8-2011

3D minutiae extraction in 3D fingerprint scans.

Sara Shafaei
University of Louisville

3D MINUTIAE EXTRACTION IN 3D FINGERPRINT SCANS

By

Sara Shafaei
B.Sc. 2001, Computer Engineering, University of Isfahan
M.Sc. 2004, Computer Engineering, Amirkabir University of Technology

A Dissertation
Submitted to the Faculty of the
Graduate School of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

Department of Electrical and Computer Engineering
University of Louisville
Louisville, Kentucky

August 2011

3D MINUTIAE EXTRACTION IN 3D FINGERPRINT SCANS

By

Sara Shafaei
B.Sc. 2001, Computer Engineering, University of Isfahan
M.Sc. 2004, Computer Engineering, Amirkabir University of Technology

A Dissertation Approved on

July 14, 2011

by the Following Reading and Examination Committee:

Tamer Inanc, Ph.D., Dissertation Director

Amir A. Amini, Ph.D.

Robert W. Cohn, Ph.D.

Laurence G. Hassebrook, Ph.D.

Prasanna K. Sahoo, Ph.D.

DEDICATION

To my parents

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Dr. Tamer Inanc for his guidance, support, and encouragement during the course of my PhD. I also would like to thank him for giving me the chance to work on interesting projects, and always supporting my ideas, and encouraging and guiding me for developing them. He has always been a very considerate person and it was great pleasure working with him.

I would like to express my sincere thanks to Prof. Laurence G. Hassebrook for his help and providing software and data for us. I also would like to thank him for helping and guiding me during my stay at the University of Kentucky.

I would like to thank Prof. Amir A. Amini, Prof. Robert W. Cohn, Prof. Prasanna Sahoo, for agreeing to be on my dissertation committee, for the useful consultation and the valuable comments.

I would like to thank Prof. Zurada for the joint seminars between our lab and his lab, which gave us the opportunity to hear about other student research and present our research.

I would like to thank Flasscan3D LLC for proving the data for us.

I would like to thank ARCS lab members for their friendship and support. I also thank all my friends for having a good time.

Very special thanks to my family for their encouragement and support, without which this dissertation and research would not have been possible. My deepest gratitude to my parents and sisters.

ABSTRACT

3D MINUTIAE EXTRACTION IN 3D FINGERPRINT SCANS

Sara Shafaei

July, 14, 2011

Traditionally, fingerprint image acquisition was based on contact. However the conventional touch-based fingerprint acquisition introduces some problems such as distortions and deformations to the fingerprint image. The most recent technology for fingerprint acquisition is touchless or 3D live scans introducing higher quality fingerprint scans. However, there is a need to develop new algorithms to match 3D fingerprints. In this dissertation, a novel methodology is proposed to extract minutiae in the 3D fingerprint scans. The output can be used for 3D fingerprint matching.

The proposed method is based on curvature analysis of the surface. The method used to extract minutiae includes the following steps: smoothing; computing the principal curvature; ridges and ravines detection and tracing; cleaning and connecting ridges and ravines; and minutiae detection. First, the ridges and ravines are detected using curvature tensors. Then, ridges and ravines are traced. Post-processing is performed to obtain clean and connected ridges and ravines based on fingerprint pattern. Finally, minutiae are detected using a graph theory concept.

A quality map is also introduced for 3D fingerprint scans. Since a degraded area may occur during the scanning process, especially at the edge of the finger-

print, it is critical to be able to determine these areas. Spurious minutiae can be filtered out after applying the quality map. The algorithm is applied to the 3D fingerprint database and the result is very encouraging. To the best of our knowledge, this is the first minutiae extraction methodology proposed for 3D fingerprint scans.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Person identification and authentication are fundamental activities in a fast-moving, modern society. Traditional identification methods such as ID cards, ATM cards and PIN codes do not meet the demands of this wide-scale connectivity. Automated biometrics provide efficient solutions to these modern identification problems and eliminate common problems such as forged or stolen personal identification numbers (PINs) and illegally copied keys. It can also be used for identification purposes involving security access systems. A biometric system is a pattern-recognition system that recognizes a person based on specific physiological or behavioral characteristics that the person possesses [11]. Some of the physiological biometrics are fingerprint, face, iris, retinal, hand geometry, DNA and vascular pattern; some of the behavioral biometrics are signature, keystroke dynamics and vocal behavior.

The fingerprint has been one of the most successful biometrics used for personal identification. Fingerprint recognition systems are widely used in the field of biometrics because of their uniqueness, stability and universality. Each individual has unique fingerprints, and no two people have yet been found with the same fingerprints. A fingerprint is the pattern of ridges and valleys on the finger tip. Fingerprints have been used in forensic applications for a long time and, recently, in computer-automated identification and authentication. Fingerprint acquisition, for many years, has been accomplished by first pressing an inked finger on paper and then converting the image into digital form. Recent developments in fingerprint acquisition technology have resulted in inkless or livescan fingerprint scanners; compared to the ink and paper-based methods this technology is

easy to use. The contact-based fingerprint image acquisition includes: ink , capacitive, ultrasonic, pyroelectric, thermal, and optoelectronic approaches [12–15]. The most recent technology for fingerprint acquisition is touchless or 3D live scan, which uses more than one camera that surround the finger for acquisition of a 3D fingerprint.

Many existing fingerprint technologies are used either by rolling or pressing an ink-covered fingertip on the paper surface or touching or rolling a finger on a glass surface of a special device. In both cases, the user must place her/his finger on the hard surface. Since the surface of the finger is not flat, the user should press down the fingerprint area onto the sensor surface to get a good quality fingerprint image. This pressure introduces physical distortions and inconsistencies on the image. These distortions are usually non-linear in arbitrary direction and strength. Moreover, the distortion occurs globally, while its parameters can change locally [16]. All of these factors make it difficult to estimate the distortion accurately, and deformation errors remain in the fingerprint image. Therefore, the image changes with every impression, and there are different versions of the fingerprint images for the same finger. Hence, the relative location of the minutiae, size and quality of fingerprint are different, and this highly affects fingerprint recognition. A good number of algorithms have been proposed to obtain better results. Despite all efforts to enhance the performance, there is an innate problem of distortion due to the pressure of contact with the solid surface. Although many algorithms have been suggested to overcome this problem [17], there are still large amounts of cost and error.

There are also latent fingerprint problems [18]. The latent fingerprint is the trace of fingerprint on the surface of the sensor. This can cause hygienic problems and forgery use, such as the fingerprint faking. Additionally, a contact-based fingerprint usually results in partial or degraded images caused by improper fingerprint placement, smearing, or sensor noise from a tear of surface coating. Image imperfections generate errors in determining the coordinates of each true minutia

and relative orientation of the image and produce spurious minutiae. All these facts remarkably decrease the recognition system reliability.

In order to address the above-mentioned problems with touch-based fingerprint technology, a new generation of touchless live scan devices that generate 3D representation of fingerprints has been introduced to the market [2,16,18–22]. They are able to capture a nail-to-nail fingerprint without any surface touching . Since this new technology can capture a fingerprint image without any contact, it overcomes many of the aforementioned problems. Due to using cameras, touchless fingerprint devices have several advantages such as avoiding plastic distortion, avoiding latent fingerprints, reducing hygienic problems and capturing a large image area quickly. The combination of distortion-free fingerprints and a large image area are most desirable to get many minutiae in the same relative location and orientation in each image [18]. 3D touchless systems can also provide new information about the finger ridges and valleys, and information related to the 3D dimensions and shape of the finger. This new information can be used for more accurate automatic and manual techniques for fingerprint analysis.

Parziale et al [21,22] proposed a multi-camera touchless fingerprint scanner which acquires different finger views that are combined together to provide a 3D representation of the fingerprint. Due to the lack of contact between the finger and surface, the acquired images preserve the fingerprints "ground-truth" without skin deformation during acquisition [22]. However, employing the shape-from-silhouette scanning technique, the ridge information is obtained from the surface reflection variation information. Therefore, the fingerprint is sensitive to surface color, surface reflectance, geometric factors and some other effects.

In an attempt to build such a system, Flashscan3D LLC and the University of Kentucky [23] have been developing a non-contact, 3D finger scanning system. A prototype scanner is shown in Figure 1. This system uses multiple, high-resolution, commodity, digital cameras and utilizes Structured Light Illumination (SLI) through phase measuring profilometry (PMP) [23–25] as a means of acquir-

ing 3D scans of the fingers with sufficient high resolution so as to record 3D ridge-depth information. SLI is a popular sensing technique that calculates the 3D geometry of a target by illuminating the target with structured projection patterns and capturing and decoding the reflected image to obtain the projector co-ordinates corresponding to the image pixel. Conventional SLI systems use multiple patterns for reliable and accurate 3D reconstruction such as (i) binary patterns, (ii) gray-coded patterns, and (iii) PMP. The PMP technique has the advantage of high accuracy and simple implementation, requiring as low as three projection frames. It also requires no point-matching or image enhancement to obtain the fringe distortion, making it suitable for a pipelined or parallel processing implementation. In PMP, the light pattern projected is a sine-wave pattern shifted several times with the captured light pattern [23]. Figure 2 demonstrates 3D fingerprint acquisition using the PMP technique.

In this research, the 3D fingerprint scans were used from the system developed by Flashscan3D LLC and the University of Kentucky [23] as the input to the algorithms in the following chapters. Figure 3 demonstrates a sample 3D fingerprint scan.

After obtaining the 3D data, new algorithms must be developed to match 3D fingerprints, which might be done in two ways. The first method requires that the fingerprint is first unwrapped from a 3D scan into a 2D rolled equivalent fingerprint. Having the unwrapped 2D fingerprints, conventional 2D matching methods can be applied. The benefit of this method is the compatibility with the existing legacy rolled images.

However, a major disadvantage of transforming 3D fingerprint images into 2D images is that unwrapping results in information loss and distortion in the final 2D image. To overcome these disadvantages, a better approach is to perform fingerprint matching directly in 3D. High accuracy in fingerprint matching might be achieved by direct comparison of the fingerprints in 3D.

Fingerprint matching is an extremely difficult task because of large intra-

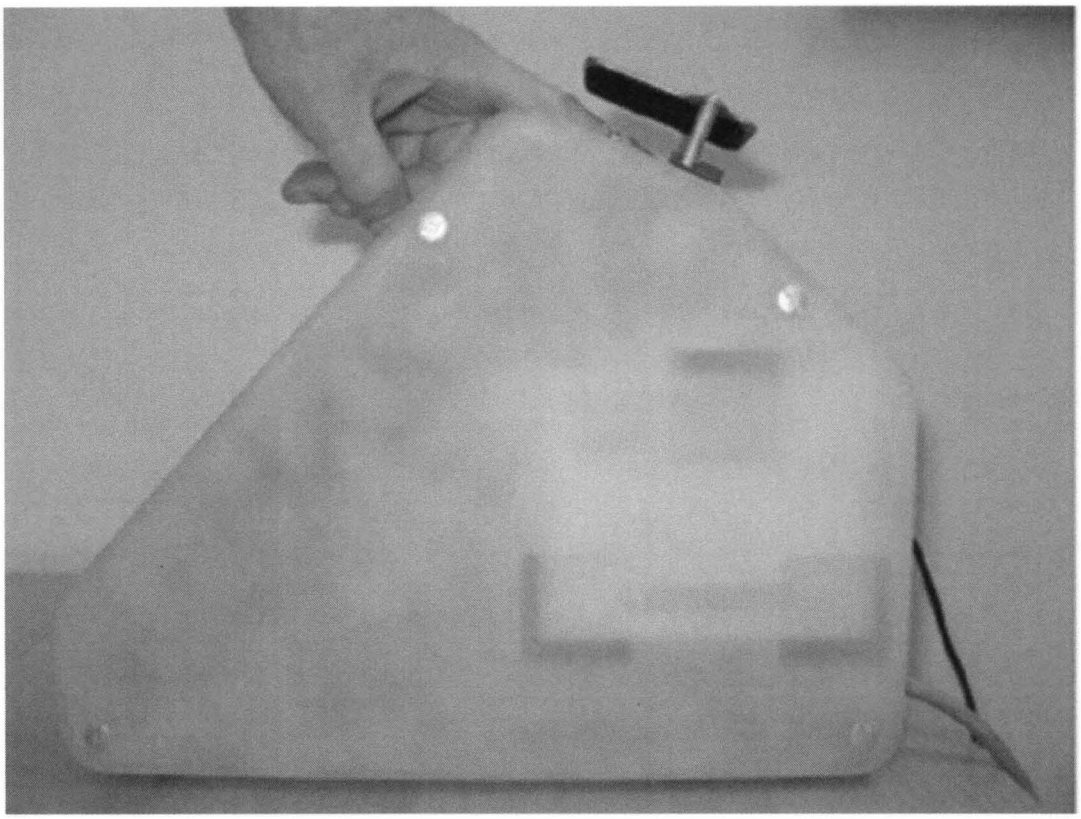


FIGURE 1 – Prototype scanner for 3D fingerprint scanning [1].

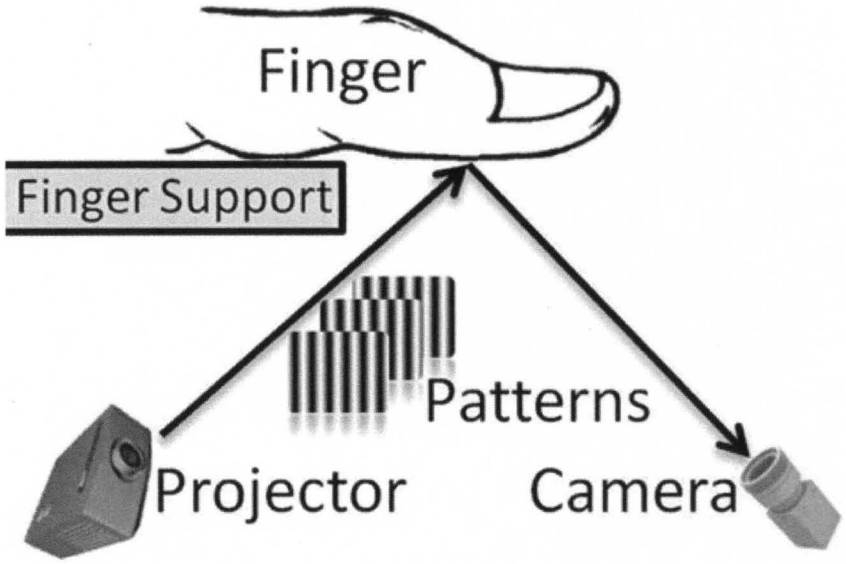


FIGURE 2 – 3D fingerprint acquisition using PMP technique [2].

FIGURE 3 – 3D fingerprint scan [2].

class variation and small inter-class variation. The problem of 2D fingerprints has been studied in the literature, and a large number of approaches have been proposed. An overview of the current 2D minutia-based matching is provided in Chapter II, which includes minutiae definition, 2D minutiae extraction and matching algorithms. As it is demonstrated in Chapter II, most automatic 2D systems for fingerprint matching are based on minutiae matching [3]. Minutiae, or Galton's characteristics [26], are local discontinuities in the fingerprint pattern. A thorough explanation of minutiae is defined in Chapter II.

The first step to develop a 3D minutiae-based fingerprint matching is minutiae detection in the 3D fingerprint. Since minutiae are the key features for fingerprint identification, it is important to detect them precisely. In this research, the focus is on the minutiae detection in 3D fingerprint scans. Chapter III proposes an unwrapping method to unwrap a 3D scan into a 2D rolled equivalent, and then apply the conventional minutiae detector to detect minutiae. However, as it is explained earlier, this method has some disadvantages. Therefore, the main focus of

this research is on minutiae detection directly in 3D scans, which is explained in Chapter IV. A quality map for 3D fingerprint scans is proposed in Chapter V, which can be used separately or along with minutiae detection to remove low quality minutiae. To the best of our knowledge, there are currently no 3D minutiae detection methodologies that exist. Chapter VI includes testing and evaluation of the proposed algorithm. Finally, a conclusion and future works are given in Chapter VII.

3D finger imaging not only brings the biometric industry forward but also sets the stage for the future by enabling more advanced, more reliable and completely new methods for liveness detection and fingerprint matching.

CHAPTER II

2D FINGERPRINT MATCHING

Fingerprint matching is an extremely difficult task because of large intra-class variation and small inter-class variation. Intra-class variation, which is a different impression of the same finger, is usually caused by rotation and displacement of finger impression on the sensor, non-linear distortion, sensor noise or skin condition. The problem of fingerprint has been studied in the literature, and a large number of approaches have been proposed, which can roughly be classified in three categories: correlation-based, minutiae-based, and ridge feature-based approaches.

Correlation-based matching: The query and template fingerprint images are superimposed and spatially correlated, so degree of similarity can be estimated. Since the rotation and displacement parameters are usually unknown, correlation should be computed for all possible parameters. Therefore, it is computationally expensive. Additionally, some problems such as non-linear distortion and noise reduce global correlation between two fingerprints. Thus, local correlation is proposed to solve these problems. Local correlation is done locally in a region of interest such as regions with high curvature or around minutiae.

Minutiae-based matching: The most common and well-known approach for matching fingerprints is minutiae-based matching. In this approach, the minutiae are first extracted from the query and template fingerprint images and saved in two point sets, and then matching two minutiae sets are usually considered as a point pattern matching problem. The similarity between them is proportional to the number of matching minutiae pair.

Ridge feature-based matching: These approaches compare fingerprints based on features extracted from ridge patterns. In fact, correlation-based and minutiae-based matching can be considered as a sub-class of feature-based matching, since pixel intensity and minutiae are features of the ridge pattern. Other common features used in papers are size of finger print and shape of fingerprint silhouette; number, type and position of singularities; spatial relationship and geometrical attributes of ridge lines; shape features; global and local texture information; sweat pores; and fractal features.

A. Minutiae-Based Matching

As it is mentioned in the previous section, minutiae-based matching is the most popular matching approach. Minutiae-based fingerprint representation has an advantage in helping privacy issues since the original image cannot be reconstructed from using only the minutiae information. Minutiae are relatively stable and robust to contrast, image resolutions and global distortion when compared to other representations. However, extracting the minutiae from a poor quality image is not an easy task. Today, most of the automatic fingerprint recognition systems are developed to use minutiae for their fingerprint representations [27]. This section starts with the thorough definition of minutiae and then continue with minutiae extraction methods and finally minutiae matching methods.

1. What are the Minutiae?

Minutiae are the local discontinuities of the local ridge structures in the fingerprint pattern. Since Sir Francis Galton (1822-1922) was the first person who categorized minutiae and observed the structures and permanence of minutiae, minutiae are also called "Galton details." There are about 150 different types of minutiae [28] (the most common types are shown in Figure 4). The American Na-

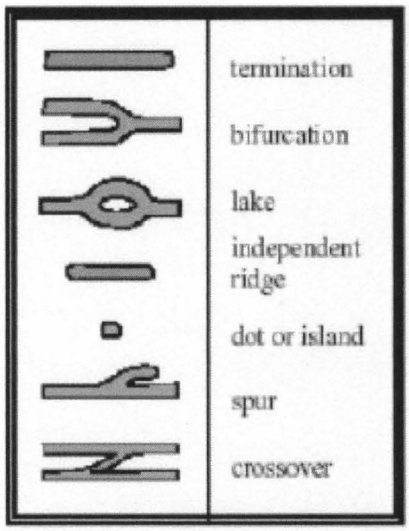


FIGURE 4 – different minutiae type [3]

tional Standards Institute (ANSI) proposes four classes of minutiae [3] : ending, bifurcation, crossover (trifurcation), and undetermined. The FBI only considers ending and bifurcation in its model. Fortunately, nearly all the minutiae types are definable in terms of two fundamental types: ridge endings and bifurcations. So the matching systems usually do not use other types of minutiae. A ridge ending is defined as the point where the ridge ends abruptly, and the ridge bifurcation is the point where two ridges merge together (see Figure 5). The minutiae-based fingerprint representation that is proposed by ANSI-NIST includes minutiae location, which is x and y coordination and orientation [29]. The minutia orientation is defined as the direction of the ridge on which it resides (Figure 5).

2. Minutiae Extraction

There are two main approaches to extract the minutiae from the fingerprint image:

- Binarization-based extraction
- Gray-scale based approach

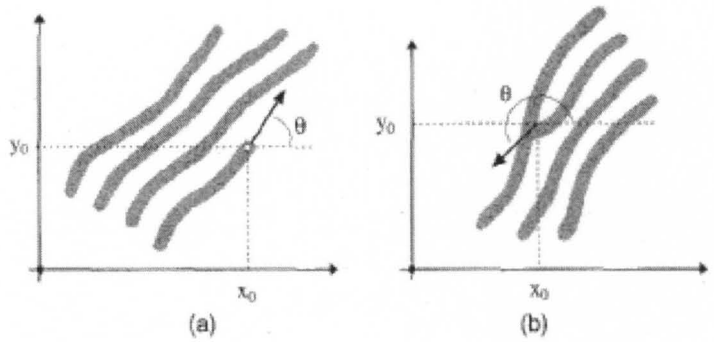


FIGURE 5 – (a) ridge ending (b) ridge bifurcation (x_0, y_0) is coordinate of minutia and θ is orientation of minutia [3]

a. Binarization-Based Minutiae Extraction Most minutiae extraction methods are based on the binary image where there are only two levels of interest: black and white. Most implementations include the steps which are shown in Figure 6). In the following paragraphs, each step is explained.

Binarization is the process of transforming a gray-level image to a binary image, which improves the contrast between the ridges and valleys in the fingerprint image and consequently eases the minutiae extraction process. The easiest method uses global threshold T, and then it assigns one to each pixel that has an intensity value greater than threshold T and zero to each pixel that has an intensity value lower than threshold t. A global threshold is not enough to have correct binarization because of the contrast variation in the fingerprint images. Therefore, a local threshold is preferred in general, which requires that the threshold is adjusted to the local characteristic of the image. A local threshold method cannot always guarantee acceptable results. For instance, it does not work very well with a poor quality fingerprint image.

Ratha et al. [30] propose a binarization approach based on the peak detection in the cross section gray-level profiles orthogonal to the ridge flow orientation. They consider a 16x16 oriented window around each pixel, in which the pixel of interest is on the center, and the orientation of the window is perpendicular to the local ridge orientation. A gray-level profile of pixel intensities is obtained from

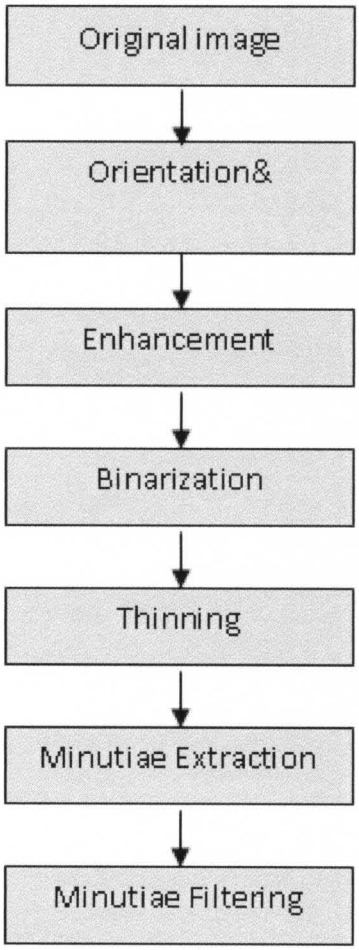


FIGURE 6 – Minutiae extraction stages

the central segment of the window. Before projection, the profile is smoothed by averaging the pixels along the direction of the ridge in the window. The picks in the profile and the two neighboring pixels on each side of the picks are taken as foreground and the rest of it as background. The result image is a binary image.

Cavusoglu et al. [31] use a local threshold for binarization. They divide the fingerprint image into 9x9 blocks, then they compute mean gray value as the local threshold value for the pixels in the block. Kim and park [4] propose a binarization technique based on convex threshold. Their algorithm is composed of four steps. First they estimate local ridge orientation using the modified least mean square orientation estimation algorithm. Next, they quantize the local ridge orientation into four directions (0,45,-45,90). Then a 2-D filter is applied on the image for the fingerprint ridge structure regularization. Finally, convex threshold detect the ridge. The fingerprint of ridge direction is considered as an one-dimensional signal (see Figure 7), where the ridges are convex and valleys are concave. Figure 7 illustrates the process for ridge detection. If the mean value of left and right pixels of the current center pixel is lower than the current pixel, then the pixel is a ridge point.

Garris et al. [8] and Watson et al. [32] propose a directional binarization technique. In this approach, each pixel is analyzed to determine whether to assign a black or white pixel. A pixel is assigned to white if there is no detectable ridge flow for the local block. If there is a detected ridge flow in the pixel's local block, then an orientated window is used to analyze the neighboring pixel intensity of the pixel. The center of the window is on the pixel of interest, and its rows are parallel to the local ridge direction. The average row sum is obtained and compared against the central row sum. If the central row sum is less than the window's average row sum, a pixel would be white; otherwise, it is black.

Zhang and Xiao [5] propose an approach that simulates the human exercise for finding uniform areas. An algorithm is developed to distinguish the dark/light image area, which generates a robust uniform region (see Figure 8). Then bina-

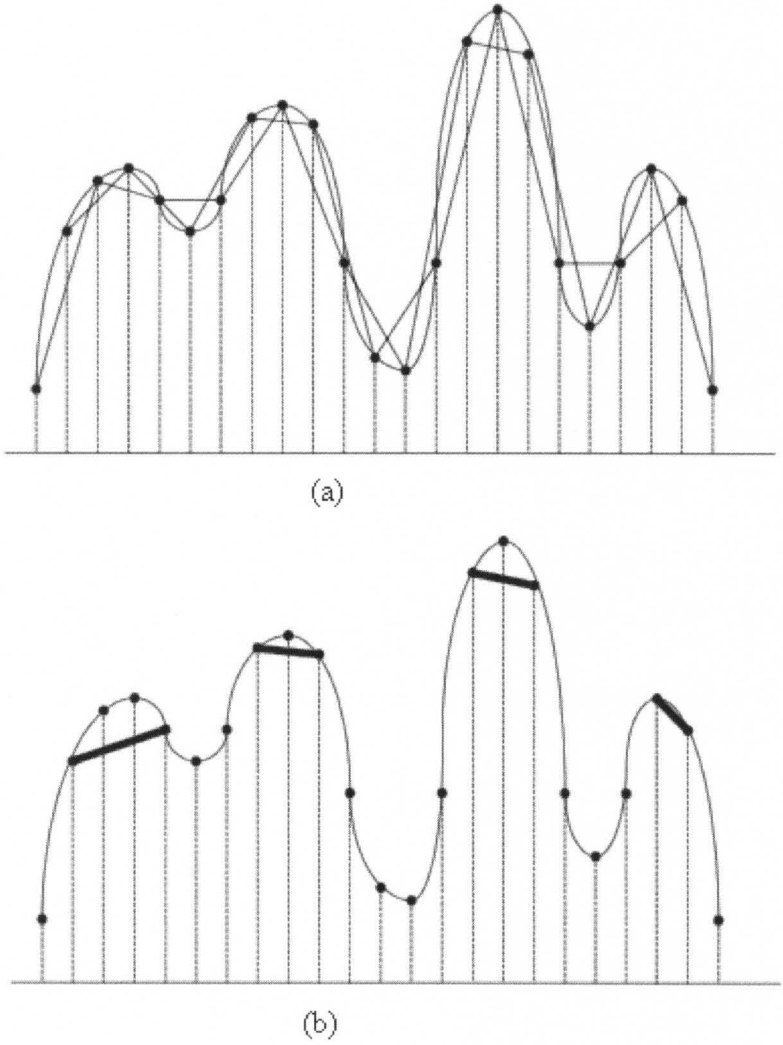


FIGURE 7 – (a) convex detection process (b) result of convex detection [4]

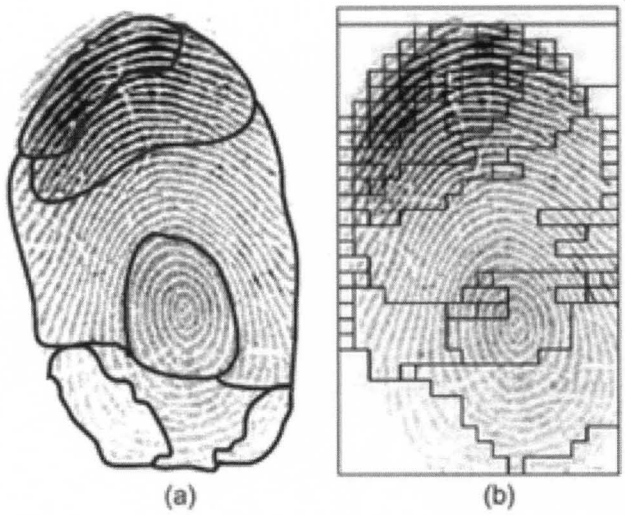


FIGURE 8 – (An example of region generation: (a) region divided based on human vision; (b) image divided by region generation algorithm [5]

rization is performed for each generated region individually by calculating a local threshold for each region and using a traditional thresholding method. Domeniconi et al. [9] introduce a binarization method using a differential geometry tool. The main idea is to view ridges line as a sequence of maximum and saddle points. A Hessian matrix is computed for each pixel; its elements are second-order derivative of tow-dimensional surface S around a pixel P. The eigenvalues of the Hessian matrix are principal curvatures (maximum and minimum curvature) of the surface. Then Pixel P is a local maximum if both eigenvalues are negative, and it is a saddle point if one of them is negative and the other one positive.

Bartunek et al. [33] presents an approach for binarization using frequency domain. A method is developed for automatically determining the proper size of the local area which is obtained by analyzing the entire fingerprint image in the frequency domain. Frequency analysis is also used in the local areas to design directional filters.

The binarization-based minutiae extraction methods usually apply an intermediate thinning step after the binarization step to obtain the skeletons of fingerprint ridges, which reduces the width of ridges to one pixel while keeping the con-

nectivity of the original shape. It is critical to develop thinning algorithms without generating spurious minutiae. The thinning algorithms tend to generate hairy-like artifacts along the skeleton which produce the spurious minutiae. Also, holes in ridges will produce two spurious bifurcations in a skeleton, or noise will result in two spurious ridge endings. Therefore, various regularization techniques are proposed between binarization and thinning stages to fill holes, remove small breaks and eliminate ridge bridges and other artifacts. Most of thinning algorithms use morphological operators [3].

Fingerprint thinning is usually implemented via morphological operations such as erosion and dilation. The following equation explains the mathematical definition of erosion. The erosion of A by element B is denoted by:

$$A \ominus B = \{z \in E | B_z \subseteq A\} \tag{1}$$

Where E is Euclidean space and B_z is the translation of B by the vector z, $B_z = \{b + z | b \in B\}$

Sometimes using erosion might cause some features to be corrupted, so there is another function called dilation. Its mathematical definition is shown in the following equation:

$$A \oplus B = \{z \in E | (B^s)_z \bigcap A \neq \emptyset\} \tag{2}$$

where B_s denotes the symmetric of B, that is, $B^z = \{x \in E | -x \in B\}$ Two other operators that might be used are opening and closing. They are respectively denoted by the following equations:

$$A \circ B = (A \ominus B) \oplus B \tag{3}$$

$$A \bullet B = (A \oplus B) \ominus B \tag{4}$$

Ji et al. [34] proposed an image thinning algorithm using a template-based pulsed-coupled neural network (PCNN). The proposed PCNN is a single-layer, 2-D neural

network. There is a one-to-one relationship between neurons in the network and pixels in the image. Then a coarse-to-fine skeletonize is introduced. It removes the pixels from the object's edge iteratively. In the first step, it removes four pixel patterns from the edge of the image, and in the second step, it removes all possible three pixel patterns. They also introduce a thinning constraint schema by ridge orientation which avoids the spikes. Constraint forces the network to do thinning in the direction of orthogonal to local ridge orientation by indicating that a neuron can fire only along the constrained thinning direction.

Bazen and Gerez [35] introduce a thinning algorithm by encoding each 3×3 pixel neighborhood in a 9-bit integer. They use this code as an index of the lookup table to find the binary value after the current thinning step. Two different lookup tables are used for a different thinning direction. They use them iteratively, until the final skeleton is obtained. Hongbin et al. [36] propose a thinning approach based on mathematical morphology, which uses hit-miss transformation for thinning the fingerprint. Ratha et al. [30] eliminate the spikes using an adaptive morphological filtering. They use a box-shaped open morphological operator with a structuring element. The orientation of the structuring element is perpendicular to local ridge orientation.

Ahmed and Ward [37] propose a rule-based approach for thinning. The method is rotation invariant and guarantees symmetrical thinning and high speed. It uses 20 rules which are applied simultaneously to each pixel in the image and thins objects to their central lines. The algorithm is iterative. At each iteration, it deletes every point that lies on the outer boundaries of the symbol, as long as the width of the symbol is more than one pixel wide. For this purpose, the algorithm uses a set of rules over a 3×3 pixel neighborhood of the pixel to be considered for deletion, but it cannot handle 2-pixel wide lines very well. Patil et al. [6] propose a modified version of an algorithm, which also takes care of the thin zigzag diagonal line with a 2-pixel width that is not considered in [37]. The method is based on 21 thinning rules plus 4 diagonal rules, which are applied in parallel to every pixel

FIGURE 9 – (a) Original image (b) Tinning by Ahmed et al's algorithm (c) Thinning by Pattil et al's algorithm [6]

in each iteration. The results of the two aforementioned algorithms are shown in Figure 9.

Once a binary skeleton of a fingerprint is obtained, minutiae detection in the thinned image is relatively easy. The most commonly employed method of minutiae extraction is the Crossing Number (CN) concept. The minutiae are extracted by scanning the local neighborhood of each pixel in the image using a 3×3 window. A ridge ending point has only one neighbor in the window. A bifurcation point possesses more than two neighbors, and an intermediate ridge point has two neighbors. Therefore, minutiae detection is implemented by scanning the thinned fingerprint and counting the CN. The CN number can be defined as follows [38]:

$$CN = 0.5 \sum_{i=1}^{8} |P_i - P_{i+1}|, P_9 = P_1 \tag{5}$$

where P_i is the pixel value in the neighborhood of I. The eight neighboring pixels for pixel I are shown in Figure 10.

After the CN for a ridge pixel has been computed, the pixel can then be classified according to the property of its CN value. (Figure 11)

- If $CN = 2, p$ is an intermediate ridge point
- If $CN = 1, p$ is a ridge end
- If $CN >= 3, p$ is a bifurcation, crossover or some other complex minutia

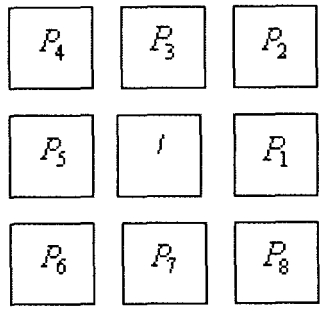


FIGURE 10 – The eight neighboring pixels for pixel I

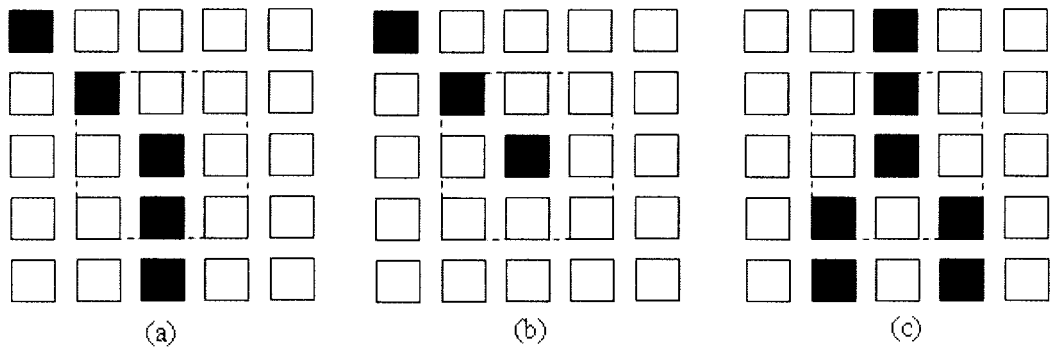


FIGURE 11 – (a) $CN = 2$, ridge intermediate point (b) $CN = 1$, ridge ending (c) $CN = 3$, ridge bifurcation

Many papers use the aforementioned method for extracting minutiae from the skeleton [30,39–42]. Bartunek et al. [43] propose a method for minutiae extraction from the skeleton using the neural network. A four-layered, fully-connected, feed-forward neural network is implemented to detect and classify minutiae. This network has 25 neurons in the input layer, two hidden layers with 25 neurons in each of them and 3 neurons in output layers, and the activation function is a bipolar sigmoid function. The network is trained for three different pattern classes: termination, bifurcation and non-minutiae, with a 5×5 window as training data. The algorithm that is used for learning is back-propagation. Then the neural network is trained and tested on the 23 fingerprints. Leung et al. [44] also use a three-layer perception neural network with a back-propagation learning technique for extracting minutiae from the skeleton.

Miao et al. [7] introduce a method based on principal curve to extract minutiae. The principal graph algorithm is used to obtain the fingerprint's skeleton which consists of a set of principal curves. Then, the endings of principal curves are analyzed to extract the minutiae. For this purpose, they check the first and the last points of each of the principal curves and count how many principal curves share these endings. If the ending is found only in one of them, then it is a ridge end, and if it is found in three of them, then it is a ridge bifurcation. In Figure 12 five principal curves (AB, BC, BD, CE, CF) are shown. A is a ridge end because it is found in only one principal curve (AB), and B is bifurcation because it is found in three principal curves (AB, BC, BD).

There are some proposed approaches which extract the minutiae directly from a binary image without any intermediate thinning. Weber [45] proposes a rule-based algorithm to extract the minutiae from a thick binary image. The algorithm follows a ridge from a starting point to a termination point which can be a bifurcation or a ridge end. Jung-Hwan Shin et al. [46] extract minutiae based on representing the ridge structure of a fingerprint image as a run length code (RLC). After preprocessing and segmentation, the fingerprint image is converted to a bi-

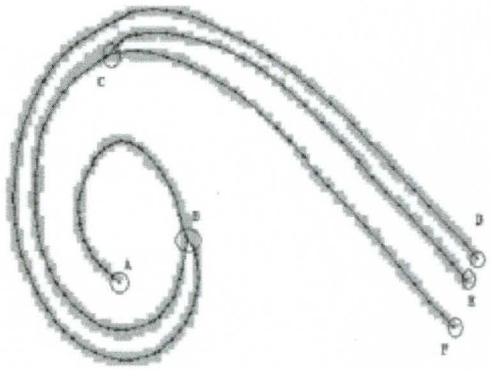


FIGURE 12 – Principal curves of a complex ridge [7]

nary image. Then, a binarized fingerprint image is entirely represented by a linked list of its runs. The minutiae are detected by searching for the termination points or bifurcation points of ridges in the RLC.

Garris et al. (NIST) [8] use a series of pixel patterns to extract the minutiae from binraized fingerprint images. The right-most pattern in Figure 13 represents the family of ridge ending patterns which are scanned vertically. Ridge ending candidates are determined by scanning the consecutive pair of pixels in the fingerprint searching for the pattern-matched sequences. Figure 14 illustrates a series of minutiae patterns used to detect the ridge endings and bifurcations in the binary fingerprint image. Since the mechanism of this minutiae detection method is totally different from a skeletonization-based minutiae detection method, specific minutiae filtering methods are also designed. The NIST algorithm is applied on two different fingerprint images (inked and digital). The minutiae detection results are shown in Figure 15; the red color represents the best quality, and the blue color represents the worst quality. Minutiae with bad quality can be avoided.

b. Direct Gray-Scale Minutiae Extraction Binarization-based minutiae extraction causes some problems such as losing information during the binarization, producing spurious minutiae and being time-consuming. It is also unsatisfactory when applied to a low-quality image such as a broken ridge. To overcome the

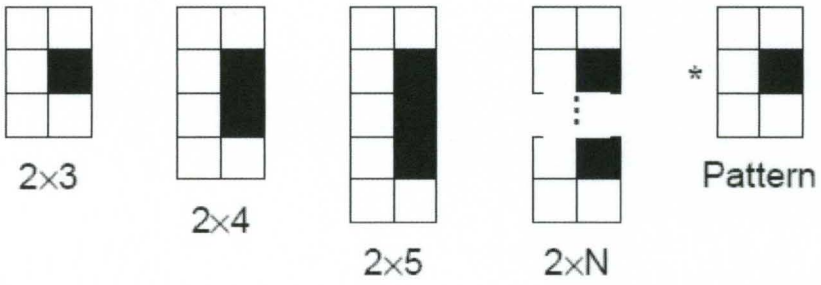


FIGURE 13 – Pixel pattern used to detect ridge ending [8]

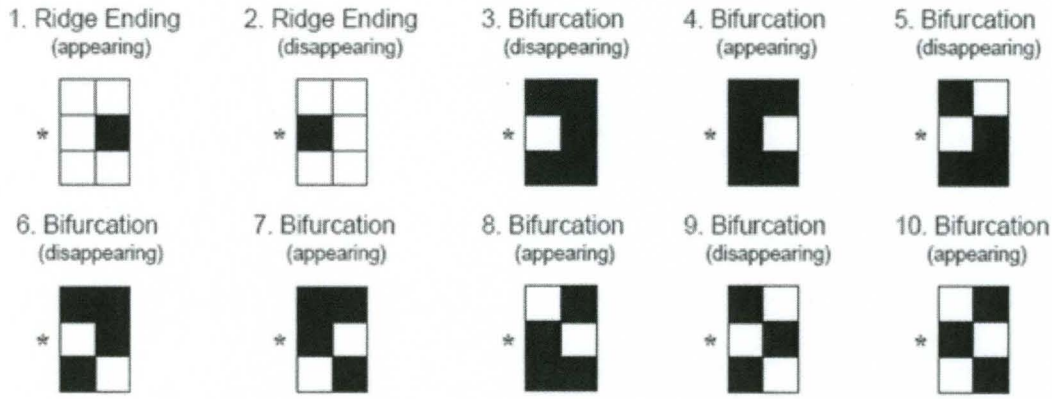
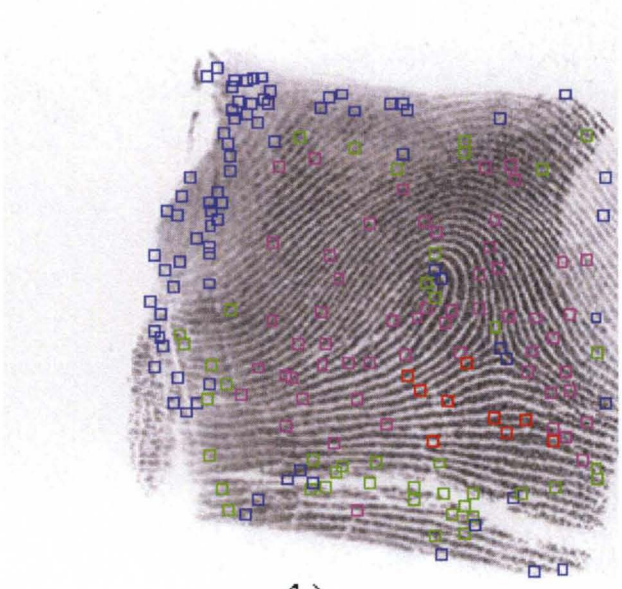


FIGURE 14 – Pixel patterns used to detect minutiae [8]

(a)



(b)

FIGURE 15 – Minutiae extraction using NIST algorithm. Fingerprint acquisition is performed by UK university (a) digital fingerprint (b) inked fingerprint

problems, some approaches have been proposed, which extract minutiae directly from gray-scale extraction methods.

Maio and Maltoni [9] introduce a method based on ridge line following to extract the minutiae from a gray-scale image. The algorithm uses the concept that a ridge line is composed of a set of pixels with local maxima along one direction. Therefore, at each step, the algorithm finds a local maximum which is related to a section orthogonal to the ridge direction. The ridge line can be obtained by connecting them. Therefore, the algorithm begins with a starting point (x_c, y_c) and starting direction θ_c, and then computes a new point (x_t, y_t) by tracing the ridge along the θ_c with step of μ pixels from current pixel. The point (x_t, y_t) is only an approximation of the next ridge point. So, to compute the exact location of it, a new section Ω with a median of (x_t, y_t) is obtained, with a direction that is orthogonal to θ_c and its length is $2\rho + 1$. The next ridge point (x_n, y_n) is computed by analyzing the gray-scale profile of the section set Ω and finding a local maximum in it. This point would be used as a new starting point and the local ridge direction at it as the new starting direction of the next iteration (Figure 16). The parameters μ and ρ are determined according to the average thickness of the ridge. The tracing is executed in the direction of a ridge and stops when a ridge line terminates or intersects with other ridges. Jiang et al. [47] improve Maio and Maltoni's method by determining a dynamic value for the ridge following step μ according to the change of ridge contrast and bending level. A large step μ is used when there is low variation of intensity along the segment and the bending level of the local ridge is low. Otherwise, a small step μ is taken in the presence of high intensity variation and high ridge bending, which respectively represent possible ridge termination and ridge bifurcation.

Chang and Fan [48] introduce an algorithm to extract the ridges' locations directly from a gray-scale image, which is based on gray-level histogram decomposition. The ridge, raving and background are determined by a statistical analysis of the trimodal distribution. The algorithm considers the effect of background in

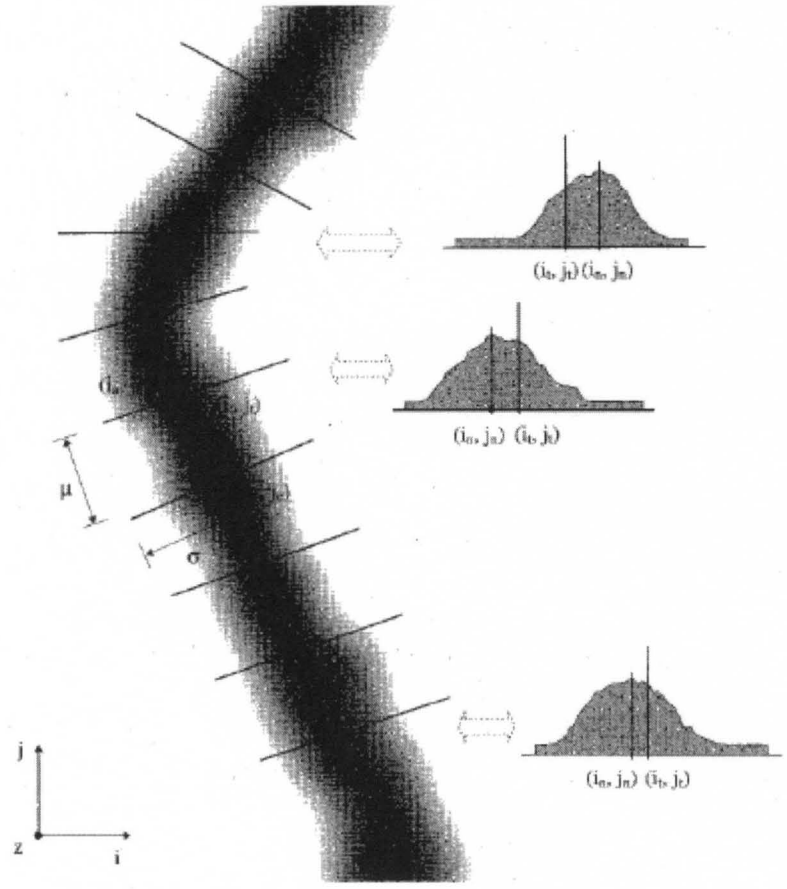


FIGURE 16 – Some ridge following steps. Some sections are shown on the right [9]

a fingerprint image; therefore, it is working for a complete fingerprint, as opposed to Maio and Maltoni's algorithm, which only works for a partial fingerprint. Since the algorithm does not need any preprocessing stage, the computational time is low. Montesanto et al. [49] use three algorithms to detect minutiae in gray-scale fingerprint images. The first algorithm is based on a sequential method which traces a ridge in a gray-scale image. The ridges are reproduced on another digital image using a polynomial to combine all of the ridge points which are found by the algorithm. The second algorithm is also a ridge following algorithm which is a reactive agent, and the third one extracts minutiae using a multi-layer neural network from a gray-scale image. The results of the three algorithms are combined to match fingerprints.

Liu et al. [50] propose an approach based on tracking the relationship between the ridges and valleys in the gray-scale image. They use the concept that the ridges and the valleys are roughly parallel to each other in a local neighborhood. Therefore, the relationship between adjacent ridges and valleys remains the same until they reach to minutiae, such as bifurcation and ridge end. The relationship is detected by tracking neighboring ridges and valleys along the local direction of the ridge which is obtained from orientation image, and the relationship is represented by vector with three points located in a line orthogonal to the local direction. One of the points is the center of a ridge and the others are the centers of the two adjacent valleys.

Leung et al. [51] propose a neural network-based method. A set of oriented Gabor filters is first applied to fingerprint image, and the outputs are given as input into a three-layered back-propagation neural network to find the existence of minutiae. Fronthaler and Bigun [10] use Linear Symmetry (LS) filter to detect the minutiae based on the concept that minutiae are local discontinuities of the LS vector field. Two types of symmetries (parabolic and linear symmetry) model and extract the local structure in a fingerprint. After the calculation of linear and parabolic symmetry, extra steps are applied to extract minutiae reliably. Hence, the

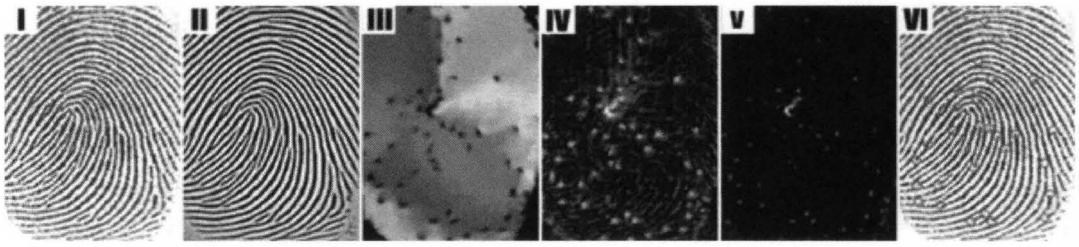


FIGURE 17 – Minutia point detection process [10]

selectivity parabolic filter is improved with the following equation:

$$PS_i = PS.(1 - |LS|) \tag{6}$$

In Figure 17 the minutiae detection process is shown. The first and second images are original and enhanced fingerprint images respectively. The image displayed in image III is linear symmetry, and parabolic symmetry is shown in image IV. The resulting sharpened magnitudes after applying equation 6 is shown in image V. Then all points which have filter response below a certain threshold are set to zero, and remaining points are searched for finding minutiae. The point is determined as minutiae if it is fully surrounded by high linear symmetry. It can be done by checking whether the average linear symmetry on a ring around a minutia candidate is above a certain threshold. Detected minutiae are shown in image VI.

All the minutiae points detected with the above methods are not always true minutiae. False minutiae may be introduced by factors such as a noisy image and image artifacts created by the thinning process. A post-processing stage called minutiae filtering is usually necessary to eliminate spurious minutiae. The elimination is based on their structural characteristics or minutiae filtering in the gray-scale domain. Some examples of false minutiae are shown in Figure 18, which include the spur, hole, triangle and spike structures [29].

3. Minutiae-based matching

Most common fingerprint matching algorithms are Minutiae-based matching algorithms. This approach attempts to get the similarity degree between two

FIGURE 18 – Examples of typical false minutiae structures.

minutiae sets which are first extracted from the query and stored template by one of the methods that was previously explained. Then an algorithm matches the relative placement of the minutiae set in the query fingerprint with stored template and returns a binary decision (matched/non-matched) or a similarity score to indicate how similar the two fingerprints are. The minutiae can be represented by a number of attributes: i) x-coordinate, ii) y-coordinate, iii) local ridge orientation iv) type (e.g., ridge end, ridge bifurcation), and so on. A triplet $m = \{x, y, \theta\}$ is the most common representation of a minutia, where (x, y) is the location coordinate and θ is the minutia angle. If T and Q are the indication of the template and the query fingerprint, they can be represented as:

$$T = \{m_1, m_2, ..., m_n\}, m_i = \{x_i, y_i, \theta_i\}, i = 1...n \tag{7}$$

$$Q = \{m_1^{'}, m_2^{'}, ..., m_n^{'}\}, m_j^{'} = \{x_j^{'}, y_j^{'}, \theta_j^{'}\}, j = 1...n \tag{8}$$

Where n and f are the number of minutiae in T and Q, respectively. A minutia $m_j^{'}$ in T matches the minutia m_i in Q, if their spatial distance is smaller than a certain threshold r_0 and their orientation difference is smaller than the angular threshold θ_0. They can be represented by the following equations:

$$\sqrt{(x_i - x_j^{'}) + (y_i - y_j^{'})} \leq r_0 min(|(\theta_i - \theta_j^{'})|, 360 - |(\theta_i - \theta_j^{'})|) < \theta_0 \tag{9}$$

Since the alignment parameters are unknown, the matching problem is hard. Various techniques have been proposed in the literature to solve the problem. The

easiest technique is a brute force approach which computes all the possible solutions, and the complexity of it is exponential in the number of minutiae. A few brute force approaches have been proposed in literature. Minutiae-based methods may make the more sophisticated computation to search for the best correspondence of minutiae pairs or ridge pairs, which usually use point pattern matching algorithm, such as relaxation, energy minimization and Hough transform. It may also use some features like core or delta minutiae point to estimate the alignment parameter. Usually, the geometric transformation (e.g. displacement, rotation, scale) is used to make the alignment between two fingerprints.

One of the popular approaches for minutiae matching is the Hough transform-based approach [52] which converts point pattern matching to a problem of locating the highest peak in the discrete Hough space of transformation parameter. Ratha et al. [53] propose a Hough transform-based minutiae matching approach which creates a four-dimensional parameter space include translation, rotation and scale $(\triangle x, \triangle y, \theta, s)$. The parameter space is discretized into small cells, then a four-dimensional array A is utilized for accumulating the evidence of transformation parameters. The parameter values with the highest evidence are used for computing the geometric transformation. Chang et al. [54] introduce another version of Hough transform-based approach which use a line segment between two minutiae to compute transformation parameter space and accumulate the evidence.

In the relaxation approach [55], the confidence level of each corresponding pair iteratively is adjusted based on its consistency with other pairs until a certain criterion is satisfied. Although different versions of this approach are proposed in the literature, these algorithms are slow because of their iterative nature. Energy minimization is another approach to point matching. This approach defines a cost function based on an initial set of possible correspondences, and then optimal solutions are obtained using an appropriate optimization algorithm such as genetic algorithm [56] and simulated annealing [57]. These methods are usually very slow.

Another approach for minutiae-based matching is alignment-based match-

ing algorithm, which pre-aligns template and input fingerprint before the minutiae matching. Jain et al. [58] propose a pre-alignment-based minutiae matching. They use the fact that each minutia in a fingerprint is associated with a ridge. During minutiae extraction, the associated ridge (represented as a planar curve) with the minutiae is also recorded. Ridges matching between all possible pairs of ridges on query and template fingerprint is performed until a pair is found in which their matching score is within a certain threshold. The pair found is used to extract the parameter for pre-alignment. All the minutiae are converted to polar coordinates, and they are then translated into symbolic strings, where the correspondence between minutiae can be obtained by a dynamic programming algorithm.

Wegstein [59] introduces an approach that pre-alignment is performed with respect to the core positions and the average direction of two regions on the two sides of the core. Since a database is created by the transformed template, speed of identification is significantly improved.

CHAPTER III

UNWRAPPING A 3D FINGERPRINT TO A 2D ROLLED EQUIVALENT FINGERPRINT

As mentioned in Chapter I, the first method to detect minutiae is to unwrap the 3D fingerprint scan to a 2D rolled equivalent image and to use conventional 2D minutiae detectors. Also, 3D touchless fingerprint images need to be compatible with the legacy rolled images. Therefore, this chapter proposes an algorithm to convert the 3D fingerprint scan to a 2D rolled equivalent fingerprint.

A. Previous Work

There have been recently some algorithms proposed to convert 3D fingerprint images to unwrapped 2D images. Chen et al. [60] propose two methods to unwrap a 3D fingerprint scan, namely, parametric and non-parametric. The parametric approach projects a 3D fingerprint to the parametric model and then unwraps the model. They used a cylinder as the parametric model. Since a cylindrical model is the closest model to the finger shape, it is a reasonable choice for parametric unwrapping of 3D fingerprints. The transformation in this method is often straightforward. The texture of the fingerprint is projected onto the cylinder which surrounds the finger, and then the 2D fingerprint is obtained by flattening the cylinder. Each point (x, y, z) in the fingerprint is transformed to the cylindrical coordinate (θ, z), where $\theta = tan^{-1}(x/y)$. One of the shortcomings of this approach is that it does not preserve the relative distance between the points on the fingerprint surface, which introduces a horizontal distortion to the flattened fingerprint.

Another method proposed in [60] is a non-parametric algorithm. In this method, the unwrapping directly applies to the fingerprint without projecting it

to a special model. The approach locally unfolds the finger surface. In fact, a 3D fingerprint is divided into thin horizontal parallel sections and each section is unfolded separately. Linear interpolation is used to obtain more slices between the main slices which results in a more smooth fingerprint. Finally, points are regenerated using linear interpolation for each horizontal slice to map the slice from 3D to 2D. The regenerating of the point for unwrapping starts from the center and goes to the nail side. The non-parametric method generates better results than the parametric method since it preserves the relative distance between minutiae in the fingerprint. The main shortcoming of this method is that it does not use depth information and it only uses albedo.

Fatehpuria et al. [61] propose another approach to extract a 2D rolled equivalent fingerprint from a 3D fingerprint. They first extract the smooth surface of the 3D fingerprint by smoothing the ridges and valleys by a weighted, non-linear, least square algorithm. The weights are obtained by a Gaussian function. Then the smoothed 3D surface is transformed to the 2D unfolded surface using the "springs algorithm" proposed by Atkins et al. [62]. The texture of the fingerprint (ridges and valleys) is calculated by taking the difference between the original 3D surface and the smoothed 3D surface. Therefore, the final, 2D rolled equivalent fingerprint is obtained by putting the texture onto the unfolded surface which is extracted by the "springs algorithm." Wang et al. [63] propose an approach using both depth and albedo information. First the ridges and valleys information is extracted by fitting different size circles along the vertical direction. By putting these circles together, a tube form of 3D fingerprint will be obtained. Then, the 2D fingerprint is obtained by flattening the tube. Also, the albedo flattened 3D fingerprint is obtained by extracting the ridges information from the albedo image. Hassebrook et al. [64] fuse the depth and albedo unraveled fingerprints to achieve higher quality.

B. Proposed Algorithm to Unwrap 3D Fingerprints to 2D Images

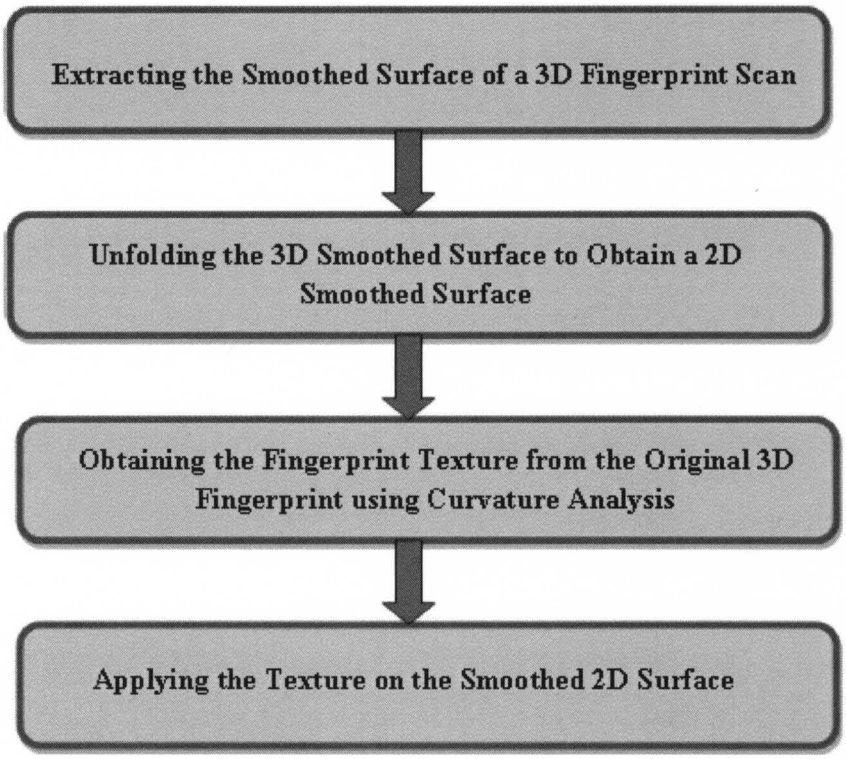


FIGURE 19 – Schematic flow chart of our algorithm for unwrapping a 3D fingerprint.

In this research, a new algorithm based on curvature analysis of the 3D surface is proposed to unwrap the 3D fingerprint. The steps are shown in Figure 19.

C. Extracting the Smoothed Fingerprint Surface from a 3D Fingerprint Scan

The first step in this approach is to extract the smoothed surface of the 3-D fingerprint. The purpose is to obtain the 3D finger shape without any ridge and valley. To satisfy this purpose, a weighted linear least square algorithm is used, whose weights are calculated by a Gaussian function. At each 3D point of the 3D fingerprint scan, a plane is fitted to the point under consideration and the points in the neighborhood of it. Therefore, a $N \times N$ window centered at the point of interest is considered and the plane is fitted to the points inside the window using the weighted linear least square method. The points close to the point of interest or center of the window are given higher weights, and points that are further are

given lower weights.

$$S = min_{a,b,c} \sum_{i=1}^{N^2} w_i.(z_i - (ax_i + by_i + c))^2 \tag{10}$$

w_i is the weight of i^{th} point. N^2 is number of points in the $N \times N$. The weight of each point shows its influence on the plane fitting. Closer points to the center have a higher weight than the further points. The Gaussian function is used to calculate the weights:

$$w_i = e^{-d_i^2/\sigma^2} \tag{11}$$

where d_i is the Euclidean distance between the i^{th} point inside the window and the window's center point. Equation (10) can be minimized by setting partial derivatives of the function $\sum_{i=1}^{N} w_i.(z_i-(ax_i+by_i+c))^2$ to zero. A linear system of equations is obtained by taking partial derivatives with respect to the unknown coefficients a, b and c. Therefore, coefficients can be obtained from the following formula:

$$C = (X^T W X)^{-1} X^T W Z \tag{12}$$

C includes coefficients; X includes all of the x and y coordinates of the points; Z includes z coordinates of the points, and W includes the weights. The result of smoothing is shown in Figure 20.

D. Unfolding the 3D Smoothed Surface

Once a smoothed approximation of the fingerprint shape is obtained by the aforementioned method, the next step is unfolding the 3D smoothed surface to obtain the 2D rolled-equivalent fingerprint image. This is performed by applying the "springs algorithm" proposed by Atkins et al. [62]. They introduce a halftone post-processing technique which rearranges image pixels to get a smoother rendition.

The algorithm assumes virtual springs between a point under consideration and its neighboring points. The point would move to a new location based on

(a) termination (b) bifurcation

FIGURE 20 – (a) Original 3D fingerprint scan (b) Smoothed 3D fingerprint scan

minimizing the energy in the springs. This energy is defined by the cost function. Brown et al. [65] also use the spring method to unwrap an old document, and Provot [66] uses the mass spring system to model rigid cloth deformation. The same idea is applied here to unwrap the 3D surface to the flattened, 2D surface. A physical-based modeling approach is used same as in [61].

The cloud point is considered a mechanical system, in which each point has some mass, and it is connected to the 8-connected neighbors with virtual springs. Each spring has a relax length and it has minimum energy when it has its relax length. Therefore, to have minimum energy in the springs, they are required to compress or stretch to reach their relax length. An iterative algorithm is implemented to calculate the displacement of the points. Each iteration consists of one pass over all the points. Displacement is only applied to the point under consideration, and other points remain fixed. To calculate the total energy in each point, the energy that is stored in the springs, which connect the point to its neighbors, is added together. The magnitude of the energy of each spring is obtained by squar-

ing the magnitude of the displacement between the current length of the spring and its relaxed length, and the sign of energy is determined by subtracting the relax length from the current length. More details can be found in [62].

E. Curvature Analysis to Obtain the Fingerprint Texture and Applying it on the Unfolded 2D Surface

After the rolling, the fingerprint texture (ridges and valleys) needs to be obtained and apply it to the 2D rolled equivalent fingerprint surface. The curvature concept is utilized to detect a ridge on the 3D fingerprint surface. Therefore, each point is determined whether or not to be on a ridge. If the point is a ridge point the corresponding point in the 2D rolled equivalent fingerprint is assigned black color which shows a ridge pixel on the fingerprint.

Since the image captured by the 3D scanner has some noise and gaps, a preprocessing step is required to remove them. Therefore, a median filter is first applied to the whole image to remove the sharp spikes which occur during the scanning of a finger. Then a low pass filter is applied to get a smoother surface on ridges and valleys [67], which are not smooth due to existing pores on the ridge surface and other noise which happens during the scanning.

The next step is to extract the points lying on ridge lines on the surface. There are different approaches to locate the ridges [68]. One approach is to use the Gaussian and mean curvature which is used here. They are thresholded to find these points. In 3D Euclidean space, a surface can be defined by two partial differential equations, the so-called first and second fundamental form of differential geometry.

These fundamental forms determine how to measure the length, area and the angle of the surface, and the normal surface curvature can be calculated from these two fundamental forms. The first fundamental form (I) is defined as the inner product of dx with itself [69], where dx is tangent to the surface in the direction

defined by du and dv:

$$\begin{aligned} I &= dx.dx = (x_u du + x_v dv).(x_u du + x_v dv) \\ &= (x_u.x_u)du^2 + 2(x_u.x_v)dudv + (x_v.x_v)dv^2 \\ &= Edu^2 + 2Fdudv + Gdv^2 \end{aligned} \tag{13}$$

E, F and G are first fundamental coefficients.

The second fundamental form (II) is defined as the inner product of dx and dN [69], where dN means the spatial rate of change of unit normal vector N to the surface.

$$\begin{aligned} II &= -dx.dN = -(x_u du + x_v dv).(N_u du + N_v dv) \\ &= (x_u.N_u)du^2 + 2(x_u.N_v + x_v.N_u)dudv + (x_v.N_v)dv^2 \\ &= Ldu^2 + 2Mdudv + Ndv^2 \end{aligned} \tag{14}$$

L, M and N are the second fundamental coefficients. The Gaussian and mean curvatures K and H ,respectively, are defined as:

$$K = k_1 k_2 = \frac{LN + M^2}{EG - F^2} \tag{15}$$

$$H = \frac{1}{2}(k_1 + k_2) = \frac{EN - 2FM + GL}{2(EG - F^2)} \tag{16}$$

Principal curvature k_1 and k_2 can be obtained by the following formula:

$$\begin{aligned} k_1 &= H + \sqrt{H^2 - K} \\ k_1 &= H - \sqrt{H^2 - K} \end{aligned} \tag{17}$$

By using the Gaussian and mean curvature, every point can be determined that is a ridge or valley point on the fingerprint. Then all of the ridge points are painted with the black color on the flattened rolled fingerprint surface. The shape and orientation at each point of the surface can be described by the Gaussian and mean curvature, respectively. For example, if the sign of two principal curvatures k_1 and k_2 are opposite or the Gaussian curvature K is less than zero, the surface is a saddle

at that point. Also if the points have mean curvature H greater than zero, they are ridge points. Therefore, each point on the rolled fingerprint would be replaced by black color if it is determined as a ridge point on the original 3D fingerprint. Other points remain white.

F. Results

In this section, the proposed method is applied to several 3D fingerprints which are scanned by the 3D scanning hardware developed by Flashscan3D LLC and the University of Kentucky [2].

Figure 21 shows three different 2D rolled equivalent fingerprints which are obtained by applying the proposed algorithm on the 3D scan fingerprints.

Once the 2D rolled equivalent fingerprints are obtained by our proposed method, we enhance the result by using the method proposed by Chikkerur and et al. [70] which is based on block-wise contextual filter approach in Fourier domain. The result of enhancement on images in Figure 21 are illustrated in Figure 22.

In order to determine the quality of the 2D rolled equivalent fingerprint image, NIST [1] fingerprint image software (NFIS[2]) [71] is used. It is important to analyze the fingerprint image and determine degraded areas for reliable minutiae detection. NFIS includes seven major packages. The MINDTCT package is utilized, which detects minutiae and assesses the quality of each minutiae. It also generates the quality map.

To generate the quality map, NFIS divides the image into the blocks and for each block it generates several maps (direction map, low contrast, low flow, and high curve) and integrates them into one general map. It contains 5 levels of quality; a value of 4 shows the highest quality, and the value of 0 shows the lowest quality. In the quality map grayscale image that we display here, the brighter area

[1]National Institute of Standards and Technology

[2]NIST Fingerprint Image Software

TABLE 1
MINUTIAE'S QUALITY

Color	Quality Percentage
Red	80-100
Magenta	60-80
Yellow	40-60
Green	20-40
Blue	0-20

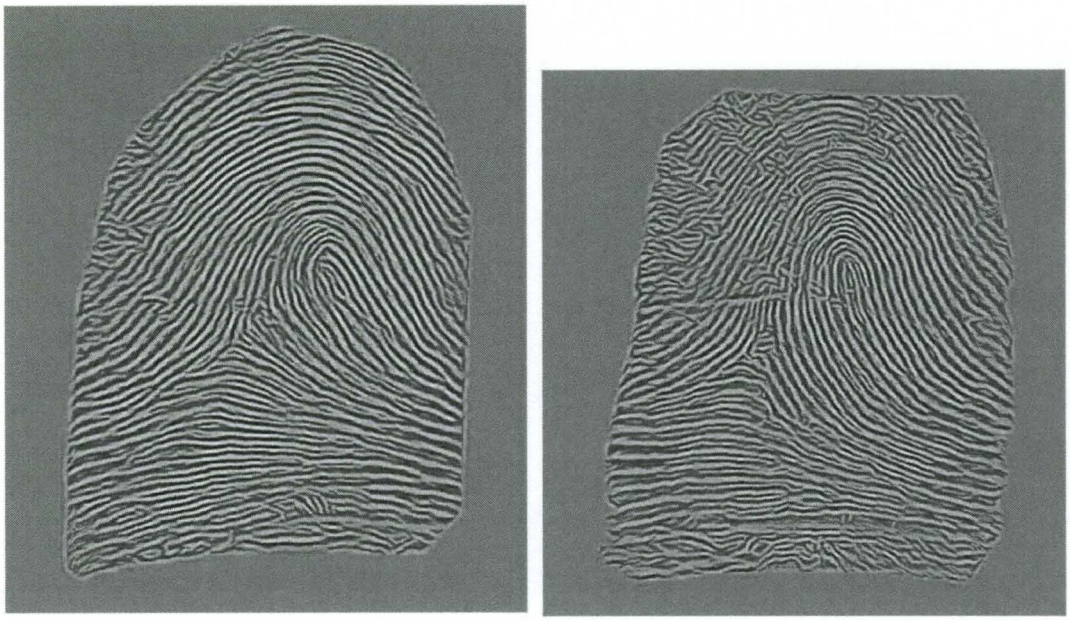
shows the better quality, and the darker area shows the worse quality. Therefore, white shows the highest quality, and black shows the lowest quality. Figure 23 shows the minutiae detected on the fingerprints in Figure 22. The color of minutiae shows the reliability and quality of the minutiae. Table 1 shows the quality value for each color. Most of the detected minutiae are red which shows the best quality (Table 1). The blue minutiae are the poor quality minutiae, and can be deleted for the matching purpose. The fingerprint quality maps are shown in Figure 24. The majority of gray scale quality map images are white which shows a good quality fingerprint image as described above.

(a) subject 1

(b) subject 2

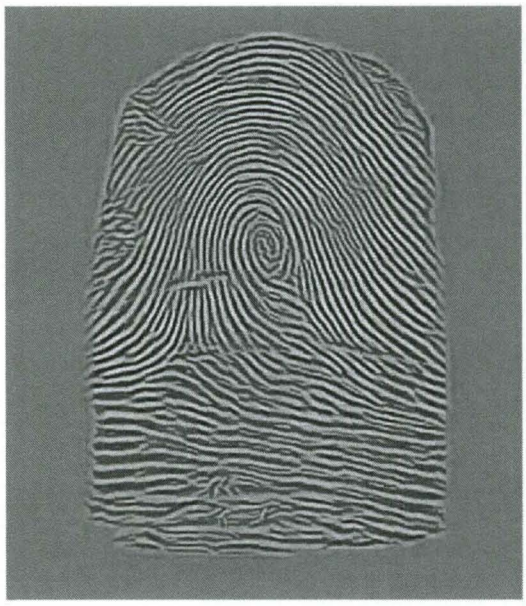
(c) subject 3

FIGURE 21 – The rolled 2-D fingerprints which are obtained by applying the proposed algorithm to the 3-D fingerprints.



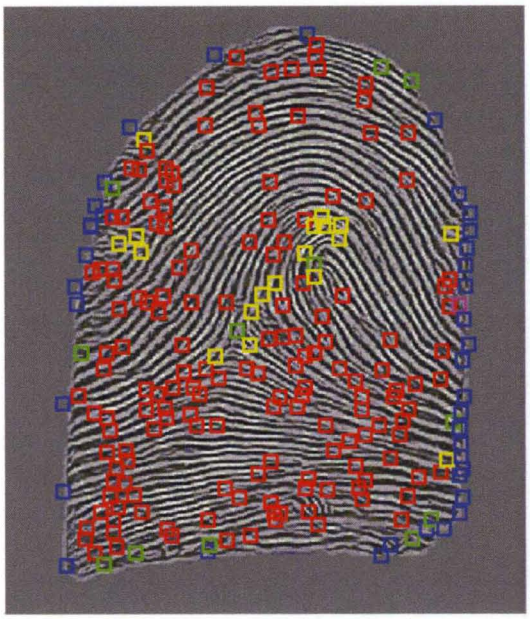
(a) subject 1

(b) subject 2



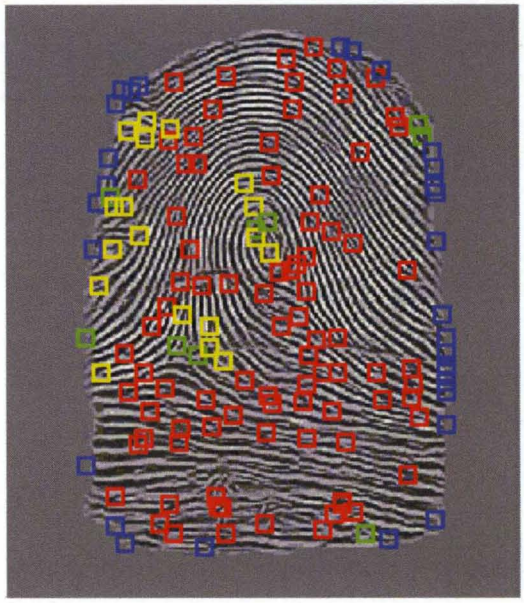
(c) subject 3

FIGURE 22 – Enhanced 2D rolled equivalent fingerprints.



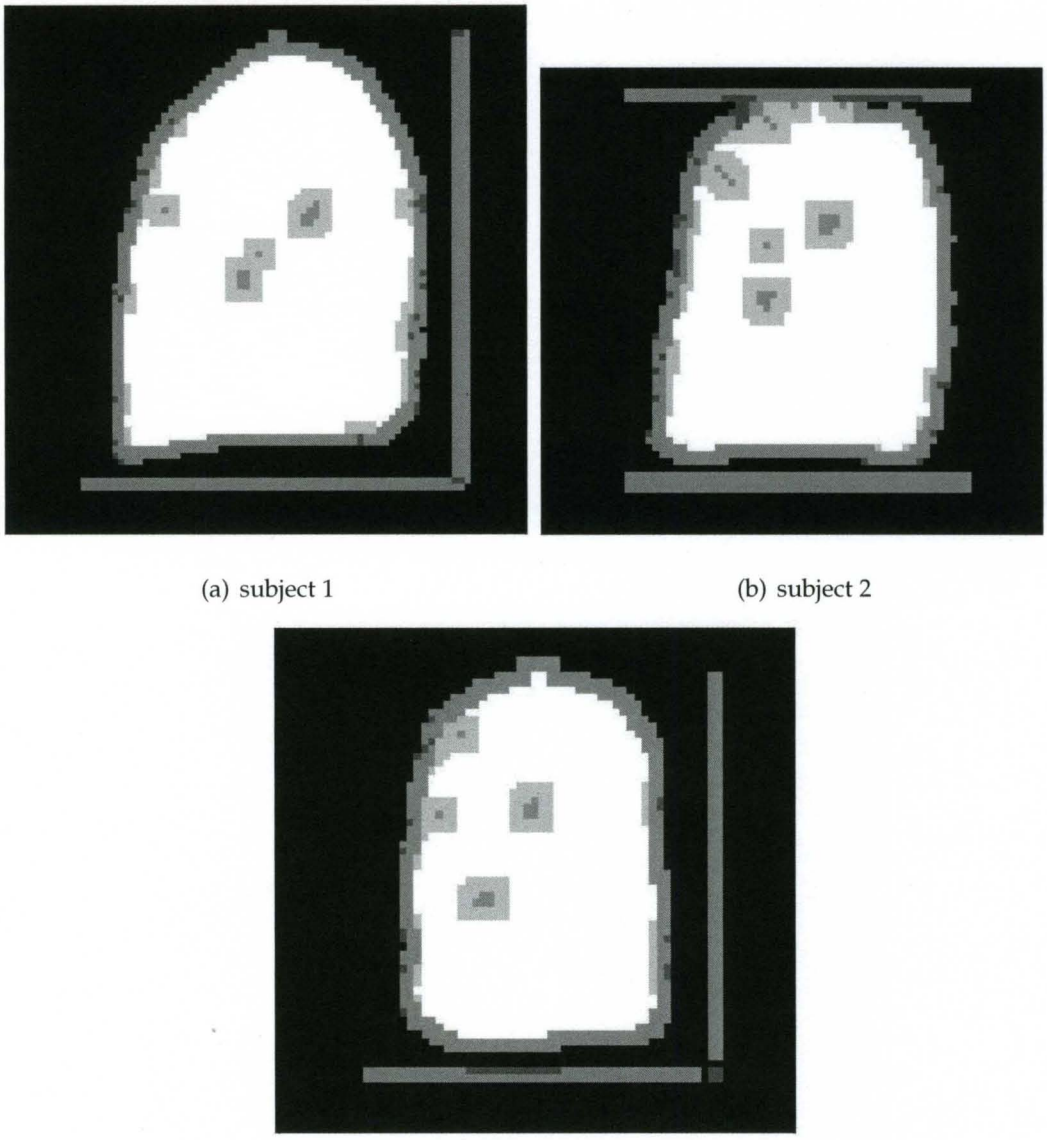
(a) subject 1

(b) subject 2



(c) subject 3

FIGURE 23 – Detected minutiae on the fingerprints. The color of minutiae shows the reliability and quality of the minutiae (Table 1)



(a) subject 1

(b) subject 2

(c) subject 3

FIGURE 24 – Quality map where white shows the best quality and gray shows the bad quality; worst quality is shown in black.

CHAPTER IV

MINUTIAE DETECTION IN 3D FINGERPRINT SCANS

As it is mentioned in the second chapter, the first step to develop 3D minutiae-based fingerprint matching is minutiae detection in the 3D fingerprint. Minutiae extraction is a crucial process in 2D conventional fingerprints as well. Minutiae are the key features for fingerprint identification, and it is important to develop an algorithm to detect them precisely. The focus of this research is mainly the minutiae detection in 3D fingerprint scans. This chapter proposes a novel methodology to find 3D minutiae in 3D fingerprint scans. To the best of our knowledge, there are currently no 3D minutiae detection methodologies that exist for 3D fingerprint scans.

A. Overview of the proposed Algorithm

In this method, first differential geometry concepts are used and curvature analysis of the surface in order to detect ridges and ravines in the 3D surface. Practical detection of the crest lines is a difficult computational task since it requires a high-quality estimation of the curvature tensors and their derivatives. In order to detect them more accurately, an implicit surface can be fitted globally or locally to the mesh. Global fitting leads to more accurate detection; however, local fitting is much faster, and results can be satisfactory. Here, the local fitting is used. Having detected ridges and ravines, minutiae can be detected using graph theory concepts. Finally, a quality map is developed for 3D fingerprint scans. The false minutiae are removed by applying the quality map. The steps of the proposed algorithm are shown in Figure 25.

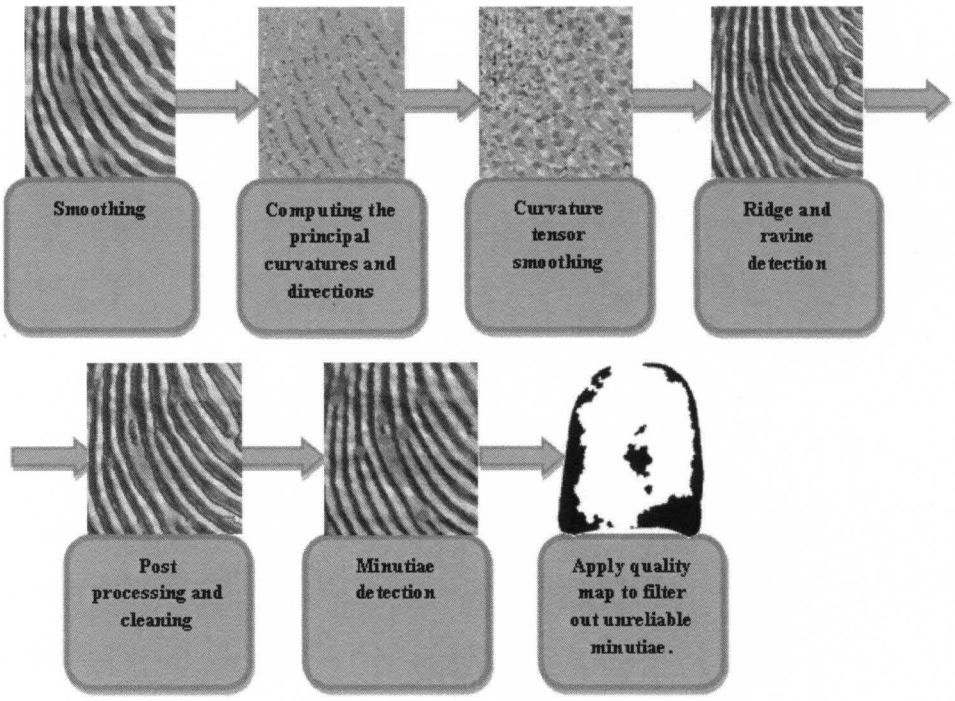


FIGURE 25 – Schematic flow chart of our algorithm for 3D minutiae detection.

B. Smoothing

Since the image captured by the 3D scanner has some noise and gaps, and the curvature extrema are sensitive to even small variations, a smoothing procedure is applied to the mesh before curvature extraction. This study used two kinds of smoothing. In the first called Centroid smoothing, the smoothed mesh is obtained by using adjacent triangles. For every vertex in the mesh, a one-ring neighborhood is considered. Then the arithmetic mean of the centroids of the adjacent triangles of the considered vertex is obtained as a new vertex [72]. A new mesh is formed by the new vertices, which is smoother than the old one. Figure 26 shows the original scan of a sample fingerprint; Figure 27 demonstrates the same sample after Centroid smoothing.

The second method is adaptive smoothing [73]. This method smoothes the noisy mesh while preserving the sharp features, like ridge lines. First, mesh normals are smoothed using the Gaussian filter, then the mesh vertex positions are

modified in order to fit the mesh to the field of smoothed normals. Figure 28 shows the result after adaptive smoothing.

C. Computing the Principal Curvatures and Directions

This section describes the methods that have been developed in order to calculate principal curvatures on meshes. There are three basic steps before curvature calculation: normal estimation, local coordinates, and surface fitting.

1. Normal Estimation

The first step to compute the principal curvatures of the surface mesh is the computing of the normal vector for each vertex on the mesh. In general, a triangle mesh does not have the normals associated with it, and therefore, these must be estimated as well. This normal vector approximates the real surface normal vector at that vertex. The normal vector at the vertex can be computed as the average of the face normals for the faces adjacent to the vertex, with various weightings applied. It may also be computed as the normal to the plane that best fits the vertex and some number of nearby vertices. Here, the unit normal for each vertex is computed by using Nelson Max's method [74]. In this method, a normal vector at each vertex is computed as a normalized weighted sum of the normals of triangles incident to the vertex. The weight of each normal depends on the size of the relevant facet. The larger weights are assigned to smaller facets, and the smaller weights are assigned to the larger facets.

2. Local Coordinate System

Before delving into curvature estimation, a local orthonormal coordinate system should be build. Many parameterization methods utilize a local 3D coordinate frame with its origin at the vertex. A local coordinate system is formed

FIGURE 26 – Original 3D fingerprint scan.

FIGURE 27 – Fingerprint scan after Centroid smoothing.

FIGURE 28 – Fingerprint scan after Adaptive smoothing.

by the normal vector and two arbitrary orthonormal vectors in the plane through the considered vertex. Transforming it to such a local coordinate system does not restrict the curvature calculation, but does simplify the solution of the equations defining the surface representation. All of the calculations should be done in the local coordinate system.

3. Surface Fitting Method

Continuous methods rely on fitting a polynomial surface locally to the vertex and to some neighboring points. The idea is to somehow best approximate locally the underlying surface that was the source of the triangulation. The main discriminatory factor between fitting methods is the function chosen to model the local surface shape. The chosen function is fit separately at each vertex of the mesh, and the coefficients of the function are computed. A local 3D coordinate centered at the vertex is used for parameterization. Curvatures are then calculated by interrogating the polynomial surface fit rather than directly from the triangle mesh. Since differential properties are local, then this is all that is needed. This method has been one of the more popular and stable techniques for curvature estimation. Two of the most popular choices for the fitting function are tried, quadratic and cubic. They are explained in the following sections. As will be seen later in Chapter 6, the cubic fitting produces better results. Therefore, in this chapter, the output of the cubic method is used.

a. Quadratic fitting Various forms of quadratic function have been fitted to range data [75–78] and mesh representations [79, 80].The approximation is based on the fact that a surface can locally be represented as a graph of a bivariate function which is computed using the least squares approach. A general second-order polynomial with six coefficients, applied to a height function, is shown in Equation 18. Amini and Duncan [81] and Yokoya and Levine [82] used this local quadratic function for surface fitting. The same function is also used here to fit the

surface.

$$z_i = f(x_i, y_i) = Ax_i^2 + Bx_iy_i + Cy_i^2 + Dx_i + Ey_i + F \tag{18}$$

where (x_i, y_i) is the parametric location of the i th point in the tangent plane, and z_i is the height of the point above (or below) the tangent plane, measured along the normal direction . i runs from 1 to N, where N is the number of vertices being fit. The coefficients A through F are obtained using the standard least squares fit [83, 84]. The number of vertices to include in the least square fit should be decided. One approach is to use only the vertices of the one-ring neighborhood. Alternatively, the neighborhood can be expanded to include a specified number of vertices in the least-square fit. This larger number of vertices may be required based on the number of coefficients, or to improve the stability of the solution. The later method is used for including vertices in the surface fitting.

In matrix form, the coefficient vector $\vec{a}$ which minimizes the fit error is:

$$\vec{a} = (A^T A)^{-1} A^T \vec{Z} \tag{19}$$

where A is an $n \times 6$ matrix with row i being $[x_i^2 \;\; x_iy_i \;\; y_i^2 \;\; x_i \;\; y_i \;\; 1.0]$, $\vec{Z}$ is a column vector of the z_i values.

b. Cubic fitting Goldfeather et al. [85] expanded the quadric method by using a cubic fit of a system of equations formed from the coordinates and normal vectors at vertices on the neighborhood. They added normal vectors at adjacent vertices in order to create third-degree terms in the least-squares solution, which resulted in a better fitting surface.

In this method, first a cubic approximation approach is applied. A cubic polynomial is fitted to p, the vertex under consideration, and its neighboring vertices:

$$f(x, y) = \frac{A}{2}x^2 + Bxy + \frac{C}{2}y^2 + Dx^3 + Ex^2y + Fxy^2 + Gy^3 \tag{20}$$

Neighboring vertices of the vertex under consideration are obtained from

the k-link neighborhood, not including vertices, whose normal vector makes an obtuse angle with the normal vector at the vertex under consideration.

Each point in the selected neighborhood of p should fit in the surface, so Equation 20 can be rewritten:

$$(\frac{1}{2}x_i^2 \quad x_iy_i \quad \frac{1}{2}y_i^2 \quad x_i^3 \quad x_i^2y_i \quad x_iy_i^2 \quad y_i^3)b = z_i \tag{21}$$

where $b = (A \ B \ C \ D \ E \ F \ G)^T$, and $(x_i \ y_i \ z_i)$ are the coordinates of the points in the selected neighborhood of p.

In order to get more accurate results, the normal vectors calculated in the first step are also used. Therefore, two other approximation equations are added to the equation system , in which the real normal at each point should be equal to the normal at the surface of that point. $(a_i \ b_i \ c_i)$ indicates the normal vector at the point $(x_i \ y_i \ z_i)$, and normal to the surface at the same point $(x_i \ y_i \ z_i)$ is given by:

$$\begin{aligned} N(x_i, y_i) &= (f_x(x,y), f_y(x,y), -1) \\ &= (Ax_i + By_i + 3Dx_i^2 + 2Ex_iy_i + Fy_i^2, Bx_i + Cy_i + Ex_i^2 + 2Fx_iy_i \\ &+3Gy_i^2 - 1) \end{aligned} \tag{22}$$

The real normal vector can be rewritten as $(-\frac{a_i}{c_i}, -\frac{b_i}{c_i}, -1)$, and it should be equal to Equation 22 :

$$(-\frac{a_i}{c_i}, -\frac{b_i}{c_i}, -1) = (Ax_i+By_i+3Dx_i^2+2Ex_iy_i+Fy_i^2, Bx_i+Cy_i+Ex_i^2+2Fx_iy_i+3Gy_i^2-1) \tag{23}$$

which leads to the following equations:

$$(x_i \quad y_i \quad 0 \quad 3x_i^2 \quad 2x_iy_i \quad y_i^2 \quad 0)b = -\frac{a_i}{c_i} \tag{24}$$

$$(0 \quad x_i \quad y_i \quad 0 \quad x_i^2 \quad 2x_iy_i \quad 3y_i^2)b = -\frac{b_i}{c_i} \tag{25}$$

Then, a linear least-square method can be applied in order to solve the Equations 21, 24 and 25 as system $Ub = d$. Then b would be found. As it can be seen later, just A, B and C are used to drive the curvature extrema, since the curvature is only dependent on the second-degree terms in the local coordinate system.

4. Principal Curvatures and Directions

After a surface is fitted to the point and its neighbor vertices, principal curvatures and principal directions can then be extracted from the local surface.

Let P denote a point on 3D surface S. Suppose $X(u, v)$ is a local parameterization of S in a neighborhood of P. The partial derivatives of X with respect to u and v are denoted by $x_u(P)$ and $x_v(P)$. The unit normal vector $N(P)$ to the surface at point P is computed as:

$$N(P) = \frac{X_u(P) \times X_v(P)}{|X_u(P) \times X_v(P)|} \tag{26}$$

Using $(X_u(P), X_v(P), N(P))$ as a local orthogonal coordinate system, we can compute coefficients of the first fundamental form as:

$$E = X_u(P).X_u(P) \tag{27}$$

$$F = X_u(P).X_v(P) \tag{28}$$

$$G = X_v(P).X_v(P) \tag{29}$$

The coefficients of the second fundamental form are computed as:

$$e = N(P).X_{uu}(P) \tag{30}$$

$$f = N(P).X_{uv}(P) \tag{31}$$

$$g = N(P).X_{vv}(P) \tag{32}$$

The Weingarten curvature matrix at the point P can be computed as follows:

$$W = \begin{bmatrix} \frac{eG-fF}{EG-F^2} & \frac{fE-eF}{EG-F^2} \\ \frac{fG-gF}{EG-F^2} & \frac{gE-fF}{EG-F^2} \end{bmatrix} \tag{33}$$

when x_u and x_v are orthogonal unit vectors, then this becomes the symmetric matrix:

$$W = \begin{pmatrix} e & f \\ f & g \end{pmatrix} \tag{34}$$

If v is a unit vector in the tangent plane to S at P, then:

$$K_v = v^T W v \tag{35}$$

is the normal curvature of the surface at the point P in the direction of v.

Therefore, the eigenvalues γ_1 and γ_2 of the matrix W are the maximum and minimum principal curvatures of the surface at point P. The eigenvectors v_1 and v_2 are the corresponding maximum and minimum principal directions.

Since, x_u and x_v are orthogonal unit vectors, and regarding the Equation 20 as the local surface, the Weingarten matrix can be rewritten as:

$$W = \begin{pmatrix} A & B \\ B & C \end{pmatrix} \tag{36}$$

A, B and C are computed in the previous step. Having matrix W, we can obtain principal curvatures and principal directions by computing the eigenvalues and the eigenvectors of W. Principal curvatures and directions for a sample fingerprint scan are shown in Figures 29, 30, 31, 32, respectively.

D. Curvature Tensor Smoothing

In order to get a better result, a smoothing to the curvature tensors $k_{max}, k_{min}, t_{max}$ and t_{min} is also applied. In order to do so, the weighted average curvature tensor of the neighboring vertices is calculated. Weights are computed according to the distance from the neighboring vertex to the vertex under consideration. A smaller weight is assigned to the bigger distance and vice-versa. Finally, the new curvature tensor is the weighted summation of the original curvature tensor and that obtained from the averaging.

E. Ridge and Ravine Detection

Once principal curvatures and directions for each point are obtained, ridge and ravine points can be determined. Two different methods are tried in order to

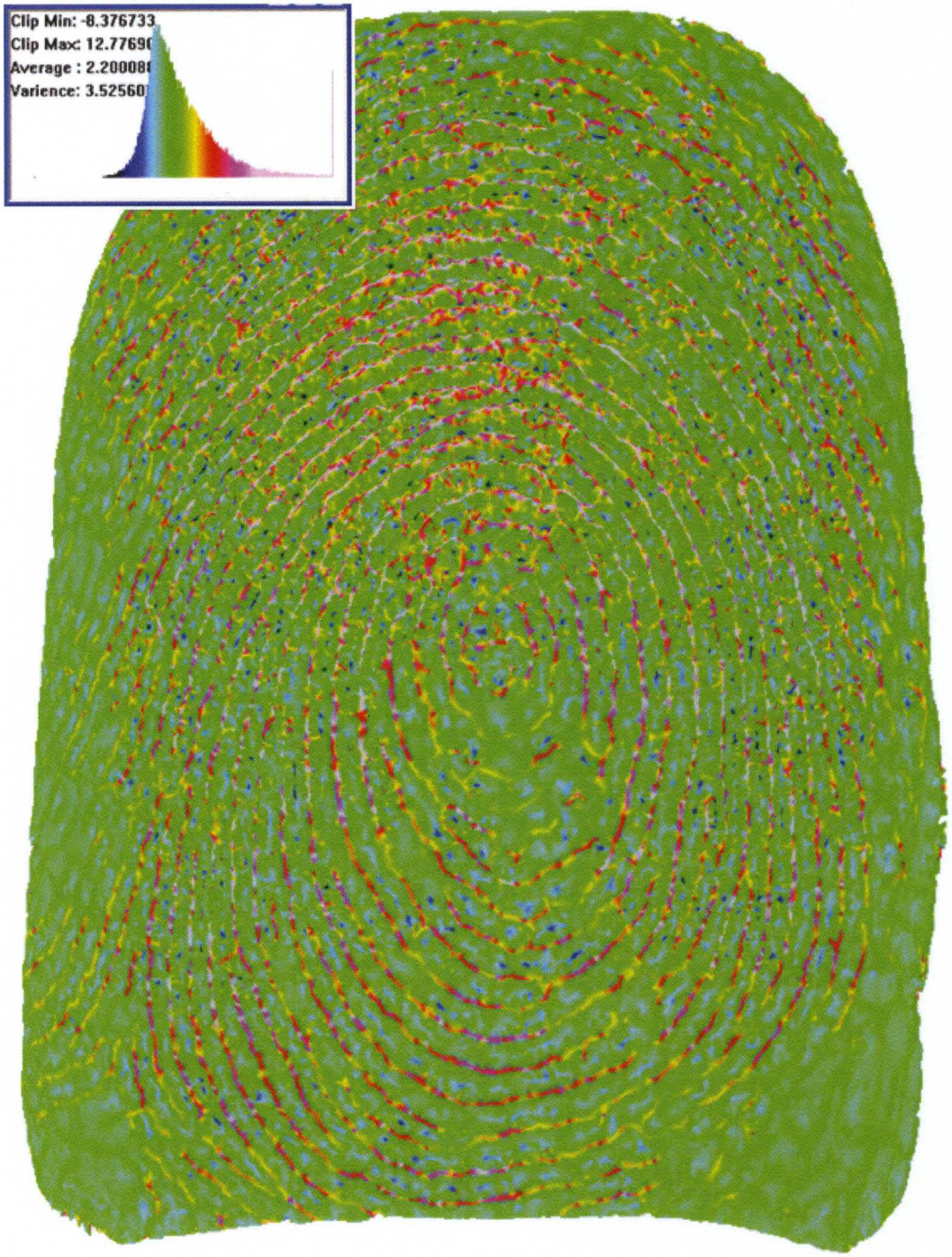


FIGURE 29 – Maximum principal curvature.

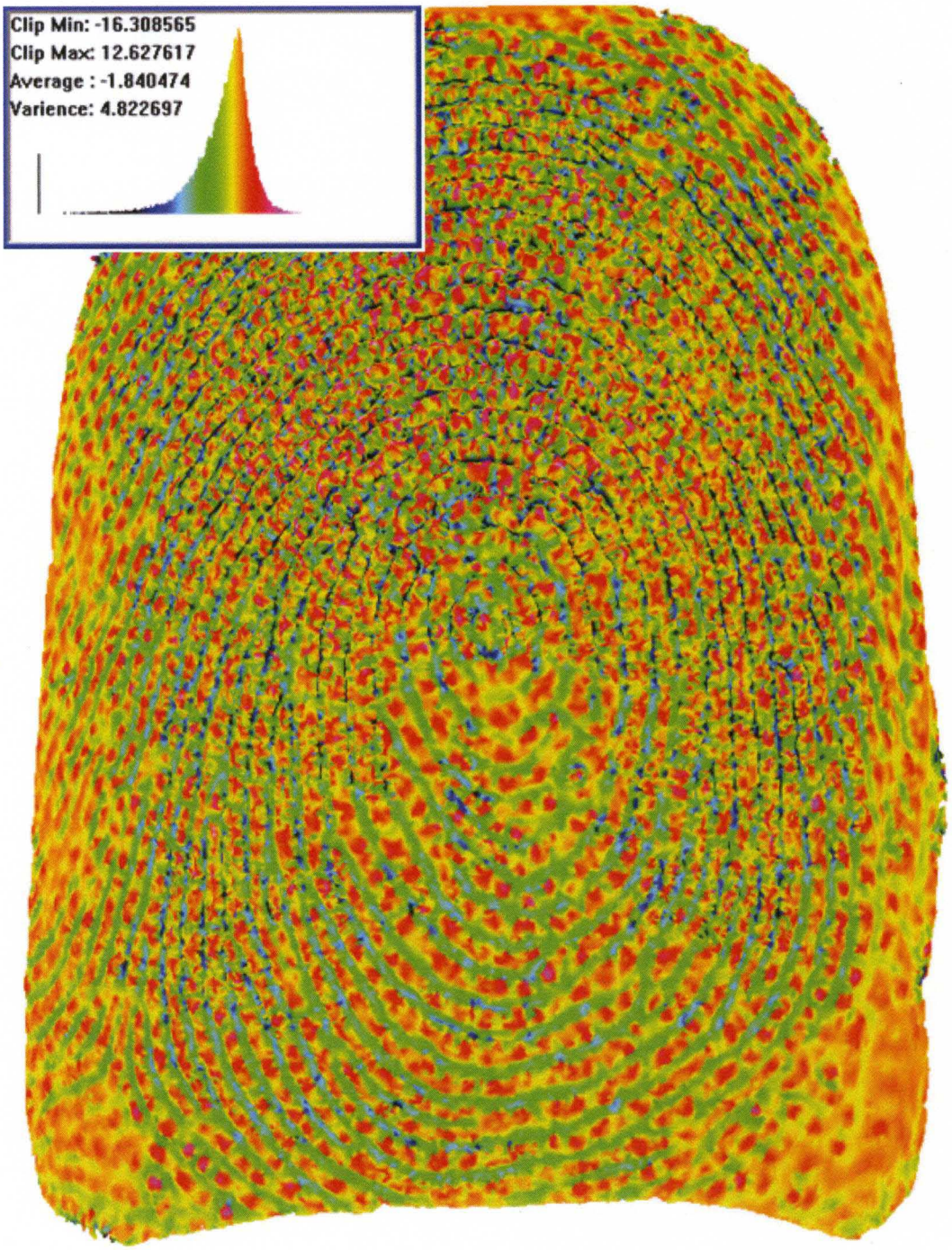


FIGURE 30 – Minimum principal curvature.

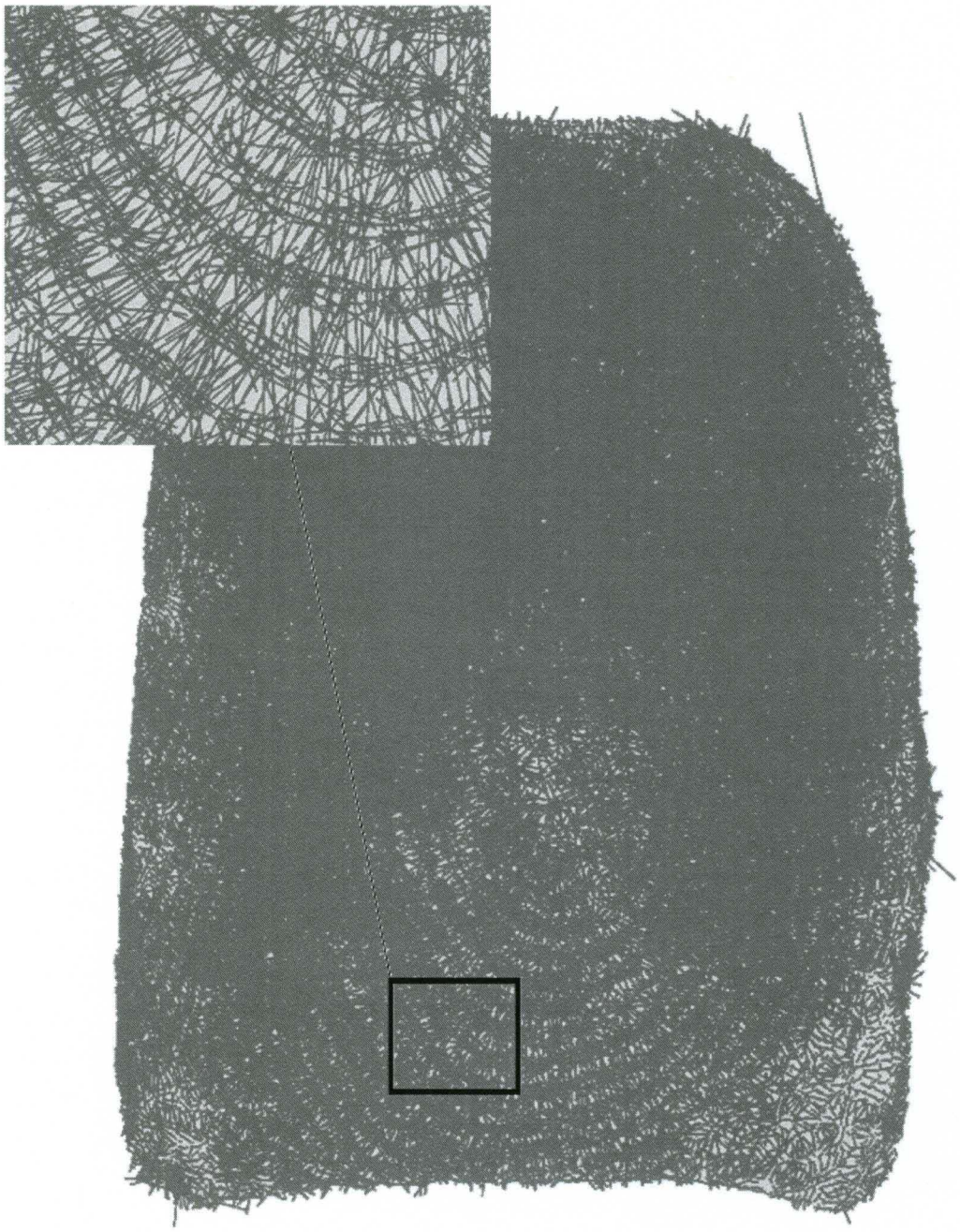


FIGURE 31 – Maximum principal direction.

FIGURE 32 – Minimum principal direction.

detect ridges and ravines. First method is the nonmaxima suppression technique. In the second method, derivatives of curvature principals are used to detect ridges and ravines. A comparison between two methods is discussed in Chapter 6.

F. First Method

The ridge points are selected by choosing the points that their maximal principal curvature attains the local positive maxima along its curvature line. The ravine points are selected by choosing the points that their minimal principal curvature attains the negative minimum along its curvature line. Ridges and ravines are dual according to the definition. Therefore, without loss of generality, only ridges are considered.

The nonmaxima suppression technique [86,87] is applied to obtain the ridge and ravine vertices. To decide whether k_{max} takes a maximum along its curvature associated with maximum principal direction t_{max} at vertex V, three steps are followed:

1- The intersection between the normal plane and the polygon that is composed by the first ring neighbors of P is obtained. The normal plane is generated by the normal vector at P and the maximal principal curvature at P.

2- These two surfaces intersect each other at two points: Q and S (Figure 33). Curvature values can be estimated at Q and S by linear interpolation. For example, curvature values at Q are obtained by linear interpolation of curvature values of V_i and V_{i+1}.

3- Finally, the maximum principal curvature at P is compared to the maximum principal curvatures at Q and S. If it is greater than both of them, then the maximum principal curvature attains a maximum at P along the normal section curve, and it is the ridge point.

The same process can be applied in order to find ravine points. Once the ridge/ ravine vertices are marked, it is also checked if they meet the following

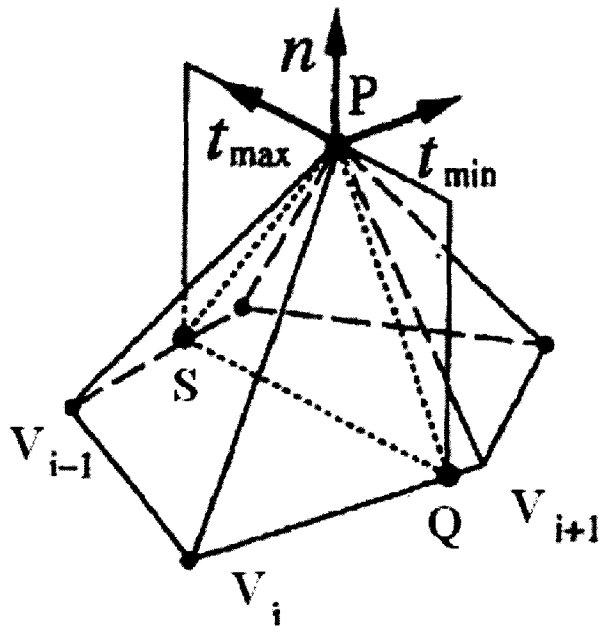


FIGURE 33 – Intersection between the normal plane and the polygon that is composed by the first ring neighbors of P.

conditions, and the vertices are unmarked if they do not. This is necessary in order to remove insignificant ridges.

$$Ridge : k_{max} > |k_{min}| \tag{37}$$

$$Ravine : k_{min} < -|k_{max}| \tag{38}$$

The final result is shown in Figure 18, in which ridge and ravine points are represented by red and blue dots, respectively.

G. Tracing the Ridge and Ravine Lines

In order to reduce fragmentation of the ridges, those vertices are marked as ridge vertices that have two ridge neighbors. Assume that V_i, $V_{i+1}$and V_{i+2} are three ordered neighbor vertices of i, in which V_{i+1} neighbors V_i and V_{i+2} as shown in Figure 35. For each non-ridge vertex i, if V_i and V_{i+2} are both ridge vertices and V_{i+1} is not a ridge vertex, then either i or V_{i+1}, whichever has bigger k_{max}, would

FIGURE 34 – Ridge (red dots) and ravine (blue dots) vertices.

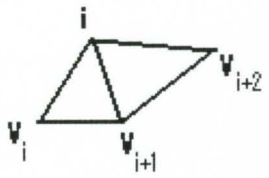


FIGURE 35 – Vertex i and three neighboring vertices.

be the new ridge vertex.

The same procedure can be applied to all the ravines. k_{min} value of V_i and V_{i+2} are compared and then choose whichever is smallest.

Having all ridge and ravine vertices, we can then trace the vertices and connect them together. In this step, the gaps between the ridges/ravines are filled. In order to do so, a simple modification of Dijkstra's algorithm [88] is applied in order to find the shortest path between two ridge ends. To generate the weights for Dijkstra's algorithm, the following formula [89] is used for ridge detection and to assign a weight to each edge in the original mesh:

$$w(P,Q) = [\frac{(k_{max}(P) - K)^2 + (k_{max}(Q) - K)^2}{2} + \frac{(k_{max}(P) - k_{max}(Q))^2}{6}]||PQ|| \quad (39)$$

where P and Q are two vertices of the edge PQ in the mesh. $||PQ||$ is the length of edge PQ. K is assigned the largest maximal curvature of the region. While Dijkstra's algorithm is being applied, all the vertices in the neighborhood are not visited. Only those ridge vertices are visited and expanded that meet the following condition:

$$Ridge : k_{max} > |k_{min}| \quad (40)$$

$$Ravine : k_{min} < -|k_{max}| \quad (41)$$

The final result is shown in Figure 36, in which red lines show ridges and blue lines show ravines.

FIGURE 36 – Tracing ridge (red) and ravine (blue) lines.

H. Second Method

In this method, curvature principals and their derivatives are used to detect ridges and ravines.

Denote S to be a given smoothed oriented surface and P to be a point on S. The maximal and minimal curvatures of S at P are called k_{max} and $k_{min}(k_{max} \geq k_{min})$. Let t_{max} and t_{min} be the corresponding principal directions which are the associated tangent directions of S at P. e_{max} and e_{min} denote the derivatives of the principal curvature along their corresponding curvature directions:

$$e_{max} = \partial k_{max}/\partial t_{max} \qquad e_{min} = \partial k_{min}/\partial t_{min} \tag{42}$$

The point at which the principal curvatures are equal to each other ($k_{max} = k_{min}$) is called umbilic. e_{max} and e_{min} are not defined at umbilic points, because the principal directions are undefined there. A non-umbilic point P is called a ridge point if k_{max} attains a local maximum at P along the corresponding principal direction t_{max} . A non-umbilic point P is called a valley point if k_{min} attains a local minimum at P along the corresponding principal direction t_{min} . Considering these definitions, the ridges and ravines are characterized as:

Ridges:

$$e_{max} = 0, \quad \partial e_{max}/\partial t_{max} < 0, \quad k_{max} > |k_{min}| \tag{43}$$

Ravines:

$$e_{min} = 0, \quad \partial e_{min}/\partial t_{min} > 0, \quad k_{min} < -|k_{max}| \tag{44}$$

Note that if surface orientation is changed, then ridges turn into the valleys and vice versa.

1. Computing e_{max} and e_{min}

The well-known formula to compute an extremality coefficient $e = \partial k/\partial \mathbf{t}$ for surface given in implicit form $F(\mathbf{X}) = 0$, $\mathbf{X} = (x_1, x_2, x_3)$ is:

$$e = \triangle k.\mathbf{t} = \frac{F_{ijl}t_i t_j t_l + 3kF_{ij}t_i t_j}{|\triangle F|} \tag{45}$$

where F_{ij} and F_{ijl} denote the second and third partial derivatives of $F(x)$, respectively. $\mathbf{t} = (t_1, t_2, t_3)$is the principal direction corresponding to a principal curvature k, $\mathbf{n} = (n_1, n_2, n_3)$ is the unit surface normal, and the Einstein summation convention is used.

Some papers like [90] use Equation 45 to compute e_{max} and e_{min}, but here we use the elegant formula from [91] which is computationally less expensive and faster:

$$e = \partial k/\partial t = \begin{pmatrix} t_1^2 \\ t_2^2 \end{pmatrix}^T \begin{pmatrix} 6D & 2E \\ 2F & 6G \end{pmatrix} \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} \tag{46}$$

where $t = (t_1, t_2)^T$ is the principal direction corresponding to the principal curvature k, and surface is locally approximated by:

$$f(x, y) = \frac{A}{2}x^2 + Bxy + \frac{C}{2}y^2 + Dx^3 + Ex^2y + Fxy^2 + Gy^3 \tag{47}$$

Equation 46 is derived from Equation 45. In our case, $F = z - f(x, y)$ where $f(x, y)$ is defined in Equation 47 and $\mathbf{n} = (0, 0, 1)$ and $\mathbf{t} = (t_1, t_2, 0)$ at the origin of coordinates. $f(x, y)$ does not contain linear terms, at the origin of coordinates, therefore Equation 45 would be :

$$e = F_{ijl}t_i t_j t_l \tag{48}$$

which results in Equation 46.

2. Tracing the ridges and valleys

Once the principal curvatures, directions and their derivatives e_{max} and e_{min} are calculated for each vertex of the mesh, we are ready to extract the crest points

with checking if the each edge of mesh contains curvature maxima or minima. To do so, we applied the approach proposed by [88].

To detect the ridge points, for each edge $e = (v_1, v_2)$, the following condition is checked:

$$k_{max}(v) > |k_{min}(v)| \quad for \quad v = v_1, v_2 \tag{49}$$

$$e_{max}(v_1)e_{min}(v_2) < 0 \tag{50}$$

$$e_{max}(v_i)[(v_{3-i} - v_i).t_{max}(v_i)] > 0 \qquad with \quad i = 1or2 \tag{51}$$

$e_{max}(v_1)e_{min}(v_2) < 0$ implies whether e_{max} has a zero crossing on edge e, and Equation 51 checks whether e_{max} obtains a maximum on edge e. If the aforementioned equations are satisfied then a linear interpolation is used to find a zero-crossing of e_{max} on edge e, which would be a ridge point.

$$p = \frac{|e_{max}(v_2)|v_1 + |e_{max}(v_1)|v_2}{|e_{max}(v_1)| + |e_{max}(v_2)|} \tag{52}$$

Same procedure can be applied to obtain valley points.

Finally, if two crest points are detected on the two edges of mesh triangle, those crest points would be connected by a straight line. If three crest points are detected on three edges of triangle, then crest points are connected by the centriod of the triangle. The final result is shown in Figure 37, in which red lines show ridges and blue lines show ravines. Since, first method gives better results, the result of first method is used for the following sections. The comparison between two methods will be discussed in Chapter VI.

I. Post Processing and Cleaning

To prevent false minutiae detection, it is necessary to clean the ridges and ravines before going to the minutiae detection step. Three different cleaning processes are applied:

- Removing the short ridges

FIGURE 37 – Ridge (red) and ravine (blue) lines for second method.

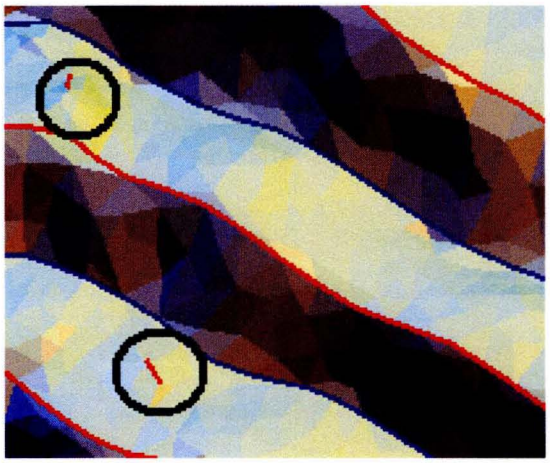


FIGURE 38 – A sample of short ridges

- Removing the short branches
- Connecting the broken ridges/ ravines

According to the ridge pattern of the fingerprint, very short ridges are not real ridges. Short ridges may be caused by some noise or other artifacts during scanning. A example of short ridges is displayed by Figure 38 . Therefore, the ridge length for each ridge is computed and compared to a specific threshold and short ridges are then deleted. Figure 42 demonstrates ridges and ravines after deleting the short ridges. The real size of the ridge length in the physical scan is used, which gives us more accurate results. Also, the length of branches in the ridges/ravines is calculated. Figure 39 illustrates a sample of short branches. Once compared to a certain threshold, then the short branches are eliminated (Figure 41).

Finally, the remaining gaps between ridges /ravines. Figure 40 shows a sample of gaps need to be filled. Here a modified Dijkstra's algorithm is applied with different conditions. First, Dijkstra's algorithm is applied to find the shortest path from one ridge/ravine end to the nearest ridge/ravine end. If the distance between them is less than a specific threshold and the direction of two ridges/ravines is almost identical, then two ridge/ravine ends are connected. The result is shown

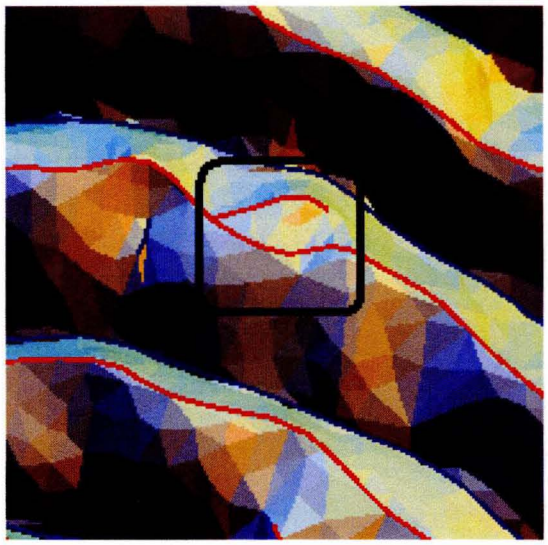


FIGURE 39 – A sample of short branch

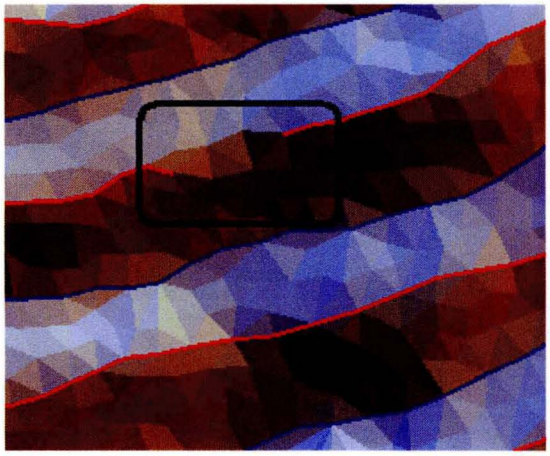


FIGURE 40 – A sample of gaps between the ridges

in Figure 43. Figure 44 shows the difference before and after post-processing in the zoomed area.

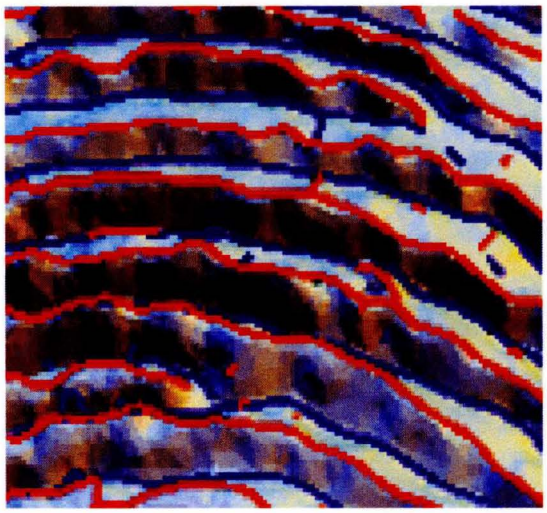
J. Minutiae Detection

After obtaining clean and connected ridges and ravines, minutiae can be detected. We look for two kinds of minutiae: termination and bifurcation. Termination happens when a ridge suddenly comes to an end; bifurcation happens when a ridge divides into two [3]. As mentioned in Chapter 2, most fingerprint

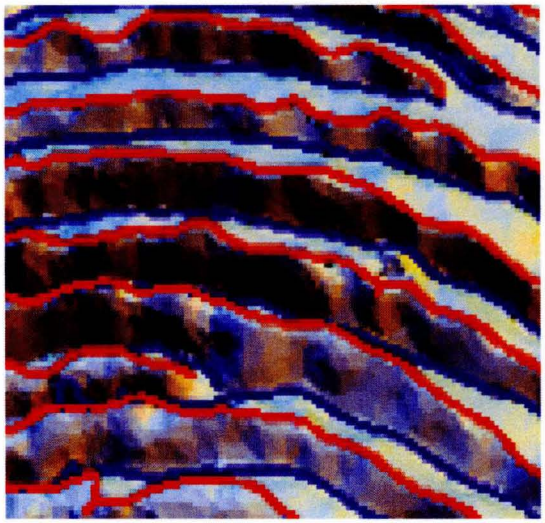
FIGURE 41 – Ridges (red lines) and ravine (blue lines) after removing short branches.

FIGURE 42 – Ridges (red lines) and ravines (blue lines) after removing short ridges.

FIGURE 43 – Ridges (red lines) and ravines (blue lines) after filling gaps in the ridges.



(a)

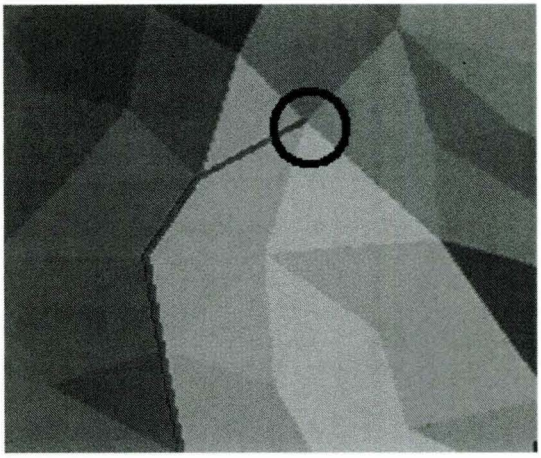


(b)

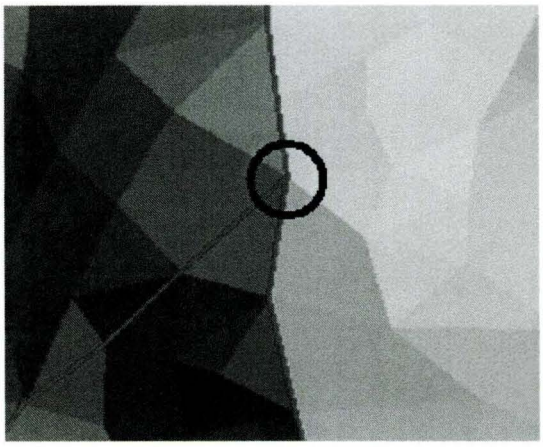
FIGURE 44 – (a) Before cleaning (b) after cleaning

matching systems are based on these two kinds of minutiae. First, the degree is computed for each vertex. The degree of a vertex is the number of ridge edges incident to the vertex. For example, a vertex with a degree of two shows that the vertex is in the middle of the ridge. Therefore, vertices with a degree of one correspond to termination minutiae as shown in Figure 45(a). Vertices that having a degree of three correspond to bifurcation minutiae as shown in Figure 45(b). All one-degree ridge vertices are marked as low quality termination; all three-degree ridge vertices are marked as low-quality bifurcation.

In order to mark them as high-quality minutiae, the termination and bifurcation in the ravines are used. Ridges and ravines are dual. Hence, a ridge termination usually happens when ravine bifurcation happens nearby (Figure 46(a)); a ridge bifurcation usually occurs close to the ravine termination (Figure 46(b)). This fact is used to obtain reliable minutiae. If a ravine bifurcation is found close to the ridge termination, then it is marked as a high quality termination. If we can find a ravine termination close to ridge bifurcation, then we mark it as a high quality bifurcation. The result is shown in Figure 47, in which red dots show reliable minutiae and blue dots show low-quality minutiae. The blue minutiae might be ignored for matching purpose. In Figure 48, the low-quality minutiae are removed and only the high-quality minutiae are demonstrated.

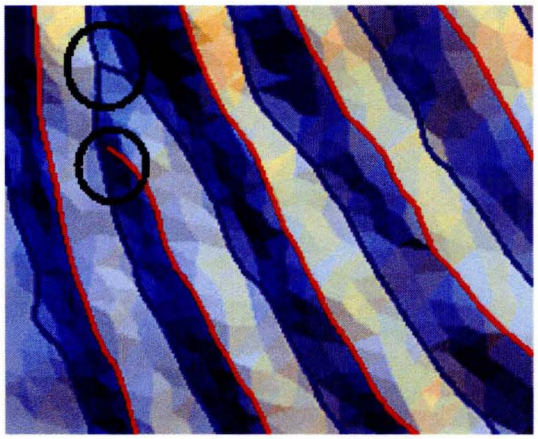


(a) termination



(b) bifurcation

FIGURE 45 – (a) Vertex with degree of one (termination). (b) Vertex with degree of three (bifurcation).



(a) termination

(b) bifurcation

FIGURE 46 – (a) Ridge termination (red line) close to ravine bifurcation (blue line). (b) Ridge bifurcation close to ravine termination.

FIGURE 47 – Minutiae detection. Red dots demonstrate reliable minutiae and blue dots show low-quality minutiae.

FIGURE 48 – Minutiae detection. After removing the low-quality minutiae.

CHAPTER V

QUALITY MAP

As seen in Figure 48, there are lots of false minutiae, especially at the edge of the fingerprint. Since degraded areas may happen during the scanning process, especially at the edge of the fingerprint, it is critical to be able to determine these areas. Therefore, false minutiae can be avoided. In order to identify the bad quality area, we propose a method that measures the quality of 3D fingerprint scans. We define fingerprint quality to mean the quality of detected minutiae. Several characteristics are designed to convey information that measures the quality in the 3D fingerprint scans. These include detecting regions with low-depth information and low-flow direction. Such characteristics represent unstable areas in the fingerprint scan where detected minutiae are unreliable. Also they can be used to represent the levels of quality in the 3D scan. Before defining such characteristics, it is necessary to divide the 3D scan into blocks.

A. Blocking

In order to analyze the quality of the fingerprint locally, the 3D scan is divided into blocks. All vertices inside the block are assigned the same value. First, it must be determined how many vertices for each block are necessary to reliably derive the desired characteristics. Different block sizes are tried and thirty vertices gave the best results.

In the first step of the blocking process, each vertex is checked. If it is not already assigned to the block, then the current block number is assigned to the vertex. Next, the first-ring neighborhood of the vertex under consideration is checked.

Any of the vertices that are not assigned to another block are assigned the same block number of the vertex under consideration. The number of vertices in each block needs to be equal to the threshold number for the block size. Therefore, if the number of vertices is less than the block size, then the second-ring neighborhood is checked and the same process is applied. This will continue in the next neighborhood untill the number of vertices is equal to the block size, or there are no non-assigned vertices in the neighborhood. Hence, the size of some blocks is less than the threshold. This may result in small blocks, which have inaccurate results. In order to solve this problem, these small blocks are merged to the neighbor block with the smallest size. The result of blocking is shown in Figure 49.

B. Low-depth information map

The purpose of this map is to represent areas of the 3D scan that have sufficient ridge structure. Clearly visible and well-formed ridges and ravines are necessary to reliably detect minutiae. Sometimes, there is no sufficient depth information to create a well-formed ridge structure. This especially happens at the edge of fingerprint scans. The minutiae that are detected in these areas are not reliable. To avoid detecting the false minutiae in these areas, we need to identify them. For this purpose, the following steps are proposed, which include: curvature tensor detection, ridge and ravine detection, identifying the depth value for each vertex , and finally obtaining the value for the whole block to assign to each vertex. Using blocks instead of each vertex individually gives a more accurate and smooth results.

Curvature tensor for each vertex is obtained in Section C of Chapter 4; ridge and ravine detection is explained in Section E of Chapter 4. Once the ridge and ravine points are computed, the following conditions are checked for each ridge and ravine vertex:

$$ridge\ \ vertex: \qquad k_{max} > T \tag{53}$$

FIGURE 49 – Dividing the 3D fingerprint scan into blocks. Size of the blocks are 30 vertices. The blocks are shown by different color.

$$ravine\ vertex: \qquad k_{min} < -T \tag{54}$$

where T is the threshold according to the physical normal ridge depth of a fingerprint.

If these conditions are met for the vertex, then the vertex is marked as a good vertex, otherwise it is marked as a bad vertex. The result for the block is obtained by taking the average of the value for all vertices inside the block. Afterwards, the result is assigned to all vertices inside the block. Therefore, all vertices inside each block have the same quality.

It is desirable to share data used to compute the results with neighboring blocks. This way, part of the mesh that contributed to one block's results is included in the neighboring block's results as well, which helps minimize discontinuity. This smoothing can be implemented by taking a weighted averaging between the block under consideration and the neighboring blocks. The result for a low-depth information map is shown in Figure 50. The low-quality areas are represented by black; high-quality areas are shown by white.

C. Low-direction flow

The purpose of this map is to detect the areas in which ridge directions are not well-formed, where there are no clearly defined ridges. Minutiae detection within these areas is problematic. A map called the low-direction flow map is computed, in which blocks of sufficiently low flow-direction are flagged.

One of the fundamental steps for extracting a low-direction flow map is to derive a directional ridge flow map. The map is computed using principal directions and principal curvatures of the vertex. The following formulas are used in order to extract the ridge and ravine directions:

$$ridge\ direction[i] = \frac{\partial k_{max}[i]}{\partial t_{min}[i]} \times t_{max}[i] - \frac{\partial k_{max}[i]}{\partial t_{max}[i]} \times t_{min}[i] \tag{55}$$

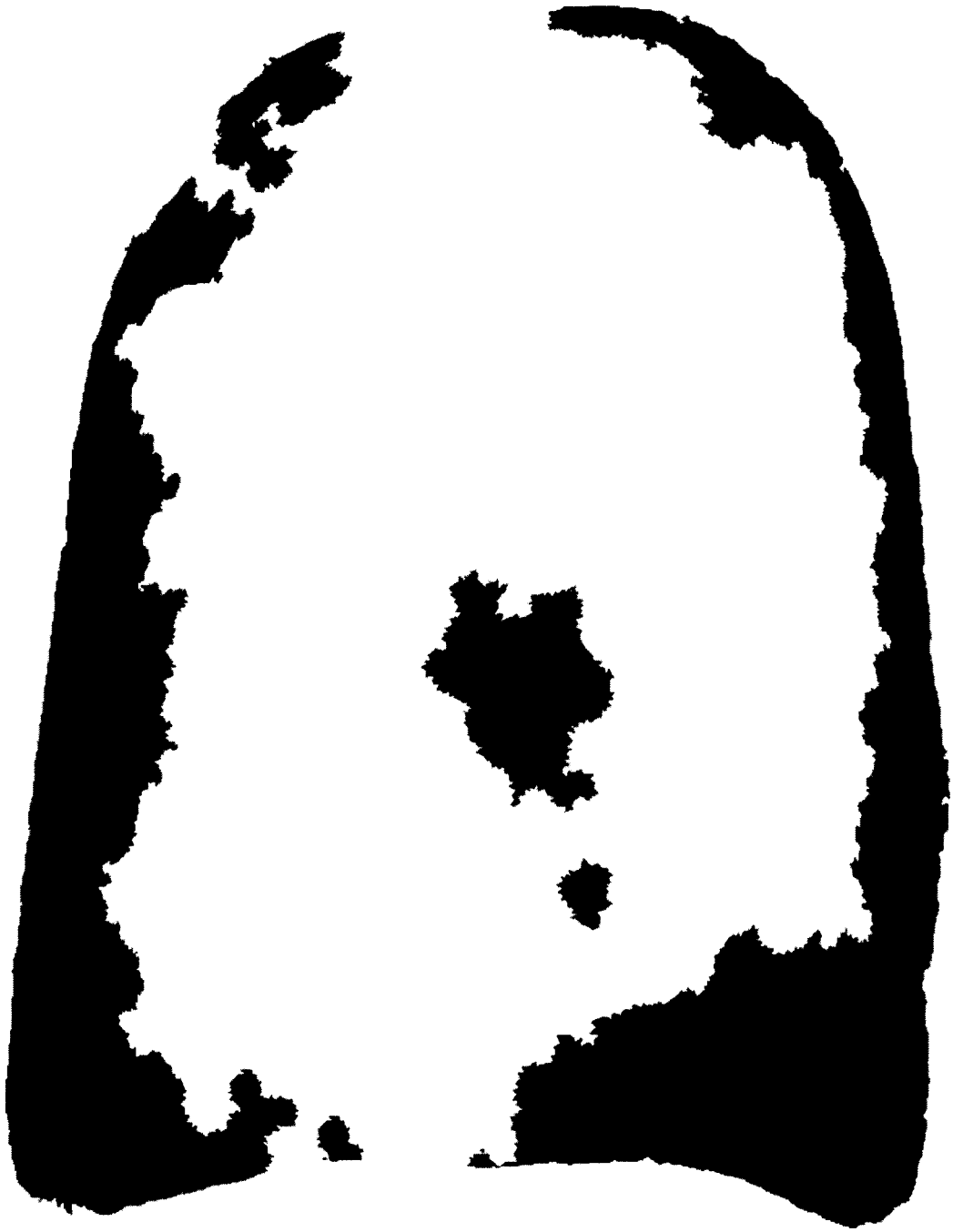


FIGURE 50 – Low-depth information map. White indicates good quality,while black shows poor quality.

$$ravine\ \ direction[i] = \frac{\partial k_{min}[i]}{\partial t_{max}[i]} \times t_{min}[i] - \frac{\partial k_{min}[i]}{\partial t_{min}[i]} \times t_{max}[i] \tag{56}$$

where i is the vertex under consideration.

The direction map is shown in Figure 51. Once the ridge directions for the ridge vertices and the ravine directions for the ravine vertices are optained, then the next step is to compare these directions inside a block to find low-direction flow blocks. In order to do so, all of the vertices inside a block are checked. Aligned vertices and non-aligned vertices are recognized; the average is taken over the vertices and the result for the block is obtained. Finally, the result is assigned to all of the vertices inside the block.

In order to determine the aligned and non-aligned vertices, the direction of the vertex is compared to the direction of other vertices inside the block. If the angle between them is less than a certain threshold, then the vertex is marked as an aligned vertex. Otherwise, it is marked as a non- aligned vertex. Again, like a low-depth information map, the results in the neighboring blocks are used for the block. For this purpose, a weighted averaging between the block under consideration and the neighboring blocks is applied. The result of low-flow direction is shown in Figure 52. The black area shows poor quality and the white part shows good quality.

D. Overall quality map

The final quality map is produced by integrating the low-depth information map and the low flow direction map into one general map. Figure 53 shows the final result. The quality assigned to a specific block is determined based on its proximity to blocks flagged in these various maps. Finally, the quality map is applied to the detected minutiae. The result is shown in Figure 54. Red shows reliable minutiae detected in good quality areas and black indicates unreliable minutiae detected in low-quality areas.

FIGURE 51 – Direction map.

FIGURE 52 – Low-flow direction map. White shows good quality and black indicates poor quality.

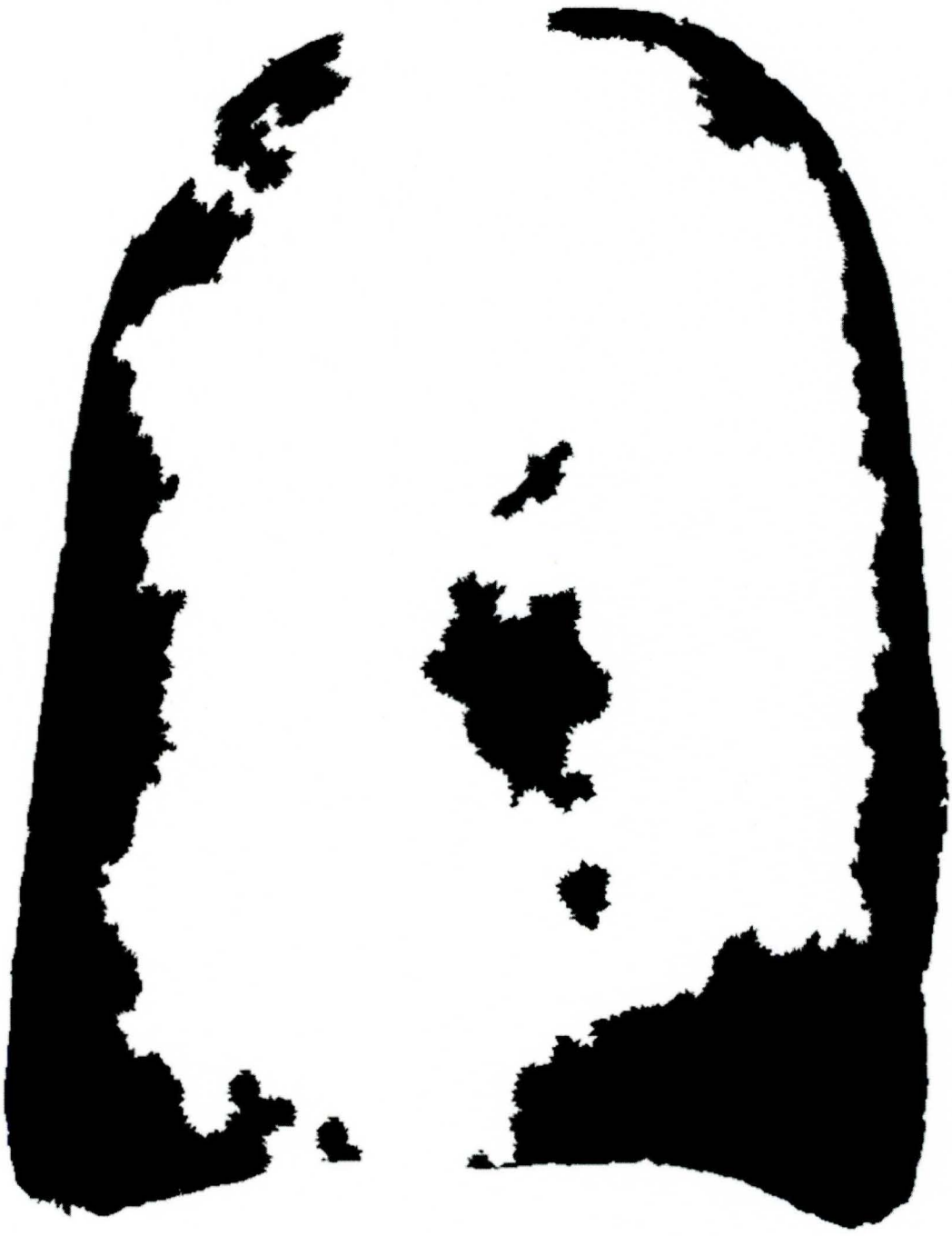


FIGURE 53 – Overall quality map. White indicates good quality and black refers to poor quality.

FIGURE 54 – Minutiae detection after applying quality map. Red dots show reliable minutiae detected in high quality areas and black dots demonstrate minutiae detected associated with poor-quality areas.

CHAPTER VI

TESTING AND EVALUATION

The proposed 3D minutiae extraction procedure has been tested on a 3D fingerprint database consisting of 100 fingerprint scans. All fingers were scanned using the SLI 3D fingerprint scanner provided by Flashscan3D LLC and the University of Kentucky [2]. The camera resolution of the scanner was 1392 pixels $\times$ 1040 pixels (H $\times$W), where, depending on the depth, the lateral spacing between points typically varies from 20 to 25 μm.

Row data is in Mat5 format, which includes five files that end with: *C.bmp, *I.bmp, *X.byt, *Y.byt, *Z.byt. The first file is the color image, sometimes called the texture map in BMP format. The second file is the "Indicator" or "quality" matrix. The format for this is also in BMP. This Indicator matrix is black and white in value. The remaining three files are X, Y, and Z. They contain no header information and use one float value for each element. The X, Y and Z matrices contain the X, Y and Z coordinates, respectively. Using a simple Matlab code, they were converted to pts format, which can be used as input for the triangulation process. The method proposed by [92] is used to make the triangle mesh from data points. The method is based on the Garland-Heckbert local quadric error minimization strategy.

In this chapter, first the results of the two different fitting methods: quadratic and cubic are compared. Then, the comparison is performed between two different methods for detecting ridges and ravines, which are explained in Chapter IV.

A. Statistical results

The efficiency of the minutiae extraction can be measured by how well it

TABLE 2
MEAN AND STANDARD DEVIATION OF THE SENSITIVITY AND SPECIFICITY FOR CUBIC FITTING

	Sensitivity(%)	Specificity(%)
Mean Value	88.3	79.5
Standard Deviation	6.8	9.1

meets the objective of detecting the minutiae from the original image by a fingerprint expert. The proposed algorithm is tested by using two quantity measures, namely Sensitivity and Specificity, which indicate the ability of the algorithm to detect the genuine minutiae and remove the false minutiae respectively. Sensitivity and specificity are defined as follows [93]:

$$Sensitivity = 1 - \frac{Missed\ Minutiae}{Ground\ Truth\ Minutiae} \tag{57}$$

$$Specificity = 1 - \frac{False\ Minutiae}{Ground\ Truth\ Minutiae} \tag{58}$$

Missed minutiae are those that could not be detected by the algorithm; false minutiae are that do not exist. The "ground truth" consists of the fingerprint minutiae manually detected in each fingerprint image, as exampled in Figure 55.

The sensitivity and specificity were measured over the entire database; the mean and standard deviation were calculated. Table 2 presents the results of the cubic fitting along with the first method ridge and ravine detection. Results for quadratic fitting and first method of ridge and ravine detection are reported in Table 3. Studying Tables 2 and 3 reveals that cubic fitting gives better results. Therefore, the cubic fitting is chosen in order to compare the two methods for detecting ridges and ravines.

The sensitivity and specificity were also obtained for the second method of detecting ridges and ravines; the results are reported in Table 4. Comparing Tables 2 and 4 shows that the best results were observed for the first method of ridge and ravine detection along the cubic fitting. The high values of sensitivity and

TABLE 3
MEAN AND STANDARD DEVIATION OF THE SENSITIVITY AND SPECIFICITY FOR QUADRATIC FITTING

	Sensitivity(%)	Specificity(%)
Mean Value	75.8	43.2
Standard Deviation	8.3	14.2

TABLE 4
MEAN AND STANDARD DEVIATION OF THE SENSITIVITY AND SPECIFICITY FOR SECOND METHOD OF RIDGE AND RAVINE DETECTION

	Sensitivity(%)	Specificity(%)
Mean Value	55.8	71.6
Standard Deviation	13.9	11.5

specificity suggest the effectiveness of the proposed technique.

B. Visual considerations

Figure 55 shows the minutiae manually detected on a fingerprint of the database. The minutiae located in regions with poor contrast are neglected, where minutiae detection cannot be performed even manually. Figure 56 shows the automatic extraction through the proposed algorithm, mentioned in this research on the same fingerprint. The Figure 56 was obtained by using cubic fitting, along with the first method for detecting ridges and ravines. In this figure, only the good quality minutiae in the high quality area are demonstrated. Comparing these two pictures, there are one missed minutiae and two false minutiae. Sensitivity and specificity for this Figure are 0.97 and 0.94, respectively.

Figure 57 demonstrates detected ridges and ravines for the same sample used in Figure 55 by using quadratic fitting. Red lines show ridges and blue lines show ravines. The quality map for this fingerprint is shown in Figure 58, in which white indicates high-quality areas, and poor-quality areas are shown by

FIGURE 55 – The minutiae manually detected on a fingerprint.

FIGURE 56 – Automatic extraction through our algorithm.

black. Figure 59 demonstrates minutiae detection on this sample after applying the quality map to the detected minutiae. Black dots illustrate minutiae associated with poor-quality regions; red dots indicate high-quality minutiae associated with high quality regions. Finally, Figure 60 shows detected minutiae after removing the low-quality minutiae. Sensitivity and specificity for this sample are 0.83 and 0.40, respectively.

Figures 61 - 64 illustrate the results for the second method of detecting ridges and ravines as well as cubic fitting. Figure 61 demonstrates ridge lines (red) and ravine lines (blue). The quality map for this method is shown in Figure 62, in which white shows high-quality areas, and poor-quality areas are displayed in black. Figure 64 demonstrates minutiae detection after applying a quality map, in which black dots show minutiae associated with poor-quality regions while red dots show high-quality minutiae associated with high-quality regions. Finally, Figure 64 shows detected minutiae after removing the low-quality minutiae. Sensitivity and specificity for this sample are 0.64 and 0.70, respectively.

There are more samples from the database shown in Figures 65 - 72. Figures 65 and 66 show the ridges(red) and ravines(blue) after post processing and cleaning. The quality maps for these samples are demonstrated in Figures 67 and 68 with white for high-quality areas and black for poor-quality areas. Figures 69 and 70 illustrate detected minutiae after applying the quality map. Black minutiae are associated with poor-quality areas and red minutiae with high-quality area. Finally, Figures 71 and 72 show the final results after removing low-quality minutiae.

C. Algorithm Performance

The algorithms have been executed on an Intel(R)Xeon(R) CPU 5140 (2.33 GHz) PC with 8 GB RAM. These algorithms are compiled by using Microsoft Visual C++ 2008. For each algorithm, the following statistics have been evaluated on the whole database: average execution time and standard deviation of the ex-

FIGURE 57 – Ridges (red) and ravines (blue) lines detected using quadratic fitting.

FIGURE 58 – Quality map for the fingerprint in Figure 57. High-quality area (white) and poor-quality area (black).

FIGURE 59 – Minutiae detection for the fingerprint in Figure 57. Black dots show minutiae associated with poor-quality areas and red dots show minutiae associated with high-quality areas.

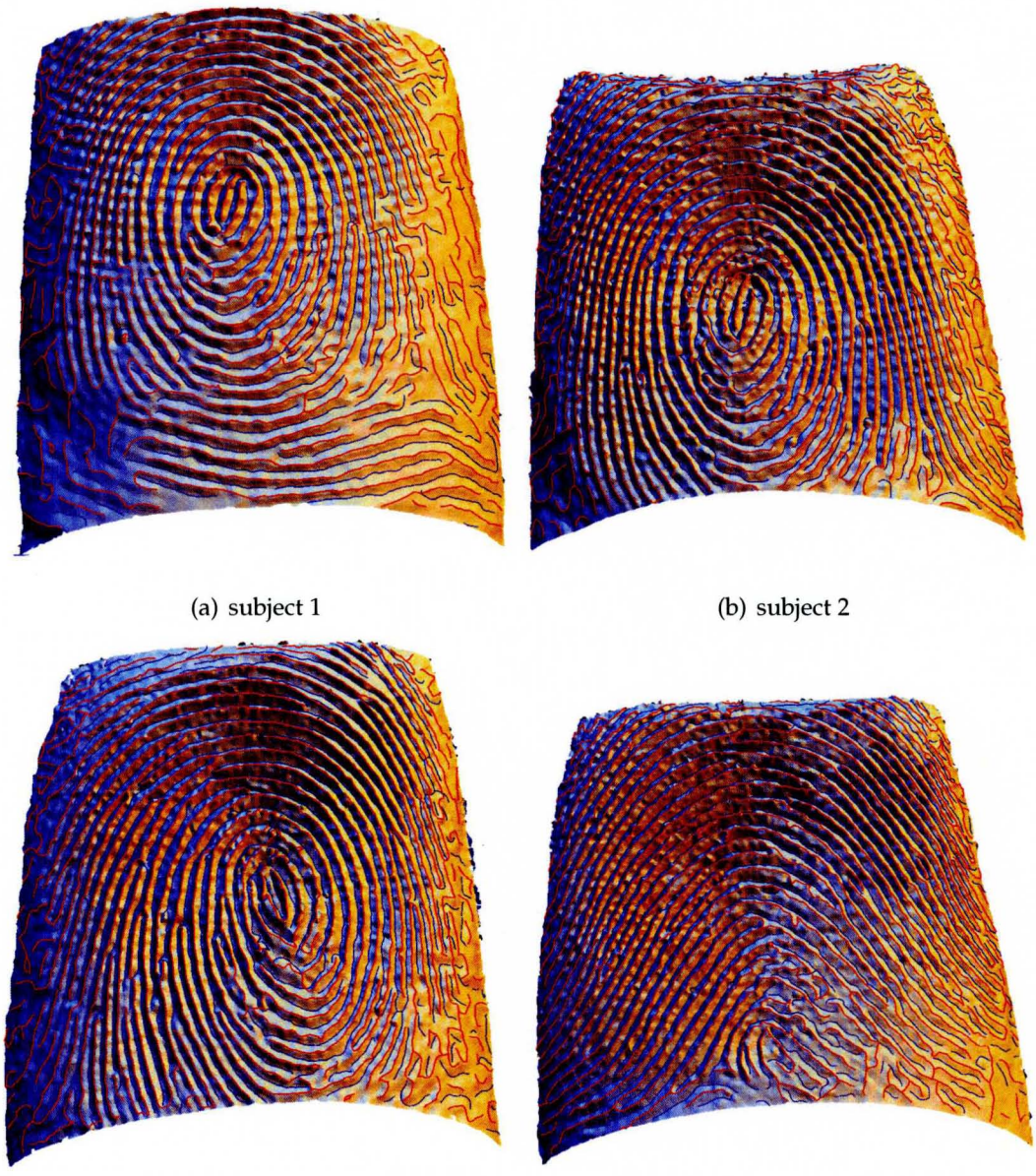
FIGURE 60 – Minutiae detection after removing the low-quality minutiae of Figure 59.

FIGURE 61 – Ridges (red) and ravines (blue) lines of Figure 37 after post processing.

FIGURE 62 – The quality map for the fingerprint in Figure 61. High-quality areas displayed in white and poor-quality areas displayed in black.

FIGURE 63 – Minutiae detection for the fingerprint in Figure 61. Black dots indicate minutiae associated with poor-quality areas; red dots refer to minutiae associated with high-quality areas.

FIGURE 64 – Minutiae detection after removing the low-quality minutiae of Figure 63.



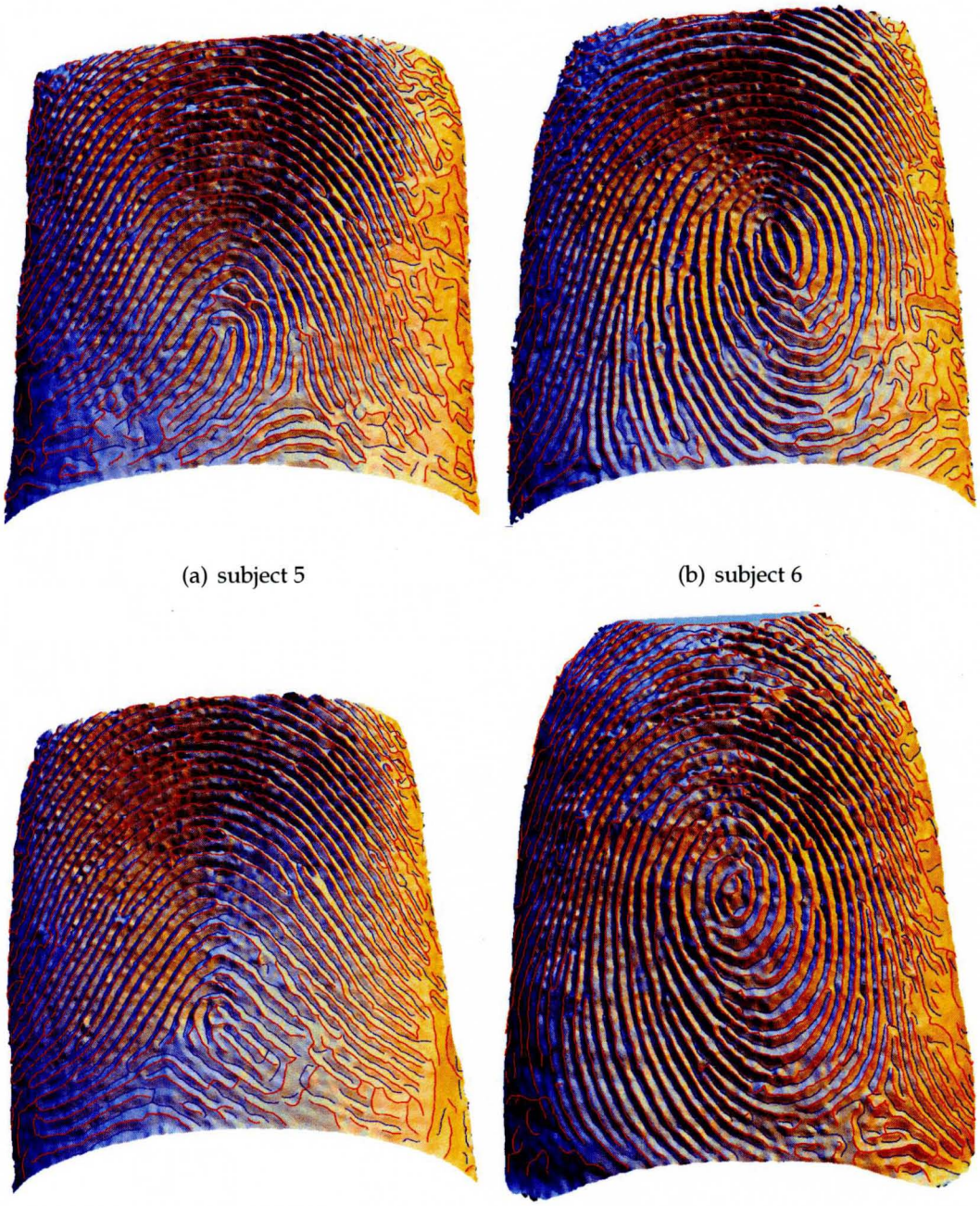
(a) subject 1

(b) subject 2

(c) subject 3

(d) subject 4

FIGURE 65 – Ridges (red) and ravines (blue) lines after post processing for four samples of the database.



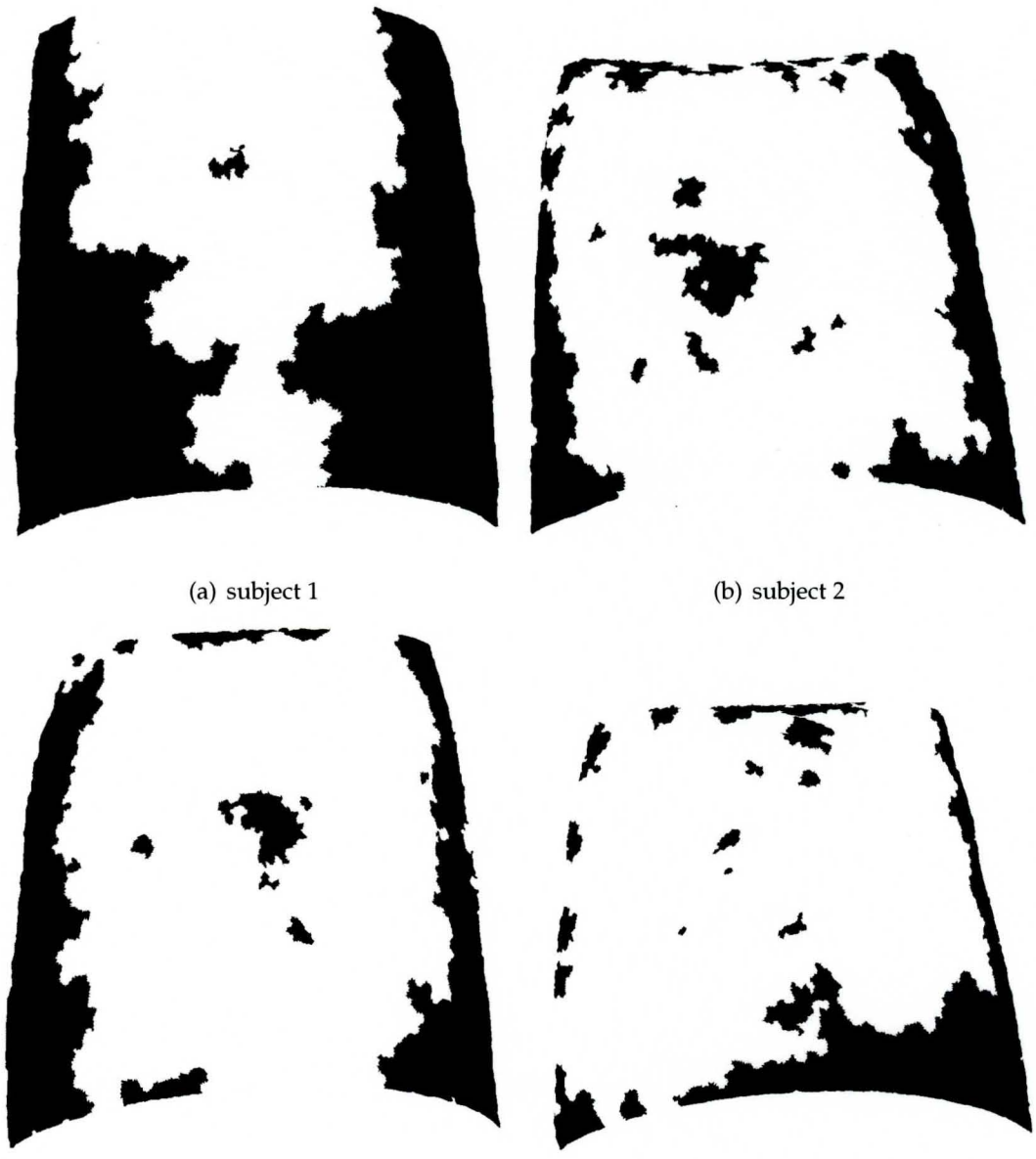
(a) subject 5

(b) subject 6

(c) subject 7

(d) subject 8

FIGURE 66 – Ridges (red) and ravines (blue) lines after post processing for four samples of the database.



(a) subject 1 (b) subject 2

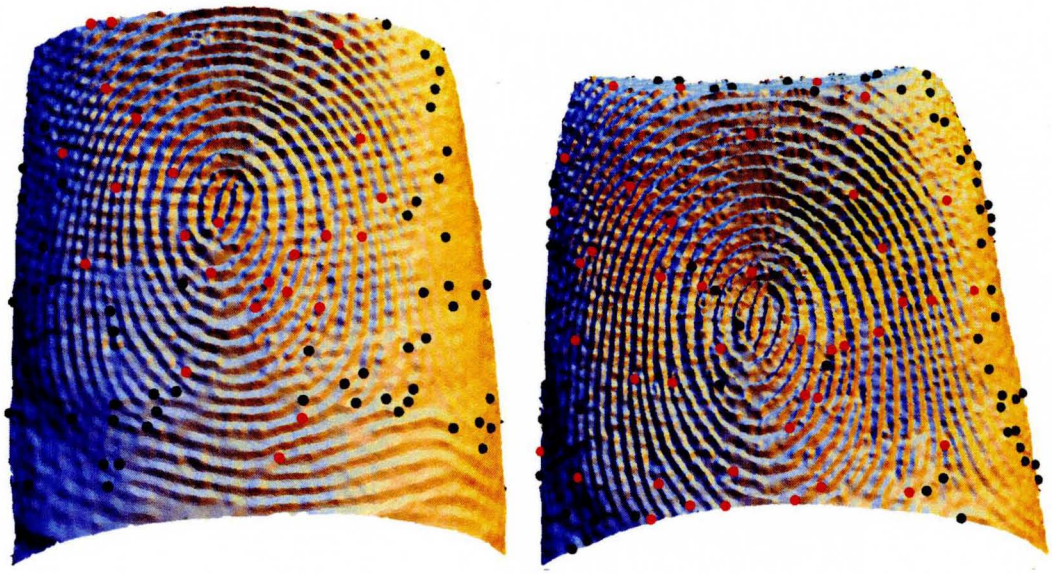
(c) subject 3 (d) subject 4

FIGURE 67 – The quality map for four samples of the database. High-quality areas displayed in white and poor-quality areas displayed in black.

(a) subject 5 (b) subject 6

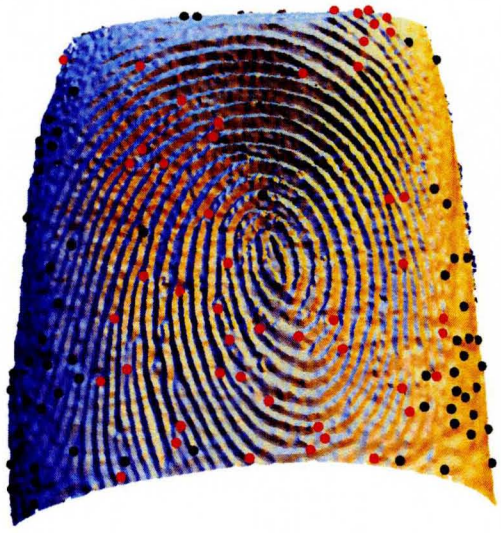
(c) subject 7 (d) subject 8

FIGURE 68 – The quality map for four samples of the database. High-quality areas displayed in white and poor-quality areas displayed in black.

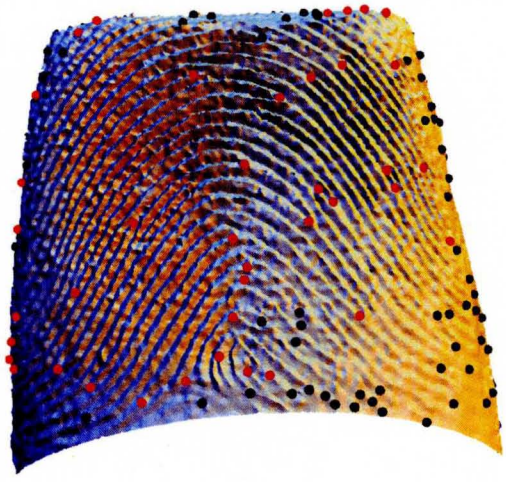


(a) subject 1

(b) subject 2

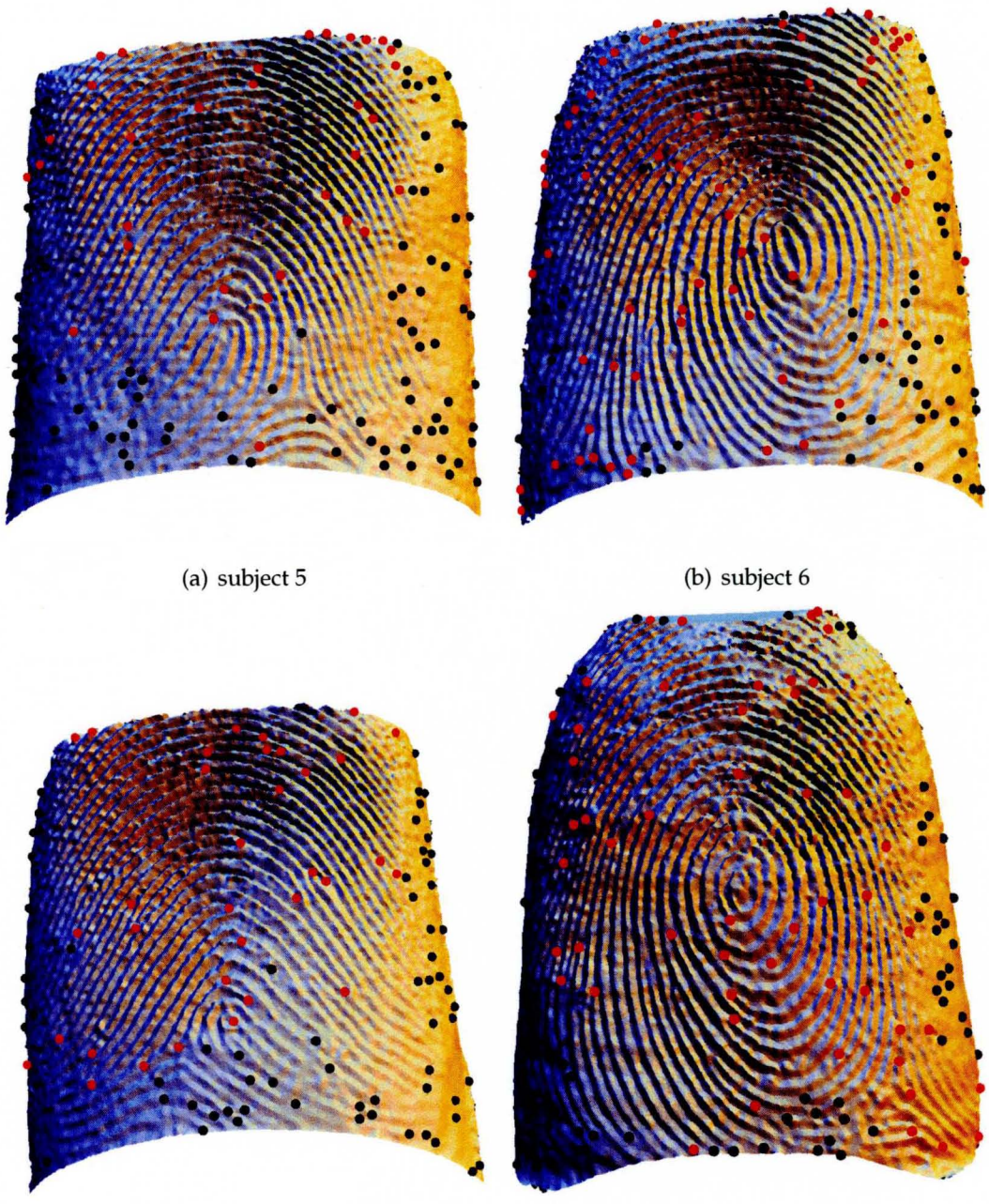


(c) subject 3



(d) subject 4

FIGURE 69 – Minutiae detection for four samples of the database. Black dots indicate minutiae associated with poor-quality areas; red dots refer to minutiae associated with high-quality areas.



(a) subject 5 (b) subject 6

(c) subject 7 (d) subject 8

FIGURE 70 – Minutiae detection for four samples of the database. Black dots indicate minutiae associated with poor-quality areas; red dots refer to minutiae associated with high-quality areas.

(a) subject 1

(b) subject 2

(c) subject 3

(d) subject 4

FIGURE 71 – Minutiae detection after removing the low-quality minutiae for four samples of the database.

(a) subject 5

(b) subject 6

(c) subject 7

(d) subject 8

FIGURE 72 – Minutiae detection after removing the low-quality minutiae for four samples of the database.

TABLE 5
ALGORITHM EXECUTION TIME

Algorithm	Average Execution time	Standard deviation
Adaptive Smoothing	18.71	1.29
Centroid Smoothing	0.105	0.023
Normal Computation	0.062	0.004
Principal Curvature	10.906	1.046
Tensor Smoothing	22.75	1.935
Ridge and Ravine Detection	0.297	0.047
Remove Short Branches	0.016	0.003
Remove Short Ridges	0.015	0.003
Connecting Broken Ridges	0.828	0.052
Minutiae Detection	0.309	0.040
Quality Map	1.484	0.622

ecution time. The execution times for each step of the extraction procedure are reported in Table 5. Adaptive smoothing and tensor smoothing are the most CPU-expensive.

CHAPTER VII

CONCLUSION AND FUTURE WORK

This dissertation has proposed a novel framework used to detect minutiae in 3D fingerprint scans. A series of algorithms were introduced that extract minutiae in 3D scans, and can be used for 3D fingerprint matching. To the best of our knowledge, this is the first minutiae detector for 3D fingerprint scans.

With 3D fingerprint scans becoming popular, there is a need to develop algorithms that detect minutiae directly in 3D and use them as input for 3D matching. Minutiae are the key features in fingerprint identification. After minutiae detection, matching can be reduced to the point matching problem.

A. Conclusion

In Chapter I, first the problems with existing contact-based fingerprint technologies is studied. These problems include : physical distortions and inconsistencies to the final image, hygienic problems, fingerprint faking, etc. In order to address these aforementioned problems, the new technology for fingerprint acquisition is introduced to the market, which is a touchless, or 3D live scan, of the finger. Due to using cameras, touchless fingerprint devices have several advantages, such as avoiding plastic distortion, avoiding latent fingerprints, reducing hygienic problems and capturing a large image area quickly. In an attempt to build such a system, Flashscan3D LLC and the University of Kentucky have been developing a non-contact, 3D finger scanning system. This system uses multiple, high-resolution, commodity, digital cameras and utilizes Structured Light Illumination (SLI). In this dissertation, the 3D fingerprint scans from the system developed by

Flashscan3D LLC and the University of Kentucky are used to test the proposed algorithms.

In Chapter II, we studied an existing matching method for conventional 2D fingerprints. This study led us to the fact that most fingerprint print matching systems are based on minutiae detection. Therefore, the first step in order to develop 3D minutiae-based fingerprint matching is minutiae detection in the 3D fingerprint. Minutiae extraction is a crucial process in 2D conventional fingerprints as well. Minutiae are the key features for fingerprint identification and it is important to develop an algorithm that detects them precisely. Hence, research in this study focuses on minutiae detection in 3D fingerprint system. Two methodologies are proposed that extract minutiae. The first method is to convert a 3D fingerprint scan to a 2D rolled equivalent fingerprint. Then, minutiae detection is performed using conventional existing 2D minutiae detection tools. This method is introduced in Chapter III. The proposed algorithm is based on curvature analysis of the 3D surface. First, the smooth surface of the 3D fingerprint is extracted. Then, the smoothed 3D surface is transformed into the 2D unrolled surface. Ridge and valley information are extracted from the 3D fingerprint by using curvature analysis of the 3D surface. Finally, ridge and valley information were put onto the unrolled surface. The benefit of this method is the compatibility with the existing 2D fingerprint images. However, the problem with this method is that unwrapping often results in information loss and distortion of the final 2D image. Therefore, another method that detects minutiae directly in 3D fingerprint is proposed. The main focus of our research is on this second method, which is explained in Chapter IV.

A set of algorithms suitable for the extraction of minutiae from 3D fingerprint scans are presented. The proposed method to extract minutiae includes the following steps: smoothing; computing the curvature tensors; ridges and ravines detection and tracing; cleaning and connecting ridges and ravines; and minutiae detection. Since the curvature extrema are sensitive to even small variations, a smoothing procedure is applied to the mesh before curvature extraction. Two

kinds of smoothing are used: Centroid and Adaptive. Then, principal curvatures are extracted from the surface by using differential geometry concepts and curvature analysis of the surface. In order to detect them more accurately, an implicit surface can be fitted locally to the mesh. Practical detection of the crest lines is a difficult computational task since it requires a high-quality estimation of the curvature tensor and their derivatives. Two different methods are applied in order to detect ridges and ravines on the surface. The First method is the nonmaxima suppression technique. In the second method, derivatives of curvature principals are used in order to detect ridges and ravines. The derivatives of curvature principals are computed by using an elegant formula that is much faster than the well-known formula. After detecting ridge and ravine vertices, they are traced to ridge and ravine lines.

It should be noted that in order to prevent false minutiae detection, it is necessary to clean the ridges and ravines before going to the minutiae detection step. Three different cleaning processes are applied:

- Removing the short ridges
- Removing the short branches
- Connecting the broken ridges/ ravines

Finally, minutiae are extracted using graph theory concepts. They are classified as reliable and unreliable minutiae. Unreliable minutiae might be ignored in the final results.

Chapter V introduces a quality map for 3D fingerprint scans. Since degraded area may happen during the scanning process, especially at the edge of the fingerprint, it is critical to be able to determine these areas. The quality map assigns the low quality to the area that has a poor contrast or very noisy ridges and valleys. Spurious minutiae can be filtered out after applying quality map. Several characteristics are designed to convey information to measure the quality in the

3D fingerprint scans. These include detecting regions with low-depth information and low-flow direction. These characteristics represent unstable areas in the fingerprint scan where detected minutiae are unreliable. This map might be also used to determine the levels of quality in the 3D scans.

Testing algorithms on database and evaluation is discussed in Chapter VI. The proposed method is tested using 3D fingerprint database including 100 3D fingerprint scans. The statistical analysis of the results obtained after quality map shows the effective reduction of spurious minutiae. The efficiency of the minutiae extraction is measured by two quantity measures namely sensitivity and specificity, which indicate the ability of the algorithm to detect the genuine minutiae and remove the false minutiae respectively. The high values of sensitivity and specificity in the end illustrate the encouraging performance of the proposed method.

B. Future works

In the future work, detected minutiae can be used for matching 3D fingerprint. The similarity degree might be obtained between two minutiae sets which are first extracted from the query and stored template by our algorithm. Then an algorithm matches the relative placement of the minutiae set in the query fingerprint with stored template and returns a binary decision (matched/non-matched) or a similarity score to indicate how similar the two fingerprints are. The 3D minutiae set are similar to 2D but having extra z coordinate. Therefore, 2D matching algorithm can be used with adding Z coordinate. Hence, the best correspondence of minutiae pairs can be searched by algorithms such as point pattern matching algorithm, relaxation, energy minimization and Hough transform.

Another recommendation for future work is to develop an enhancement that improves the 3D fingerprint scan quality. In general, a single fingerprint scan contains regions of:

- Well-defined regions, where ridges are clearly differentiated from each other;

- Recoverable regions, where ridges are corrupted by a small amount of gaps and noise, and so on, but they are still visible and the neighboring regions provide sufficient information about their true structure.
- Unrecoverable regions, where ridges are corrupted by such a severe amount of noise and distortion.

The goal of an enhancement algorithm is to improve the clarity of the ridge structures in the recoverable regions. One suggestion might be 3D Gabor filters. Gabor filters have both frequency-selective and orientation-selective properties and have optimal joint resolution in both spatial and frequency domains.

REFERENCES

[1] W. Yongchang, H. Qi, A. Fatehpuria, and D.L. Hassebrook, L.G.; Lau. Data acquisition and quality analysis of 3-dimensional fingerprints. In *Biometrics, Identity and Security (BIdS), 2009 International Conference on*, pages 1–9, 2009.

[2] Yongchang Wang, Laurence G. Hassebrook, and Daniel L. Lau. Data acquisition and processing of 3-d fingerprints. *IEEE Transactions on Information Forensics and Security*, 5(4):750–760, 2010.

[3] D. Maltoni, D. Maio, A.K. Jain, and S. Prabhakar. *Handbook of Fingerprint Recognition*. Springer, 2009.

[4] D.-H. Kim and R.-H. Park. Fingerprint binarization using convex threshold. In *Computer Graphics and Imaging*, pages 224–227, 2003.

[5] Yuheng Zhang and Qinghan Xiao. An optimized approach for fingerprint binarization. In *IJCNN*, pages 391–395, 2006.

[6] Pradeep M. Patil, Shekhar R. Suralkar, and Faiyaz B. Sheikh. Rotation invariant thinning algorithm to detect ridge bifurcations for fingerprint identification. In *ICTAI*, pages 634–641, 2005.

[7] D.Q. Miao, Q.S. Tang, and W.J. Fu. Fingerprint minutiae extraction based on principal curves. 28(16):2184–2189, December 2007.

[8] M. D. Garris, C. I. Watson, R. M. McCabe, and C. L. Wilson. *User's guide to NIST fingerprint image software (NFIS)*. National Institute of Standards and Technology, 2002.

[9] D. Maio and D. Maltoni. Direct gray-scale minutiae detection in fingerprints. 19(1):27–40, January 1997.

[10] H. Fronthaler, K. Kollreider, and J. Bigun. Local feature extraction in fingerprints by complex filtering. page 77, 2005.

[11] A.K. Jain. Biometric recognition: Overview and recent advances. pages 13–19, 2007.

[12] R. Hashido, A. Suzuki, A. Iwata, T. Okmoto, Y. Satoh, and M. Inoue. Aa capacitive fingerprint sensor chip using low-temperature poly-si tfts on a glass substrate and a novel and unique sensing method. *IEEE J. Solid-State Circuits*, 38(2):274280, February 2003.

[13] N. Sato. Novel surface structure and its fabrication process for mems fingerprint sensor. *IEEE Trans. Electron Devices*, 52(5):10261032, May 2005.

[14] B. Charlot, F. Parrain, N. Galy, S. Basrour, and B. Courtois. A sweeping mode integrated fingerprint sensor with 256 tactile microbeams. *J. Microelectromech. Syst.*, 13(4):636644, August 2004.

[15] M. Faundez-Zanuy. Are inkless fingerprint sensors suitable for mobile use. *IEEE Aerosp. Electron. Syst. Mag.*, 19(4):1721, April 2004.

[16] Song Y., Lee C., and Kim J. A new scheme for touchless fingerprint recognition system. In *Proc. Int. Symp. Intell. Signal Process. Commun. Syst.*, pages 524–527, 2004.

[17] A.M. Bazen and S.H. Gerez. Elastic minutiae matching by means of thin-plate spline models. pages II: 985–988, 2002.

[18] C.H. Lee, S.H. Lee, and J.H. Kim. A study of touchless fingerprint recognition system. pages 358–365, 2006.

[19] R. Rowe, S. Corcoran, K. Nixon, and R. Ostrom. Multi-spectral imaging for biometrics. In *Proc. SPIE Conf. Spectral Imaging: Instrumentation, Application, and Analysis*, page 9099, Mar 2005.

[20] Mitsubishi touchless fingerprint sensor, October 2006.

[21] TBS. http://www.tbsinc.com/, 2007.

[22] G. Parziale, E. Diaz Santana, and R. Hauke. The surround imagertm: A multi-camera touchless device to acquire 3d rolled-equivalent fingerprints. pages 244–250, 2006.

[23] V. G. Yalla and L. Hassebrook. Very-high resolution 3d surface scanning using multi-frequency phase measuring profilometry. In *SPIE Defense and Security*, pages 44–53, 2005.

[24] V. Srinivasan, H. Liu, and M. Halioua. Automated phase-measuring profilometry of 3d diffuse objects. *Applied Optics*, 23(18):3105–3108, 1984.

[25] J. Li, L. G. Hassebrook, and C. Guan. Optimized twofrequency phase measuring profilometry light-sensor temporal-noise sensitivity. *J. Opt. Soc. Am*, 20:106–115, 2003.

[26] F. Galton. *FingerPrints*. SLondon: Macmillan, 1892.

[27] T.Y. Jea. *Minutiae-based partial fingerprint recognition*. PhD thesis, 2005.

[28] A. K. Jain, Lin Hong, S. Pankanti, and R. Bolle. An identity-authentication system using fingerprints. *Proceedings of the IEEE*, 85(9):1365–1388, 1997.

[29] Q.H. Xiao and H. Raafat. Fingerprint image post-processing: A combined statistical and structural approach. 24(10):985–992, 1991.

[30] N.K. Ratha, S.Y. Chen, and A.K. Jain. Adaptive flow orientation-based feature-extraction in fingerprint images. 28(11):1657–1672, November 1995.

[31] Abdullah Çavuşoğlu and Salih Görgünoğlu. Technical communication: A fast fingerprint image enhancement algorithm using a parabolic mask. *Comput. Electr. Eng.*, 34:250–256, May 2008.

[32] Craig I. Watson, Michael D. Garris, Elham Tabassi, Charles L. Wilson, R. Michael McCabe, and Stanley Janet. *User's guide to NIST fingerprint image software 2 (NFIS2).* National Institute of Standards and Technology, 2002.

[33] Josef Strm Bartunek, Mikael Nilsson, Jrgen Nordberg, and Ingvar Claesson. Adaptive fingerprint binarization by frequency domain analysis. In *Conf. IEEE Signal System and Computers*, pages 598–602, 2006.

[34] L. Ji, Z. Yi, L. Shang, and X. Pu. Binary fingerprint image thinning using template-based pcnns. 37(5):1407–1413, October 2007.

[35] A.M.Bazen and S.H.Gerez. Achievement and challenges in fingerprint recognition. In *In Biometric Solutions for Authentication in an e-World*, pages 23–57, 2002.

[36] Pu Hongbin, Chen Junali, and Zhang Yashe. Fingerprint thinning algorithm based on mathematical morphology. In *IEEE 8th International Conference on Electronic Measurement and Instruments*, pages 2618 – 2621, 2007.

[37] M. Ahmed and R. Ward. A rotation invariant rule-based thinning algorithm for character recognition. 24(12):1672–1678, December 2002.

[38] H. Tamura. A comparison of line thinning algorithms from digital geometry viewpoint. pages 715–719, 1978.

[39] L.R. Palmer, M.S. Al-Tarawneh, S.S. Dlay, and W.L. Woo. Efficient fingerprint feature extraction: Algorithm and performance evaluation. In *IEEE 6th International Symposium on Communication Systems*, pages 581–584, 2008.

[40] S.A. Sudiro, M. Paindavoine, and M. Kusuma. Simple fingerprint minutiae extraction algorithm using crossing number on valley structure. In *IEEE Workshop on Automatic Identification Advanced Technologies*, pages 41–44, 2007.

[41] J. Zalev and R. Sedaghat. Automatic fingerprint recognition algorithm. In *IEEE Conference on Electrical and Computer Engineering*, pages 2263–2266, 2004.

[42] Jiyi Li and Guangshun Shi. A novel palmprint feature processing method based on skeleton image. In *Proceedings of the 2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*, SITIS '08, pages 221–228, Washington, DC, USA, 2008. IEEE Computer Society.

[43] Josef Strm Bartunek, Mikael Nilsson, Jrgen Nordberg, and Ingvar Claesson. Neural network based minutiae extraction from skeletonized fingerprints. In *TENCON 2006 IEEE Region 10 Conference*, pages 1–4, 2006.

[44] W.F. Leung, S.H. Leung, W.H. Lau, and A. Luk. Fingerprint recognition using neural network. In *Proceedings of the IEEE Workshop Neural Networks for Signal Processing*, pages 226–235, 1991.

[45] D.M. Weber. A cost effective fingerprint verification algorithm for commercial applications. In *In Proc. South African Symp. on Communication and Signal Processing*, pages 99–104, 1992.

[46] J.H. Shin, H.Y. Hwang, and S.I. Chien. Detecting fingerprint minutiae by run length encoding scheme. 39(6):1140–1154, June 2006.

[47] X.D. Jiang, W.Y. Yau, and W. Ser. Detecting the fingerprint minutiae by adaptive tracing the gray-level ridge. 34(5):999–1013, May 2001.

[48] C.J. Lee and S.D. Wang. Fingerprint feature reduction by principal gabor basis function. 34(11):2245–2248, November 2001.

[49] A. Montesanto, P. Baldassarri, G. Vallesi, and G. Tascini. Fingerprints recognition using minutiae extraction: a fuzzy approach. pages 229–234, 2007.

[50] J. Liu, Z. Huang, and K. Chan. Direct minutiae extraction from gray-level fingerprint image by relationship examination. pages Vol II: 427–430, 2000.

[51] M. Leung, W. Engeler, and P. Frank. Fingerprint image processing using neural network. In *In Proc. IEEE Region 10 Conf. on Computer and Communication Systems*, pages 582–586, 1990.

[52] P. V. C. Hough. Machine analysis of bubble chamber pictures. 1959.

[53] Nalini K. Ratha, Kalle Karu, Shaoyun Chen, and Anil K. Jain. A real-time matching system for large fingerprint databases. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(8):799–813, 1996.

[54] Shih-Hsu Chang, Fang-Hsuan Cheng, Wen-Hsing Hsu, and Guo-Zua Wu. Fast algorithm for point pattern matching: Invariant to translations, rotations and scale changes. *Pattern Recognition*, 30(2):311–320, 1997.

[55] S. anade and A. Rosenfeld. Point pattern matching by relaxation. 12(4):269–275, 1980.

[56] N. Ansari, M. H. Chen, and E. S. H. Hou. *A Genetic Algorithm for Point Pattern Matching*. Chapt. 13, B. Soucek and the IRIS Group, eds., Dynamic, Genetic, and Chaotic Programming. New York: John Wiley and Sons, 1992.

[57] A. Pikaz and I. Dinstein. Matching of partially occluded planar curves. *PR*, 28:199–209, 1995.

[58] A.K. Jain, L. Hong, and R.M. Bolle. Online fingerprint verification. 19(4):302–314, April 1997.

[59] Joseph H. Wegstein. An Automated Fingerprint Identification System. NBS special publication 500-89, U.S. Dept. of Commerce, Washington, DC 20234, February 1982.

[60] Yi Chen, Eva Diaz-santana, and Anil K. Jain. 3d touchless fingerprints: Compatibility with legacy rolled images. In *Proceeding of Biometric Symposium, Biometric Consortium Conference*, pages 1–6, 2006.

[61] Abhishika Fatehpuria, Daniel L. Lau, and Laurence G. Hassebrook. Acquiring a 2-d rolled equivalent fingerprint image from a non-contact 3-d finger scan. biometric technology for human identification. In *Proceedings of SPIE, the International Society for Optical Engineering*, pages 2020C–1–62020C–8, 2006.

[62] C. Brian Atkins, Jan P. Allebach, and Charles A. Bouman. Halftone postprocessing for improved rendition of highlights and shadows. *J. Elec. Imaging*, 9:151–158, 2000.

[63] Yongchang Wang, Daniel L. Lau, and Laurence G. Hassebrook. Fit-sphere unwrapping and performance analysis of 3d fingerprints. *Applied Optics*, 49(4):592–600, 2010.

[64] Laurence G. Hassebrook, Yongchang Wang, , and Daniel L. Lau. Fusion of unraveled 3d fingerprint. *Submitted to IEEE Transactions on Information Forensics and Security.*

[65] Michael S. Brown and W. Brent Seales. Image restoration of arbitrarily warped documents. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26:1295–1306, 2004.

[66] Xavier Provot Institut and Xavier Provot. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *In Graphics Interface*, pages 147–154, 1996.

[67] Mohammad H. Mahoor and Mohamed Abdel-Mottaleb. Face recognition based on 3d ridge images obtained from range data. *Pattern Recogn.*, 42:445–451, March 2009.

[68] A.M. Lopez, F. Lumbreras, J. Serrat, and J.J. Villanueva. Evaluation of methods for ridge and valley detection. 21(4):327–335, April 1999.

[69] M.P. Do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, 1976.

[70] J. Li, S. Tulyakov, and V. Govindaraju. Verifying fingerprint match by local correlation methods. pages 1–5, 2007.

[71] E. Tabassi, C. L.Wilson, and C. I.Watson. Fingerprint image quality. Technical Report NISTIR 7151, National Institute of Standards and Technology, August 2004.

[72] Shin Yoshizawa, Alexander G. Belyaev, and Hans-Peter Seidel. Fast and robust detection of crest lines on meshes. In *Symposium on Solid and Physical Modeling*, pages 227–232, 2005.

[73] Yutaka Ohtake, Alexander G. Belyaev, and Hans-Peter Seidel. Mesh smoothing by adaptive and anisotropic gaussian filter applied to mesh normals. In *VMV*, pages 203–210, 2002.

[74] Nelson Max. Weights for computing vertex normals from facet normals. *J. Graph. Tools*, 4:1–6, Mar 1999.

[75] P.J. Flynn and A.K. Jain. On reliable curvature estimation. pages 110–116, 1989.

[76] N.N. Abdelmalek. Algebraic error analysis for surface curvatures and segmentation of 3-d range images. *PR*, 23:807–817, 1990.

[77] E.M. Stockely and S.Y. Wu. Surface parameterization and curvature measurement of arbitrary 3-d objects: Five practical methods. 14(8):833–840, August 1992.

[78] A.M. McIvor and R.J. Valkenburg. A comparison of local surface geometry estimation methods. 10(1):17–26, 1997.

[79] B. Hamann. *Curvature approximation for triangulated surfaces*, pages 139–153. Springer-Verlag, London, UK, 1993.

[80] Dereck S. Meek and Desmond J. Walton. On surface normal and gaussian curvature approximations given data sampled from a smooth surface. *Computer Aided Geometric Design*, 17:521–543, 2000.

[81] A.A. Amini and J.S. Duncan. Differential geometry for characterizing 3d shape change proc. In *SPIE: Math. Methods in Medical Imaging*, volume 1786, pages 170–181, 1992.

[82] N. Yokoya and M.D. Levine. Range image segmentation based on differential geometry: A hybrid approach. 11(6):643–649, June 1989.

[83] V. Pratt. Direct least-squares fitting of algebraic surfaces. *SIGGRAPH*, 87:145–152.

[84] A.W. Fitzgibbon, M. Pilu, and R.B. Fisher. Direct least square fitting of ellipses. 21(5):476–480, May 1999.

[85] Jack Goldfeather and Victoria Interrante. A novel cubic-order algorithm for approximating principal direction vectors. *ACM Trans. Graph.*, 23(1):45–63, 2004.

[86] Alexander G. Belyaev, Yutaka Ohtake, and Kasumi Abe. Detection of ridges and ravines on range images and triangular meshes. *Vision Geometry IX*, 4117:146–154, 2000.

[87] O. Faugeras. *Three-Dimensional Computer Vision*, chapter 4: Edge Detection. MIT Press, 1993.

[88] Yutaka Ohtake, Alexander G. Belyaev, and Hans-Peter Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph.*, 23(3):609–612, 2004.

[89] N. Khaneja, M.I. Miller, and U. Grenander. Dynamic programming generation of curves on brain surfaces. 20(11):1260–1265, November 1998.

[90] O. Monga, N. Armande, and P. Montesinos. Thin nets and crest lines: Application to satellite data and medical images. 67(3):285–295, September 1997.

[91] Shin Yoshizawa. *Computational Differential Geometry Tools for Surface Interrogation, Fairing, and Design*. PhD thesis, Saarland University, 2006. MPI Informatik.

[92] Yutaka Ohtake, Alexander Belyaev, and Hans peter Seidel. An integrating approach to meshing scattered point data. In *Symposium on Solid Modeling and Applications*, pages 61–69, 2005.

[93] B.G. Sherlock, D.M. Monro, and K. Millard. Fingerprint enhancement by directional fourier filtering. 141(2):87–94, Apr 1994.

CURRICULUM VITAE

C. CONTACT INFORMATION

Sara Shafaei

775 Theodore Burnett ct., Apt. 5
Louisville, Kentucky, 40217, USA.
1-217-417-7092
sara.shafaei@louisville.edu

D. RESEARCH INTERESTS

Image Processing, Pattern Recognition, Computer Vision, Biometrics, 3D Fingerprint Recognition, Human Body Tracking

E. EDUCATION

University of Louisville, Louisville, Kentucky USA

Ph.D., Electrical & Computer Engineering, August, 2011

- Dissertation Topic: "3D Minutiae Extraction in 3D Fingerprint Scans"
- GPA: 3.92
- Advisor: Tamer Inanc

Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran

M.Sc., Computer Engineering (Artificial Intelligence), February, 2004

- Dissertation Topic: "Human body part tracking for physiotherapy virtual training"

University of Isfahan, Isfahan, Iran

B.Sc., Computer Engineering, August, 2001

- Thesis Title: "Design and Implementation of Persian chat on the web"

F. Research Experience

Research Assistant, University of Louisville, 2006-2008, 2009-2010

- Unwrapping 3D fingerprint scans to rolled equivalent fingerprints
- Matching rolled equivalent fingerprints
- Minutiae detection in 3D fingerprint scans
- Quality map for 3D fingerprint scans

Research Assistant, Amirkabir University of Technology, 2001-2004

- Human body tracking
- Physiotherapy virtual training using human body tracking

G. Industrial Experience

Researcher, Niroo Research Institute (NRI), Power systems control and dispatching research center, Tehran, Iran, http://www.nri.ac.ir, 2003-2004

- Preparing technical specification for Tase-2 Protocol

Software Engineer, Setareh Talaei Kish Company, Tehran, Iran, http://www.stkish.com, 2001 - 2002

- Analyses, Design and Implementation of Inventory management Software

Software Engineer, Nirou Chlor Co. (P.V.T), Isfahan, Iran, http://www.nirouchlor.com, 2000-2001

- Analyses, Design and Implementation of data management software for Datalogers using Access and Visual Basic
- Web site design for Datalogers

H. Teaching Experience

Teaching Assistant, University of Louisville, Electrical and Computer engineering Dept., 2008-2009

- Digital Signal Processing
- Fundamental Autonomous Robots
- Control System Principals

Instructor, Bu-Ali Sina University, Computer engineering Dept., 2004-2006

- Advance Programming with C++
- Introduction to object oriented programming
- Software Engineering
- Discrete Mathematics
- Programming with Pascal
- Algorithm Foundation
- Supervision of B.Sc. thesis

Teaching Assistant, Amirkabir University of Technology, Computer Engineering and Information Technology Dept., 2003

- Symbolic Logic

I. Honors and Awards

- Member of the Institute of Electrical and Electronics Engineers (IEEE).
- Doctoral Dissertation Completion Award, 2011.
- The Theobald Scholarship Award, 2011.
- Certificate of Completion for attending PLAN events, 2011.

J. Publications

1. **Sara Shafaei** and Tamer Inanc, "Minutiae detection in 3D fingerprint scans." Submitted to IEEE Transaction on image processing.

2. **Sara Shafaei**, Tamer Inanc, and Laurence G. Hassebrookand Aly A. Farag. " A new approach to unwrap a 3-D fingerprint to a 2-D rolled equivalent fingerprint", Proceedings of the 3rd IEEE international conference on Biometrics: Theory, applications and systems, Washington, DC, USA , Pages: 370-374, 2009.

3. **Sara Shafaei** and Mohammad Rahmati "Human Body Tracking For Physiotherapy Virtual Training", International Conference on Computer Vision Theory and Applications(visapp), Setbal, Portugal, 2006.

4. Mohammad Rahmati, **Sara Shafaei**, "A Virtual Training System for Physiotherapy Applications" The 4th IASTED International Conference on Visualization, Imaging, and Image Processing(VIIP), Marbella, Spain,2004.

5. **Sara Shafaei**, Mohammad Rahmati, "Physiotherapy Virtual Training By Computer Vision Approach", Third International Workshop on Virtual Rehabilitation(IWVR), Lausanne, Switzerland, 2004.

K. PATENT

- University of Louisville Research Disclosure No. 10040, "3D (3-Dimensional) Minutiae Detection in 3D Fingerprint Scans," submitted November 3, 2010, U.S. Provisional Patent Application to be filed in near future.

L. COMPUTER SKILLS

- Operating Systems: Windows XP, LINUX
- Computer Languages: C/C++, Visual C++, Visual Basic, Pascal
- Visualization Libraries: The visualization toolkit (VTK) and OpenGL
- GUI Libraries: MFC (Microsoft Visual C++) and MATLAB GUI
- Computational Libraries: Computational Geometry Algorithms Library(CGAL)
- Scientific Applications: MATLAB/Simulink
- Technical Drawing: PhotoImpact, PhotoShop
- Software Engineering Application: RUP and Rational Rose
- Database: SQL Server, Microsoft Access
- Internet Development: HTML, Asp, VB script, java script

M. MEMBERSHIP

- The president of the Iranian Student Organization (ISO) Louisville Unit
- Member of Society of Women Engineers (SWE)