# Adaptive flow orientation based feature extraction in fingerprint images*

Nalini K. Ratha, Shaoyun Chen and Anil K. Jain
Department of Computer Science
Michigan State University
East Lansing, MI 48824
{ratha, chenshao, jain} @cps.msu.edu

February 12, 1995

**Abstract**

*A reliable method for extracting structural features from fingerprint images is presented. Viewing fingerprint images as a textured image, an orientation flow field is computed. Rest of the stages in the algorithm use the flow field to design adaptive filters for the input image. To accurately locate ridges, a waveform projection-based ridge segmentation algorithm is used. The ridge skeleton image is obtained and smoothed using morphological operators to detect the features. A large number of spurious features from the detected set of minutiae is deleted by a postprocessing stage. The performance of the proposed algorithm has been evaluated by computing a "goodness index" (GI) which compares the results of automatic extraction with manually extracted ground truth. The significance of the observed GI values is determined by comparing the index for a set of fingerprints against the GI values obtained under a baseline distribution. The detected features are observed to be reliable and accurate.*

**Keywords: Fingerprints, feature extraction, texture, flow orientation, minutiae, segmentation, skeleton**

*Research supported by a contract from the Institute for Defense Analyses.

# 1 Introduction

Fingerprint matching is the most popular biometric technique used in automatic personal identification [1]. Law enforcement agencies use it routinely for criminal identification. Now, it is also being used in several other applications such as access control for high security installations, credit card usage verification, and employee identification [1]. The main reason for the popularity of fingerprints as a form of identification is that the fingerprint of a person is unique and remains invariant with his/her age. The law enforcement agencies have developed a standardized method for manually matching rolled fingerprints and latent or partial fingerprints (lifted from the scene of crime). However, the manual matching of fingerprints is a highly tedious task for the following reasons. As the features used for matching are rather small in size compared to the image size, a human expert often has to use a magnifying glass to get a better view of the fingerprint impression. The fingerprint matching complexity is a function of the size of the image database, which can vary from a few hundred records to several million records. Even though the standard Henry formula [2] for fingerprint recognition can be used to cut down the search time, manual matching can still take several days in some cases. These problems can be easily overcome by automating the fingerprint-based identification process.

An automatic fingerprint identification system (AFIS) consists of various processing stages as shown in Figure 1. For the purpose of automation, a suitable representation (feature extraction) of fingerprints is essential. This representation should have the following desirable properties:

1. Retain the discriminating power (uniqueness) of each fingerprint at several levels of resolution (detail).

2. Easily computable.

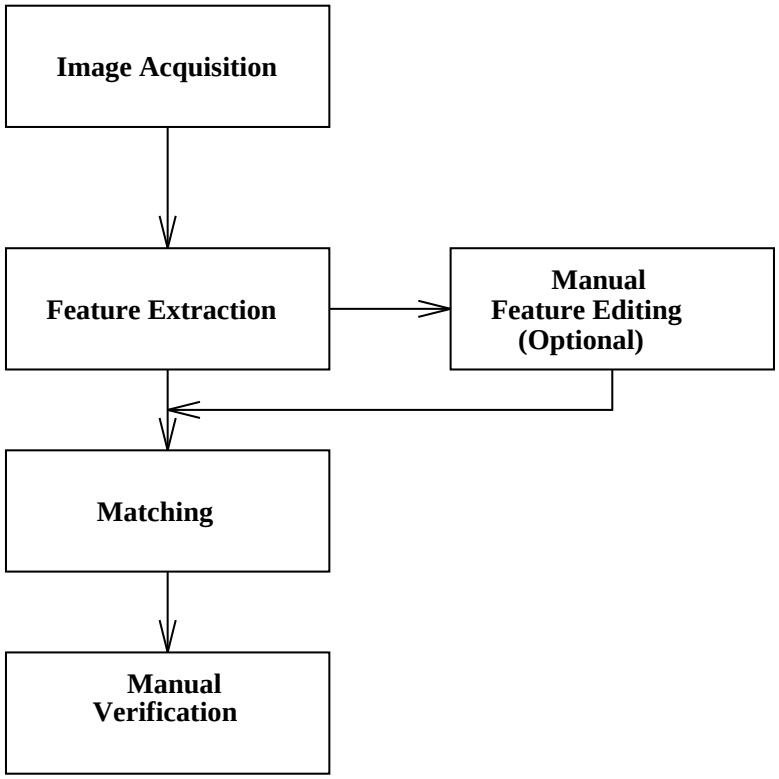



Figure 1: Stages in an AFIS

3. Amenable to automated matching algorithms.

4. Stable and invariant to noise and distortions.

5. Efficient and compact representation.

The compactness property of representation often constrains its discriminating power. Clearly, the raw digital image of a fingerprint itself does not meet these representational requirements. Hence, high-level structural features are extracted from the image for the purpose of representation and matching.

The ridges and valleys in a fingerprint alternate, flowing in a local constant direction (see Figure 2). A closer analysis of the fingerprint reveals that the ridges (or the valleys) exhibit anomalies of various kinds, such as ridge bifurcations, ridge endings, short ridges, and ridge crossovers. Eighteen

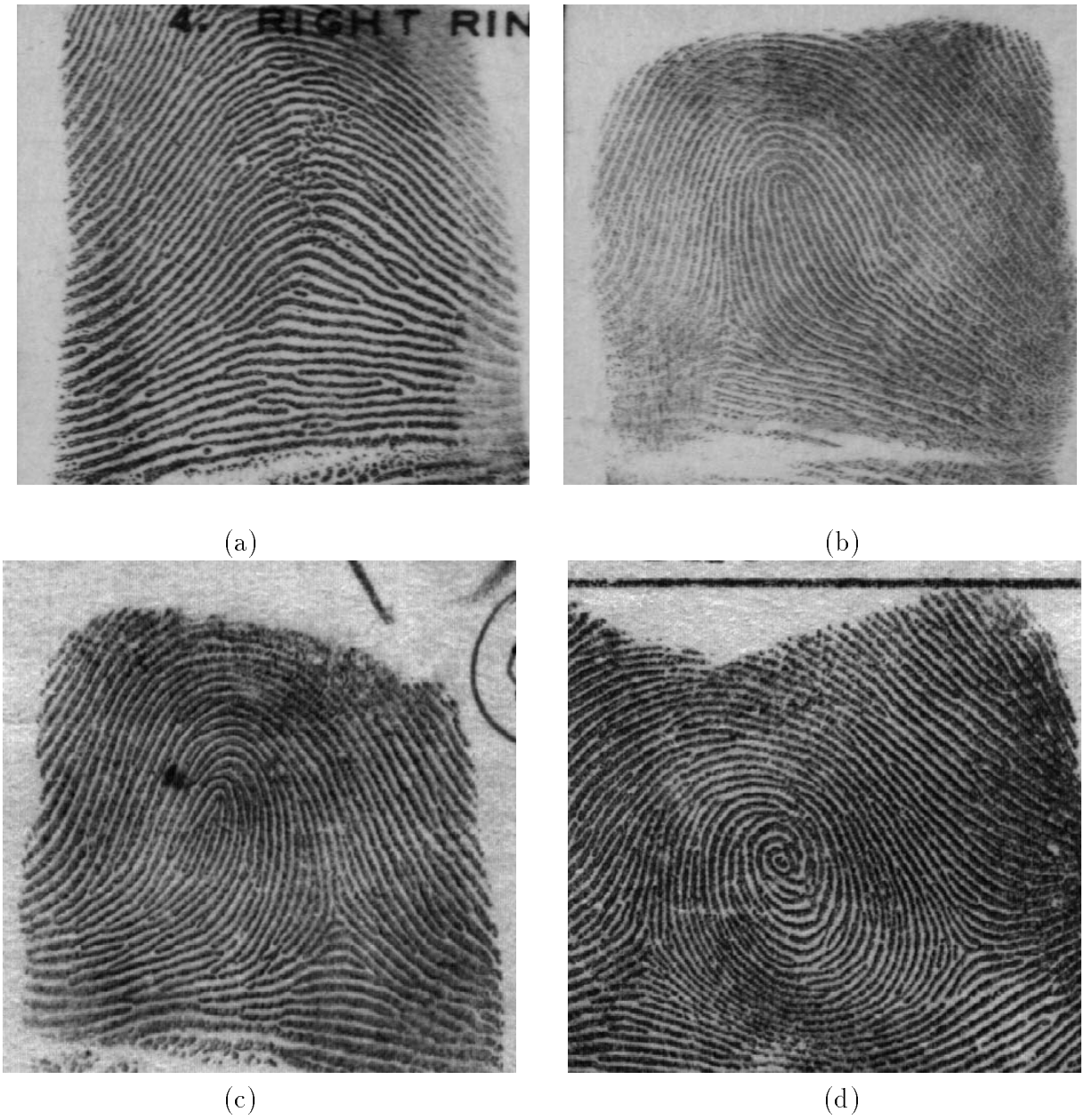



Figure 2: Gray level fingerprint images of different types of patterns: (a) Arch; (b) Left loop; (c) Right loop; (d) Whorl.

different types of fingerprint features have been enumerated in [3]. Collectively, these features are called *minutiae*. For automatic feature extraction and matching, the set of fingerprint features is restricted to two types of minutiae: *ridge endings* and *ridge bifurcations*. Ridge endings and bifurcations are shown in Figure 3. We do not make any distinction between these two feature types

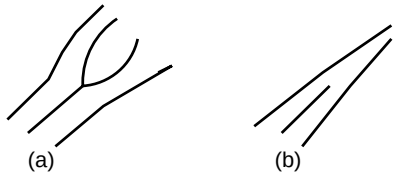



Figure 3: Two commonly used fingerprint features: (a) Ridge bifurcation; (b) Ridge ending.

since various data acquisition conditions such as inking, finger pressure, and lighting conditions can easily change one type of feature into another. More complex fingerprint features can be expressed as a combination of these two basic features. For example, an *enclosure* can be considered as a collection of two bifurcations and a *short ridge* can be considered as a pair of ridge endings as shown in Figure 4. In a good quality rolled fingerprint image, there are about 70 to 80 minutiae points and in a latent fingerprint the number of minutiae is much less (approximately 20 to 30).

Commercially available fingerprint identification systems typically use ridge bifurcations and ridge endings as features. Because of the large size of the fingerprint database and the noisy

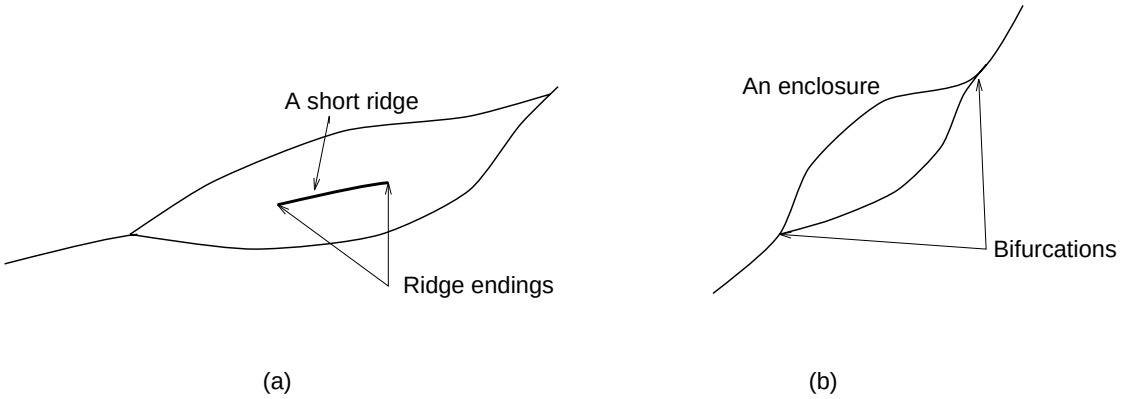



Figure 4: Complex features as a combination of simple features: (a) Short ridge; (b) Enclosure.

fingerprints encountered in practice, it is very difficult to achieve a reliable one-to-one matching in all the test cases. Therefore, the commercial systems provide a ranked list of possible matches (usually the top ten matches) which are then verified by a human expert. Details of commercial fingerprint recognition systems from NEC, PRINTRAK, and MORPHO are presented in [2].

One of the main problems in extracting structural features is due to the presence of noise in the fingerprint image. Commonly used methods for taking fingerprint impressions involve applying an uniform layer of ink on the finger and rolling the finger on paper. This causes the following types of problems: (i) over-inked areas of the finger create smudgy areas in the image, (ii) breaks in ridges are created by under-inked areas, and (iii) the skin being elastic in nature can change the positional characteristics of the fingerprint features depending upon the pressure being applied on the fingers. Although inkless methods for taking fingerprint impressions are now available, these methods still suffer from the positional shifting caused by the skin elasticity. The non-cooperative attitude of suspects or criminals also leads to smearing in parts of the fingerprint impressions. Thus, a substantial amount of research reported in the literature on fingerprint identification is devoted to image enhancement techniques.

This paper proposes a reliable method for feature extraction from fingerprint images. The matching stage uses the position and orientation of these features, and the total number of such features. As a result, the accuracy of feature extraction is crucial in the overall success of fingerprint matching. Reliable and robust features can make matching algorithms simpler, and the manual verification stage redundant. The orientation field of the input gray level fingerprint image plays an important role in our algorithms to design adaptive filters. A new method, based on projection of the image in the direction of the orientation field, for segmenting the ridges from the fingerprint image is described. The quality of the extracted features is evaluated quantitatively.

The rest of the paper is organized as follows. In Section 2, we briefly summarize the previous work reported in the literature. The details of our feature extraction algorithm which includes the segmentation of ridges, minutia extraction and feature postprocessing are presented in Section 3. We have tested our algorithm on a number of input images. The results of feature extraction using the proposed algorithm are analyzed in Section 4. A quantitative method to evaluate the performance of the algorithm is also presented in Section 4. Our conclusions and plans for future work are described in Section 5.

## 2 Background and Related work

The structural features which are commonly extracted from the gray level input fingerprint image are the ridge bifurcations and ridge endings. Each of the two features has three components, namely, the x-coordinate, the y-coordinate, and the local ridge direction at the feature location as shown in Figure 5. Many other features have been derived from this basic three-dimensional feature vector [4].

Current research in the design of AFIS research is being carried out in the following areas: (i) preprocessing, (ii) feature extraction, (iii) matching algorithms, (iv) compression of fingerprint images, and (v) special-purpose architectures for real-time feature extraction and matching. A substantial amount of research in fingerprint analysis has been reported in the literature. We provide a brief survey of some recently published literature in preprocessing and feature extraction. A brief history of automation efforts in fingerprint recognition is available in [2]. The papers by Sherlock et al. [5], Mehtre [6], Coetzee and Botha [7], Hung [8], Xiao and Raafat [9], O'Gorman and Nickerson [10] are relevant to our proposed feature extraction scheme.

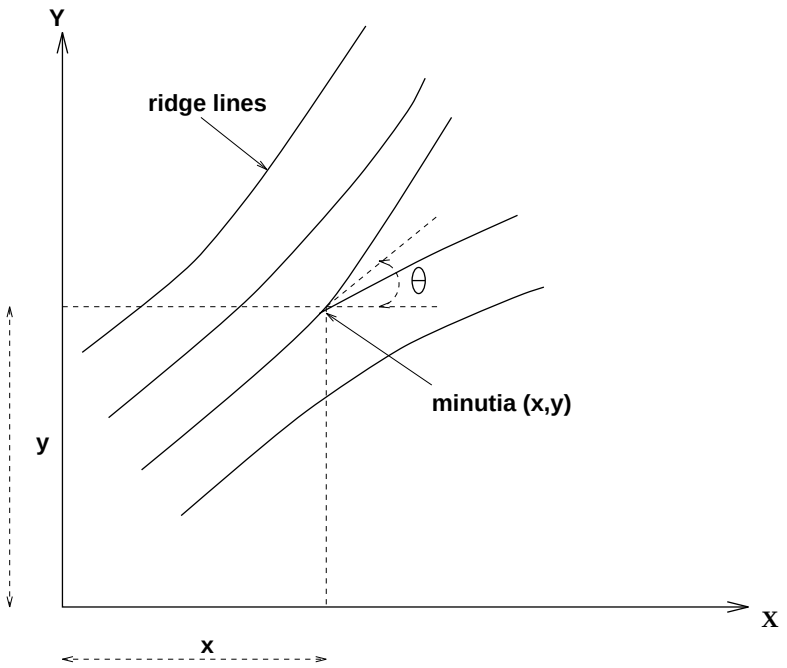



Figure 5: Components of a minutiae feature.

Sherlock et al. [5] enhance fingerprint images by a directional Fourier filtering. The direction of the filtering is decided by the local ridge orientation. A $32 \times 32$ window is used to obtain a projection of the pattern in 16 directions. The projection with the maximum variance is the desired ridge direction for the window. The result of the enhancement is compared with feature extraction techniques used in a system currently used by the UK Home office. Performance evaluation is carried out by comparing features obtained with the enhancements proposed by this method with the features obtained using the available software in the Home office system. Mehtre [6] computes the directional image, representing the local ridge direction, in a block of size $16 \times 16$ pixels. For this purpose, local gray level intensity variances along eight different directions are computed. The direction with the least variance is the desired ridge direction. A set of eight $7 \times 7$ convolution masks is applied to the input image for ridge enhancement. The fingerprint area is segmented from the background before applying standard locally adaptive thresholding and thinning operators.

Features are obtained based on the computation of the connection number (CN) described in [11]. A post-processing stage based on a set of heuristics eliminates the spurious minutiae.

Coetzee and Botha [7] obtain the ridges by using the Marr-Hildreth edge operator. This edge map along with the gray scale image is used to binarize the fingerprint image. The thresholded image is smoothed before applying the thinning operation. The directional image is computed in a fashion similar to the one described in [6]. No feature extraction stage is described. Xiao and Raafat [9] assume that the skeleton image has already been derived from the fingerprint images. They describe methods to identify spurious minutiae and eliminate them using the structural definition of minutiae. For each minutia, statistics of ridge width and ridge attributes such as ridge length, ridge direction and minutiae direction are used to decide the spurious minutiae. Hung [8] enhances fingerprint images by equalizing the ridge widths. The input image is assumed to be a binary image. Directional enhancement of ridges is done after estimating the local direction in a small window using a method similar to the one in [6]. The enhancement process has two steps: (i) direction-oriented ridge shrinking, followed by (ii) direction-oriented ridge expanding. The skeleton of the enhanced image is obtained by Baja's algorithm. This paper also describes methods for detecting bridges and breaks as separate features.

The main theme of O'Gorman and Nickerson's [10] work is to design filters for fingerprint image enhancement. The $k \times k$ mask coefficients are generated based on the local ridge orientation. Only three orientation directions are used. Four model parameters derived from ridge width ($W_{max}$, $W_{min}$), valley width ($\overline{W}_{max}$, $\overline{W}_{min}$), and minimum radius of curvature are used to describe a fingerprint. It is assumed that $W_{max} + W_{min} = \overline{W}_{max} + \overline{W}_{min}$. The mask is convolved with the input image. The enhanced image is binarized and postprocessed. An application-specific integrated circuit (ASIC) has been designed to meet the computing requirements of this algorithm. No

description of feature extraction or postprocessing is given.

To summarize, most of the published approaches for feature extraction use local ridge directions and a locally adaptive thresholding method. To improve fingerprint image quality, directional ridge enhancement is also commonly employed. The thinning step involves a standard operator. Few published papers describe a methodology to evaluate the performance of image enhancement and feature extraction stages. Often, only portions of the overall feature extraction module are implemented. Various approaches described in the literature can be compared based on the following factors:

1. Does the method describe all the stages of feature extraction?
2. What kind of input does it handle?
3. How is the local ridge direction computed?
4. What are the preprocessing/enhancement steps?
5. What is the binarization/segmentation approach?
6. Which thinning operator is used?
7. Is there a postprocessing stage?
8. Does the system describe a performance evaluation methodology?

Table 1 contains a comparison of the six papers reviewed earlier with reference to our approach. The first column in the table describes the features used in the comparison. The next six columns describe the scheme adopted in each of the six papers. The last column describes the proposed method.

| Method / Factor | Coetzee [7] | Hung [8] | Mehtre [6] | O'Gorman [10] | Sherlock [5] | Xiao [9] | Proposed method |
|---|---|---|---|---|---|---|---|
| All stages | Yes | No | Yes | No | No | No | Yes |
| Input Image | Grayscale | Binary | Grayscale | Grayscale | Grayscale | Skeleton | Grayscale |
| Direction Computation | Variance Based | Sum of local directions of 6 neighbors in one of the 16 directions | Intensity variance in 8 directions | N.D. | Project along 16 directions and pick maximum variance direction | N.A. | Orientation field based |
| Preprocessing | Marr-Hildreth edge operator | Equalize ridge widths | Special $7\times7$ directional convolution masks | Ridge shrinking and ridge expanding | Directional Fourier filters | N.A. | Gaussian directed along the block direction |
| Binarization | Locally adaptive | N. A. | Locally adaptive | Locally Adaptive | Locally Adaptive | N.A. | Projection based |
| Thinning | Not used | Arcelli and Baja's algorithm | Standard | N. D. | N. D. | N. A. | HIPS package supported |
| Postprocessing | Not used | N. D. | Yes | N.D. | N.D. | Ridge direction, ridge length based | Yes |
| Performance evaluation | No | No | No | No | Yes | No | Yes, with ground truth |

Note: N. D. - Not described in the paper; N.A. - Not applicable.

Table 1: Comparative Analysis of selected feature extraction approaches.

While our approach uses many of the well-known ideas proposed in the earlier studies, the ridge flow orientations form the basis for adapting parameters in all the stages of our feature extraction algorithm. We also propose a technique for performance evaluation of the feature extraction process by computing a goodness index (GI) with reference to the ground truth for a set of one hundred fingerprint images.

# 3 Proposed Algorithm

The salient features of our approach for feature extraction can be described as follows. We view a fingerprint image as a flow pattern with a definite texture. An orientation field for the flow texture is computed [12]. To accurately determine the local orientation field, the input image is divided into equal-sized blocks (windows) of $16 \times 16$ pixels. Each block is processed independently. The gray level projection along a scanline perpendicular to the local orientation field provides the maximum variance. We locate the ridges using the peaks and the variance in this projection. The ridges are thinned and the resulting skeleton image is enhanced using an adaptive morphological filter. The feature extraction stage applies a set of masks to the thinned and enhanced ridge image. The postprocessing stage deletes noisy feature points. The schematic flow of the proposed feature extraction algorithm is shown in figure 6. The overall process can be divided into three main operations; (i) preprocessing and segmentation, (ii) thinning and feature extraction, and (iii) postprocessing. The details of various stages in feature extraction are described in this section.

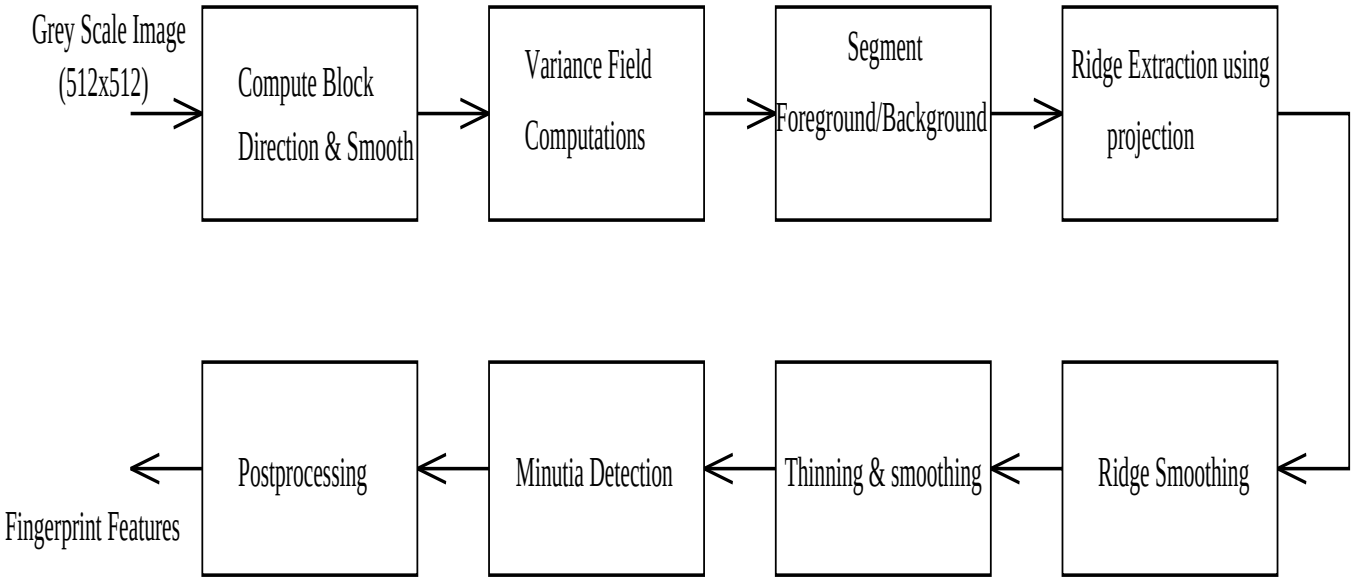



Figure 6: Stages in the proposed feature extraction algorithm.

## 3.1 Preprocessing and Segmentation

The purpose of preprocessing and segmentation is to obtain a binary segmented fingerprint ridge image from an input grayscale fingerprint image, where the ridges have a value '1' (white) and rest of the image has value '0'. This is achieved through the following four steps: (i) computation of orientation field, (ii) foreground/background separation, (iii) ridge segmentation, and (iv) directional smoothing of ridges.

### 3.1.1 Computation of orientation field

Fingerprint images can be considered as an oriented texture pattern. As per the taxonomy described in [12], fingerprints can be classified as a weakly-ordered texture. The orientation field [12] is used to compute the optimal dominant ridge direction in each $16 \times 16$ window or block. Following steps are involved in the computation of the orientation field for each window.

1. Compute the gradient of the smoothed block. Let $G_x(i,j)$ and $G_y(i,j)$ be the gradient magnitude in $x$ and $y$ directions, respectively, at pixel $(i,j)$ obtained using $3 \times 3$ Sobel masks.

2. Obtain the dominant direction in a $16 \times 16$ block using the following equation:

$$\theta_d = \frac{1}{2} tan^{-1} \left( \frac{\sum_{i=1}^{16} \sum_{j=1}^{16} 2G_x(i,j)G_y(i,j)}{\sum_{i=1}^{16} \sum_{j=1}^{16} (G_x(i,j)^2 - G_y(i,j)^2)} \right), G_x \neq 0 \; and \; G_y \neq 0 \quad (1)$$

Note that if either $G_x$ or $G_y$ is zero then the estimate of the dominant direction is trivial ($0^o$ or $90^o$). The angle $\theta_d$ is quantized into 16 directions. The orientation field obtained using this method is shown in Figure 7. The orientation field serves to select the parameters of adaptive filters in subsequent stages. Ridge directions have also been used in deciding the pattern class of the input fingerprint image [13].

### 3.1.2 Foreground/Background Segmentation

A fingerprint image usually consists of a region of interest (ridges and valleys of fingerprint impressions) along with a printed rectangular bounding box, smudgy patches of ink, and blurred areas of the pattern and background. We need to segment the fingerprint area (foreground) to avoid extraction of features in noisy and background areas of the fingerprint. We compute the variance of gray levels in a direction orthogonal to the orientation field in each block. The underlying assumption is that the noisy image regions have no directional dependence, where as regions of interest (fingerprint area) exhibit a very high variance in a direction orthogonal to the orientation of the pattern and a very low variance along the ridges. In other words, the background has low variance in all the directions. Since our computation of the orientation field is quite robust, we use this

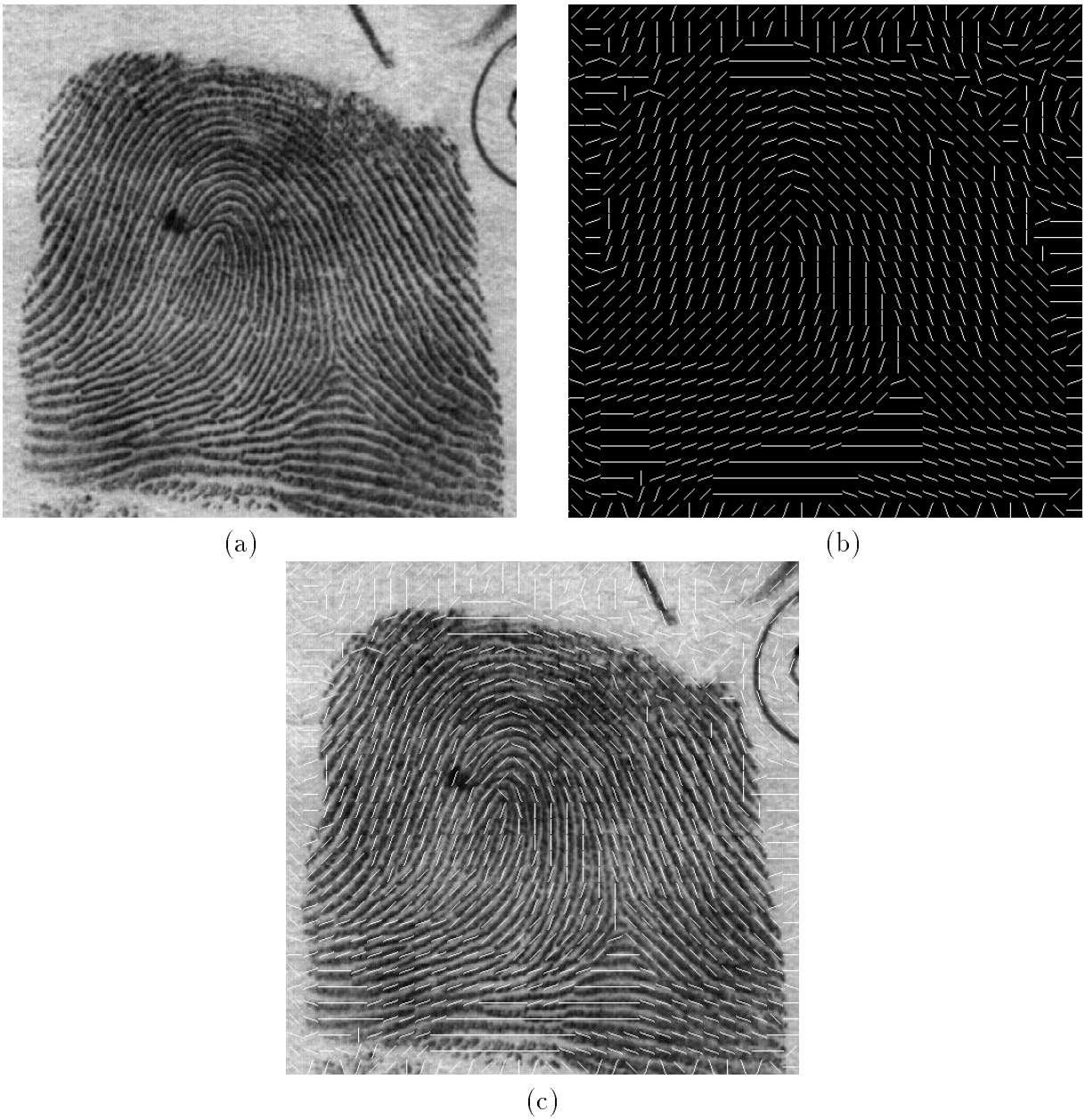


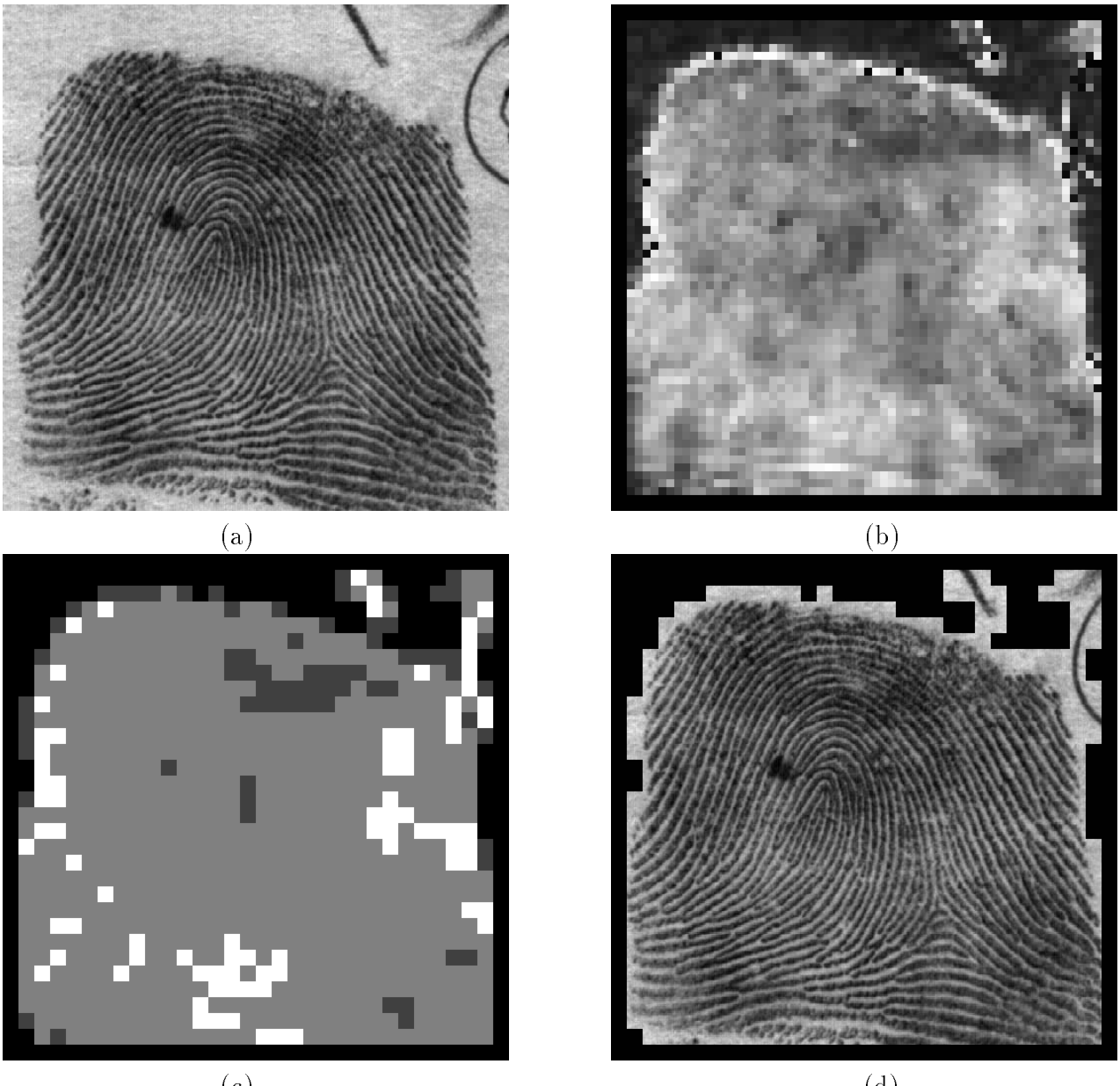
Figure 7: Computation of orientation field; (a) input fingerprint image ($512 \times 512$); (b) orientation field (for each $16 \times 16$ window); (c) orientation field superimposed on the input image.

information directly in the segmentation process. Mehtre [6] uses the variance at every pixel in a set of known directions to decide if the pixel is in the foreground.

The variance can also be used to decide the 'quality' of the fingerprint image in terms of the image contrast of the block ($16 \times 16$ subimage) under consideration. The quality field value for a window is defined to have one of the following four values: "good", "medium", "poor", and "background". A high contrast area gets the value "good" and a low contrast area is assigned the value "poor". This quality field is used in performance evaluation of our feature extraction algorithm. The variance field for the input image in Figure 8(a) is shown in Figure 8(b) and the corresponding quality field is shown in Figure 8(c). The segmented image is shown in 8(d). A high gray value in the quality field image implies a better quality of that image block. We assign the same quality value to all the pixels in a block. Figure 8(d) shows that our algorithm eliminates most of the background in the fingerprint image.

### 3.1.3 Ridge Segmentation

After the foreground and background regions have been identified, the next step is to locate the ridges. A new technique has been implemented to locate the ridges. Consider an image window (in our case $16 \times 16$ pixels) and its projection in the direction orthogonal to the orientation field for the window. A ridge center maps itself as a peak in the projection. The projection waveform facilitates the detection of ridge pixels. Two neighboring pixels on either side of the peak are also retained along the direction perpendicular to the orientation field. For an ideal model of the ridges as shown in Figure 9(a), we should get a projection waveform shown in Figure 9(b). The waveform for a $16 \times 16$ window of a real fingerprint image (Figure 9(c)) is shown in Figure 9(d). Before projecting the image, the image is smoothed using a 1-dimensional averaging mask on each line oriented along



(a) (b)

(c) (d)

Figure 8: Foreground/background segmentation: (a) original image; (b) variance field; (c) quality image; (d) segmented image.

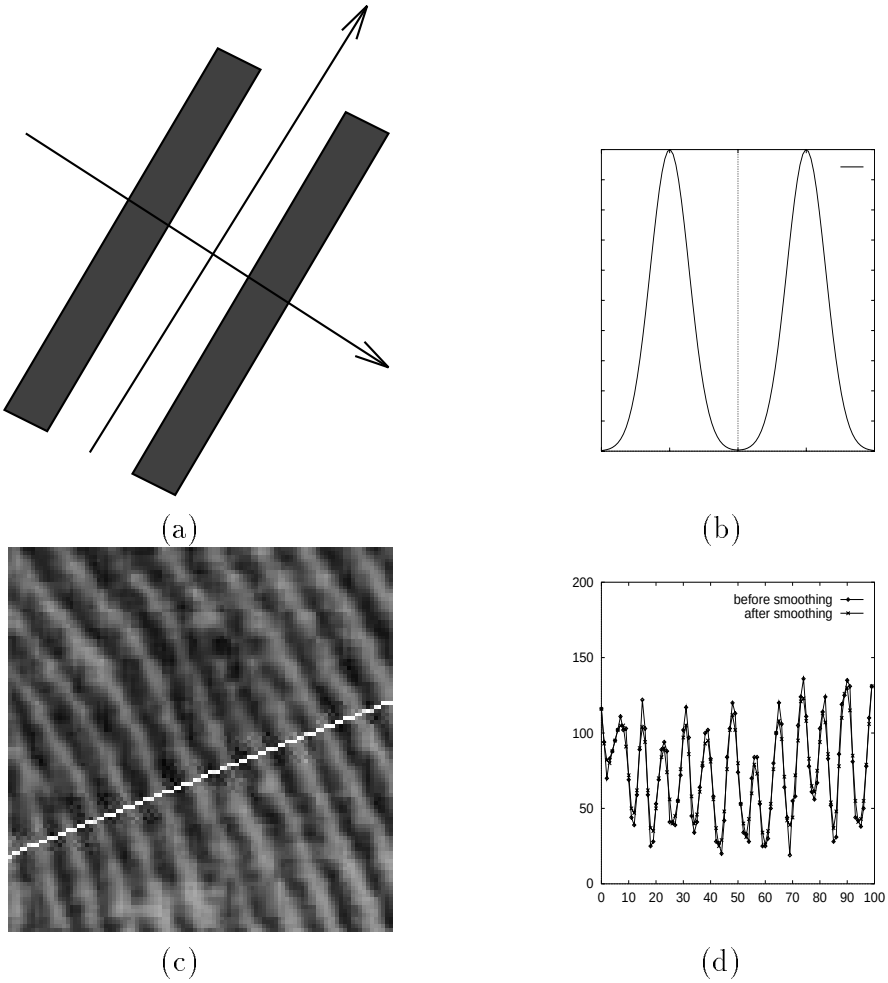
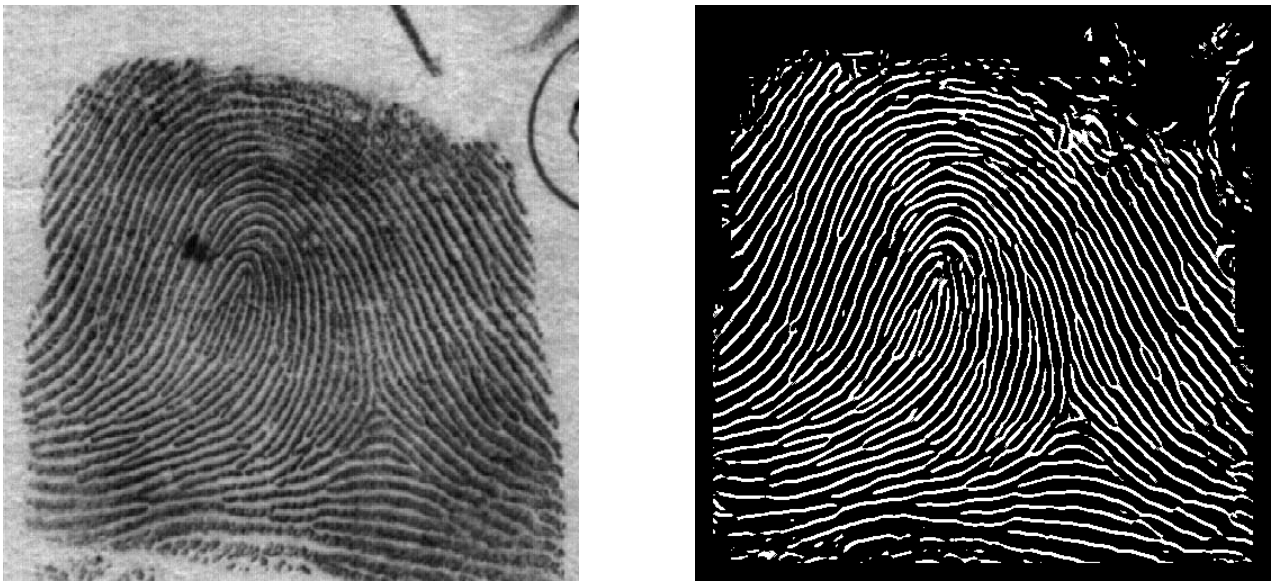



Figure 9: Ridge segmentation: (a) ideal model for ridges; (b) projection waveform for the ridges in (a); (c) a typical $16 \times 16$ window from a fingerprint image. Also shown is the axis orthogonal to the ridge direction; (d) projection waveform for ridges in (c).

a direction orthogonal to the orientation field of the window.

Sherlock et al. [5] used several different projections to determine the local ridge orientation where as we use the orientation field to obtain a single projection. A commonly used technique for segmentation is to threshold the image to obtain a binary image [6]. A fuzzy thresholding algorithm for locating ridges in fingerprints was proposed in [14]. The thresholding technique uses a large window to ensure that at least one ridge and one valley is included in a window at every pixel, but does not use the directional information. Our approach is much better than a locally



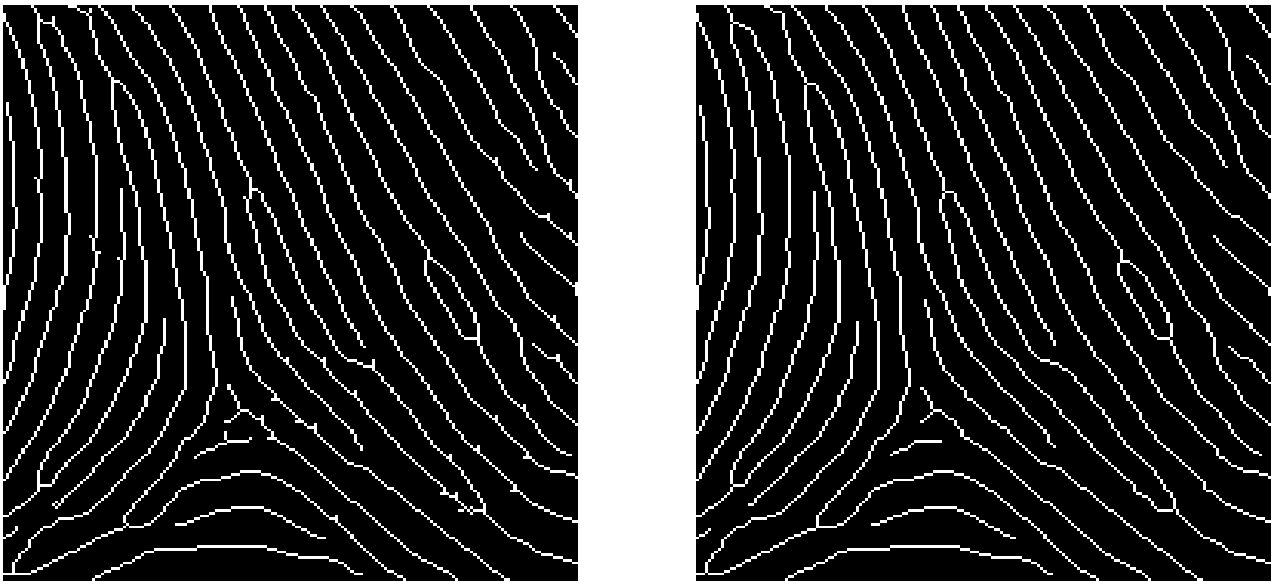
(a) (b)

Figure 10: Segmented ridges: (a) input fingerprint image (same as Figure 1(b)); (b) identified ridges.

adaptive thresholding scheme both in terms of the computational efficiency and performance by appropriately using the orientation field. The ridge pixels are assigned a value '1' (white) and the remaining pixels are assigned a value '0'. Figure 10 shows that the resulting binary ridge image is very good for feature extraction purposes even though the full widths of the ridges are not retained.

### 3.1.4 Directional Smoothing

Once the ridges are located, directional smoothing is applied to smooth the ridges. A $3 \times 7$ mask is placed along the orientation field for each window. The mask containing all '1's enables us to count the number of '1's in the mask area. If the count of '1's is more than 25% of the total number of pixels (21 in this case), then the ridge point is retained. The size of the mask was determined empirically.



(a) (b)

Figure 11: Thinned ridges: (a) before spike removal; (b) after spike removal.

## 3.2 Minutiae Extraction

The binary ridge image needs further processing before the minutiae features can be extracted. The first step is to thin the ridges so that they are single-pixel wide. A skeletonization method described in [15] and available in the HIPS library [16] is used. Unfortunately, the ridge boundary aberrations have an adverse impact on the skeleton, resulting in "hairy" growths (spikes) which lead to spurious ridge bifurcations and endings. Hence, the skeleton needs to be smoothed before minutiae points can be extracted. The spikes are eliminated using an adaptive morphological filtering. The filter used is a binary "open" operator with a box-shaped structuring element with all '1's of size $3 \times 3$. The structuring element is rotated in the direction orthogonal to the orientation field in the window. The ridge skeletons before spike removal and after spike removal are shown in Figure 11.

Locating minutia points in the thinned (skeleton) image is relatively easy. A count of the number of "on" neighbors at a point of interest in a $3 \times 3$ window is sufficient for this purpose; this is similar to the connection number described in [11]. A ridge end point has only one neighbor in the window

and a ridge bifurcation has at least three neighbors. All the ridge end points and ridge bifurcation points detected with this method are not always true features, but the method does seem to identify most of the true feature points. A postprocessing stage filters out the undesired feature points based on their structural characteristics.

## 3.3 Postprocessing

The preprocessing stage does not eliminate all possible defects in the input gray scale fingerprint image. For example, ridge breaks due to insufficient amount of ink and ridge cross-connections due to overinking are not totally eliminated. In fact, the preprocessing stage itself occasionally introduces some artifacts which later lead to spurious features. The postprocessing stage eliminates spurious feature points based on the structural and spatial relationships of the minutiae. For instance, two minutiae in a real fingerprint cannot occur within a very short distance of each other. The following heuristics are used to validate minutia points found in section 3.2.

1. Ridge break elimination: Two end points with the same orientation and within a distance threshold $T_1$ are eliminated.

2. Spike elimination: An end point which is connected to a bifurcation point and is also within a distance threshold $T_2$ is eliminated.

3. Boundary effects: The minutiae detected within a specified border of the boundary of the foreground areas are deleted.

The connectivity property contained in the second heuristics is verified by tracking the ridges starting from a ridge bifurcation. The tracking directions are shown in Figure 12.

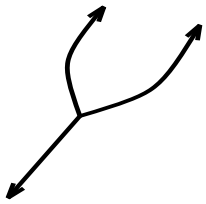


Figure 12: Three ridge tracking directions.

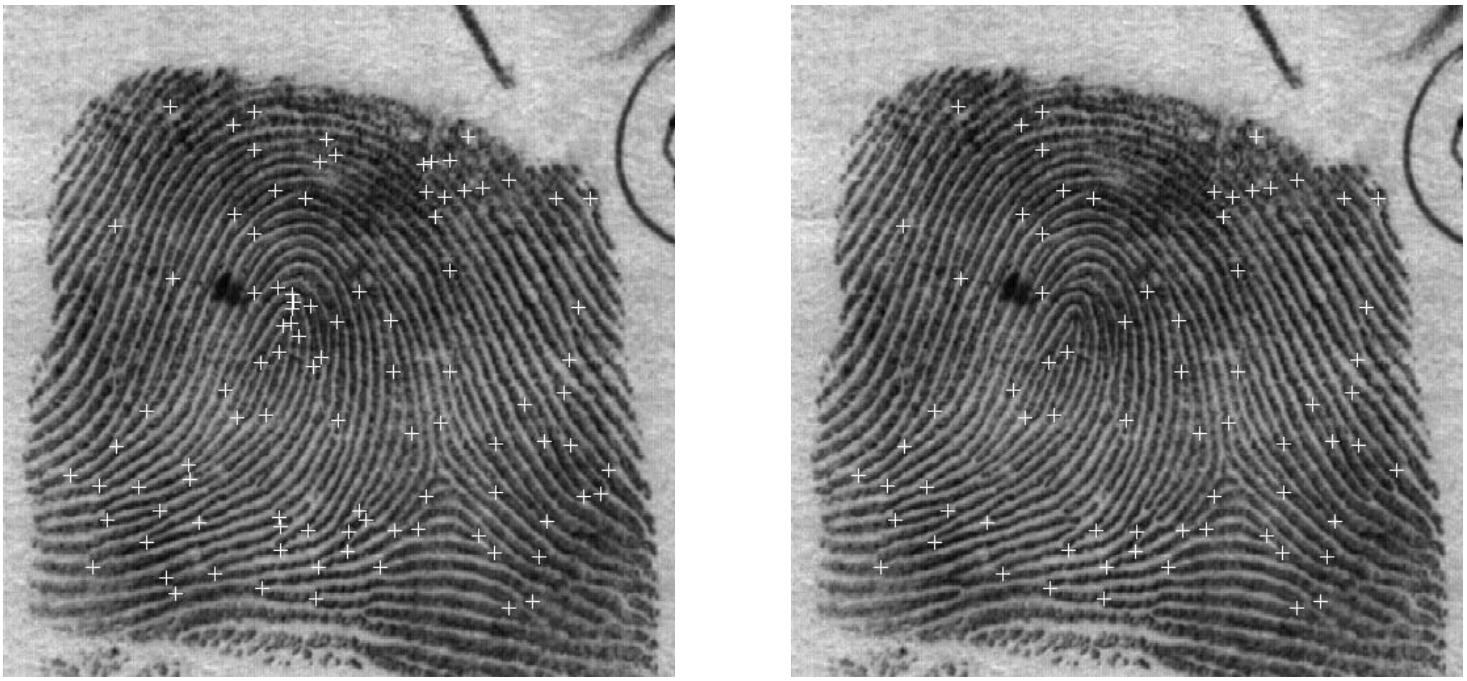


Figure 13: Extracted minutiae points: (a) before postprocessing; (b) after postprocessing.

A large number of spurious minutiae get deleted with these rules. The feature points detected before postprocessing and after postprocessing are shown in Figure 13. The number of feature points in Figures 13(a) and 13(b) are 97 and 71, respectively.

## 3.4 Algorithmic Parameters

In our implementation, we have used the following parameters in the various stages described in the previous sections:

1. Window (block) size: $16 \times 16$ pixels, resulting in 1,024 total windows in an input image of size $512 \times 512$.

2. Number of quantized directions in orientation field: 16.

3. Ridge smoothing mask size: $7 \times 3$.

4. Threshold on sum of variances in a window to decide background/foreground: 2,500.

5. Size of the structuring element: $3 \times 3$.

6. Parameters in postprocessing: $T_1$=10, $T_2$= 15, border size=32.

# 4 Experimental Results

The feature extraction algorithm described above has been implemented and tested on 100 fingerprint images of varying quality. The results of intermediate stages and the detected minutiae features for a typical fingerprint are shown in Figure 14.

Currently, the entire feature extraction module runs on a SPARCstation 20 model 30 with a total execution time of 32.2 seconds for a $512 \times 512$ gray-level image. Table 2 shows the execution times for the important steps in the algorithm.

The algorithmic parameters such as the variance of the Gaussian smoothing windows, the size of the structuring element, and the number of directions in the orientation field were empirically determined by running the algorithm on a set of test images.

Visually, the results of ridge segmentation and final feature detection are quite acceptable for matching purposes. In the next section, we will describe a quantitative method to evaluate the quality of the extracted features.

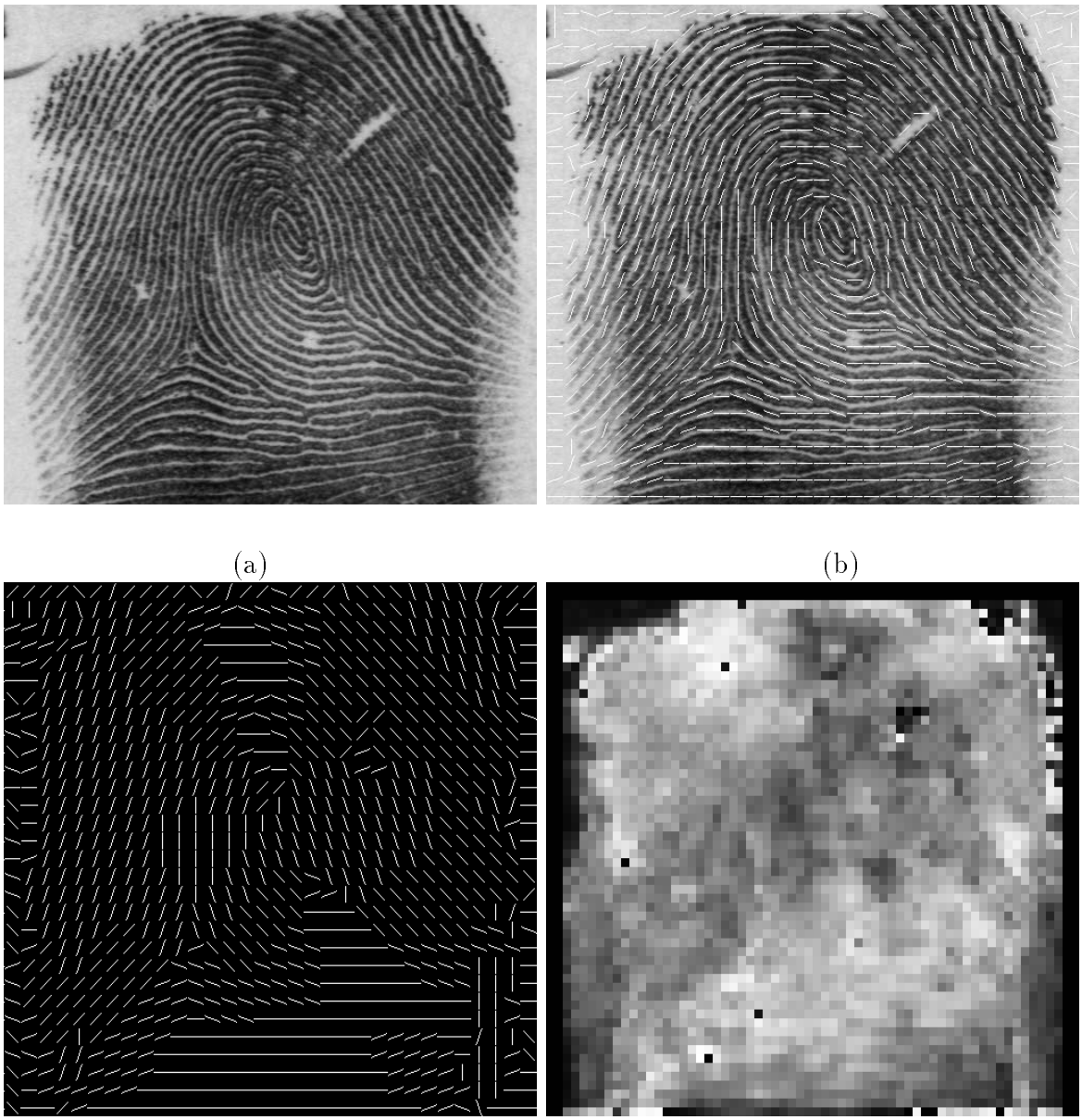
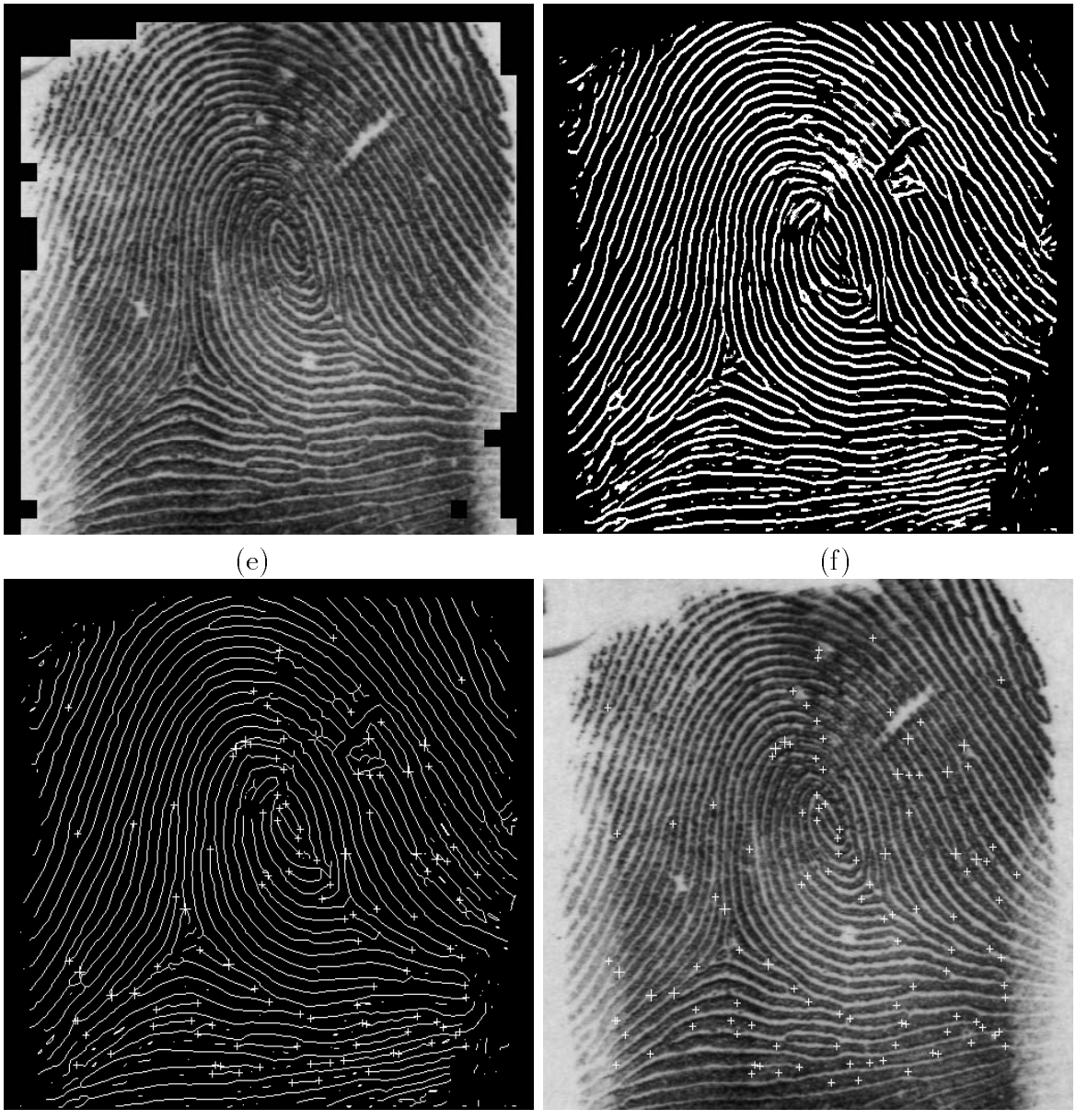



Figure 14: Results of various stages in feature extraction: (a) original image; (b) orientation field; (c) smoothed orientation field; (d) variance field (contd.)



(e) (f)

(g) (h)

Figure 14: (Contd.) Results of various stages in feature extraction: (e) foreground/background segmentation; (f) ridges after masking out the background; (g) skeleton with minutiae points marked; (h) minutiae superimposed on the input gray level image.

| Step | Time (in seconds) |
|---|---|
| Flow Orientation | 1.25 |
| Smoothing, Variance computation, and Image Enhancement | 11.9 |
| Ridge Detection | 5.1 |
| Thinning | 5.0 |
| Morphological Filtering | 5.5 |
| Minutiae detection and Postprocessing | 1.8 |

Table 2: Execution times for important steps in the algorithm.

## 4.1 Performance Evaluation

The performance of our feature extraction algorithm has been evaluated by comparing the detected minutiae with the set of minutiae obtained from the same image by a human expert (ground truth). Although this is a laborious process, in our opinion, it provides an essentially unbiased performance evaluation measure. Note that different human experts will often find different sets of minutiae points in the same fingerprint image and that is one of the motivations for developing an automatic feature extraction algorithm. Let $\mathbf{F_a} = (\mathbf{f_a^1}, \mathbf{f_a^2}, \ldots \mathbf{f_a^N})$ be the set of $N$ minutiae points detected by the algorithm and $\mathbf{F}_\mathrm{g} = (\mathbf{f_g^1}, \mathbf{f_g^2}, \ldots \mathbf{f_g^M})$ be the set of $M$ minutiae points in the ground truth for a given fingerprint image. For the purpose of computing the goodness index (GI), the following definitions are needed.

- Paired minutia: A minutia detected by the algorithm, $\mathbf{f_a}$, and a ground truth minutia, $\mathbf{f_g}$, are said to be paired if $\mathbf{f_a}$ lies within a tolerance box centered around $\mathbf{f_g}$. A graphical representation is shown in Figure 15. In our experiments, the size of the tolerance box is $8 \times 8$.

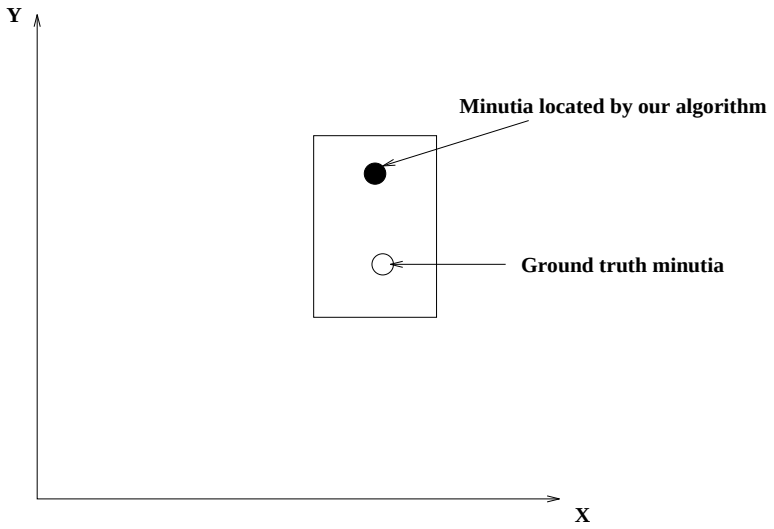



Figure 15: Pairing of minutiae using a tolerance box.

- Missing minutia: A minutia which has to be inserted in the set $\mathbf{F_a}$ (not detected by the algorithm), to allow it to be paired with some minutiae present in the set $\mathbf{F}_\mathrm{g}$.

- Spurious minutia: A minutia in the set $\mathbf{F_a}$ that needs to be deleted because it cannot be paired with any ground truth minutia in $\mathbf{F}_\mathrm{g}$.

The goodness index (GI) is defined by the following equation.

$$GI = \frac{\sum_{i=1}^{L} Q_i[P_i - D_i - I_i]}{\sum_{i=1}^{L} Q_i M_i}, \tag{2}$$

where

L = number of $16 \times 16$ windows in the input image,

$P_i$ = number of minutiae paired in the $i^{th}$ window,

$Q_i$ = quality factor of the $i^{th}$ window (good = 4, medium = 2, poor = 1),

$D_i$ = number of deleted minutiae in the $i^{th}$ window,

$I_i$ = number of inserted minutiae in the $i^{th}$ window,

$M_i$ = number of ground truth minutiae in the $i^{th}$ window.

Define variables P, D, I, and M as follows:

$$P = \sum_i P_i, \quad D = \sum_i D_i, \quad I = \sum_i I_i, \quad \text{and} \quad M = \sum_i M_i.$$

Note that we have chosen a value of 4 instead of 3 for the quality index $Q_i$ of the "good" areas to give larger weight to correct and incorrect matches in the good quality portions of fingerprint images.

The *maximum* value of GI equals +1 which is obtained when $D_i$=$I_i$=0 and $N_i$=$M_i$, i.e., all the detected minutiae are paired. This index penalizes both the missing minutiae and spurious minutiae. Note that the number of matches in each window is weighted by its quality. Thus, a mistake (missing or spurious minutiae) in a good-contrast window is weighted more heavily compared to the same mistake in a noisy area of the fingerprint image. If we assume that the total number of detected minutiae in a window is at most twice the "true" number of minutiae, then the *minimum* value of GI is -3. This value is obtained when $N_i$=0, $D_i$=2 $*$ $M_i$, and $I_i$=$M_i$, i.e., no detected minutiae could be paired and the number of detected minutiae is two times the number of ground truth minutiae.

A large value of GI for a fingerprint image implies that the feature extraction algorithm has done a good job on that image. The GI values for a representative subset of 10 fingerprint images is shown in Table 3. The average value of GI using the proposed feature extraction method is 0.24. The maximum and minimum values of GI obtained on this dataset are 0.48 and 0.1, respectively.

How significant are these observed values of GI? In order to determine the significance of the observed values, we compared them against the GI values obtained under a baseline distribution [17]. We have used the following procedure to obtain the baseline distribution of GI.

| Fingerprint number | P | D | I | M | GI | $P^B$ | $D^B$ | $I^B$ | $GI^B$ |
|---|---|---|---|---|---|---|---|---|---|
| u1421 | 57 | 16 | 9 | 62 | 0.48 | 15 | 58 | 51 | -1.24 |
| f023 | 40 | 23 | 2 | 42 | 0.475 | 5 | 58 | 37 | -2.06 |
| f013 | 50 | 23 | 9 | 59 | 0.34 | 14 | 59 | 45 | -1.30 |
| s018 | 43 | 22 | 9 | 52 | 0.285 | 7 | 58 | 45 | -1.94 |
| u1420 | 50 | 20 | 18 | 68 | 0.263 | 9 | 61 | 59 | -1.19 |
| u1373 | 48 | 35 | 11 | 59 | 0.135 | 13 | 70 | 46 | -1.375 |
| s024 | 30 | 18 | 8 | 38 | 0.118 | 2 | 46 | 36 | -1.64 |
| s23 | 36 | 21 | 10 | 49 | 0.11 | 6 | 51 | 43 | -1.69 |
| u13a7 | 36 | 21 | 10 | 46 | 0.102 | 7 | 50 | 39 | -2.38 |
| f09 | 49 | 28 | 17 | 66 | 0.10 | 8 | 69 | 58 | -1.68 |

Table 3: GI values for a sample of 10 fingerprint images.

1. Generate a pair of random integers (x, y), x, y $\epsilon$ [0, 511]. Note that the image size is $512 \times 512$.

2. If (x, y) is a foreground point in the input image, then accept the point as a random minutiae point.

3. If the total number of random minutiae points is equal to the number of minutiae points obtained from the feature extraction algorithm, then stop; otherwise go to step 1.

The values of GI computed with a set of random number of $N$ points with the fixed $M$ ground truth points can be used to evaluate the observed value of GI. The baseline distribution for the fingerprint numbered 'u1421' (based on 100 sets of random points) is shown in Figure 16. For the ten fingerprints used in Table 3, we also provide the values of baseline GI (denoted as $GI^B$). Note that the variables $P^B$, $D^B$, and $I^B$ correspond to P, D, B, respectively, for the baseline distribution. The values of $GI^B$ varied from -2.38 to -1.19 with an average value of -1.65. (Note that the average GI value is 0.24). Comparing the values of $GI^B$ with the values of GI computed from the extracted minutiae, we can conclude that the proposed feature extraction method is robust and accurate.

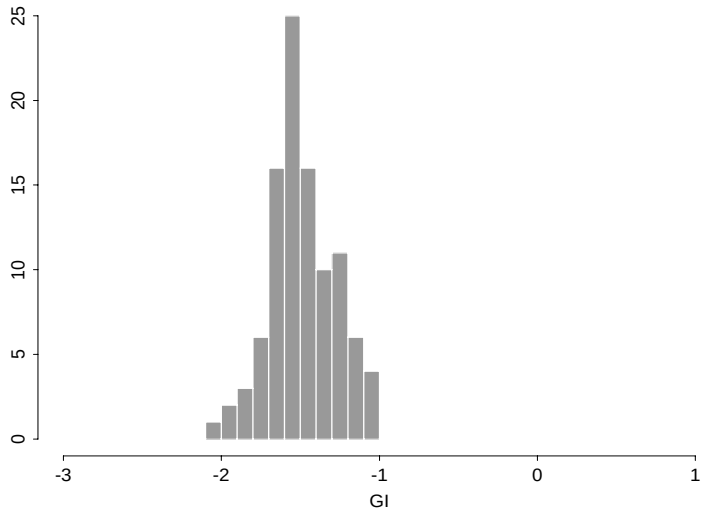



Figure 16: Baseline distribution for fingerprint image 'u1421'.

# 5 Conclusions and Future Work

We have proposed a new method for robust feature extraction from fingerprint images based on ridge flow orientations. The main contributions of this paper are: (i) a novel segmentation method, (ii) an adaptive enhancement of the thinned image, and (iii) quantitative performance evaluation. The input image quality did not adversely affect the performance of our technique. Ridge segmentation based on peak detection of the projected waveform and morphological filtering results in a good skeleton image. A performance evaluation technique has been described which compares the detected features with the ground truth. The algorithm has been tested on one hundred fingerprint images and the goodness index has been computed to substantiate our claim of robustness. We are currently in the process of running the feature extraction algorithm on the NIST fingerprint database [18], which contains several thousand fingerprint images.

In order for the proposed method to be acceptable for commercial use, the execution time of the algorithm must be substantially reduced. We are currently porting the feature extraction

algorithm on Splash 2 — a Field Programmable Gate Array (FPGA)-based array processor [19]. A FPGA-based fingerprint matching algorithm has already been implemented [20].

# References


[1] B. Miller. Vital signs of identity. *IEEE Spectrum*, 31(2):22–30, February 1994.

[2] Henry C. Lee and R. E. Gaensslen, editors. *Advances in Fingerprint Technology*. Elsevier, New York, 1991.

[3] Federal Bureau of Investigation, U. S. Government Printing Office, Washington, D. C. *The Science of Fingerprints: Classification and Uses*, 1984.

[4] Application briefs: Computer graphics in the detective business. *IEEE Computer Graphics and Applications*, 5(4):14–17, April 1985.

[5] B. G. Sherlock, D. M. Monro, and K. Millard. Fingerprint enhancement by directional Fourier filtering. *IEE Proc. Vis. Image Signal Processing*, 141(2):87–94, April 1994.

[6] B. M. Mehtre. Fingerprint image analysis for automatic identification. *Machine Vision and Applications*, 6:124–139, 1993.

[7] Louis Coetzee and Elizabeth C. Botha. Fingerprint recognition in low quality images. *Pattern Recognition*, 26(10):1441–1460, October 1993.

[8] D. C. Douglas Hung. Enhancement and feature purification of fingerprint images. *Pattern Recognition*, 26(11):1661–1671, November 1993.

[9] Q. Xiao and H. Raafat. Fingerprint image postprocessing: A combined statistical and structural approach. *Pattern Recognition*, 24(10):985–992, 1991.

[10] L. O'Gorman and J. V. Nickerson. An approach to fingerprint filter design. *Pattern Recognition*, 22(1):29–38, 1989.

[11] H. Tamura. A comparision of line thinning algorithms from digital geometry viewpoint. In *Proc. of 4th. Int. Joint Conf. on Pattern Recognition, Kyoto, Japan*, pages 715–719, 1978.

[12] A. Ravishankar Rao. *A Taxonomy for Texture Description and Identification*. Springer-Verlag, New York, 1990.

[13] M. Kawagoe and A. Tojo. Fingerprint pattern classification. *Pattern Recognition*, 17(3):295–303, 1984.

[14] M. R. Verma, A. K. Majumdar, and B. Chatterjee. Edge detection in fingerprints. *Pattern Recognition*, 20(5):513–523, 1987.

[15] T. Sakai, M. Nagao, and H. Matsushima. Extraction of invariant picture sub-structures by computer. *Computer Graphics and Image Processing*, 1(1):81–96, 1972.

[16] SharpImage Software, New York. *The HIPS Image Processing Software*, 1993.

[17] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Englewood Cliffs, New Jersy, 1988.

[18] C. I. Watson. *NIST special database 9: Mated Fingerprint Card Pairs*. Advanced Systems Division, Image Recognition Group, National Institute for Standards and Technology, February 1993.

[19] Jeffrey M. Arnold, Duncan A. Buell, and Elaine G. Davis. Splash 2. In *Proceedings 4th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 316–322, 1992.

[20] N. K. Ratha, A. K. Jain, and D. T. Rover. Fingerprint matching on Splash 2. In D. A. Buelll, J. M. Arnold, and W. J. Kleinfelder, editors, *Splash 2: FPGAs in a Custom Computing Machine*, To be published. IEEE Computer Society Press, IEEE, 1995.