the School of Automation, Indian Institute of Science, Bangalore, India as a project assistant in 1973 Since 1974 he is working in the Centre for Information Processing, Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore, India His present interests are pattern recognition and digital signal processing