



**Integrating Fingerprint and Top-View Finger Image
for Personal Identification**

Panyayot Chaikan

**A Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy in Computer Engineering
Prince of Songkla University**

2010

Copyright of Prince of Songkla University

Thesis Title Integrating Fingerprint and Top-View Finger Image
 for Personal Identification

Author Mr. Panyayot Chaikan

Major Program Computer Engineering

Major Advisor

.....
Assoc. Prof. Dr. Montri Karnjanadecha

Co-advisor

.....
Asst. Prof. Dr. Thanate Khaorapapong

Examining Committee:

.....Chairperson
(Assoc. Prof. Dr. Chusak Limsakul)

.....
Assoc. Prof. Dr. Montri Karnjanadecha

.....
Asst. Prof. Dr. Thanate Khaorapapong

.....
(Asst. Prof. Dr. Pornchai Phukpattaranont)

.....
(Assoc. Prof. Dr. Kosin Chamnongthai)

The Graduate School, Prince of Songkla University, has approved this thesis as partial fulfillment of the requirements for the Doctor of Philosophy Degree in Computer Engineering.

.....
(Assoc. Prof. Dr. Krerckchai Thongnoo)
Dean of Graduate School

Thesis Title Integrating Fingerprint and Top-View Finger Image
for Personal Identification

Author Mr. Panyayot Chaikan

Major Program Doctor of Philosophy (Computer Engineering)

Academic Year 2009

ABSTRACT

This thesis presents the use of top-view finger imaging to increase the accuracy of fingerprint recognition systems. While the user is touching a fingerprint sensor, a top-view finger image is captured using a CCD camera without requiring the user to carry out any additional work. The acquired gray scale finger image is preprocessed to enhance its edges, the skin furrows, and the nail shape, before the image is filtered by a bank of Oriented-Filters. A square tessellation is applied to the filtered image to create a feature map, called a NailCode. The NailCode is used in the matching process by employing a Euclidean distance computation. A combination between the NailCode and the fingerprint matcher is done at the decision level by means of a likelihood ratio. Measured at an equal error rate, the system error is reduced by 17.68% in the verification mode, and 6.82% in the identification mode. Top-view finger imaging also reduces the possibility of fraud by having recognition rely on more than one feature. Since NailCode alone gives lower accuracy than fingerprinting, it is suggested that top-view finger imaging should not be used alone to verify or identify individuals, especially when high security is required. It should be employed in conjunction with fingerprinting to improve overall recognition accuracy.

Keywords: fingerprints, top-view finger images, multimodal biometrics, Nailcode

ACKNOWLEDGMENTS

First of all, I would like to express my guidance sincere gratitude to my advisor, Assoc. Prof. Dr. Montri Karnjanadecha, for his attention, insight, encourage, guided, and support during this research and for being available at anytime to response my questions, which has been valuable.

I would like to thank the co-advisor, Asst. Prof. Dr. Thanate Khaorapapong for his assistance, helpful comments, and insightful suggestions.

I especially acknowledge the Thai government for financially supporting me during my Ph.D. program.

I am grateful to Dr. Andrew Davison for his kind help in polishing the language of my published papers.

I am grateful to Mr. Robert Elz and Dr. Andrew Davison for their kind help in polishing the language of this thesis.

I would like to express my sincere thank to Asst. Prof. Damarong Kalawdee for his kind help which made me having enough time writing this thesis. Working with him has been a great learning experience.

Finally, my greatest thanks are to my beloved family and parents whose caring, understand, and possible attitude have encouraged me to go forward during difficult times. Whose never-ending love and support made the completion of this work completed and my dream of a graduate education comes true.

Panyayot Chaikan

CONTENTS

Chapter 1. Introduction	1
1.1 Background and Rationale	1
1.2 Literature Review	2
1.3 Objectives	5
1.4 Scope of the Thesis	5
1.5 Organization of the Thesis	6
Chapter 2. Research Methodology	7
2.1 Theoretical Background	7
2.2 Method	19
2.3 Materials and Equipments	50
2.4 Summary	50
Chapter 3. Experimental Result	51
3.1 Top-View finger image acquisition	51
3.2 Top-view finger image processing and feature extraction	52
3.3 Top-view finger image and time variances	58
3.4 Test database of the top-view finger images and fingerprints	59
3.5 Fingerprint preprocessing	59
3.6 Fingerprint post-processing	63
3.7 Decision fusion	72
3.8 System Performance	74
3.9 Summary	78
Chapter 4. Conclusions and Future Works	79
4.1 Discussions	79
4.2 Research contributions	81
4.2 Future work	81
References	83
Appendix A. Test Database	89
Appendix B. List of Publications	129
Vitae	155

LIST OF TABLES

Table		Page
2.1.	Set operations applied to pixels.	8
2.2.	Examples of multimodal biometric systems.	15
3.1.	The system accuracy according to each value of window size (w).	53
3.2.	The system accuracy according to each value of V .	54
3.3.	The system accuracy according to each value of H .	54
3.4.	The class separation of a system for different sizes of an adaptive threshold's window.	55
3.5.	Identification accuracy of person recognition with different reference-point location errors.	57
3.6.	Identification accuracy with different reference point markings.	58
3.7.	Result of computation time required by each type of Gabor filter.	63
3.8.	Average computing time for one test on a 2.4 GHz Pentium 4.	76
3.9.	Equal Error Rate of the tested configurations.	78

LIST OF FIGURES

Figure		Page
1.1.	Top-view image and fingerprint recognition system.	1
2.1.	Image coordinate.	7
2.2.	Sequence of structuring elements used for thinning.	9
2.3.	Convolution masks of Prewitt and Sobel operators.	11
2.4.	The system FAR and FRR for a given threshold, τ .	14
2.5.	Receiver Operating Characteristic (ROC) curve.	15
2.6.	A fingerprint image.	16
2.7.	Four different types of fingerprint sensor	16
2.8.	Examples of the 4 Minutia types.	18
2.9.	Block diagram of the whole system.	20
2.10.	System hardware used to digitize fingerprint and top-view finger images.	21
2.11.	Flowchart of the preprocessing algorithm.	22
2.12.	Images obtained in each preprocessing step.	23
2.13.	Finger images filtered at 0° and 90° .	26
2.14.	The derivation of φ for different conditions.	27
2.15.	Kernel of Oriented Filters with different θ values.	28
2.16.	Reference point location and square tessellation on the filtered image.	29
2.17.	Top-view finger images for each step of the reference point detection algorithm.	31
2.18.	Eight points for reference point error compensation.	32
2.19.	Reference point detection algorithm.	32
2.20.	Our SMM algorithm.	34
2.21.	Oriented window and x-signature.	37
2.22.	The derivation of the coordinate (u, v) from the oriented window of size $l \times w$.	39
2.23.	Fingerprint ridges after thinning operation.	41
2.24.	Minutiae extraction window.	42

Figure	Page
2.25. Three ridge lines directed outward from the bifurcation point and the most suitable direction to be used as a bifurcation direction.	43
2.26. Flowchart of the fingerprint minutiae post-processing.	44
2.27. The comparison of the fingerprint ridge before and after small black hole deletion.	44
2.28. Fingerprint image before spur deletion (left) and after spur deletion (right).	45
2.29. Example of two spurious bifurcations around the real bifurcation.	46
2.30. (a) direction of line following to detect spurious termination point; (b) before removing spurious termination points; (c) after spurious termination points have been removed.	47
2.31. Decision fusion between the Top-View and fingerprint matcher.	48
3.1 Top-View finger image obtained from CCD camera.	51
3.2. Top-view finger image acquisition.	52
3.3. The binarized version of the top-view finger images due to different window size of an adaptive threshold.	53
3.4. Finger image captures at different times: (a) the initial image; (b) the same finger captured after 990 days had passed.	58
3.5. Results of the fingerprint orientation field due to different w_ϕ of the low pass filter: (a) image of good quality; (b) orientation field before smoothing; (c) orientation field after smoothing with a low-pass filter of 3×3 ; (d) orientation field after smoothing with a low-pass filter of 5×5 .	60
3.6. Results of the fingerprint orientation field due to different w_ϕ of the low pass filter: (a) image of poor quality; (b) orientation field before smoothing; (c) orientation field after smoothing with a low-pass filter of 3×3 ; (d) orientation field after smoothing with a low-pass filter of 5×5 .	61

Figure		Page
3.7.	X-signature due to different regions of fingerprint: (a) where a ridge ending was found; (b) where no ridge line appears; (c) where no minutia appear.	62
3.8.	The fingerprint image (left) and the obtained minutiae mask image (right).	62
3.9.	Result of the fingerprint image manipulated by a Gabor filter with different δ_x and δ_y values: (a) original gray scale fingerprint of good quality; (b) after filtered with of δ_x and δ_x of 12; (c) after filtered with of δ_x and δ_x of 20; (d) after filtered with of δ_x and δ_x of 30.	64
3.10.	Result of the fingerprint image manipulated by a Gabor filter with different δ_x and δ_y values: (a) original gray scale fingerprint of poor quality; (b) after filtered with of δ_x and δ_x of 12; (c) after filtered with of δ_x and δ_x of 20; (d) after filtered with of δ_x and δ_x of 30.	65
3.11.	Result of the fingerprint image manipulated by a Gabor filter with different δ_x and δ_y values: (a) gray scale fingerprint of non-uniform pressure of the finger; (b) after filtered with of δ_x and δ_x of 12; (c) after filtered with of δ_x and δ_x of 20; (d) after filtered with of δ_x and δ_x of 30.	66
3.12.	Result of the binarized fingerprint image due to different threshold values: (a) original gray scale fingerprint of good quality; (b) after binarized using threshold value of -800; (c) after binarized using threshold value of -400; (d) after binarized using threshold value of 0.	67
3.13.	Result of the binarized fingerprint image due to different threshold values: (a) original gray scale fingerprint of poor quality; (b) after binarized using threshold value of -800; (c) after binarized using threshold value of -400; (d) after binarized using threshold value of 0.	68

Figure	Page
3.14. Result of the binarized fingerprint image due to different threshold values: (a) gray scale fingerprint of non-uniform pressure of the finger; (b) using threshold value of -800; (c) using threshold value of -400; (d) using threshold value of 0.	69
3.15. Result of the fingerprint post-processing steps: (a) binarized fingerprint image; (b) after small black holes have been removed; (c) after small white ridges have been removed; (d) after thinning process.	70
3.16. Result of the fingerprint post-processing steps: (a) m-connectivity type fingerprint image; (b) after spurs have been deleted; (c) minutiae extracted from spurs deleted image; (d) minutiae after fake termination points have been deleted.	71
3.17. Result of too close minutiae deletion; (a) before deletion (b) after too close minutiae are deleted.	72
3.18. The estimated bottom-view matching score distribution.	73
3.19. The estimated top-view matching score distribution.	73
3.20. Verification performance of individual matchers.	75
3.21. Verification performance of all combinations.	75
3.22. Performance in the identification mode.	77

LIST OF ABBREVIATIONS

ROC	Receiver Operating Characteristic
GAR	Genuine Acceptance Rate
FAR	False Acceptance Rate
FRR	False Rejection Rate
EER	Equal Error Rate
FMR	False Match Rate
FNMR	False Non Match Rate
LED	Light-Emitting Diode
CCD	Charge-Coupled Device

CHAPTER 1

INTRODUCTION

1.1 Background and Rationale

Fingerprinting is ubiquitous because of its uniqueness and time invariance [1]. As a biometric feature, fingerprints offer high accuracy even when cheap sensors are utilized. However, fingerprint recognition accuracy has reached a limit which is difficult to surpass. One approach is multimodal biometrics, which combines multiple human features in the recognition process. For example, Hong and Jain employ the face in conjunction with fingerprints [2], Jain *et al.* use speech, face, and fingerprints [3], Marcialis and Roli utilize two different fingerprint sensors [4], while Prabhakar and Jain examine two fingers [5]. All these methods augment recognition accuracy, with the drawback that the additional features increase the complexity of user interaction with the system.

Our approach rests on the idea that the skin wrinkles and furrows on top of each person's fingers are different, along with the size and shape of the fingers and finger nails. Utilizing these attributes will increase the accuracy of a multimodal biometric system without requiring extra work by the user. The details can be captured with a small, inexpensive camera positioned above the fingerprint sensor, as shown in Figure 1.1. Top-view finger imaging also reduces the possibility of fraud by having recognition rely on more than one feature.

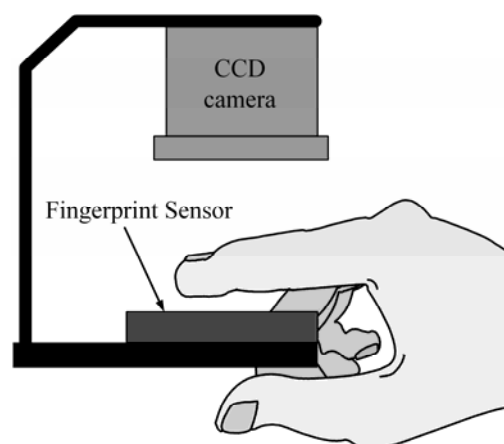


Figure 1.1 Top-view image and fingerprint recognition system.

1.2 Literature Review

Many papers report on the use of biometric features to augment the accuracy of the fingerprint recognition systems. Some use data from the finger image alone. The following papers are relevant to our work.

1.2.1 *Evaluation of personal identification system by transillumination imaging of a finger* [6]

This paper proposes the use of the finger's translucent property, through illumination by an array of near-infrared LEDs. A CCD camera captures the blood vessel structure of the finger. Since the blood vessel pattern is time invariant and unique, it does not change due to scars that might occur on the skin. Matching is done by a correlation operation. The authors claim that the system needs low computation time and has a low error rate.

1.2.2 *Vein Pattern Recognitions by Moment invariants* [7]

The paper uses a dyadic wavelet transformation to extract the structure of a finger vein. The input images are manipulated in a wavelet domain, any noise removed via soft-thresholding denoising, and matching achieved using Hausdorff distance. The authors claim that their technique is more robust for feature extraction than a line-tracking operation.

1.2.3 *Personal identification using finger knuckle orientation features* [8]

The paper proposes the use of a finger knuckle image since it offers many skin lines and creases. A finite Radon transform (FRT) is employed to detect random lines and creases. The system gives an equal error rate of 1.14%. The idea behind this paper is similar to our thesis, but only uses a data from the knuckle; the nail shape and the finger width is not exploited.

1.2.4 *Finger image identification method for personal verification* [9]

An image of the entire bottom-view finger (including fingerprint) is utilized, which contains the front surface of the finger from the fingertip to the second joint of the finger. The main feature is extracted from the distance between the second joint by projecting the image in the direction parallel to the finger. The method needs low computation time, but requires a sheet prism and an expensive TV raster scan camera which gives a resolution of 10 lines/mm.

1.2.5 ***PDE-based Finger Image Denoising*** [10]

The knuckle and wrinkle line of the front surface of a finger is employed. Edge detection is used to extract finger lines, including the shape of the finger. The idea behind this paper is similar to our thesis but uses data from the bottom-view of the finger. The authors focus on preprocessing, and the matching process is not discussed.

1.2.6 ***Integrating Faces and Fingerprints for Personal Identification*** [2]

This paper proposes the combination of face and fingerprint to construct an identification system. Fingerprints offer high accuracy but require much computation time. Face recognition needs very low computation time, and face indexing mechanisms for large database searching already exist, but recognition accuracy is much lower than for fingerprints. The paper uses the face to search over a large database, and the best five candidates are obtained. Fingerprint verification is applied to the candidates to find the best match. A final decision employs decision fusion.

1.2.7 ***On combining classifiers*** [11]

A theoretical framework for combining classifiers is proposed which uses different input patterns. Many combination schemes are suggested, such as a sum rule, product rule, min rule, max rule and majority vote rule. The authors demonstrate that the sum rule outperforms all the other combination rules.

1.2.8 ***A multimodal Biometric System using Fingerprint, Face, and Speech*** [3]

This paper uses fingerprint, face, and speech together. Although three matchers are utilized, a majority vote rule is not used to make a final decision. Instead of each matching module making a decision, they simply return a matching score. The likelihood ratio of an imposter and the genuine joint probability density functions are calculated. The imposter and genuine distributions of each matcher must be estimated.

1.2.9 ***Decision-level fusion in fingerprint verification*** [5]

A combination of different fingerprint matching algorithms augment the verification accuracy, with a likelihood ratio used to make a final decision. The authors demonstrate that the combination scheme using a likelihood ratio outperforms both sum and product rules. They claim that this combination is optimal in the Neyman-Pearson sense when sufficient data is available to obtain the distribution of the classifier outputs. The use of more than one finger is also demonstrated, and the performance gain is better than the use of different matching algorithms. They

demonstrate that the combination of the best and the weakest classifiers give better results than combining the best two classifiers.

1.2.10 *Fingerprint verification by fusion of optical and capacitive sensors* [4]

Two fingerprint sensors obtain two images from the same finger, and a string matching algorithm is implemented for both images, with a decision made at the decision level. Two types of combination scheme are used. The first one simply calculates the average of both matching scores while the other uses a gradient descent algorithm to find an appropriate weighting value for each score. The latter combination scheme gives better accuracy than the first one. Although the use of more than one feature improves recognition accuracy, the approach requires a high degree of user interaction with the system.

1.2.11 *Fingerprint Image Enhancement: algorithm and performance evaluation* [12]

This paper introduces the use of a Gabor filter, one of the most accepted algorithm for fingerprint preprocessing, for low-pass filtering along the ridge orientation, while performing band-pass filtering along the direction orthogonal to the ridge. To use a Gabor filter, the ridge orientation and its frequency must be estimated. The effect of noise is considerably reduced using this method.

1.2.12 *Fingerprint minutiae extraction from skeletonized binary images* [13]

This paper proposes a systematic method for fingerprint post processing. One fingerprint contains only 40-60 real minutiae, but a preprocessed fingerprint might contains up to 2000-3000 minutiae. Most spurious minutiae can be deleted in a pre-filtering stage. Skeleton enhancement deletes spurious minutiae arising from bridged or spur-like ridges. Topological validation verifies bifurcation and termination points. The paper also introduces the separation of highly reliable minutiae from less reliable minutiae.

1.3 Objectives

The primary goal of this thesis is to propose a new technique for increasing the accuracy of fingerprint recognition systems without burdening the user with extra tasks unlike other multimodal biometric systems. By using biometric features extracted from a top-view finger image, in conjunction with fingerprinting, multimodal biometric can be implemented. The algorithm should not require much computation time, and the combined system has an acceptable response time.

The second objective is to investigate the possibility of fingerprint indexing. If the new top-view method requires less computation than fingerprint matching, then it might be useful as a way of reducing the search space, thus increasing fingerprint matching speed.

1.4 Scope of the Thesis

Top-view finger imaging has not been proposed before, so this thesis will act as a feasibility study for using top-view finger imaging to recognize people. Feature extraction and matching for top-view finger imaging must be designed for good recognition accuracy while requiring low computation time. It must be combined with fingerprinting to construct a multimodal biometric system. The system's performance must be evaluated in terms of recognition accuracy and computation time to ensure that it can be implemented on a conventional personal computer without requiring extra hardware acceleration.

To reduce the time to build the software, a well known fingerprint preprocessing and matching algorithm will be utilized. A cheap CCD camera with a resolution of 640×480 pixels will be used to capture the top-view finger image in a light-controlled environment to avoid non-uniform illumination. Measured from the fingertip, a captured top-view finger image will contain no more than 2 inches of the finger digit. The implemented system will only utilize a touch-based fingerprint sensor.

The wrinkles on a finger usually increase over time, but this thesis will not deal with this issue. It is very likely that the top-view feature can help fraud reduction, but we will not pay much attention to this area.

1.5 Organization of the Thesis

The organization is as follows:

- This chapter introduced the subject and scope of the thesis.
- Chapter 2 describes the theoretical background to our work, and the proposed top-view feature extraction and matching mechanism.
- Chapter 3 explains the experimental result for our implemented system hardware and software.
- Finally, Chapter 4 discusses the outcomes of this thesis, and supplies conclusions and possible future work.

CHAPTER 2

RESEARCH METHODOLOGY

This chapter offers a the theoretical background of image processing and biometrics, followed by methods used to implement the system's hardware and software, and ends with the materials needed to implement the system.

2.1 Theoretical Background

2.1.1 Image coordinates

An image is stored in a rectangular shape, where the left most rectangular corner is the origin point $(0,0)$, as shown in Figure 2.1. Any pixel in an image of size $m \times n$ can be accessed using a Cartesian coordinate. Let $p(i, j)$ represents the pixel in the i^{th} row and j^{th} column of the image.

In this thesis, a top-view finger image is stored as a 8-bit grayscale. This means that pixel intensity can vary from 0 to 255, with the value 0 represents the weakest intensity (black), while 255 represents the strongest intensity (white).

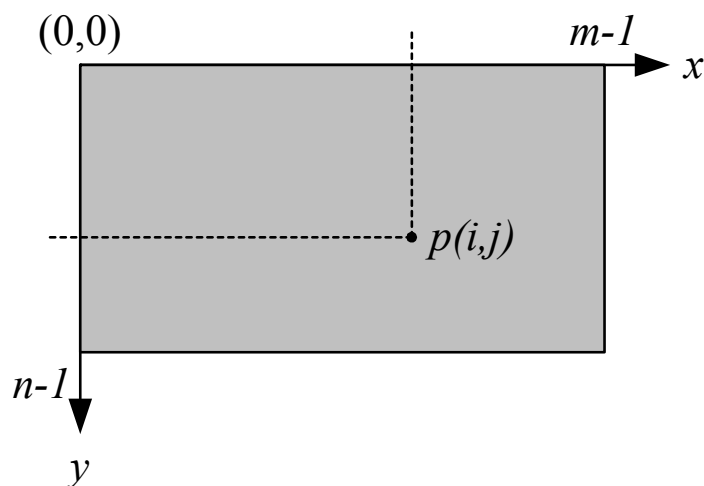


Figure 2.1. Image coordinate.

2.1.2 Set Operations and the Image

Let \cap and \cup be the intersection and union operations that will be applied to equal-sized binary images. Let $'$ denote the unary complement operation. Suppose 1 and 0 stand for a white and black pixel respectively, then the results of pixel-wise operations are shown in Table 2.1.

Table 2.1. Set operations applied to pixels.

A	B	$A \cap B$	$A \cup B$	A'
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

2.1.3 Image Thinning [14]

Thinning is the process of deriving a shape skeleton, which is one pixel thick and has a distance symmetrically to its boundary. There are many algorithms to find the skeleton of a basic shape; in this thesis, a morphological operation is used.

The morphological thinning operation, denoted \otimes , is derived by performing a Hit-or-Miss transformation(\circledast) to the input image, A , using a structuring element, B ;

$$A \otimes B = A \cap (A \circledast B)' \quad (2.1)$$

To ensure that the input image is thinned symmetrically, the input image A must be thinned using a sequence of structuring elements as follows:

$$A \otimes \{B\} = ((\dots((A \otimes B^1) \otimes B^2) \otimes B^3) \dots) \otimes B^n, \quad (2.2)$$

where $\{B\} = \{B^1, B^2, B^3, \dots, B^n\}$, and B^i is obtained by performing clockwise rotation of B^{i-1} , as shown in Figure 2.2. B^9 is identical to B^1 , and B^{10} is identical to B^2 and so on. Equation 2.2 means that the Hit-or-Miss operation must be performed on the input image A with the structuring element B^1 and then the result further modified using the structuring element B^2 , and so on, until no further changes occur to the image.

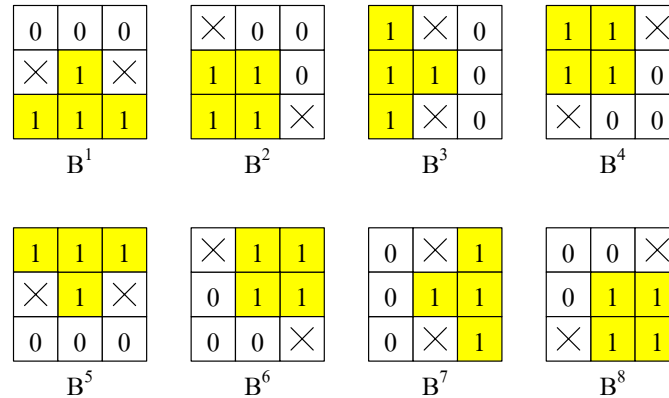


Figure 2.2. Sequence of structuring elements used for thinning. The \times 's stand for "don't care" values.

2.1.4 Adaptive Threshold [14]

Thresholding is a standard method for image segmentation, which separates an object from its background by specifying a threshold value, T . A pixel whose intensity is greater than the threshold value is claimed as an object pixel; otherwise, it becomes a background pixel. Thresholding can also be used to convert a grayscale image into a binary image. We define a binary image, $b(x,y)$, as:

$$b(x,y) = \begin{cases} 1 & \text{if } p(x,y) > T \\ 0 & \text{otherwise.} \end{cases} \quad (2.3)$$

If there is more than one object in the image, multilevel thresholding can be used. For example, if two objects exist in the image, two thresholds, T_1 and T_2 , are utilized. The image pixel $p(x,y)$ is classified using:

$$p(x,y) \in \begin{cases} \text{background} & \text{if } p(x,y) \leq T_1 \\ \text{object1} & \text{if } T_1 < p(x,y) \leq T_2 \\ \text{object2} & \text{otherwise.} \end{cases} \quad (2.4)$$

The process is called global thresholding if the threshold value (or values when multilevel thresholding is used) is applied across every pixel of the image. It is called adaptive thresholding when different threshold values are applied to different areas of the image, thereby dividing the image into sub-images. Different threshold values may be used to segment each subimage.

An appropriate threshold value can be obtained manually or automatically. Automatic thresholding is carried out as follows:

- 1) The input image is divided into subimages of size $B \times B$.
- 2) An initial thresholding value T is estimated from the average gray level of the subimage of interest.
- 3) A subimage is segmented using T , creating two groups of pixels; G_1 contains all the pixels with intensity values greater than T , while G_2 contains all the pixels with intensity value $\leq T$.
- 4) Average intensity values of all pixels in group G_1 and G_2 are calculated, producing μ_1 and μ_2 .
- 5) A new threshold value is calculated using:

$$T = 0.5(\mu_1 + \mu_2) \quad (2.5)$$

- 6) The pixels in the subimage are segmented with the new threshold T using:

$$p(x, y) = \begin{cases} 0 & \text{if } p(x, y) > T(x, y) \\ 255 & \text{otherwise.} \end{cases} \quad (2.6)$$

2.1.5 Gradient operator [14]

The gradient of an image, $f(x, y)$, at coordinate (x, y) is defined as a 2-D column vector:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}. \quad (2.7)$$

The gradient magnitude is calculated using:

$$\nabla f = \sqrt{G_x^2 + G_y^2} \quad (2.8)$$

To reduce the computational requirements of equation 2.8, the gradient magnitude can be estimated using:

$$\nabla f \approx |G_x| + |G_y| \quad (2.9)$$

The gradient direction is calculated using:

$$\Theta = \tan^{-1} \left(\frac{G_x}{G_y} \right) \quad (2.10)$$

The implementation of the partial derivatives for $\partial f / \partial x$ and $\partial f / \partial y$ utilize the Sobel or Prewitt operators. The convolution masks of these operators are shown in Figure 2.3. The Sobel mask is more difficult to implement than the Prewitt mask but gives better noise reduction.

G_x			G_y			G_x			G_y		
-1	-1	-1	-1	0	1	-1	-2	-1	-1	0	1
0	0	0	-1	0	1	0	0	0	-2	0	2
1	1	1	-1	0	1	1	2	1	-1	0	1
Prewitt						Sobel					

Figure 2.3. Convolution masks of Prewitt and Sobel operators.

2.1.6 Canny Edge detection [15], [16]

There are many ways to find the edge of an image, which can be classified into two categories: gradient and Laplacian methods. Canny is a gradient method, first proposed by John F. Canny in 1986. It is an optimal edge detection algorithm, containing 4 main steps:

- 1) *Smoothing*: Noise is reduced by smoothing the input image using a Gaussian filter. The size of the Gaussian convolution mask can be varied depending on the degree of smoothing. A large size increases noise reduction, at the expense of removing small details on the edges.
- 2) *Gradient Calculation*: Depending on the computational requirements, the gradient operator can be utilize either the Sobel, Roberts, and Prewitt

operators. Equations 2.9-2.10 are calculated to derive the edge image, and the gradient direction. The direction is discretized into one of four possibilities (0° , 45° , 90° , and 135°). For example, an edge direction falling in the range $[22.5, 67.5)$ is set to 45° , while an edge direction falling in the range $[67.5, 112.5)$ is set to 90° .

- 3) *Nonmaxima Suppression*: If two neighbor pixels in the direction orthogonal to the gradient direction have intensities lower than the edge pixel of interest, then the two pixels are considered nonmaxima and are deleted, producing a thin line in the edge image. For example, if the gradient direction is 45 degree, the northwest and the southeast pixels must be examined.
- 4) *Hysteresis Thresholding*: The edges detected with steps 1-3 are thresholded, utilizing two threshold values T_1 and T_2 , with $T_1 > T_2$. If the gradient magnitude of an edge pixel of interest is higher than T_1 , it is left untouched. Edge pixels with gradient magnitudes lower than T_2 are deleted. An edge pixel with gradient magnitude lower than T_1 but higher than T_2 will appear in the final edge image if at least one of its neighbor pixels has a gradient magnitude greater than T_1 .

2.1.7 Euclidean Distance

Euclidean distance is the most commonly used distance measurement. In two-dimensional space, the Euclidean distance between point (x_1, y_1) and (x_2, y_2) is calculated from

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (2.11)$$

Equation 2.11 states that the Euclidean distance is the shortest path between two points. In Euclidean n -space, the Euclidean distance between point \mathbf{p} and \mathbf{q} is calculated using the formula

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^n (p_i - q_i)^2} \quad (2.12)$$

where $\mathbf{p} = (p_1, p_2, \dots, p_n)$ and $\mathbf{q} = (q_1, q_2, \dots, q_n)$.

2.1.8 Biometrics

Biometric recognition is the use of biological data to distinguish individuals. It can employ distinctive physiological data, such as the face, iris, retina, facial thermogram, fingerprint, ear, DNA, hand geometry, or hand vein. It may also utilize distinctive behavioral data, such as the person's signature, gait, or voice. Biometric technology research groups include the Biometrics Research Group at the Department of Computer Science and Engineering, Michigan State University (<http://biometrics.cse.msu.edu>, 2009), the Biometric System Laboratory at the University of Bologna, (<http://biolab.csr.unibo.it>, 2009), and the National Biometric Test Center at San Jose State University, (<http://www.engr.sjsu.edu/biometrics>, 2009).

Fingerprinting is used more often than other biometric features because it only requires low cost system components, is convenient to use, has high robustness, and individual fingerprints are unique and time invariant [1].

2.1.9 Performance Evaluation of Biometrics system

Recognition begins with a score obtained from a biometric matcher. The score may be the similarity or the difference between an input pattern and a database feature.

Let Q stand for a biometric template stored in a database, and I the input biometric feature. The input pattern is assumed to fall into one of two possible classes, ω_1 and ω_2 . ω_1 stands for the imposter class (i.e. I and Q do not come from the same individual) and ω_2 for the genuine class (i.e. I and Q do come from the same individual). Suppose that the matcher reports the similarity value, s , between I and Q . The person who supplied the input I will be classified as an intruder if s is less than a threshold value, τ , otherwise, the person will be classified as a genuine user. The decision made by the system can belong to one of four categories:

- *genuine accept*: the system matches the input to the correct database template;
- *genuine reject*: the system correctly rejects the input as an intruder, whom is not enrolled in the database;
- *false accept*: the system incorrectly matches the input to an incorrect template in the database;
- *false reject*: the system incorrectly rejects a genuine user.

To measure system performance, the FAR (False Acceptance Rate) and FRR (False Rejection Rate) system values must be calculated. They are duels of each other: if the system FAR increases, then the FRR decreases, and vice versa. Figure 2.4 shows the relationship between the system threshold value and these values.

FRR and FAR can be estimated from

$$FAR = \int_{\tau}^{+\infty} p(\omega_1 | s) ds \quad (2.13)$$

$$FRR = \int_{-\infty}^{\tau} p(\omega_2 | s) ds \quad (2.14)$$

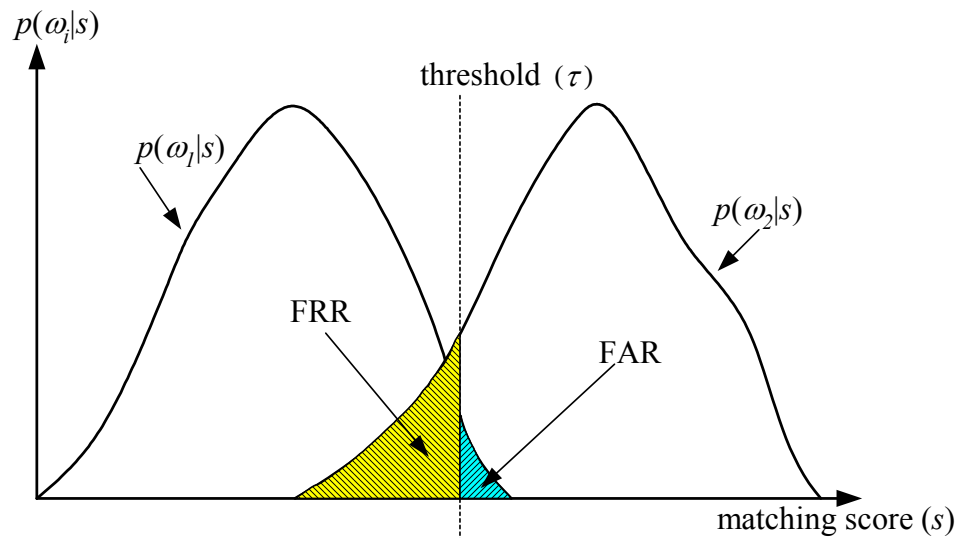


Figure 2.4. The system FAR and FRR for a given threshold, τ .

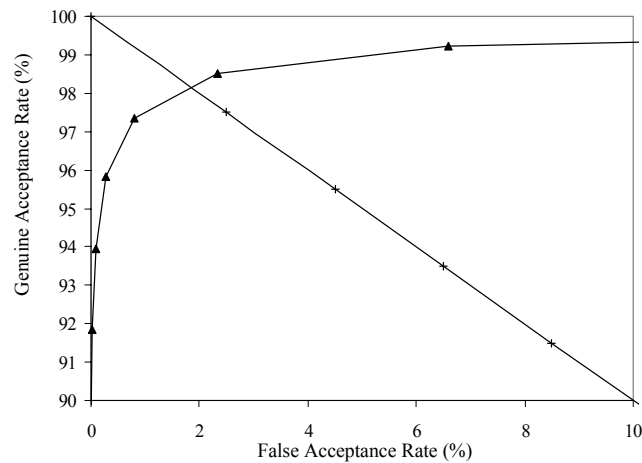


Figure 2.5. Receiver Operating Characteristic (ROC) curve

The Receiver Operating Characteristic (ROC) curve shows overall biometrics performance, as in Figure 2.5. An ROC curve plots FAR against the Genuine Acceptance Rate (GAR), which is $(1 - \text{FRR})$. Some research literature use the terms FMR (False Match Rate) and FNMR (False Non Match Rate) rather than FAR and FRR, but FAR and FRR is more common.

2.1.10 Multimodal biometrics

Different classifiers usually misclassify different input patterns [1], so overall performance is improved by employing multiple classifiers. Multimodal biometrics is simply the use of more than one biometric recognizer to identify a person, and is widely practiced, as shown in table 2.2.

Table 2.2. Examples of multimodal biometric systems.

Biometric features	Proposed by
Two different fingers	Prabhakar and Jain, 2002 [5]
Two different types of fingerprint sensors	Marcialis and Roli, 2004 [4]
Speaker and face	Brunelli and Falavigna, 1995 [17] Duc <i>et al.</i> , 1997 [18] Kitler <i>et al.</i> , 1997 [19] Choudhury <i>et al.</i> , 1999 [2] Verlinde, Chollet and Acheroy, 2001 [21] Ben-Yacoub <i>et al.</i> , 1999 [22]
Face with fingerprint and speaker	Jain, Hong and Kulkarni, 1999 [3]

2.1.11 Fingerprints

A fingerprint image replicates the epidermis structure of a person's finger. It is composed of ridges interleaved with valleys, as shown in Figure 2.6. A person's fingerprints are fully formed about 2 months before birth, and the finger ridge structure remains the same throughout the person's life [23].

There are many ways to acquire an individual's fingerprint. The simplest is by inking a finger and pressing it onto a paper. Nowadays, there are also various types of sensors, such as optical, capacitive, thermal, piezoelectric, and ultrasound devices, as shown in Figure 2.7.



Figure 2.6. A fingerprint image.



Figure 2.7. Four different types of fingerprint sensor;

- a) touch-based capacitive sensor (<http://www.biometrics-china.com>, 2009);
- b) touch-based optical sensor (<http://www.digitalpersona.com>, 2009);
- c) sweep-based capacitive sensor (<http://www.scantastik.com>, 2009);
- d) touchless optical sensor (<http://www.tst-biometrics.com>, 2009).

Fingerprint processing requires two main steps: enrollment and matching. Enrollment inserts new fingerprint data into a database, while matching deals with the comparison between the input fingerprint and fingerprints in the database. There are two modes of fingerprint recognition: verification and identification. During verification, an input fingerprint is compared with one database item to check if they come from the same finger. During identification, all the database's fingerprints must be searched to find the best match with the input. Computation time is an issue for the identification mode, especially in systems with large databases. To reduce the search space of fingerprint identification, Senior; Hao and Zong; Munir and Javed classified fingerprint into 5 categories: arch, left loop, right loop, tented arch, and whorl [24], [25], [26]. Another way of reducing the search space is by indexing [27].

The acquisition of a fingerprint by a sensor can be categorized into 3 different methods, i.e., touch, sweep, and touchless. A fingerprint image will not usually be stored in the database, instead, feature extraction is applied to the image and prominent features are stored instead. There are two types of fingerprint features: minutiae and FingerCode [28], [29], of which minutiae is the most common. There are many types of minutiae, as shown in Figure 2.8(a), but only the termination type (sometimes called endpoint) and the bifurcation type are common. Each minutia is represented by a (x, y) coordinate and the angle of the respective ridge line, as shown in Figure 2.8(b).

Before feature extraction, the supplied grayscale fingerprint must be preprocessed because most sensors lack image enhancement capabilities, (except the sensor proposed by Shigematsu *et al* [30]). The main role of preprocessing is to reduce the input image's noise, after which the image is converted into a binary image. The thinning process is applied, and a set of minutiae extracted from the result. Post-processing [3], [31] is applied to the minutiae to remove spurious results.

Some approaches do not perform minutiae extraction on a thinned binary image, but instead extract minutiae directly from the grayscale fingerprint image. This avoids the possibility of some relevant data disappearing or errors occurring during image conversion [32], [33].

Fingerprint processing can be implemented on many platforms, from microcomputers to embedded systems [34], [35]. The fingerprint sensor simply sends raw data to a processing module, which can be implemented in software alone or with software and special designed accelerator hardware. For example, Ratha *et al.* [36], Lindoso *et al.* [37] use an array processor built from FPGA for matching the

fingerprint with the large database. Although hardware offers astonishing speed ups, it is costly to build and unsuitable for small fingerprint databases.

2.1.12 Standard fingerprint databases

Four standard fingerprint databases are widely used to test system performance: NIST [38], FVC2000 [39], FVC2002 [40], and FVC2004 [41]. However, in this thesis, since two recognition features are utilized together, these databases will not be utilized for performance testing. The multimodal biometric systems of Prabhakar and Jain [5], Hong and Jain [2] and Marcialis and Roli [4] also did not use these databases for similar reasons.

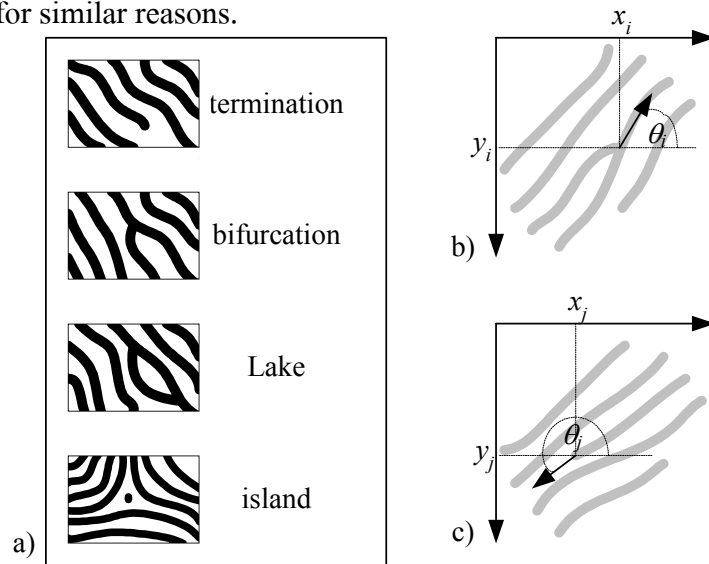


Figure 2.8. a) Examples of the 4 Minutia types; b) Feature $[x_p, y_p, \theta_p]$ extracted from a bifurcation i ; c) Feature $[x_p, y_p, \theta_p]$ extracted from a termination minutia j .

2.1.13 Fingerprint deformation

When a user presses his finger onto a sensor, the three dimensional epidermis shape is converted into two dimensional data. Finger pressure or the angle between the finger and the sensor plate may cause a non-linear deformation of the fingerprint image which varies over the image area. The deformation's effect on a (x, y) minutiae coordinate is called translation deformation; the deformation's effect on the minutiae angle is called rotational deformation. A fingerprint matching algorithm can utilize a bounding box to handle these two deformations [42]. The size of the box affects the

biometrics's system accuracy: a large bounding box tends to increase the FAR; a small bounding box tends to increase the FRR. A deformation model may help to reduce the effect of a large bounding box. For example, Bazen and Gerez propose a deformation model of fingerprint [43], Senior and Boole employ a canonical fingerprint form, and Lee *et al* utilize a method to normalize the distance between ridges [45]. All of these methods reduce the fingerprint recognition error.

2.2 Method

The system hardware must digitize the fingerprint and the corresponding top-view finger image, and the software must then process both top-view and fingerprint. Before the system performance can be evaluated, a test database of finger images must be collected. The data on each finger will be collected several times, one for enrolment into the system, and the others for testing performance.

Fingerprint processing consists of 4 main steps:

- 1) preprocessing;
- 2) feature extraction;
- 3) post processing;
- 4) matching.

Fingerprint preprocessing uses a technique proposed by Hong *et al.* [12], but feature extraction and post-processing employs our own techniques. The details of the fingerprint matching algorithms are explained in section 2.2.11.

Since top-view finger imaging has not been proposed before, new techniques for its preprocessing, feature extraction, and matching must be investigated. The proposed techniques are explained in sections 2.2.3-2.2.10.

The performance of top-view finger imaging must be evaluated on its own before combining it with standard fingerprinting in a multimodal biometric system. After the performance is known for separate top-view and fingerprint matchers, the performance of the combination can be assessed.

2.2.1 Overview of the system

The system hardware consists of an optical fingerprint sensor equipped with a CCD camera in a light controlled environment. A Pentium4 2.4 GHz personal computer interfaces with the two sensors without any additional hardware acceleration.

As shown in Figure 2.9, the system software consists of 5 parts:

- 1) *The fingerprint feature extraction module* reads an image from the fingerprint sensor, and preprocesses the image before performing feature extraction. Minutiae features are extracted, and any spurious minutiae are deleted using techniques described in section 2.2.19.
- 2) *The fingerprint matching module* calculates the similarity value between the input image and a template stored in the system database. The algorithms are explained in section 2.2.11.
- 3) *The top-view feature extraction module* reads a top-view finger image from the CCD camera. A NailCode feature map is extracted after image pre-processing.
- 4) *The top-View matching module* computes the distance between the input and the template.
- 5) *The decision fusion module* combines the matching scores from the two matchers to make a final decision.

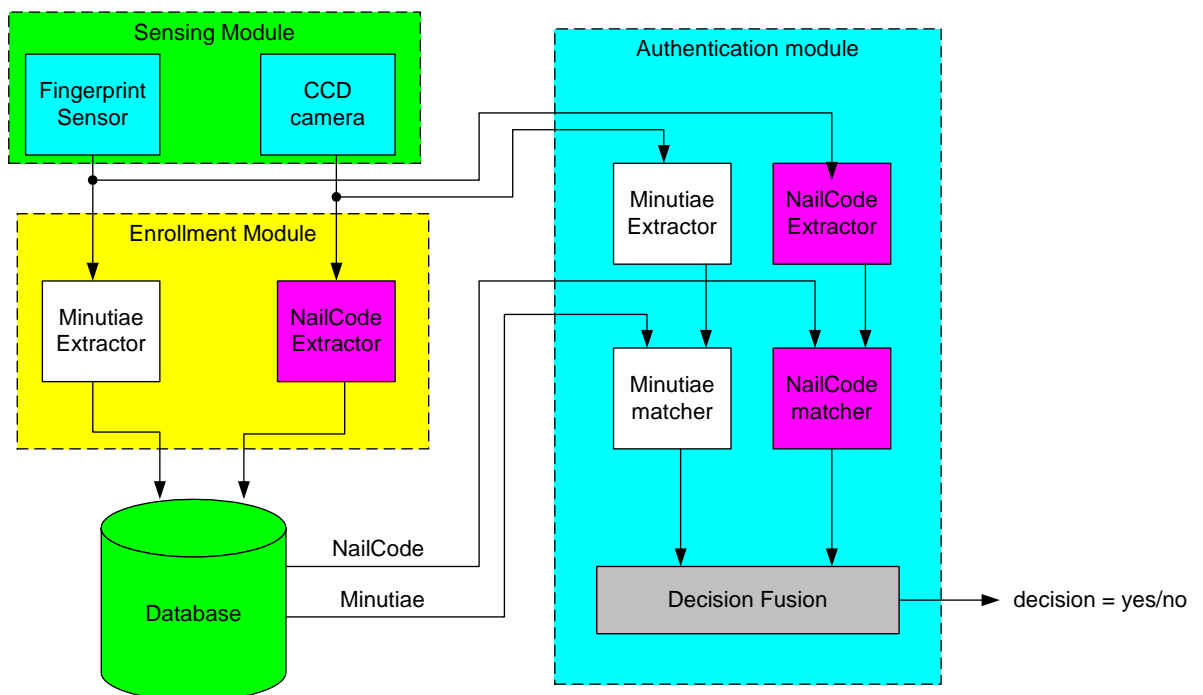


Figure 2.9. Block diagram of the whole system.

2.2.2 Top-View finger image acquisition

The acquisition hardware comprises a Creative VF0080 CCD camera (resolution 640x480) and a digital Persona UareU4000B fingerprint sensor (resolution 512 dpi). To avoid issues with non-uniform illumination, a light controlled environment is maintained, as shown in Figure 2.10, with ten blue LEDs as light sources for the top-view finger image acquisition. The optical fingerprint sensor automatically switches on a red light whenever it senses a finger pressing down. The grayscale top-view finger image is derived from the blue component of a color image obtained from the CCD camera, since the wavelength of blue is farther from red than green [14]. This means that the interference of red light with the blue component of a color image is less than with a green component. Measured from the fingertip, the captured top-view finger image contains no more than 2 inches of the finger digit. The distance between the CCD camera and the fingerprint sensor is adjusted to give the maximum amount of detail for the finger along with finger inclination detection. This setup yields an image resolution of 250 dots per inch.

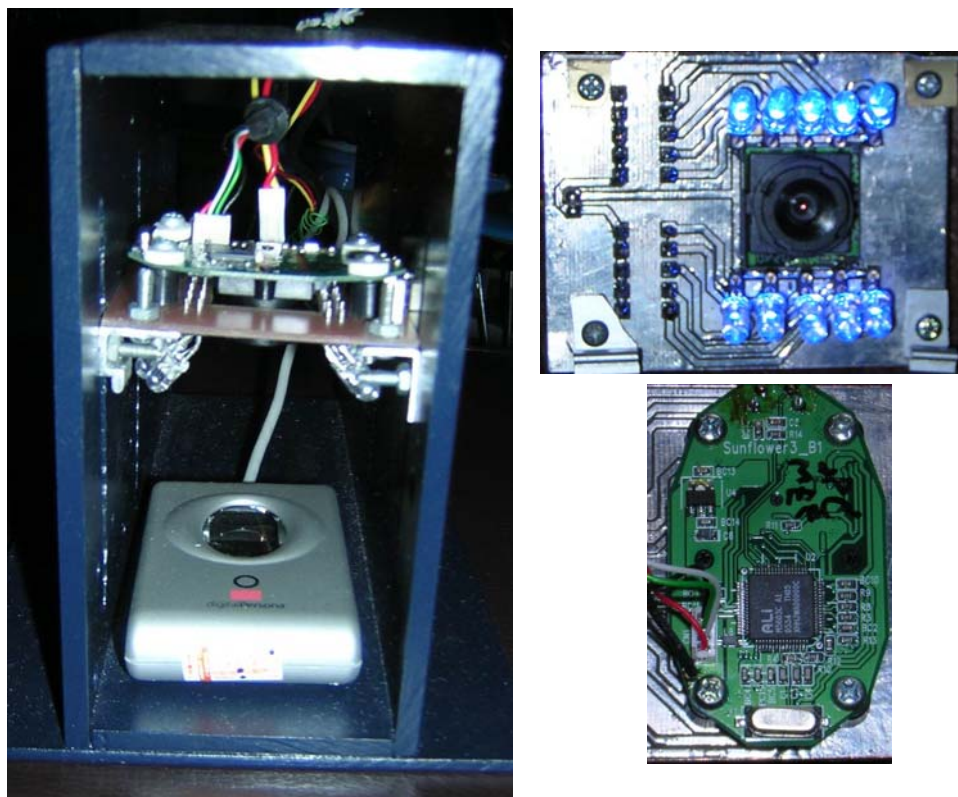


Figure 2.10. System hardware used to digitize fingerprint and top-view finger images.

2.2.3 Top-View finger image Preprocessing

The greyscale top-view finger image obtained from the CCD camera is T_G , and has size $W \times L$ (see Figure 2.12(a)). Its preprocessing flowchart is shown in Figure 2.11. The main steps include:

- 1) *Smoothing*. Due to the presence of noise and non-uniform illumination in the image, a smoothing Gaussian filter is applied to T_G .
- 2) *Binarization*. The greyscale image is converted into black and white image (black=0 and white=255) using an adaptive threshold, resulting in an image shown in Figure 2.12(b).
- 3) *Small Particle Deletion*. Small particles made up of white pixels less than the threshold value are deleted. The resulting image, T_H is shown in Figure 2.12(c).
- 4) *Background Deletion*. The background of the finger image is deleted using the method described in section 2.2.5. Figure 2.12(d) shows the resulting image.
- 5) *Finger Inclination Correction*. The image is rotated to align vertically with the x-axis of the image, as shown in Figure 2.12(e).
- 6) *Skeletonization*. A thinning operation is applied to the image to create a skeleton for the remaining lines in the image, as shown in Figure 2.12(f).
- 7) *FilterBank*. The skeletonized image is manipulated by a filterbank holding eight different filtering directions. The results are eight images ready for feature extraction. The details are explained in section 2.2.7.

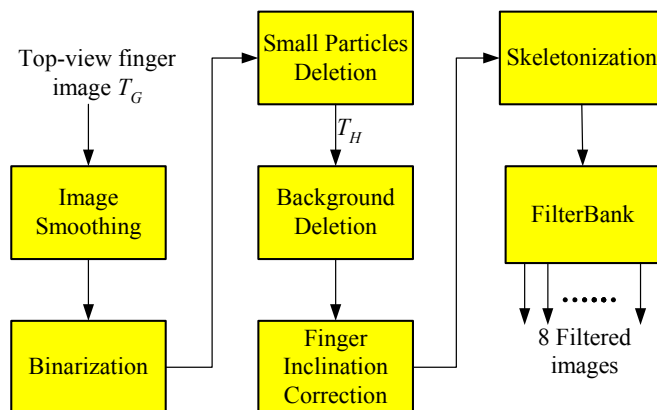


Figure 2.11. Flowchart of the preprocessing algorithm.

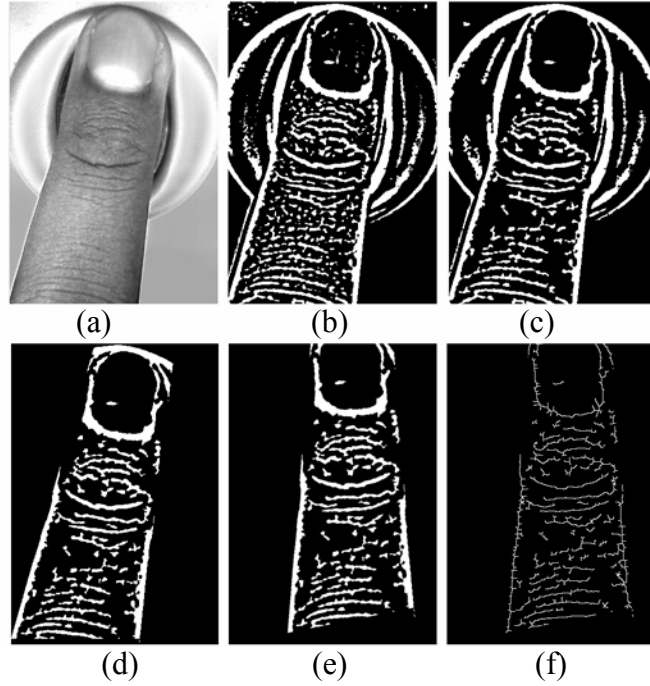


Figure 2.12. Images obtained in each preprocessing step: (a) T_G ; (b) binarized image; (c) T_H ; (d) after background deletion; (e) after inclination correction; (f) skeletonized image.

2.2.4 Finger image alignment parameter

When a finger is pressed on the fingerprint sensor, it may be up to $\pm 30^\circ$ away from the assumed vertical orientation. The inclination is detected, and the image is rotated as follows:

- 1) The Canny algorithm is applied to the T_H image, producing an edge image, T_K , which is copied into two images named T_L and T_R using the conditions:

$$T_L(x, y) = \begin{cases} T_K(x, y), & \begin{pmatrix} 0 \leq x < W/2 \\ L/2 \leq y < L \end{pmatrix} \\ 0, & \text{otherwise} \end{cases} \quad (2.15)$$

$$T_R(x, y) = \begin{cases} T_K(x, y), & \begin{pmatrix} W/2 \leq x < W \\ L/2 \leq y < L \end{pmatrix} \\ 0, & \text{otherwise.} \end{cases} \quad (2.16)$$

- 2) The parameters for the left-edge of the finger are obtained by letting C be the set of contours in T_L , where $C = \{C_1, C_2, C_3, \dots, C_k\}$ and $k = \text{number of contours}$. A line-fitting algorithm [46] is applied to each contour C_i to find its straight-line parameter S_i . For each S_i we have:

$$S_i = (V_x^i, V_y^i, X_0^i, Y_0^i), \quad i = 1..k \quad (2.17)$$

where (V_x^i, V_y^i) is a normalized vector parallel to the fitted line and (X_0^i, Y_0^i) is a point on that line [46].

- 3) The parameter S_{left} of the left-edge finger is selected from the set of S_i using the condition:

$$S_{left} = S_j \quad \text{where} \quad \left(\begin{array}{l} \text{abs}(\tan^{-1}(V_y^j / V_x^j)) \leq \pi / 6 \quad \text{and} \\ N_j > N_i \text{ for all } j \neq i, \quad i = 1..k \end{array} \right) \quad (2.18)$$

where N_i is the number of white pixels in each contour C_i .

- 4) The parameter S_{right} of the right-edge finger can be derived by applying steps 2-3 to T_R .

$$S_{left} = (V_x^l, V_y^l, X_0^l, Y_0^l) \quad (2.19)$$

$$S_{right} = (V_x^r, V_y^r, X_0^r, Y_0^r) \quad (2.20)$$

The parameter S_{left} and S_{right} are used in the background deletion and finger image inclination process.

2.2.5 Background deletion

Since the CCD camera is located above the fingerprint sensors, then the top-view finger image will include an image of the fingerprint sensor. This must be removed so that only the finger image is processed. Background deletion is achieved as follows:

- 1) Image M_L , which has the same size as T_G , is created using the condition:

$$M_L(x, y) = \begin{cases} 255 & \text{if} \left(\begin{array}{l} (m_l < 0 \text{ and } y \geq m_l x + c_l) \\ \text{or } (m_l > 0 \text{ and } y \leq m_l x + c_l) \end{array} \right) \\ 0 & \text{otherwise} \end{cases} \quad (2.21)$$

$$\text{where } m_l = \frac{V_y^l}{V_x^l} \text{ and } c_l = Y_0^l - \frac{V_y^l X_0^l}{V_x^l}.$$

- 2) Image M_R , which has the same size as M_L , is created using the condition:

$$M_R(x, y) = \begin{cases} 255 & \text{if } \left((m_r < 0 \text{ and } y \leq m_r x + c_r) \right. \\ & \left. \text{or } (m_r > 0 \text{ and } y \geq m_r x + c_r) \right) \\ 0 & \text{otherwise} \end{cases} \quad (2.22)$$

$$\text{where } m_r = \frac{V_y^r}{V_x^r} \text{ and } c_r = Y_0^r - \frac{V_y^r X_0^r}{V_x^r}.$$

- 3) The background-deleted image, T_F , is created from the operation:

$$T_F = M_R \cap M_L \cap T_H \quad (2.23)$$

where \cap is a pixelwise-intersection operation, applied to equal-sized images.

2.2.6 Finger image inclination correction

- 1) Let ρ and λ be the angles of inclination of the left and right edges of the finger respectively, defined by:

$$\rho = \tan^{-1} \left(\frac{V_y^l}{V_x^l} \right) \quad (2.24)$$

$$\lambda = \tan^{-1} \left(\frac{V_y^r}{V_x^r} \right) \quad (2.25)$$

- 2) The rotation of the finger around the origin is calculated using:

$$\varphi = \begin{cases} 0.5(\rho + \lambda) & \text{if } (\rho \times \lambda < 0) \\ 0.5(\rho + \lambda - \pi) & \text{else if } (\rho \geq 0 \text{ and } \lambda \geq 0) \\ 0.5(\rho + \lambda + \pi) & \text{otherwise.} \end{cases} \quad (2.26)$$

The value of ρ and λ range from -90° to 90° . Figure 2.14 shows all three different conditions to obtain φ .

2.2.7 Filterbank

The skeletonized top-view finger image is manipulated using a bank of oriented filters, with eight different θ values (0° , 22.5° , 45° , 67.5° , 90° , 112.5° , 135° , and 157.5°) with respect to the x -axis. The oriented filters enhance the ridge lines along the specified θ angles while blurring the lines that lie in other directions (see Figure 2.13). Figure 2.15 shows the kernel Oriented Filters for different θ values.

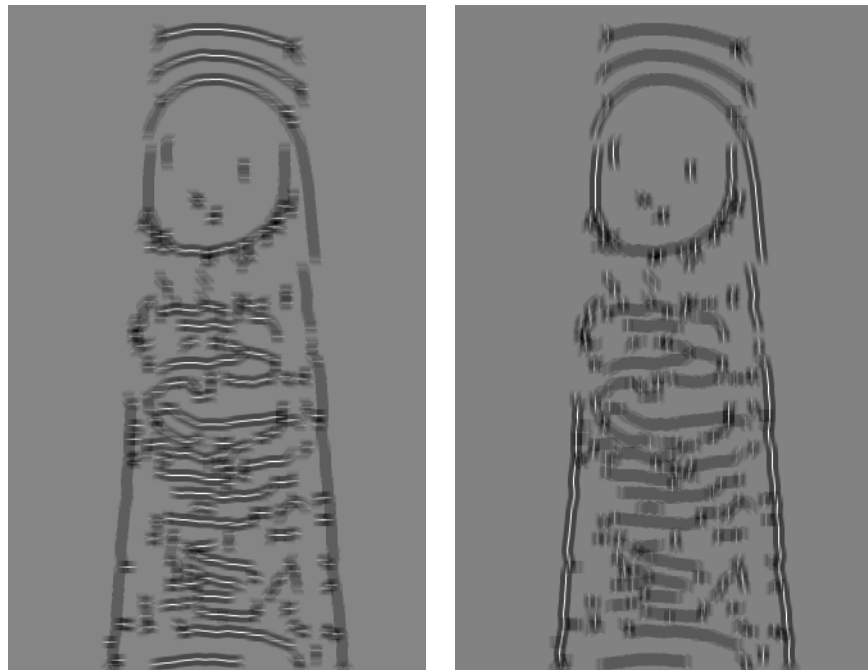


Figure 2.13. Finger images filtered at (a) 0° and (b) 90° .

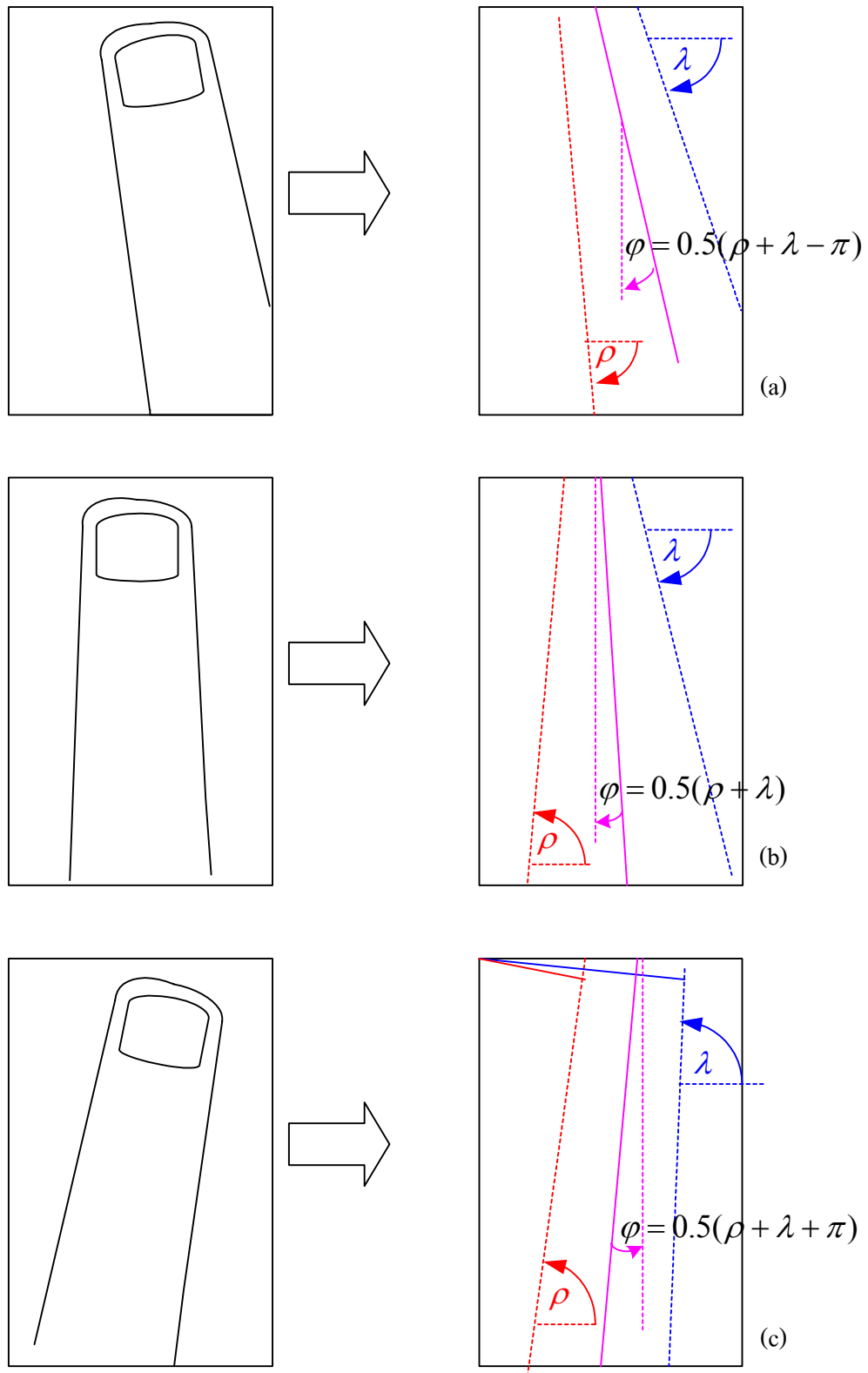


Figure 2.14 The derivation of φ for different conditions;

- (a) the signs of ρ and λ are negative;
- (b) the signs of ρ and λ are different;
- (c) the signs of ρ and λ are positive.

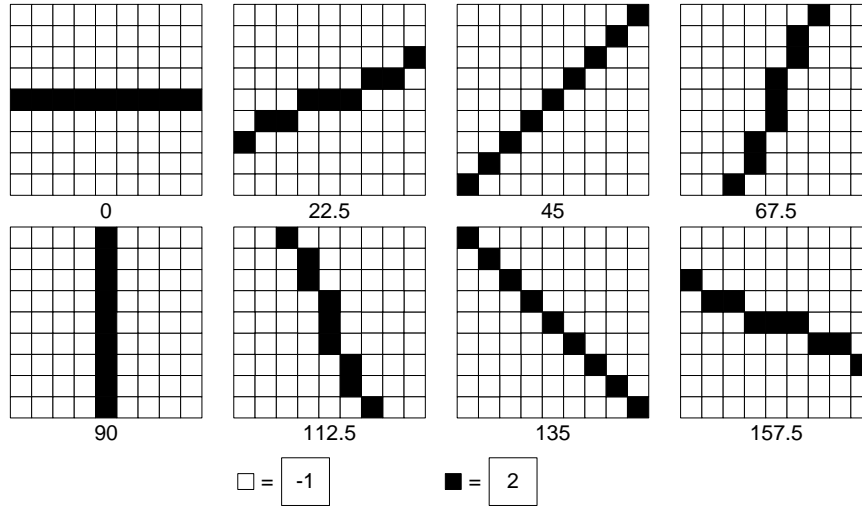


Figure 2.15. Kernel of Oriented Filters with different θ values.

2.2.8 Feature Extraction

Feature extraction is carried out as follows:

- 1) The top-view finger image reference point, located in the middle of the nail base, is obtained using the algorithm described in section 2.2.9, and shown in action in Figure 2.16(a).
- 2) θ is the degree setting on the oriented filter. The filtered image Q_θ is tessellated using the reference point (x_r, y_r) into $H \times V$ (10×15) square cells of size $w \times w$ (15×15), as shown in Fig. 2.16. $p(x, y)$ denotes the pixel intensity at location (x, y) of Q_θ . The variance for each square cell at location (h, v) is calculated using:

$$\sigma^2(h, v) = \frac{1}{w^2} \sum_{x=c}^d \sum_{y=a}^b p(x, y)^2 - \left(\frac{1}{w^2} \sum_{x=c}^d \sum_{y=a}^b p(x, y) \right)^2 \quad (2.27)$$

where

$$a = y_r + vw$$

$$b = y_r + w(v+1)$$

$$c = x_r + w(h-0.5H)$$

$$d = x_r + w(h-0.5H+1).$$

The tessellated area should cover the shape and the width of the nail base while avoiding problems with finger nail length variation. For that reason, we selected h to range from $0, 1, \dots, H-1$ and v to range from $-2, -1, 0, \dots, V-3$.

- 3) After applying step 2 to every filtered image, the extracted feature, called a NailCode, and denoted by Ψ^i , is calculated. Ψ^i for the images using reference point R_i , is defined by:

$$\Psi^i = [\Psi_{0^\circ}^i, \Psi_{22.5^\circ}^i, \Psi_{45^\circ}^i, \Psi_{67.5^\circ}^i, \Psi_{90^\circ}^i, \Psi_{112.5^\circ}^i, \Psi_{135^\circ}^i, \Psi_{157.5^\circ}^i]^T \quad (2.28)$$

where $\Psi_j^i = [\sigma_1^2, \sigma_2^2, \sigma_3^2, \dots, \sigma_{H \times V}^2]^T$.

- 4) Due to the possibility of a reference point detection error, a compensation technique is used. If R_0 is the reference point obtained by using the algorithm described in section 2.2.9, then there are eight translated versions, R_1 - R_8 , each δ pixels from R_0 , as shown in Figure 2.18. In the enrollment module, only Ψ^0 is extracted from the input top-view finger image. In the authentication module, the NailCode $N = \{\Psi^0, \Psi^1, \Psi^2, \dots, \Psi^8\}$ is extracted from the input top-view finger image.

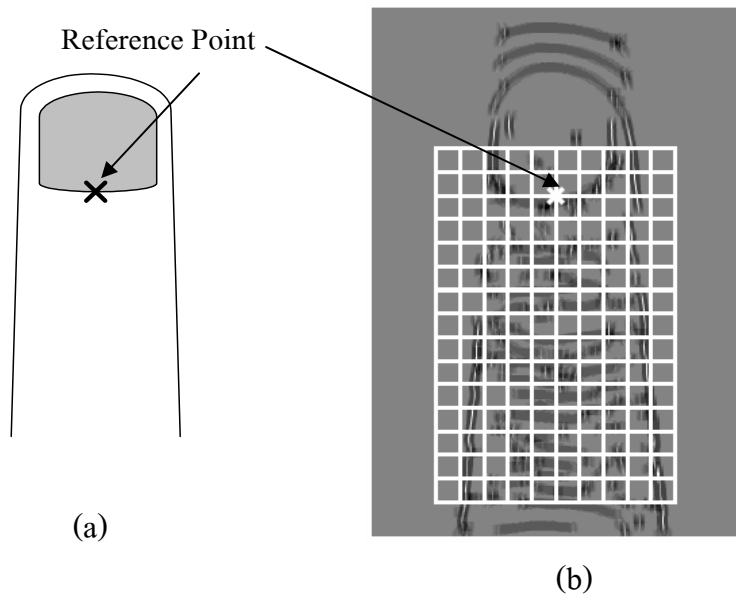


Figure 2.16. Reference point location and square tessellation on the filtered image.

2.2.9 Automatic detection of reference point location

The reference point of a top-view finger image is located at the midpoint of the finger's nail base. The steps for its detection are:

- 1) Let \cap , \cup and $'$ denote the intersection, union and inversion operations respectively. Image T_H , as shown in Figure 2.17(a), is employed to create image T_M using the condition:

$$T_M = (M_R \cap M_L \cap T_H) \cup (M_R \cap M_L)' \quad (2.29)$$

- 2) The image T_M , shown in Figure 2.17(d), is dilated and inverted before being rotated to be exactly vertical, resulting in T_p .
- 3) Let \mathcal{L} be the set of contours found in the image T_p :

$$\mathcal{L} = \{L_1, L_2, L_3, \dots, L_f\}$$

where f is the number of contours detected in the image. The reference point location can be derived by applying the algorithm described in Figure 2.19 to image T_p , using the following parameters in each iteration:

- N_i = The number of white pixels in each contour L_i
- BR_i = The bounding rectangle of each contour L_i . Each rectangle contains the parameters:

$$\{x_i^{BR}, y_i^{BR}, w_i^{BR}, h_i^{BR}\}$$

where y_i^{BR} = y coordinate of top edge of the rectangle corner

x_i^{BR} = x coordinate of left edge of the rectangle corner

w_i^{BR} = width of rectangle

h_i^{BR} = height of rectangle.

- $ratio_i = \frac{N_i}{w_i^{BR} \times h_i^{BR}}$.

As shown in Figure 2.17(e), our algorithm tries to find the largest contours in the top-view finger image which are expected to be the nail. To avoid the contours that are larger than the nail, the *ratio* of the width and the length of the bounding rectangle is calculated, and contours with a ratio less than r_{th} (0.6) are thrown away.

- 4) L_j is the selected contour obtained from the algorithm in Figure 2.19. The reference point $R(x,y)$ is computed on L_j with:

$$R(x,y) = (x_j^{BR} + 0.5w_j^{BR}, y_j^{BR} + h_j^{BR})$$

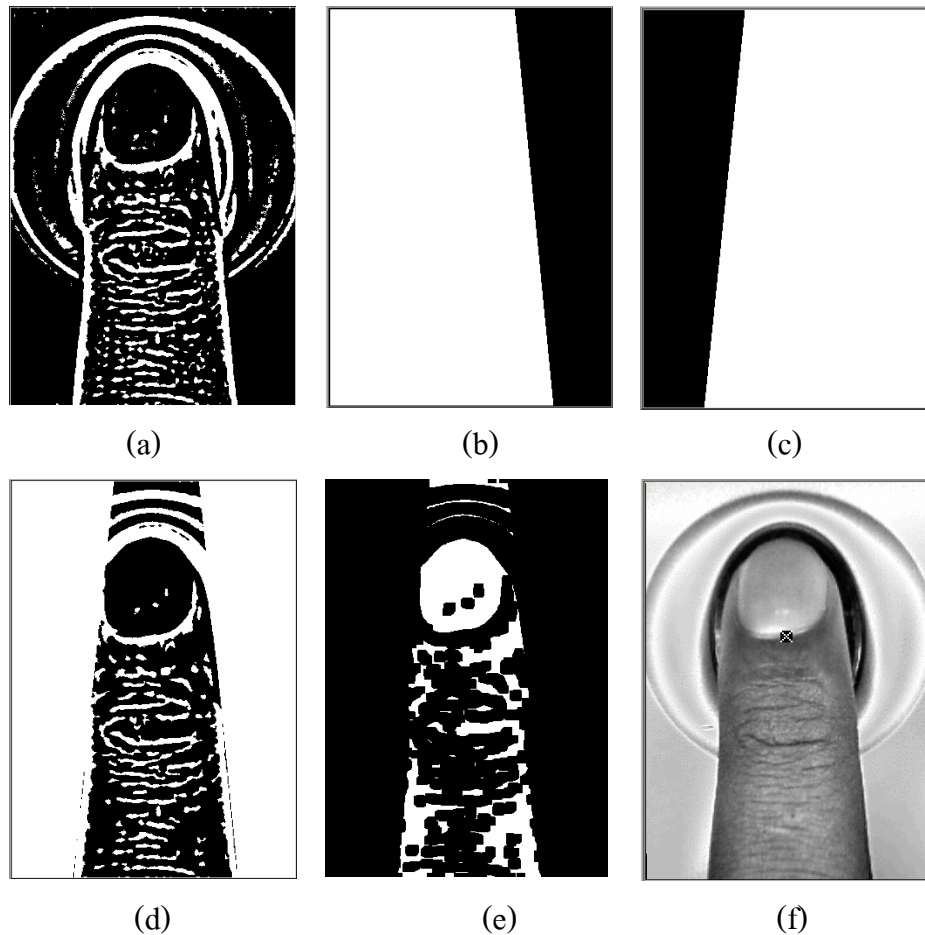


Figure 2.17. Top-view finger images for each step of the reference point detection algorithm: (a) T_H ; (b) M_R ; (c) M_L ; (d) T_M ; (e) T_P ; (f) the obtained reference point superimposes on a top-view finger image.

2.2.10 NailCode Matching

The Euclidean distance is computed as part of the matching operation. Let Ψ_T^0 be a NailCode template in the database and $N = \{\Psi_{IP}^0, \Psi_{IP}^1, \Psi_{IP}^2, \dots, \Psi_{IP}^8\}$ be the NailCode extracted from the input top-view finger image. Each E_i in the Euclidean distance $E = \{E_0, E_1, \dots, E_8\}$ is the distance between Ψ_T^0 and Ψ_{IP}^i . The matching score between the input and the template is:

$$matching_score_{top} = \min(E_0, E_1, \dots, E_8) \quad (2.30)$$

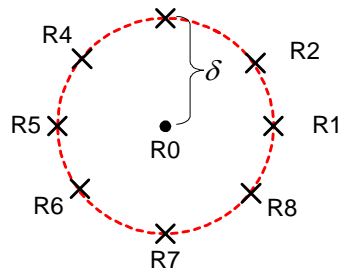


Figure 2.18. Eight points for reference point error compensation

```

num_loop=0
ref_pt_found = false
do
{
  all values in  $\mathcal{L}$  are deleted, giving  $\mathcal{L} = \{\}$ 
  find contours from image  $T_P$ , putting all the results in  $\mathcal{L}$ 
  for each contour  $L_i$ 
    if ( $N_i > \text{threshold}$  and  $\text{ratio}_i > r_{th}$ )
      ref_pt_found = true
    else
      remove  $L_i$  from  $\mathcal{L}$ 

  if (ref_pt_found = true)
    select contour  $L_j$  from  $\mathcal{L}$  with the largest ratio
  else
    perform erosion operation on image  $T_P$ 
    num_loop++
}
while (num_loop < MAX_LOOPS and ref_pt_found = false)

if (num_loop < MAX_LOOPS)
  extract the reference point from the contour  $L_j$ 
else
  the reference point can not be found, and the image is rejected

```

Figure 2.19. Reference point detection algorithm.

2.2.11 Fingerprint matching algorithms

Many fingerprint matching algorithms are proposed in the research literatures, e.g., local structure matching [47], [48], Hough Transform [36], Error Propagation [49], Energy based matching [50], Hidden Markov matching [51], and Correlation based matching [37], [52]. All of these methods can be classified into two categories: minutiae-based and texture-based. Two fingerprint matching systems based on minutiae matching are developed: the first uses Hough transform-based matching while the other uses our own algorithm. They are combined with the top-view finger image matching system as described in section 2.2.20.

2.2.12 Hough transform-based minutiae matching (HTMM)

This algorithm was proposed by Ratha *et al.* [36]. It tries to find the best transformation parameter (e.g. translation and rotation) between the input and the template minutiae. Each discretized transformation estimation is stored in an accumulator array, and the translation and rotation parameter are obtained by detecting the highest peak in the array. Since this algorithm uses an accumulator array in a similar way to a typical Hough transform, this algorithm is called Hough transform minutiae matching (HTMM). The details of this algorithm can be found in [1].

HTMM executes quickly but with low accuracy tolerances (compared to the other minutiae matching algorithms). Work by Prabhakar and Jain [5] confirm these characteristics.

2.2.13 Our proposed minutiae-based fingerprint matching (SMM)

Minutiae matching can be summarized by the following steps:

- 1) Let Z and R be the minutiae sets for the template and the input fingerprint,

$$Z = \{(x_1^Z, y_1^Z, \theta_1^Z), \dots, (x_z^Z, y_z^Z, \theta_z^Z)\}$$

$$R = \{(x_1^R, y_1^R, \theta_1^R), \dots, (x_r^R, y_r^R, \theta_r^R)\}.$$

- 2) A score table of size $z \times r$ is created, with all its values set to zero. z and r are the number of minutiae in Z and R , respectively.
- 3) Execute the algorithm shown in Figure 2.20.

Two minutiae are paired if and only if their direction distance and spatial distance are less than threshold values. The derived matching score is the number of matched minutiae between the input fingerprint and a templates. Because of its simplicity, this algorithm is called SMM (Simple Minutiae matching). The two distance values between minutiae (x^R, y^R, θ^R) and (x^P, y^P, θ^P) can be calculated using formula

$$sd = \sqrt{(x^R - x^P)^2 + (y^R - y^P)^2} \quad (2.31)$$

and

$$dd = \min(|\theta^R - \theta^P|, 2\pi - |\theta^R - \theta^P|) \quad (2.32)$$

where sd denote the spatial distance and dd represents for the direction distance.

```

for  $i=1$  to  $z$ 
  for  $j=1$  to  $r$ 
    {
      find the translation vector  $(\Delta x, \Delta y)^T$ 
      
$$\begin{bmatrix} \Delta x \\ \Delta y \\ 0 \end{bmatrix} = \begin{bmatrix} x_i^Z \\ y_i^Z \\ 0 \end{bmatrix} - \begin{bmatrix} x_j^R \\ y_j^R \\ 0 \end{bmatrix}$$

      translate all minutiae in  $R$ , storing the result in  $A$ 
      
$$\begin{bmatrix} x^A \\ y^A \\ \theta^A \end{bmatrix} = \begin{bmatrix} x^R \\ y^R \\ \theta^R \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ 0 \end{bmatrix}$$

      for  $(\Delta\theta = -\Phi; \Delta\theta \leq \Phi; \Delta\theta += \lambda)$ 
         $R^*$  is the rotated version of all minutiae in  $A$  using:
        
$$\begin{bmatrix} x^{R^*} \\ y^{R^*} \\ \theta^{R^*} \end{bmatrix} = \begin{bmatrix} \cos \Delta\theta & -\sin \Delta\theta & 0 \\ \sin \Delta\theta & \cos \Delta\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^A \\ y^A \\ \theta^A \end{bmatrix}$$

        Let  $S_{pair}^{\Delta\theta}$  be the number of paired minutiae between  $R^*$  and  $Z$ 
        
$$score\_table[i,j] = \arg \max_{\Delta\theta} S_{pair}^{\Delta\theta}$$

      }
    }
  }
  
$$matching\_score_{bim} = \arg \max_{i,j} score\_table[i,j]$$


```

Figure 2.20. Our SMM algorithm

2.2.14 Fingerprint preprocessing

In this thesis, the quality of the fingerprint image is enhanced using the algorithm proposed by Hong *et al.* [12]. The main steps of the algorithm are as follows:

- 1) Normalization: Let $I(i, j)$ and $G(i, j)$ denote the pixel intensity value of the input and output image of the normalization process respectively. M and VAR denote the estimated mean and variance of the input fingerprint I . The clarity of the fingerprint ridge structure is enhanced by applying the pre-specified mean M_0 and variance VAR_0 to the input image I using the formula:

$$G(i, j) = \begin{cases} M_0 + \sqrt{\frac{VAR_0 (I(i, j) - M)^2}{VAR}} & \text{if } I(i, j) > M \\ M_0 - \sqrt{\frac{VAR_0 (I(i, j) - M)^2}{VAR}} & \text{otherwise.} \end{cases} \quad (2.33)$$

- 2) Ridge orientation estimation: The normalized image is divided into blocks of size $w \times w$ (16×16). A local ridge orientation of each block center is estimated using the algorithm described in section 2.2.15 and the orientation image, O , is created. Although the process to derive the ridge orientation is obtained at block level, but the orientation image is defined at pixel level.
- 3) Ridge frequency estimation: The local ridge frequency of each image block is calculated using the algorithm described in section 2.2.16 and the ridge frequency image is created. As with the orientation image, the ridge frequency image is also defined at a pixel level. This value will be used in the filtering process and for segmenting the input image using a region mask.
- 4) Region mask estimation: Each input image block of size $w \times w$ must be classified into a recoverable or an unrecoverable block. This information will be used in the filtering process and feature extraction. A mask value of each block centered at pixel (i, j) is set to 255 (indicating as a recoverable block) if the ridge frequency of the corresponding block lies between $1/3$ and $1/25$; otherwise, the mask value is set to zero (indicating an unrecoverable block). This region mask is defined at the pixel level.

- 5) *Filtering*: A bank of Gabor filters is applied to the normalized fingerprint image. In this process, the input is a grayscale image, and the output is a signed 16-bit image. The purpose of this process is to perform low-pass filtering along the ridge orientation while performing band-pass filtering along the direction orthogonal to the ridge orientation [53].

2.2.15 Fingerprint orientation image [12]

An orientation image shows the local ridge orientation of each image block.

This image is obtained as follows

- 1) The normalized image, G , is divided into blocks of size $w \times w$.
- 2) The gradient $\partial_x(i, j)$ and $\partial_y(i, j)$ of $G(i, j)$ is calculated. To reduce the computational time, the simple Sobel operator is used in this thesis.
- 3) The local orientation of each block centered at pixel (i, j) is estimated using the following formula:

$$V_x(i, j) = \sum_{u=i-\frac{w}{2}}^{i+\frac{w}{2}} \sum_{v=j-\frac{w}{2}}^{j+\frac{w}{2}} 2\partial_x(u, v)\partial_y(u, v) \quad (2.34)$$

$$V_y(i, j) = \sum_{u=i-\frac{w}{2}}^{i+\frac{w}{2}} \sum_{v=j-\frac{w}{2}}^{j+\frac{w}{2}} (\partial_x^2(u, v) - \partial_y^2(u, v)) \quad (2.35)$$

$$\theta(i, j) = \frac{1}{2} \tan^{-1} \left(\frac{V_x(i, j)}{V_y(i, j)} \right) \quad (2.36)$$

where $\theta(i, j)$ represents the computed ridge orientation of each block. Note that the ridge angle of 225° and 45° are not different. For this reason, the value of $\theta(i, j)$ ranges from 0° to 179° .

- 4) The orientation field is modified using a low-pass filter to reduce the effect of noise and ridge turbidity. Before low-pass filtering, the orientation image is converted into a continuous vector field, which is defined as follows:

$$\Phi_x(i, j) = \cos(2\theta(i, j)) \quad (2.37)$$

$$\Phi_y(i, j) = \sin(2\theta(i, j)) \quad (2.38)$$

Low-pass filtering is applied to each x and y component of Φ_x and Φ_y . The filtered result, Φ'_x and Φ'_y are defined as follows:

$$\Phi'_x(i, j) = \sum_{u=-w_\phi/2}^{w_\phi/2} \sum_{v=-w_\phi/2}^{w_\phi/2} W(u, v) \Phi_x(i - uw, j - vw) \quad (2.39)$$

$$\Phi'_y(i, j) = \sum_{u=-w_\phi/2}^{w_\phi/2} \sum_{v=-w_\phi/2}^{w_\phi/2} W(u, v) \Phi_y(i - uw, j - vw) \quad (2.40)$$

where W is a two-dimensional low-pass filter of size $w_\phi \times w_\phi$, because the ridge orientation is obtained at a block level, this filter is applied to Φ'_x and Φ'_y at block level.

5) A filtered ridge orientation is obtained using the formula:

$$O(i, j) = \frac{1}{2} \tan^{-1} \left(\frac{\Phi'_y(i, j)}{\Phi'_x(i, j)} \right) \quad (2.41)$$

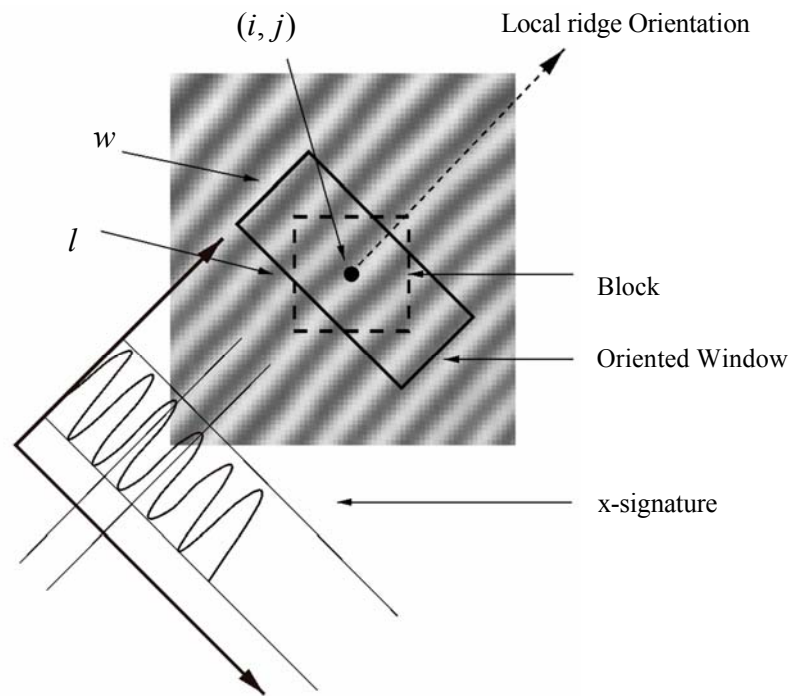


Figure 2.21. Oriented window and x-signature [12].

2.2.16 Fingerprint ridge frequency image [12]

The local ridge frequency is the reciprocal of the average inter-ridge distance in the respective block of the fingerprint image. In a grayscale fingerprint image, a local block, where no minutiae appear, contains a sinusoidal wave along the direction orthogonal to the fingerprint ridge. The steps to obtain the frequency image are as follows:

- 1) The orientation image, O , is divided into blocks of size $w \times w$.
- 2) An oriented window of size $l \times w$ is computed from each block. The orientation window centered at (i, j) is extracted for the x-signature $X[0].. X[l-1]$ using:

$$X[k] = \frac{1}{w} \sum_{d=0}^{w-1} G(u, v) \quad k = 0, 1, \dots, l-1 \quad (2.42)$$

where u is obtained from

$$u = i + \left(\frac{w}{2} - d \right) \cos \theta + \left(k - \frac{l}{2} \right) \sin \theta \quad (2.43)$$

and v is calculated from

$$v = j + \left(k - \frac{l}{2} \right) \cos \theta + \left(d - \frac{w}{2} \right) \sin \theta. \quad (2.44)$$

- 3) Let $T(i, j)$ denote the average of the number of pixels between the two consecutive valleys in the x-signature, then the local ridge frequency, $\Omega(i, j)$ can be obtained from

$$\Omega(i, j) = \frac{1}{T(i, j)} \quad (2.45)$$

As shown in Figure 2.22, any pixel in the oriented window of size $l \times w$ can be accessed using a location (u, v) . The parameters u and v can be obtained using

$$u = i - a + b \quad (2.46)$$

$$v = j - (m + g) \quad (2.47)$$

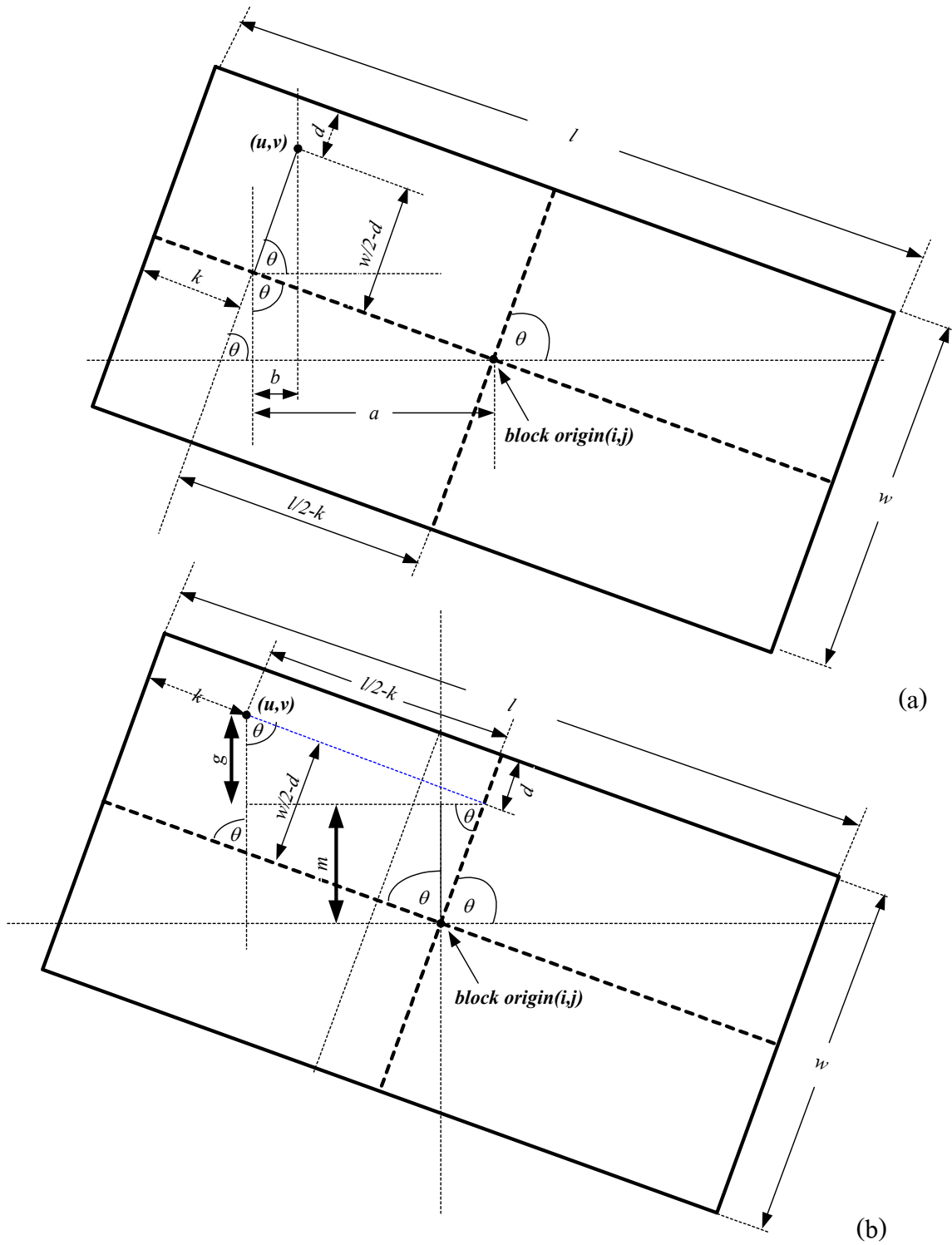


Figure 2.22. The derivation of the coordinate (u, v) from the oriented window of size $l \times w$: (a) the derivation of value u ; (b) the derivation of value v .

From Figure 2.22(a), the parameters a and b of equation 2.46 are obtained using

$$a = \left(\frac{l}{2} - k \right) \sin \theta \quad (2.48)$$

$$b = \left(\frac{w}{2} - d \right) \cos \theta \quad (2.49)$$

From Figure 2.22(b), the parameters m and g of equation 2.47 are obtained using

$$m = \left(\frac{w}{2} - d \right) \sin \theta \quad (2.50)$$

$$g = \left(\frac{l}{2} - k \right) \cos \theta \quad (2.51)$$

The substitution of equations 2.48 and 2.49 into equation 2.46 yields equation 2.43. The substitution of equations 2.50 and 2.51 into equation 2.47 yields equation 2.44.

2.2.17 Fingerprint filtering using Gabor filter [12]

The Gabor filter is a directional filter where users can select the orientation of filtration. When used with fingerprints, the two dimensional Gabor filter performs low-pass filtering along the ridge orientation while performing band-pass filtering along the direction orthogonal to the ridge orientation [53]. To apply Gabor filters to a fingerprint image, three parameters are required:

- 1) the frequency of the sinusoidal wave form,
- 2) the filtration orientation, and
- 3) the standard deviation of the Gaussian envelope along x and y axes.

Let G be a normalized fingerprint image, h be an even-symmetric Gabor filter, R be the region mask image, O be the orientation image, and F be the frequency image. The output of the Gabor filter, E , is derived using the formula:

$$E(i, j) = \begin{cases} 255 & \text{if } R(i, j) = 0 \\ \sum_{u=-w_g/2}^{w_g/2} \sum_{v=-w_g/2}^{w_g/2} h(u, v: O(i, j), F(i, j)) G(i-u, j-v) & \text{otherwise} \end{cases} \quad (2.52)$$

where w_g is the size of the Gabor filter.

An even-symmetric Gabor filter has the general form

$$h(x, y : \phi, f) = \exp \left\{ -\frac{1}{2} \left[\frac{x_\phi^2}{\delta_x^2} + \frac{y_\phi^2}{\delta_y^2} \right] \right\} \cos(2\pi f x_\phi), \quad (2.53)$$

$$x_\phi = x \cos \phi + y \sin \phi, \quad (2.54)$$

$$y_\phi = -x \sin \phi + y \cos \phi, \quad (2.55)$$

where ϕ is the orientation of the Gabor filter, f is the specified frequency, and δ_x and δ_y are the space constants of the Gaussian envelop along the x_ϕ and y_ϕ axes, respectively [53]. Choosing the values δ_x and δ_y is a trade-off, large values yield more noise reduction but allow the possibility of spurious ridges and valleys occurring.

2.2.18 Fingerprint feature extraction

The use of Gabor filter enhances the clarity of the ridge and valley structures. However, the enhanced image must be converted into the binary image using a basic global threshold which is defined as

$$b(x, y) = \begin{cases} 0 & \text{if } p(x, y) > T \\ 255 & \text{otherwise} \end{cases} \quad (2.56)$$

where $p(x, y)$ is the image after the Gabor filter has been applied and $b(x, y)$ is the resulting binary image. A thinning operation must be applied to $b(x, y)$ to obtain the skeleton of the ridge structure. The output of the thinning process is converted into m-connectivity [14], as shown in Figure 2.23.

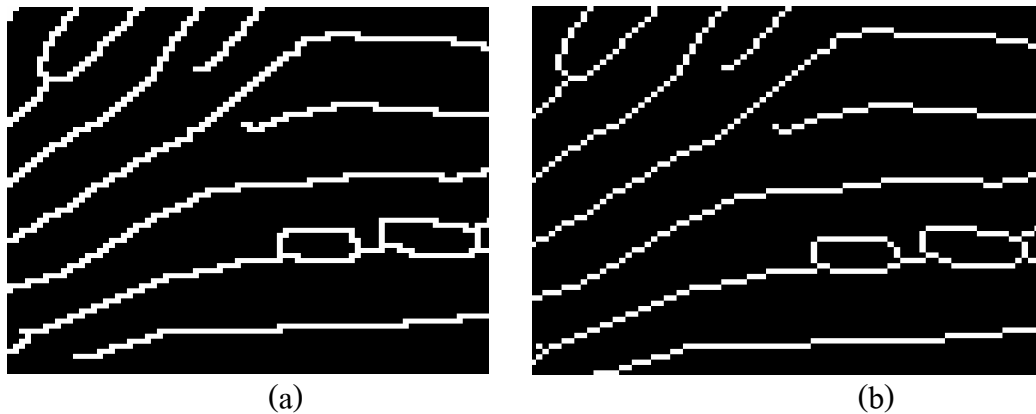


Figure 2.23. a) fingerprint ridges after thinning operation,
b) after converted to m-connectivity.

Each minutia comprises three parameters: type, direction, and location. To obtain the location of each minutia, a feature extraction window of size 3×3 , shown in Figure 2.24, is moved throughout the thinned image. Let M is the point of interest which is white pixel, and N_1, \dots, N_8 are its surrounding pixels. M is considered a location of termination point if number of its surrounding white pixels equals to 1. On the other hand, if number of surrounding white pixels is equal to 3, then M is the location of a bifurcation point.

N_1	N_2	N_3
N_8	M	N_4
N_7	N_6	N_5

Figure 2.24. Minutiae extraction window

To obtain the direction of a termination minutia, *line following* is performed on the ridge line starting from the termination point (x_t, y_t) until \mathcal{E} pixels have been visited. Let (x_s, y_s) be the stop point of the line following operation, the termination minutiae direction, denoted θ_t , can be calculated using

$$\theta_t = \tan^{-1}\left(\frac{y_s - y_t}{x_s - x_t}\right). \quad (2.57)$$

The process of deriving the direction of bifurcation is more complex than that of the termination minutia. If we have a look at a bifurcation point, there are 3 ridge lines directed outward 3 different directions. The problem arises because we need to find which direction is appropriate to be used as the bifurcation direction. Figure 2.25(a) shows that direction D_1 is better than the other two directions because A_1 is the only acute angle while the other two angles, A_2 and A_3 , are the obtuse angles. To find a bifurcation direction, the algorithm works as follows:

- 1) Ridge line following is performed outward from the bifurcation point for η pixels in each direction. Let P_1 , P_2 and P_3 stand for the stop points of each direction of ridge line following and $L_{\{i,j\}}$ stand for the line started from point P_i to P_j , then the shortest path among these 3 points is obtained using

$$L_{min} = \min(L_{\{1,2\}}, L_{\{2,3\}}, L_{\{3,1\}}) \quad (2.58)$$

- 2) Find the destination point, denoted P_d , using the following condition

$$P_d = \begin{cases} P_1 & \text{if } (L_{\{2,3\}} < L_{\{1,2\}}) \text{ and } (L_{\{2,3\}} < L_{\{3,1\}}) \\ P_2 & \text{if } (L_{\{3,1\}} < L_{\{1,2\}}) \text{ and } (L_{\{3,1\}} < L_{\{2,3\}}) \\ P_3 & \text{if } (L_{\{1,2\}} < L_{\{3,1\}}) \text{ and } (L_{\{1,2\}} < L_{\{2,3\}}) \end{cases} \quad (2.59)$$

- 3) Let (x_d, y_d) and (x_b, y_b) be the coordinates of the destination point and bifurcation point respectively, the bifurcation direction, denoted θ_b , is calculated from

$$\theta_b = \tan^{-1} \left(\frac{y_b - y_d}{x_d - x_b} \right) \quad (2.60)$$

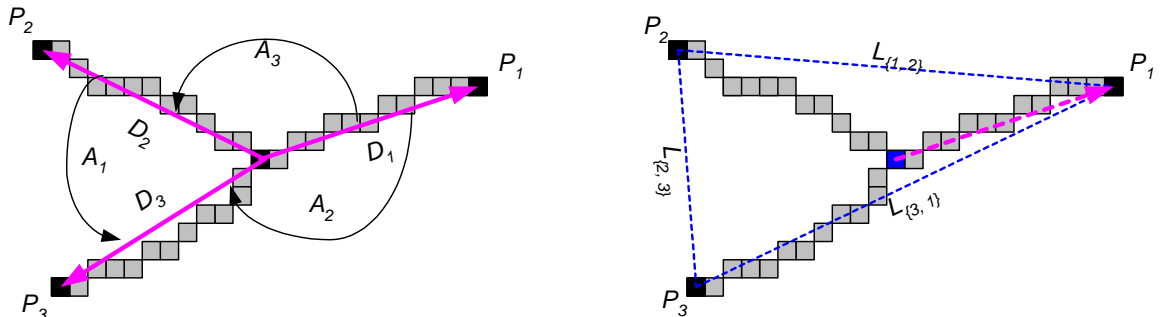


Figure 2.25. (a) three ridge lines directed outward from the bifurcation point; (b) the most suitable direction to be used as a bifurcation direction.

2.2.19 Fingerprint post-processing

Although good preprocessing algorithms are used in the system, spurious minutiae may occur if the input fingerprint has low quality or too much noise. Post processing is required to remove as many spurious minutiae as possible, while preserving real minutiae. Figure 2.26 shows the steps of the post processing, the steps in sections 2.2.19.1-2.2.19.2 deal with the fingerprint image pixel directly while the steps in section 2.2.19.2-2.2.19.4 deal with the minutiae set already extracted from the fingerprint image.

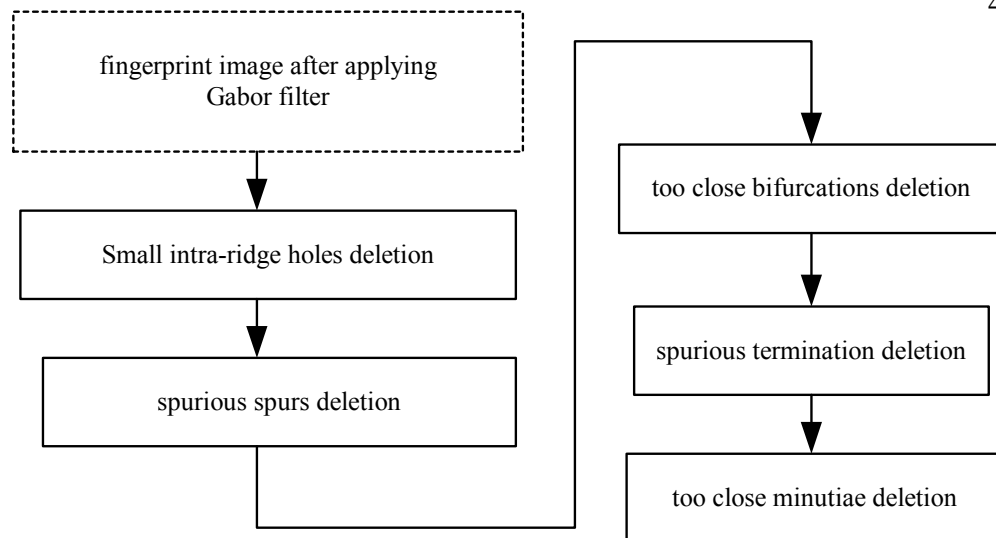


Figure 2.26. Flowchart of the fingerprint minutiae post-processing.

2.2.19.1 Small intra-ridge holes deletion

In the preprocessing, the use of a Gabor filter effectively reduces the noise, however, in some cases, small black residues may remain within the white ridge line, as shown in Figure 2.27. When thinning is applied to these ridges, an island may occur which leads to spurious minutiae. To reduce this effect, all pixels in the black contours with an area lower than a threshold value will be changed to white, this process must be applied to the fingerprint image before the thinning process.

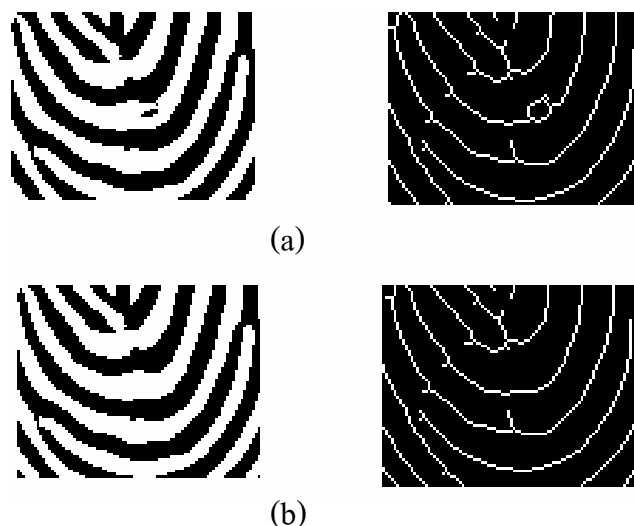


Figure 2.27. The comparison of the fingerprint ridge before and after small black hole deletion (a) the ridge shape before small black hole deletion (left) and its thinned version (right); (b) the ridge shape after small black holes are deleted (left) and its thinned version (right).

2.2.19.2 Spurious spurs deletion

When the thinning operation is performed upon a ridge line that is not of uniform thickness, spurs may occur, this leads to two spurious minutiae: termination and bifurcation. A morphological pruning operation [14] can be used to eliminate the spurs but with the side effect that some pixels at the end of the ridge line might be deleted. This means that the accuracy of the system might be affected by the translation of the termination minutia. In this thesis, spurious spurs are removed using the following steps:

- 1) Let M be the minutiae set that contains the set of bifurcation B and termination T .

$$M = \{B, T\}$$

$$B = \{B_1, B_2, \dots, B_n\}$$

$$T = \{T_1, T_2, \dots, T_k\}$$

- 2) For each bifurcation B_p , if all three neighborhood ridge pixels are found around the bifurcation point, continue with step 3, otherwise, check the next bifurcation point.
- 3) Ridge line following is performed outward from the bifurcation point for ϖ pixels in every direction. The value ϖ is derived from the local inter-ridge distance. If a ridge break is found before ridge line following is completed, all traced ridge pixels in that direction are deleted.
- 4) Steps 2 and 3 are repeated until all bifurcation points in M have been checked.
- 5) All minutiae in M are deleted before minutiae extraction is reapplied to the spur deleted image. The new minutiae in M are ready to be processed in section 2.2.19.3.

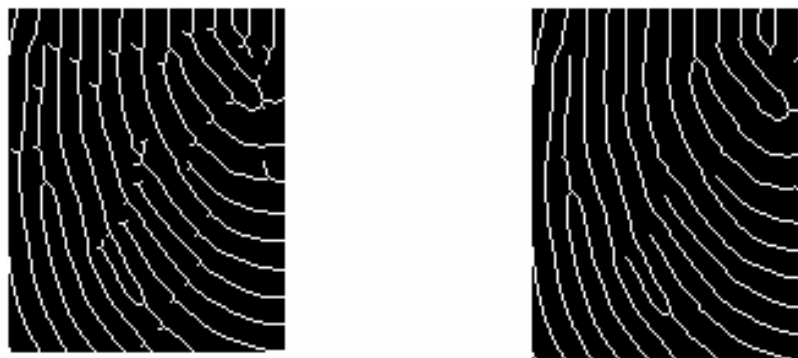


Figure 2.28. Fingerprint image before spur deletion (left) and after spur deletion (right).

2.2.19.3 Spurious bifurcation deletion

The alignment of ridge lines in some cases leads to a cluster of very close bifurcations. Figure 2.29 shows a case where two additional bifurcation points are detected near a real bifurcation point. Deleting all close minutiae would result in the real minutia vanishing. The objective is to delete all, and only, the unnecessary minutiae. Spurious bifurcation deletion is carried out as follows:

- 1) Let B is the set of bifurcations where $B = \{B_1, B_2, \dots, B_n\}$. B_i is the bifurcation point of interest selected from B .
- 2) All bifurcations around B_i that are less than 3 pixels from B_i are deleted from B .
- 3) The next bifurcation is selected from B , step 2 is repeated until all bifurcation points in B are checked.

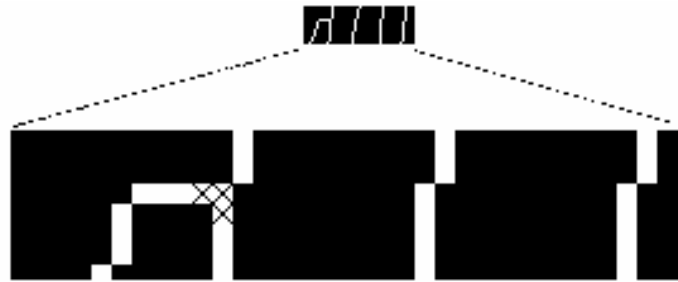


Figure 2.29. Example of two spurious bifurcations around the real bifurcation.

2.2.19.4 Spurious termination deletion

Spurious termination points arise at the joint between the fingerprint area and the background of the fingerprint image, as shown in Figure 2.30(b). Since these terminations are not real ones, but occur from the limitations of fingerprint image acquisition of each sensor, spurious terminations are removed using the algorithm modified from Farina *et al.* [13]. The process is carried out as follows:

- 1) The local ridge distance, denoted ξ , around the termination point of interest is measured.
- 2) Ridge line following is performed from the termination point for 0.5ξ .
- 3) Line following is continued for 1.5ξ , as shown in Figure 2.30(a) then neighboring ridges are searched orthogonally to the termination direction, if 2 sandwich ridges are found, continue with step 4, otherwise, that termination point is deleted.
- 4) Each detected ridge from step 3 is traced for 3ξ in the same direction as the termination of interest. If a ridge break is reached before tracing is completed, that termination is invalidated.

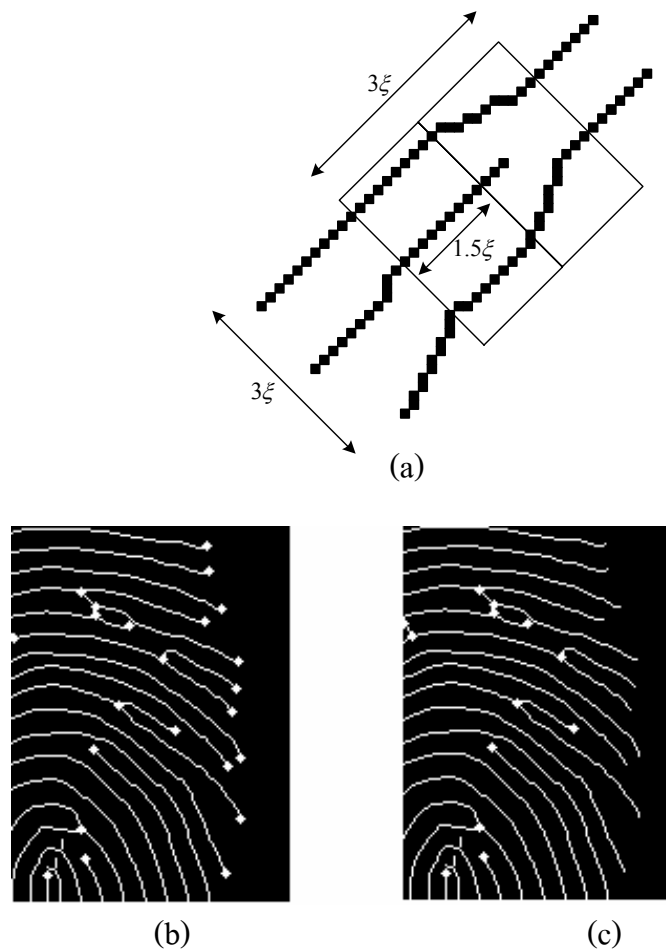


Figure 2.30. (a) direction of line following to detect spurious termination point; (b) before removing spurious termination points; (c) after spurious termination points have been removed.

2.2.20 Decision Fusion

Decision fusion results in an improvement in matching accuracy when the top-view matcher gives the wrong result while the bottom-view matcher gives the right one, or vice versa. To combine the 2 biometric features, the decision fusion is done at a confidence level. Figure 2.31 shows the block diagram of our implemented system, the decision is not performed at the two matchers. This means that the top-view and bottom-view matchers only send the matching scores to the fusion module and the decision is made in this module.

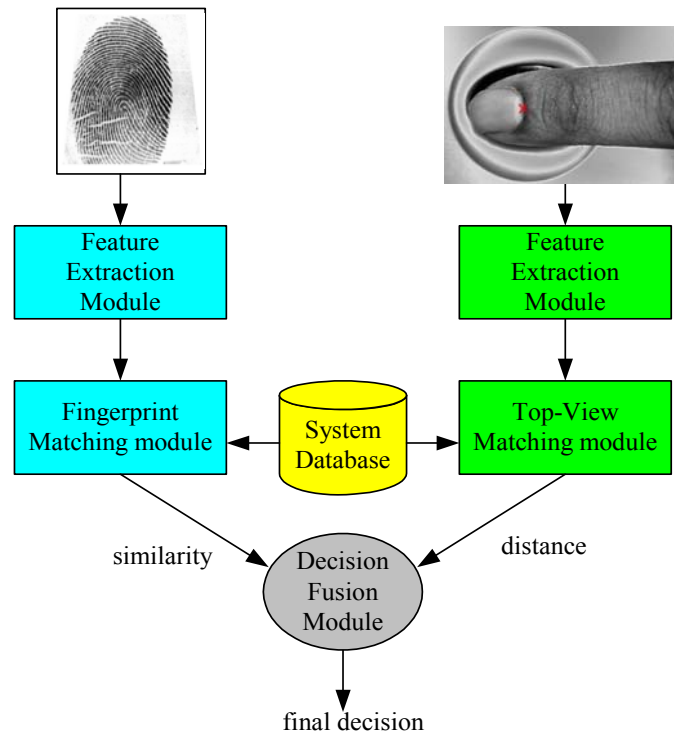


Figure 2.31 Decision fusion between the Top-View and fingerprint matcher

Suppose that the input top-view and bottom-view images are to be classified into imposter class, ω_1 , or genuine class, ω_2 . Suppose that x_1 is the distance derived from the top-view matcher and x_2 is the matching score from the bottom-view matcher.

The design goal of the biometric system depends on its application. In high security access applications, a low value of false acceptance rate (FAR) is required while forensic applications need a low value of false rejection rate (FRR). Our design goal is to minimize both FAR) and FRR. However, it is difficult to get an extremely low value for these two rates at the same time. So the approach that most designers choose to implement the system is to minimize the equal error rate (EER: the point where the system has FAR equals to FRR) [1].

If the top-view feature alone is used to verify a person, the FAR and FRR can be computed using:

$$FAR_{top} = \int_{-\infty}^{th_{top}} p(x_1 | \omega_1) dx_1 \quad (2.61)$$

$$FRR_{top} = \int_{th_{top}}^{\infty} p(x_1 | \omega_2) dx_1 \quad (2.62)$$

In contrast, if the bottom-view feature alone is used to verify a person, the FAR and FRR values can be calculated using:

$$FAR_{b_{tm}} = \int_{th_{b_{tm}}}^{\infty} p(x_2 | \omega_1) dx_2 \quad (2.63)$$

$$FRR_{b_{tm}} = \int_{-\infty}^{th_{b_{tm}}} p(x_2 | \omega_2) dx_2 \quad (2.64)$$

The values $th_{b_{tm}}$ and th_{top} are threshold values set at the top-view and bottom-view matcher, respectively. Equations 2.61-2.64 state that the threshold values of both top-view and bottom-view matchers have an effect on the FAR and FRR. When the threshold value is adjusted to increase the FAR, the FRR is decreased, and vice versa.

The final decision of the fusion module in Figure 2.31 is made using:

$$\text{decision} = \begin{cases} \omega_1 & \text{if } (L \leq \beta) \\ \omega_2 & \text{otherwise.} \end{cases} \quad (2.65)$$

The likelihood ratio, L [5], [54], can be calculated using:

$$L = \frac{p(x_1, x_2 | \omega_2)}{p(x_1, x_2 | \omega_1)}. \quad (2.66)$$

If L is high, then the input data is more likely to come from the genuine class. The input is decided to come from the genuine class if $L > \beta$, where β is an empirically determined threshold value. The joint probability in equation 2.66 is difficult to obtain directly from training data, but by assuming that each x_i is statistically independent, the joint probability density function can be estimated using: [54]

$$p(x_1, x_2, \dots, x_n | \omega_j) = \prod_{i=1}^n p(x_i | \omega_j) \quad (2.67)$$

2.3 Materials and Equipments

1. A 2.4 GHz Pentium4 Microcomputer, 2 GBytes RAM.
2. C++ Compiler.
3. OpenCV library Version 1.0.
4. Creative VF0080 CCD camera:
 - 640×480 CMOS sensor.
 - Video frame rate at up to 30 frames per second.
5. Digital Persona UareU4000B fingerprint sensor:
 - Optical based sensor.
 - 8-bit grayscale image output.
 - Resolution 512 dpi.
 - Scan capture area 14.6×12.1 mm.

2.4 Summary

This chapter discussed the theoretical background of the image processing, followed by the methods used to implement the system hardware and software. Details of the top-view finger image processing were elucidated. The fingerprint processing algorithms were also explained in this chapter. The next chapter reveals experimental results of the implemented system.

CHAPTER 3

EXPERIMENTAL RESULTS

3.1 Top-View finger image acquisition

The top-view finger image is obtained from a color CCD camera, with examples shown in figure 3.1. The top-view color image suffers interference from the red light illumination from the optical fingerprint sensor. This interference is omitted by selecting only the blue component from the obtained color image, which is used for creating an input grayscale top-view finger image in the top-view preprocessing steps.

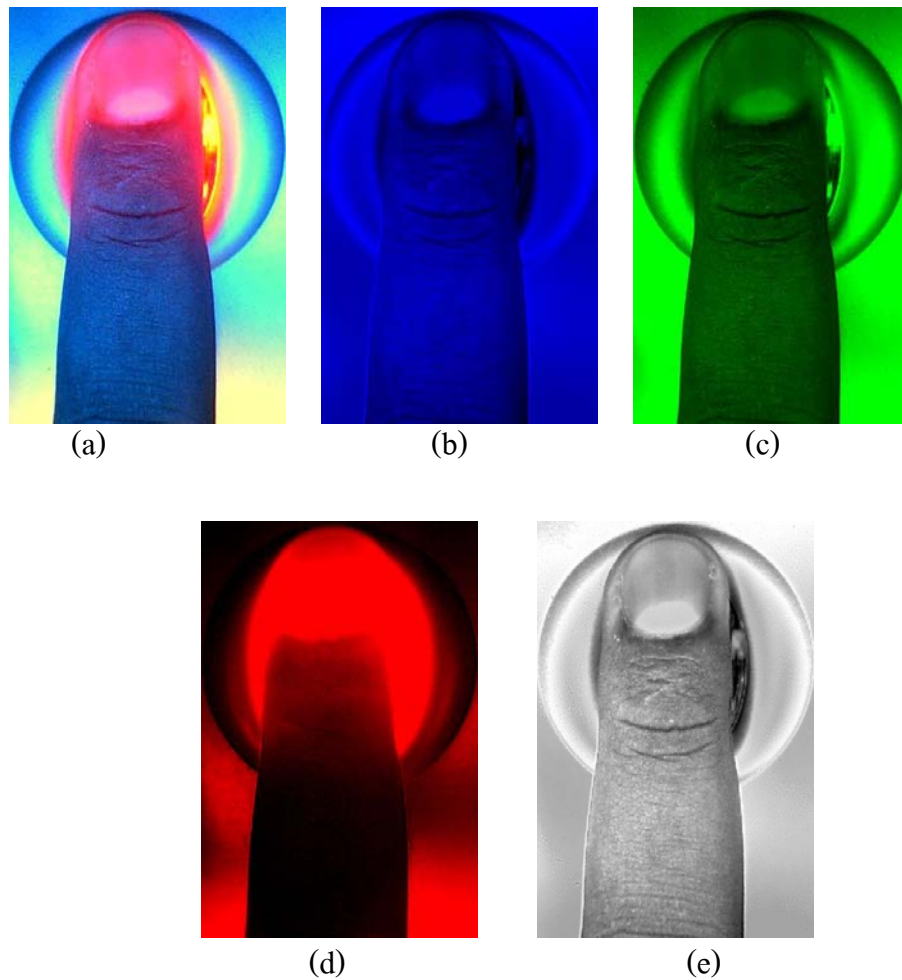


Figure 3.1 Top-View finger image obtained from CCD camera; a) color image; b) blue component; c) green component; d) red component; and e) grayscale top-view finger image.

3.2 Top-view finger image processing and feature extraction

The resolution of the image derived from a Creative VF0080 CCD camera is 640×480 pixels. The distance between the CCD camera and the fingerprint sensor is adjusted to acquire maximum detail of the finger, as shown in figure 3.2(a). The image is clockwise rotated by 90 degrees and then cropped to 326×480 pixels (see figure 3.2(b)). The obtained image is now named T_G as described in section 2.2.3.

As a trial, the HTMM transform was applied to T_G to detect the inclination of the finger. The OpenCV's *cvHoughLines2* function was used for this task. The Hough transform gives a good result, but suffered from taking a long computation time compared to the line fitting algorithm. For this reason, the line fitting algorithm has been used throughout the rest of our experiments.

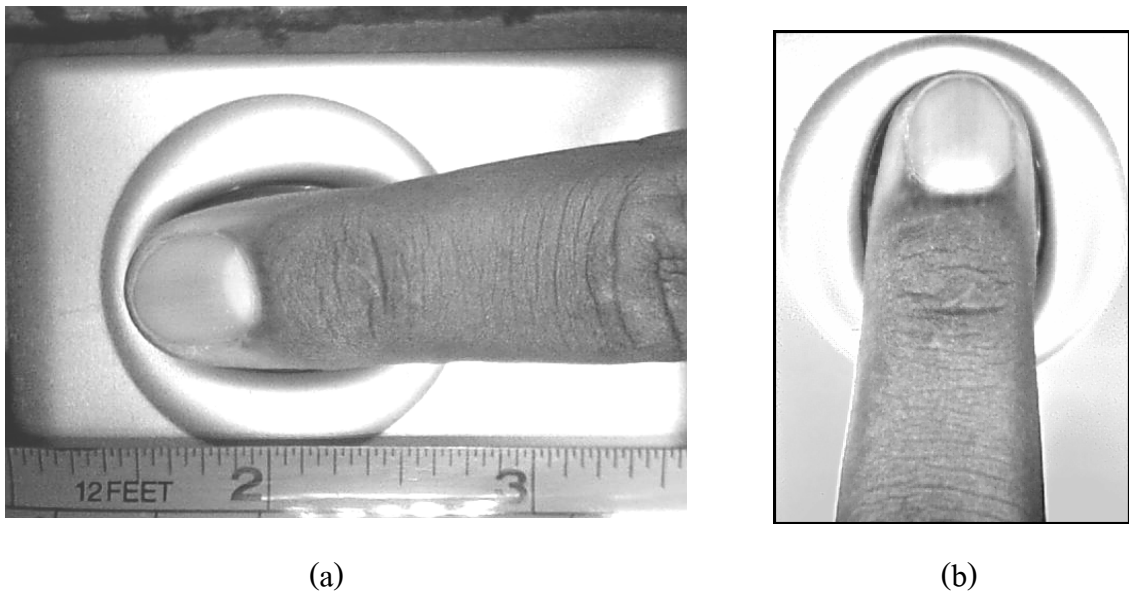


Figure 3.2. Top-view finger image acquisition; a) original image from the CCD camera; b) cropped and rotated image.

3.2.1 The parameters for top-view finger image feature extraction

As described in section 2.2.8, the NailCode is acquired by performing square tessellation of the top-view finger image. To obtain the best recognition accuracy using this feature, appropriate values of the three parameters of equation 2.27 must be evaluated:

- 1) the width of each square cell w .
- 2) the number of tessellated rows H .
- 3) the number of tessellated columns V .

To find appropriate values of these three parameters, 800 top-view finger images from 100 different fingers, with eight images per individual, were collected. One image from each individual was employed for enrolment. The remaining images were used to test the recognition accuracy. This means that there were 700 test images in the database. The reference point of each top-view finger image was manually defined.

First, several values of w were tried. Each top-view finger image was tessellated using H and V of size 10 and 14, respectively. Table 3.1 indicate that the best value of w is 15.

Table 3.1. The system accuracy according to each value of window size (w).

Window Size(pixel)	correct	incorrect	accuracy
9	527	173	75.29
11	629	71	89.86
13	662	38	94.57
15	669	31	95.57
17	668	32	95.43

Next, the appropriate value for V was sought. Table 3.2 shows that the best value of V is 15. Table 3.3 reveals that the best value of H is 10.

Table 3.2. The system accuracy according to each value of V .

Number of window in vertical axis (V)	Correct	Incorrect	Accuracy
11	653	47	93.29
12	668	32	95.43
13	668	32	95.43
14	669	31	95.57
15	671	29	95.86
16	670	30	95.71
17	667	33	95.29
18	666	34	95.14

Table 3.3. The system accuracy according to each value of H .

Number of window in horizontal axis(H)	Correct	Incorrect	Accuracy
6	657	43	93.86
7	670	30	95.71
8	670	30	95.71
9	668	32	95.43
10	671	29	95.86
11	665	35	95.00
12	662	38	94.57
13	655	45	93.57
14	653	47	93.29

3.2.2 Fine-tuning to get the best parameter of the top-view finger image preprocessing and feature extraction

As described in section 2.2.3, an adaptive threshold is applied to the grayscale top-view finger image to obtain its two-color version. Different window sizes for adaptive threshold give different results for the binarized image, as shown in figure 3.3. However, it is difficult to select an appropriate value for this parameter by

observing the result of a threshold with the naked eye. Instead we measure the *class separation (CS)* statistic [5] between the imposter and genuine class for various sizes of the adaptive threshold window. The window size that gives maximum value of the class separation is accepted as the best parameter.

As previously described in section 2.2.20, the symbols ω_1 and ω_2 represent the imposter and a genuine classes, respectively. The *CS* statistic measures how well the two classes are separated with respect to the feature vector, X^d , in a d -dimensional space, R_d , is defined as [5]

$$CS(X^d) = \int_{R^d} |p(X^d | \omega_1) - p(X^d | \omega_2)| dx \quad (3.1)$$

In order to get the class separation, the genuine and imposter distribution of a top-view matching score must be obtained. To attain both distributions, images of 200 fingers are randomly selected from the database of section 3.4. At total of 278,600 (200*199*7) matches were evaluated to estimate the imposter distribution, and 1,400 (200*7) matches were examined to approximate the genuine distribution.

To find the best value of the adaptive threshold's window size, the class separation with the window size set to 13, 15, 17, 19, 21, 23, and 25 were measured. As shown in table 3.4, the adaptive threshold with the window size of 17 gives the best class separation because it gives the largest value of *CS*.

Table 3.4 The class separation of a system for different sizes of an adaptive threshold's window.

Window Size	CS
11	1.109
13	1.404
15	1.412
17	1.474
19	1.254
21	1.157
23	1.004
25	0.801

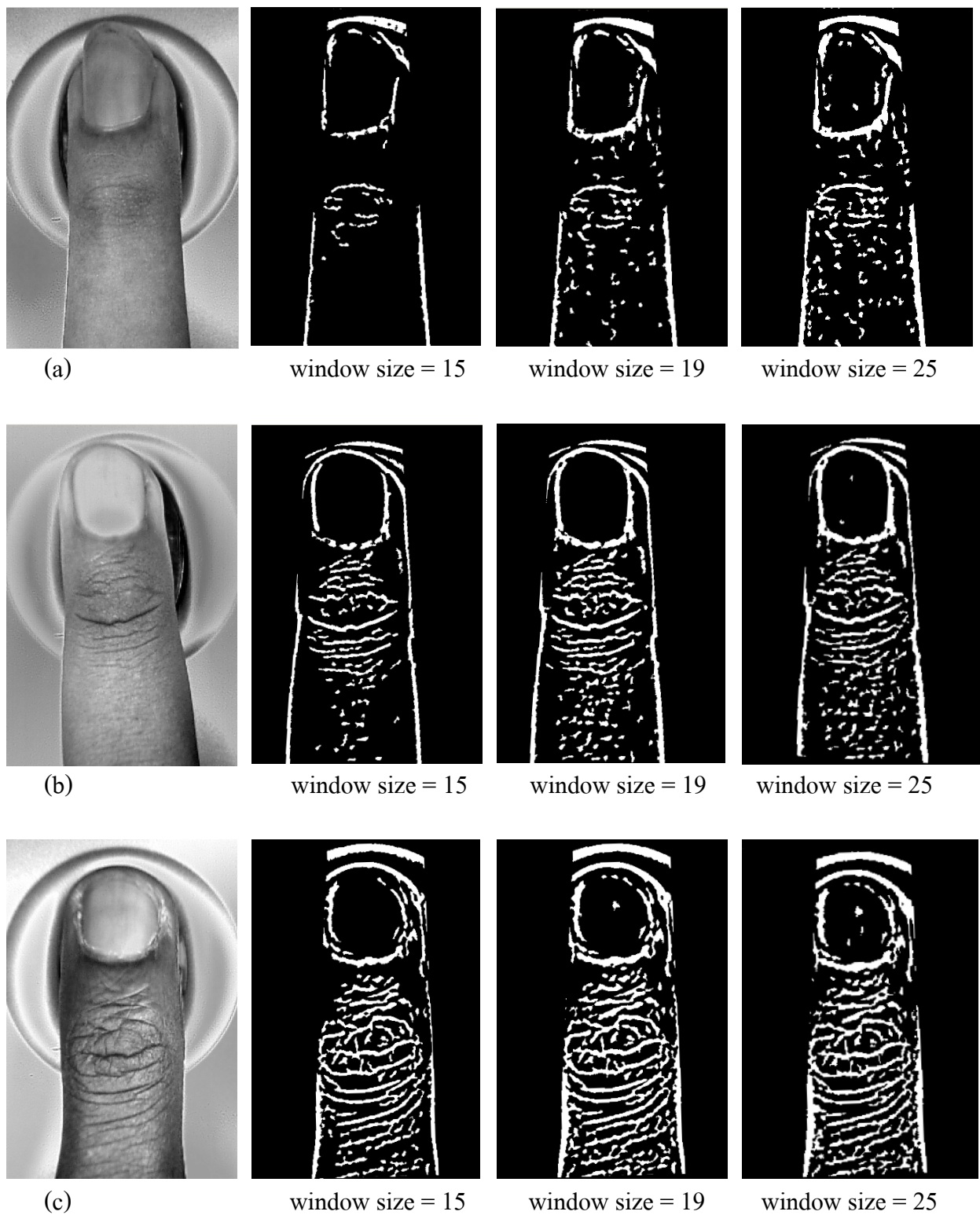


Figure 3.3. The binarized version of the top-view finger images due to different window size of an adaptive threshold; (a) finger with low detail of skin wrinkle; (b) finger of medium detail of skin wrinkle; (c) finger with high detail of skin wrinkle.

3.2.3 Top-view reference point location error and system accuracy

If the derived reference point location is incorrect, then the identification accuracy will be affected. Table 3.5 shows that an increasing reference point location error significantly reduces identification accuracy. The table was obtained by trying to tessellate the filtered image with the translated version of the reference point. This was done by moving the reference point with the distance, δ , in eight directions.

Table 3.5. Identification accuracy of person recognition with different reference-point location errors.

Reference point error δ (in pixels)	Identification accuracy (percent)
0	96.57
5	89.10
8	78.00
10	65.90
12	51.90

To test the effect of a reference point location error, 800 top-view finger images that were used to test the system in section 3.2.1 were utilized, and a proposed automatic reference point detection algorithm was tested with these images. Table 3.6 shows that only 6 test images were rejected by the proposed reference point detection algorithm. When a manually defined reference point was used, the identification accuracy was 96.57 percent, which dropped to 73.78 percent when only a single automatic reference point, R_0 , was used. Table 3.6 shows that additional points R_1 - R_8 , improved the accuracy of the system dramatically, especially when δ was 9 or 10 pixels. In those cases, the identification accuracy was 93.80 percent.

Table 3.6. Identification accuracy with different reference point markings.

Reference point marking method	Reject	Accuracy
Manual	0	96.57 %
Automatic using R_0 alone	6	73.78 %
Automatic using R_0 - R_8 ($\delta=5$)	6	87.03 %
Automatic using R_0 - R_8 ($\delta=8$)	6	92.51 %
Automatic using R_0 - R_8 ($\delta=9$)	6	93.80 %
Automatic using R_0 - R_8 ($\delta=10$)	6	93.80 %

3.3 Top-view finger image and time variances

It is well known that the wrinkles of the skin on the finger will increase over time. Images of ten different fingers captured 990 days after the day of their enrolment (see figure 3.4) have been tested to investigate the effect of time variances to the result of a NailCode matching. All of the later versions of the top-view finger images can still correctly be identified.



Figure 3.4. Finger image captures at different times: (a) the initial image; (b) the same finger captured after 990 days had passed.

3.4 Test database of the top-view finger images and fingerprints

The test database was collected from 800 different fingers. A snapshot of a finger comprises both top and bottom-views. Eight snapshots were collected for each finger: one was added to the database while the other seven were used to test system performance. This means that the test database comprises 6,400 (800×8) snapshots. This database will be used to test the system performance which will be described in sections 3.7-3.8.

3.5 Fingerprint preprocessing

The resolution of the 8-bit grayscale fingerprint derived from the sensor is 466×510 pixels. The image is first normalized and the orientation field is estimated. A low-pass filter is applied to the orientation field obtained to reduce the effect of noise. Various sizes of low-pass filter has been tested, as shown in figures 3.5 and 3.6. When applied with a good quality fingerprint, filters of size 3×3 and 5×5 give unnoticeably different results, however, a filter of size 5×5 gives better smooth orientation field. A filter of size 7×7 has been tried; however, it smoothes too much so the acquired orientation around the tented arch tends to lose detail. For this reason we chose to use a low-pass filter of size 5×5 for the rest of our experiments.

After the fingerprint orientation is acquired, the x-signature of each image block is estimated to find the local ridge frequency. An area of the fingerprint where no minutiae appear gives a nearly sinusoidal shape wave of x-signature, however an area with no fingerprint ridge gives a rectangular shape wave, as shown in figure 3.7. It is noted that the sign of the image intensity has both positive and negative values because the format of the normalized image is signed 16-bit. The average number of pixels between the two consecutive valleys in the x-signature is used for estimating the local ridge frequency. The minutiae mask image, as shown in figure 3.8, is created from the ridge frequency image. The image area where the frequency ranges from $1/3$ to $1/25$ gives the minutiae mask value of 255 (indicating a usable area), otherwise, the minutiae mask of that area will be set to 0 (indicating a non-usable area).

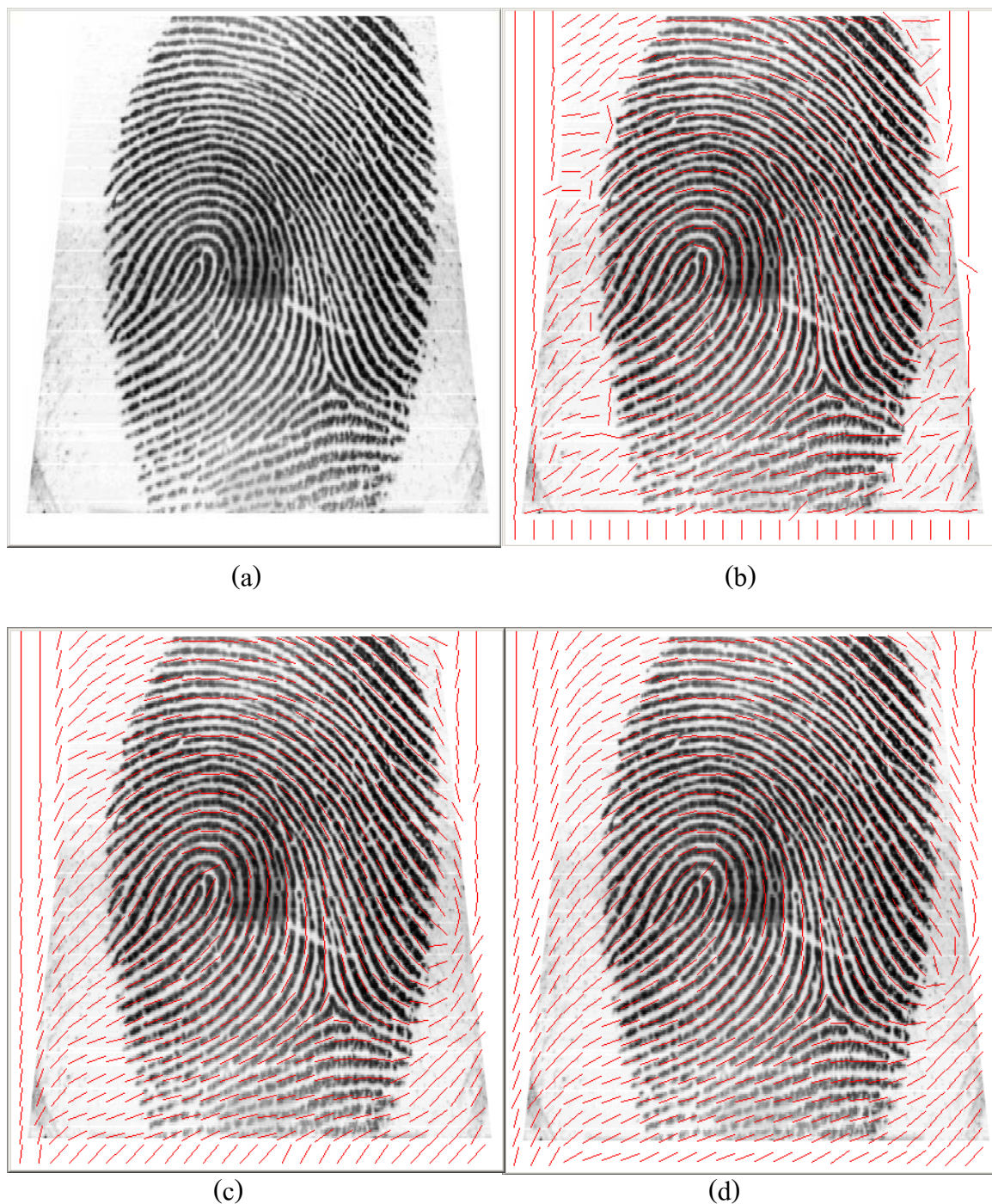


Figure 3.5. Results of the fingerprint orientation field due to different w_ϕ of the low pass filter: (a) image of good quality; (b) orientation field before smoothing; (c) orientation field after smoothing with a low-pass filter of 3×3 ; (d) orientation field after smoothing with a low-pass filter of 5×5 .

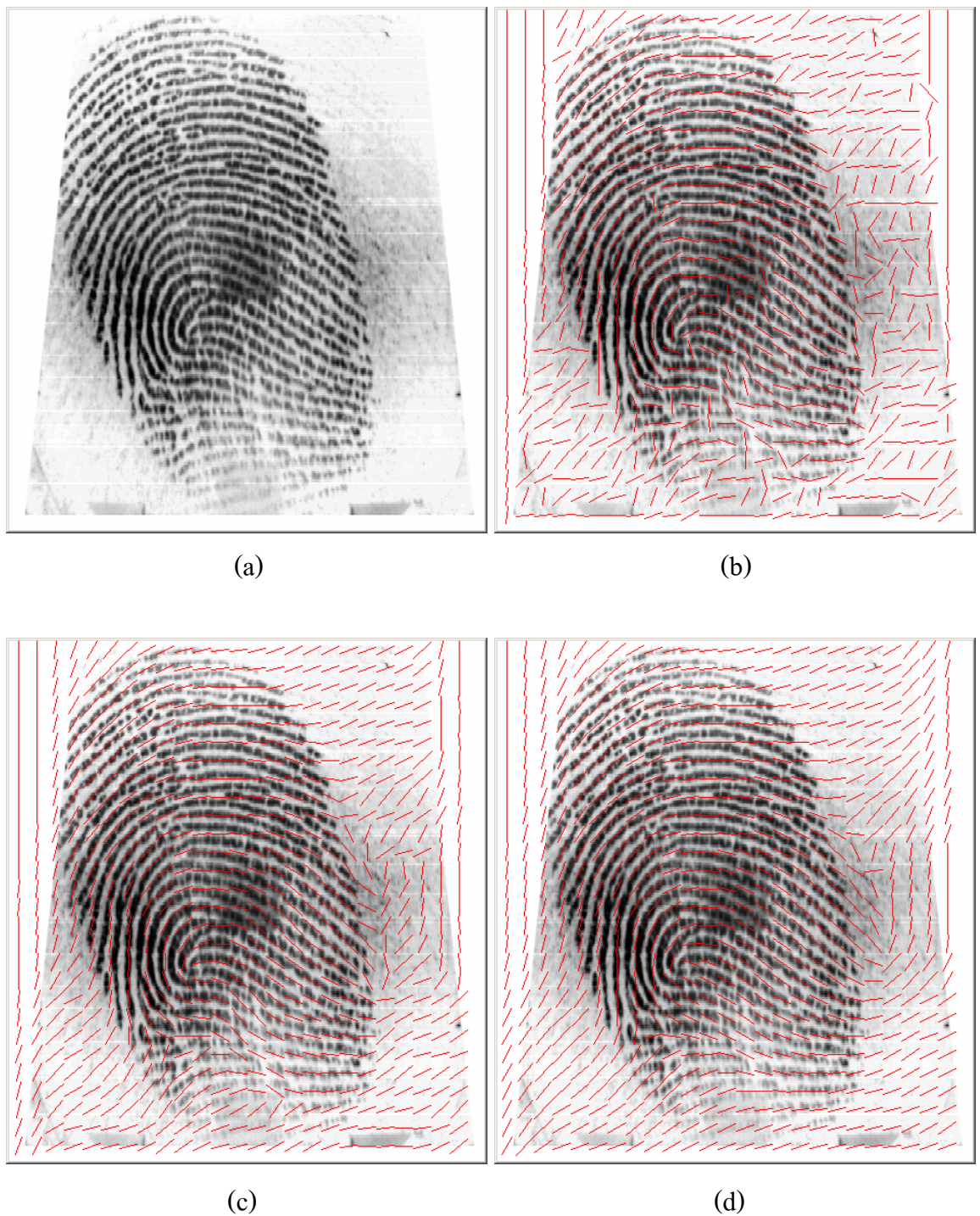


Figure 3.6. Results of the fingerprint orientation field due to different w_ϕ of the low pass filter: (a) image of poor quality; (b) orientation field before smoothing; (c) orientation field after smoothing with a low-pass filter of 3×3 ; (d) orientation field after smoothing with a low-pass filter of 5×5 .

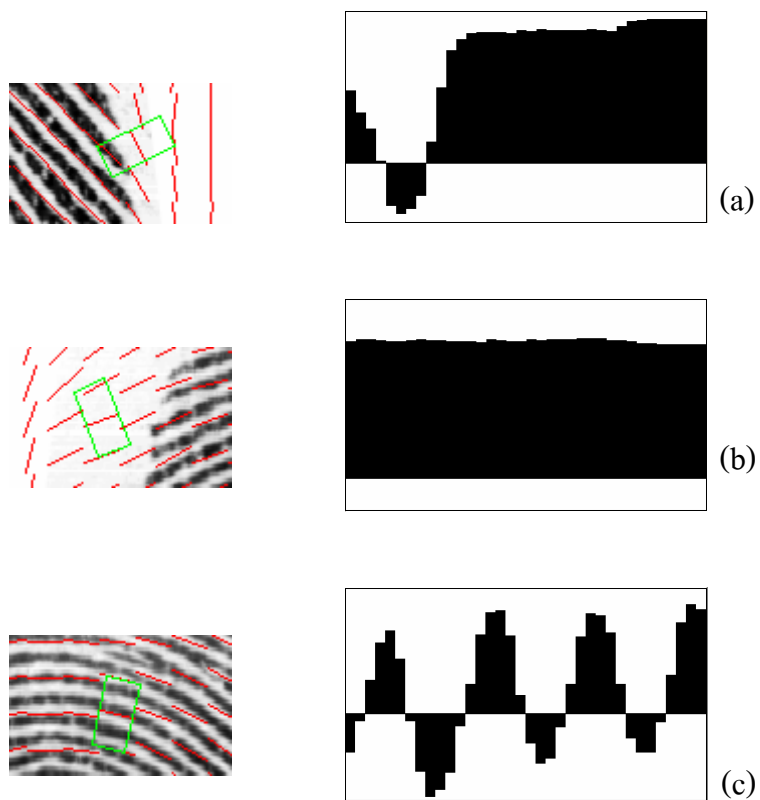


Figure 3.7. X-signature due to different regions of fingerprint: (a) where a ridge ending was found; (b) where no ridge line appears; (c) where no minutia appear.



Figure 3.8. The fingerprint image (left) and the obtained minutiae mask image (right).

When the fingerprint is manipulated by a Gabor filter, the parameters δ_x and δ_y must be defined. As shown in figures 3.9-3.11, larger values for these parameters give better noise reduction but with increased possibility of the occurrence of spurious ridges and valleys. After testing with number of fingerprint images, these values are empirically set to 20.

After the Gabor filter is applied to the fingerprint, a thresholding process is needed to convert the obtained 16-bit signed image to the binary version. Figures 3.12-3.14 demonstrate the results of different threshold values on three different qualities of fingerprint images. The threshold value is empirically set to -400.

In the preprocessing steps, the Gabor filter requires much computation time because it re-computes the trigonometric functions ($\cos \phi$ and $\sin \phi$) for every pixel of the filtering process, as described in section 2.2.17. The value ϕ is the obtained ridge orientation of the respective processing area, however, since ϕ is defined at blockwise level, it is not necessary to re-calculate these trigonometric functions every time when processing the image at the pixel level. By calculating these trigonometric functions only once for each local image block, the required computation time is substantially reduced. As shown in table 3.7, the improved version of the Gabor filter runs approximately 44 times faster than the original one.

Table 3.7. Result of computation time required by each type of Gabor filter.

Filter type	Average computation time
Conventional Gabor filter	2.726 seconds
Improved Gabor filter	0.062 seconds

3.6 Fingerprint post-processing

After preprocessing and feature extraction, post-processing steps are required. Figures 3.15-3.17 demonstrate the results of each post-processing step done to an input image of poor quality. It is noted that most of the false minutiae are deleted, however, the post processing steps also delete some real minutiae, but this happens at a very low rate.



Figure 3.9. Result of the fingerprint image manipulated by a Gabor filter with different δ_x and δ_y values: (a) original gray scale fingerprint of good quality; (b) after filtered with of δ_x and δ_y of 12; (c) after filtered with of δ_x and δ_y of 20; (d) after filtered with of δ_x and δ_y of 30.

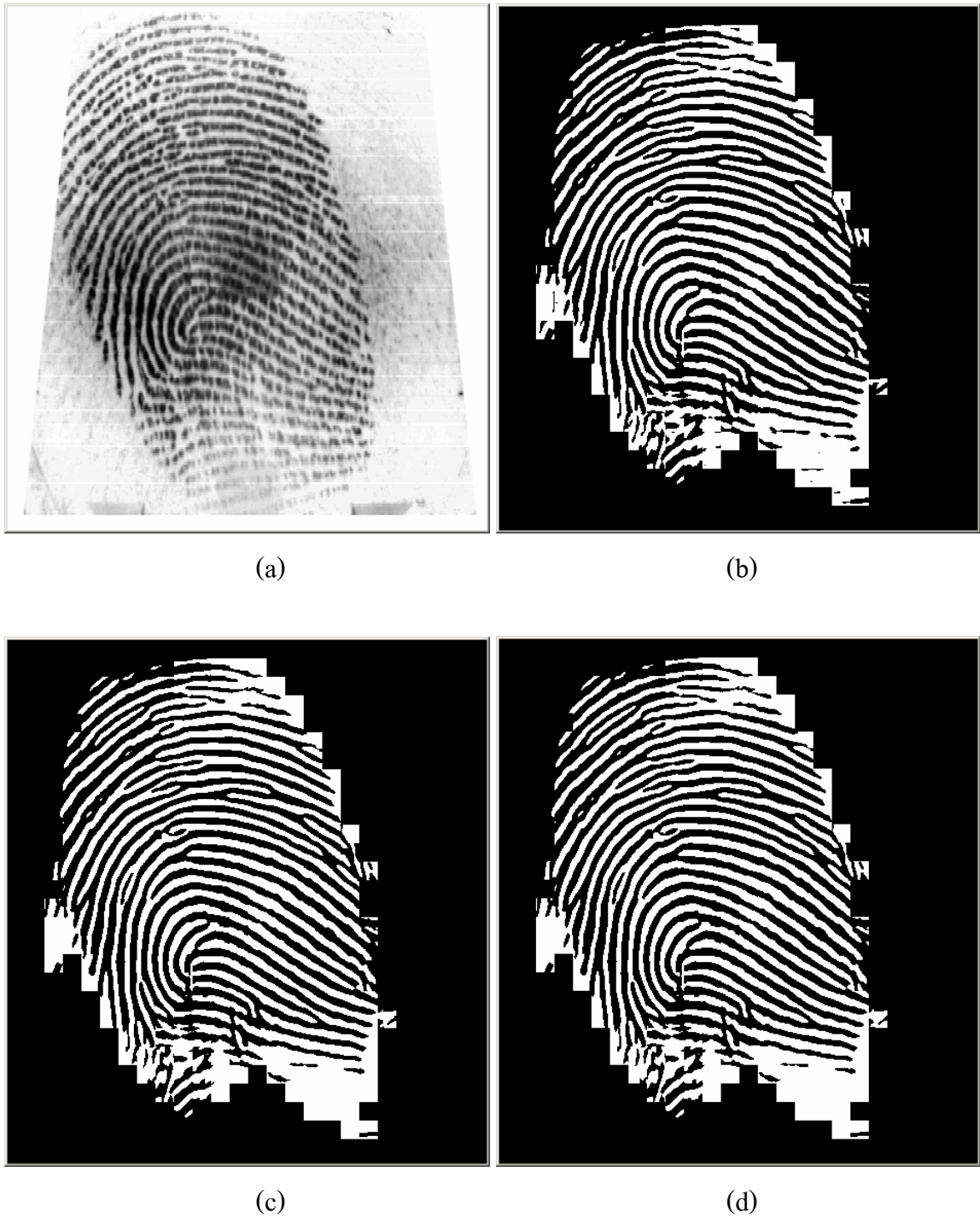


Figure 3.10. Result of the fingerprint image manipulated by a Gabor filter with different δ_x and δ_y values: (a) original gray scale fingerprint of poor quality; (b) after filtered with of δ_x and δ_y of 12; (c) after filtered with of δ_x and δ_y of 20; ; (d) after filtered with of δ_x and δ_y of 30.



Figure 3.11. Result of the fingerprint image manipulated by a Gabor filter with different δ_x and δ_y values: (a) gray scale fingerprint of non-uniform pressure of the finger; (b) after filtered with of δ_x and δ_y of 12; (c) after filtered with of δ_x and δ_y of 20; ; (d) after filtered with of δ_x and δ_y of 30.



Figure 3.12. Result of the binarized fingerprint image due to different threshold values: (a) original gray scale fingerprint of good quality; (b) after binarized using threshold value of -800; (c) after binarized using threshold value of -400; (d) after binarized using threshold value of 0.

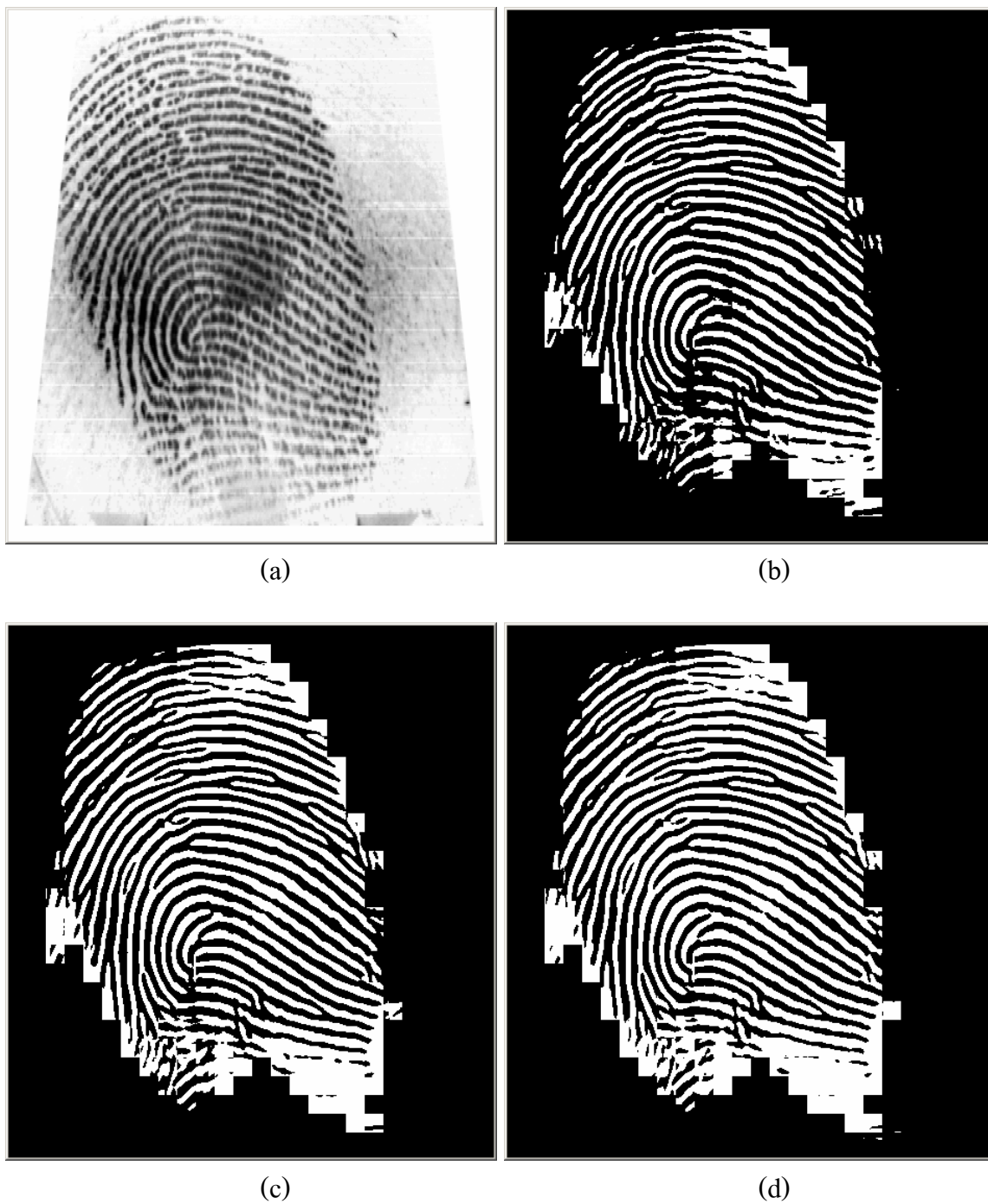


Figure 3.13. Result of the binarized fingerprint image due to different threshold values: (a) original gray scale fingerprint of poor quality; (b) after binarized using threshold value of -800; (c) after binarized using threshold value of -400; (d) after binarized using threshold value of 0.



Figure 3.14. Result of the binarized fingerprint image due to different threshold values: (a) gray scale fingerprint of non-uniform pressure of the finger; (b) using threshold value of -800; (c) using threshold value of -400; (d) using threshold value of 0.

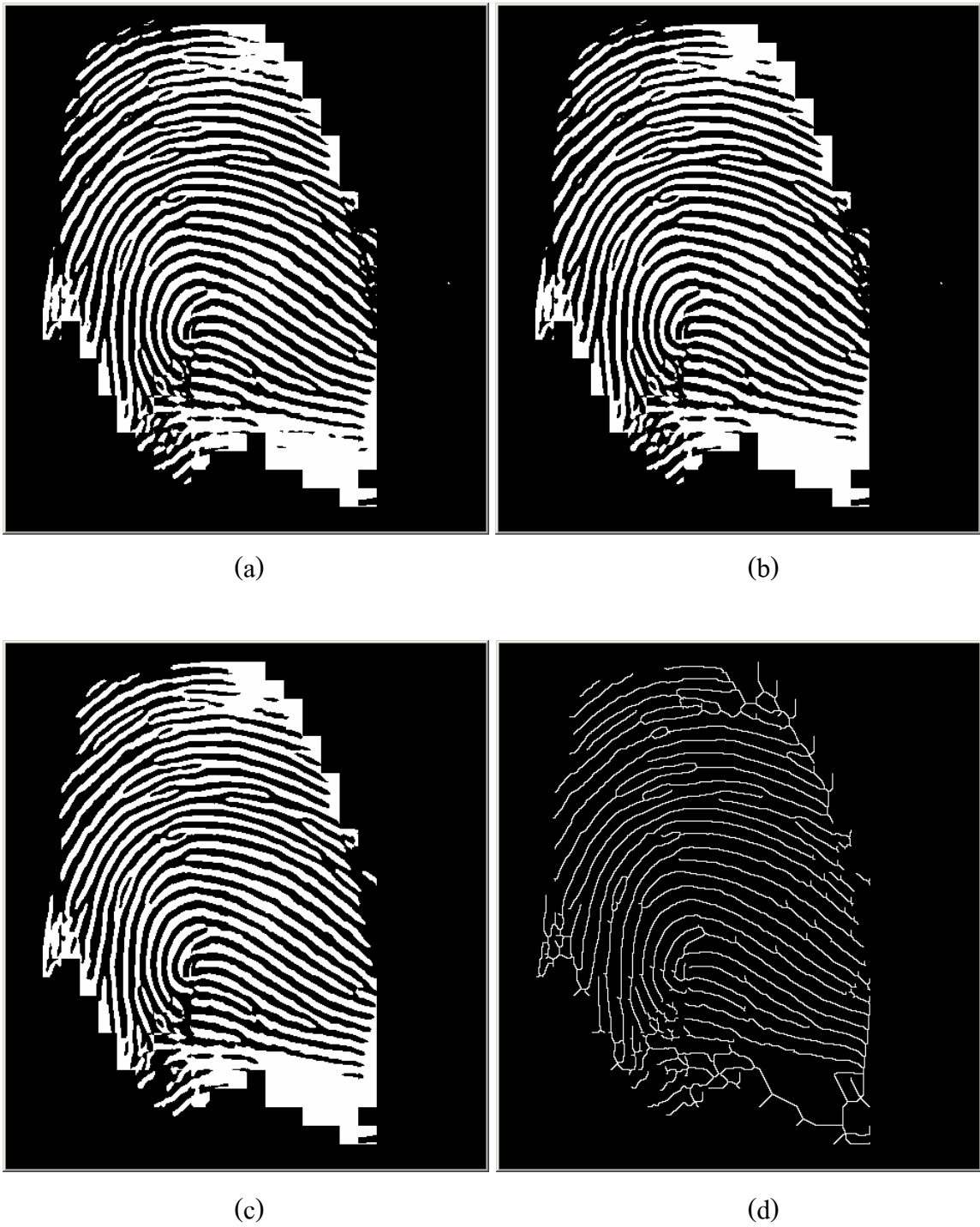


Figure 3.15. Result of the fingerprint post-processing steps: (a) binarized fingerprint image; (b) after small black holes have been removed; (c) after small white ridges have been removed; (d) after thinning process.

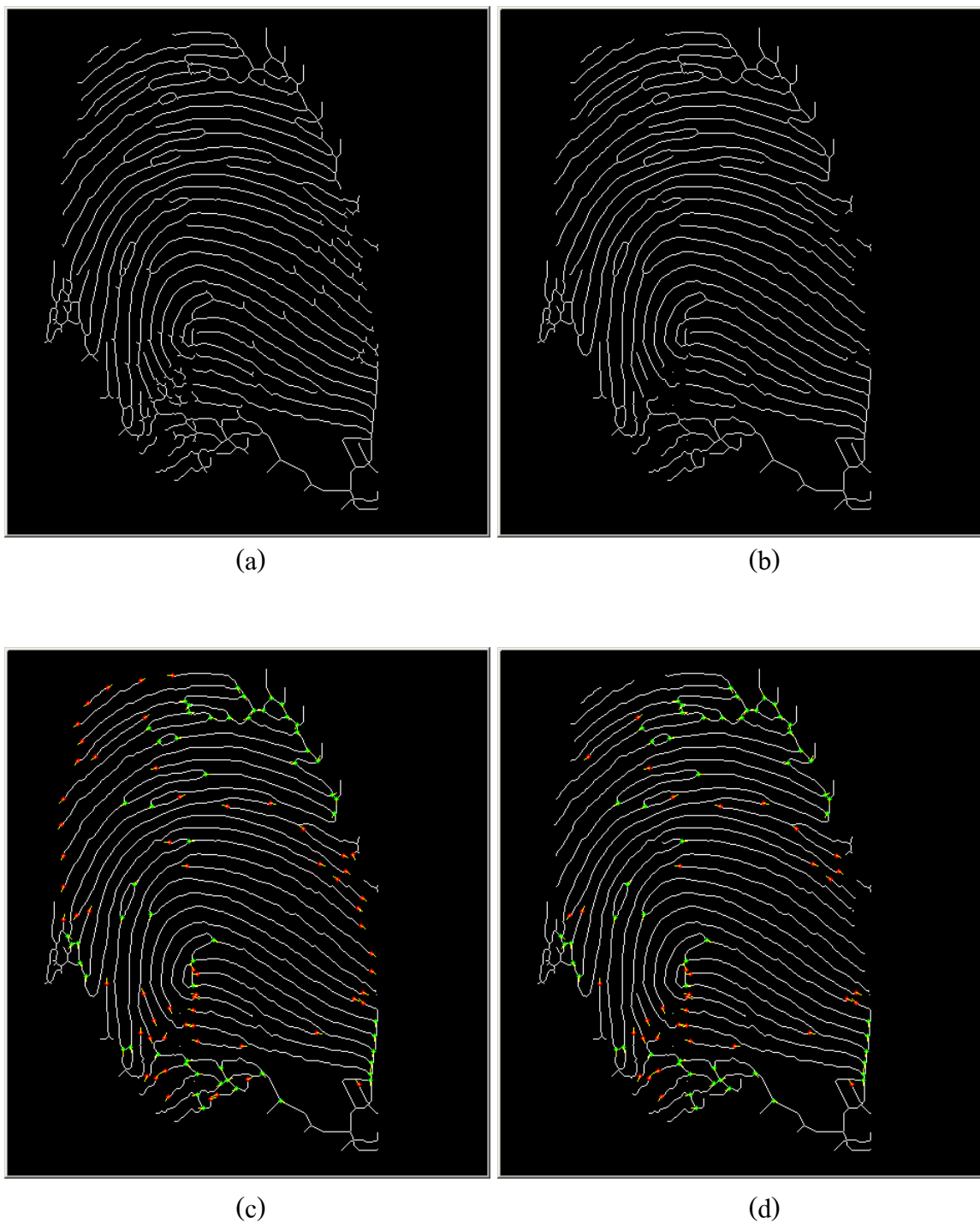


Figure 3.16. Result of the fingerprint post-processing steps: (a) m-connectivity type fingerprint image; (b) after spurs have been deleted; (c) minutiae extracted from spurs deleted image; (d) minutiae after fake termination points have been deleted.

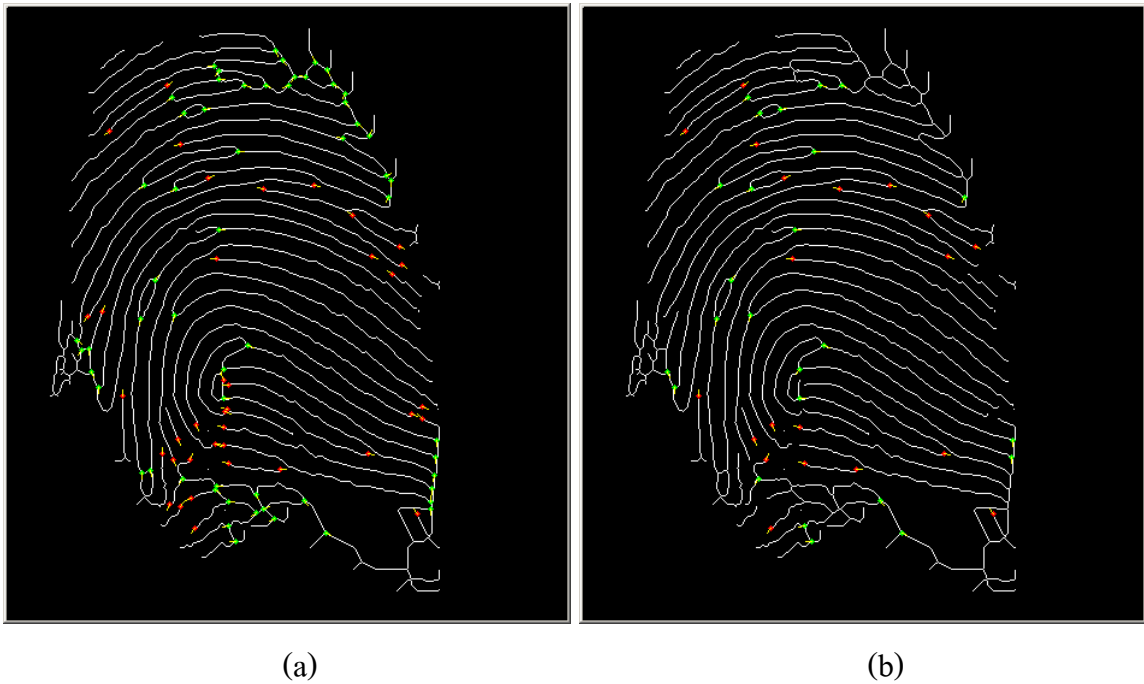


Figure 3.17 Result of too close minutiae deletion; (a) before deletion (b) after too close minutiae are deleted.

3.7 Decision Fusion

As described in section 2.2.20, the top-view and bottom-view matching score distribution must be estimated to combine both top-view and bottom-view features together. To attain both distributions, images of 300 fingers were randomly selected from the database of section 3.4. A total of 627,900 ($300 \times 299 \times 7$) matches were evaluated to estimate the imposter distribution, and 2,100 (300×7) matches were examined to approximate the genuine distribution. As previously described in section 2.2.20, the symbols ω_1 and ω_2 represent the imposter and the genuine class, respectively. x_1 is the top-view matching score derived from top-view matcher and x_2 is the matching score that comes from the bottom-view matcher. Figures 3.18 and 3.19 show the estimated distribution of the bottom-view and top-view distributions respectively.

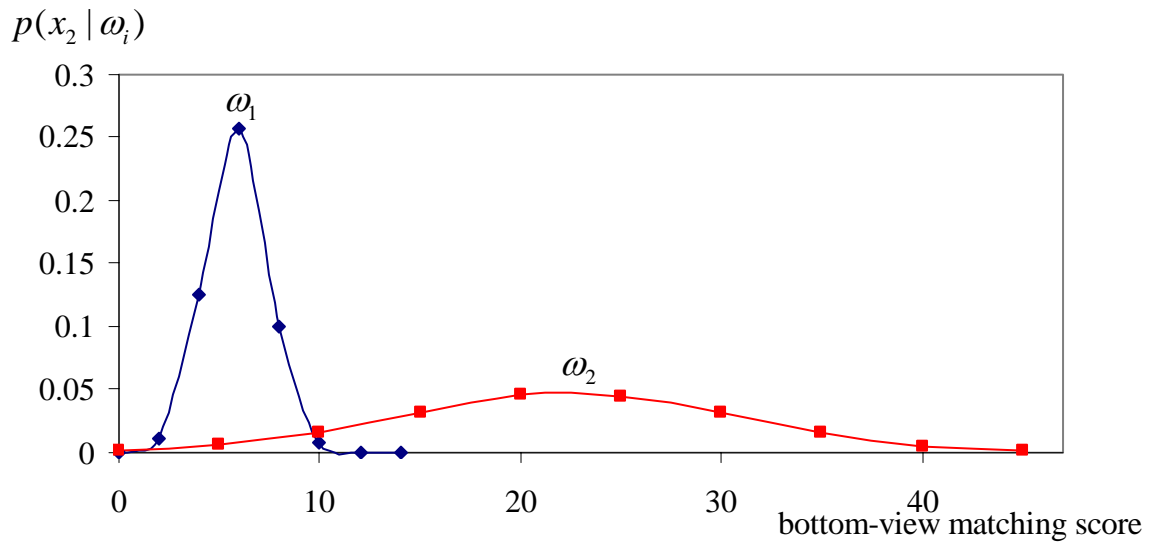


Figure 3.18. The estimated bottom-view matching score distribution.

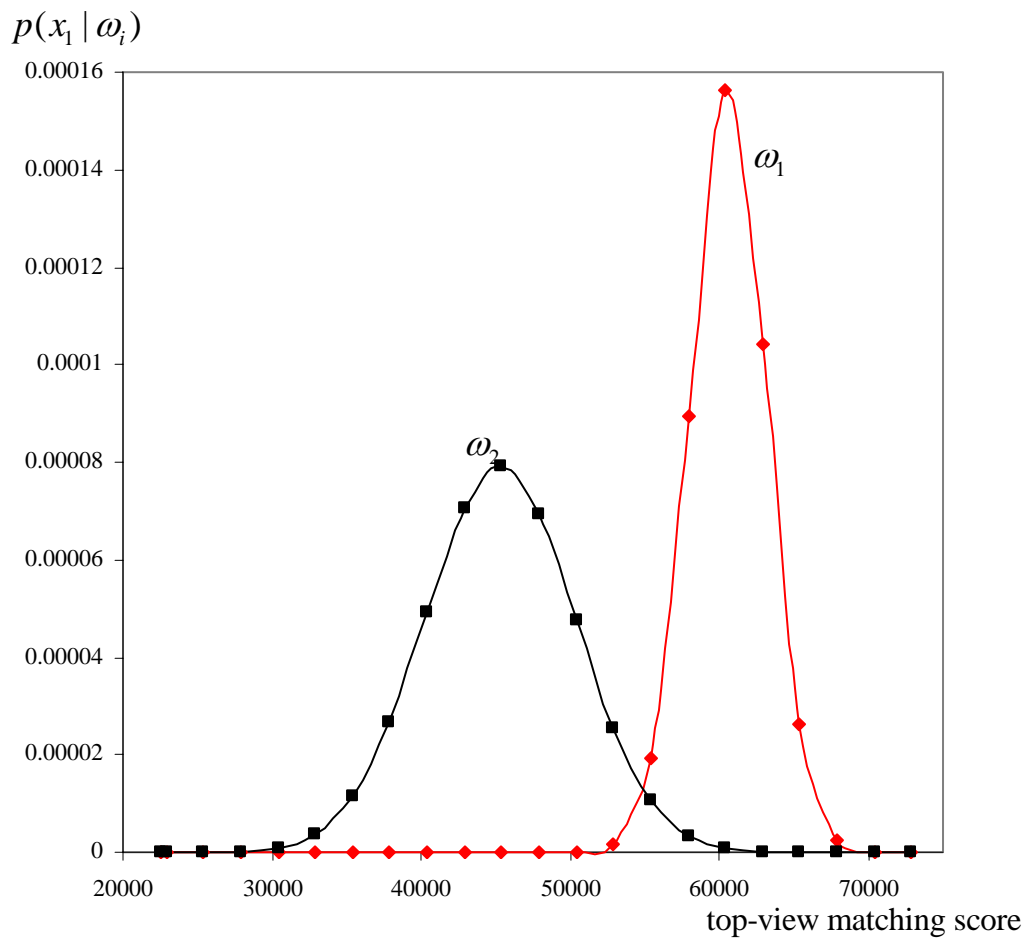


Figure 3.19. The estimated top-view matching score distribution.

3.8 System Performance

Three matchers were implemented: (1) an *HTMM* matcher using the Hough transform-based minutia matching technique, (2) a *SMM* matcher utilizing our minutiae matching algorithm as described in section 2.2.13, and (3) a *TopView* matcher which employs the NailCode feature. The scores from these three matchers were combined to make a multimodal biometric system using the algorithm described in section 2.2.20.

3.8.1 Performance in the verification mode

The FAR and FRR values were plotted on a Receiver Operating Curve (ROC) to judge the performance of the system, as shown in Figure 3.20. The genuine acceptance rate can be obtained from ROC as $1 - \text{FRR}$. For the FAR, 4,474,400 ($800 \times 799 \times 7$) matches were evaluated, and 5,600 (800×7) matches were examined to find the FRR.

The verification performance against three conditions was tested, with each condition using only one score from its respective matcher. There was no combination of these three systems. Figure 3.20 shows that the *SMM* matcher gives better accuracy than the other two matchers at every operating point. At low FARs, the *TopView* matcher gives higher accuracy than the *HTMM* matcher, but the *HTMM* matcher surpasses *TopView* at higher FAR values.

The three matchers are combined into pairs, and the likelihood ratio was used to perform the decision fusion. System accuracy increased, as shown in Figure 3.21. The combination *TopView+SMM* gives the best accuracy. Also, the *SMM* matcher alone has better accuracy than a combination of *TopView+HTMM* at all operating points with FARs lower than 7%. Since biometric systems need to operate at a low FAR value, this made us decide to use our minutia matching algorithm to improve system accuracy in the identification mode.

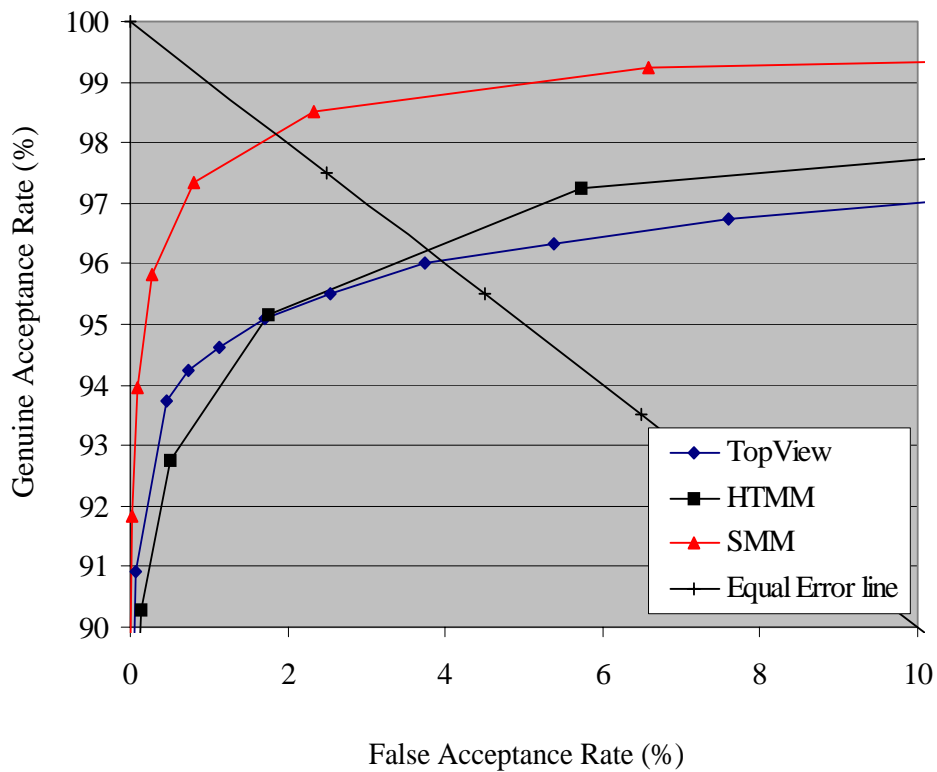


Figure 3.20. Verification performance of individual matchers.

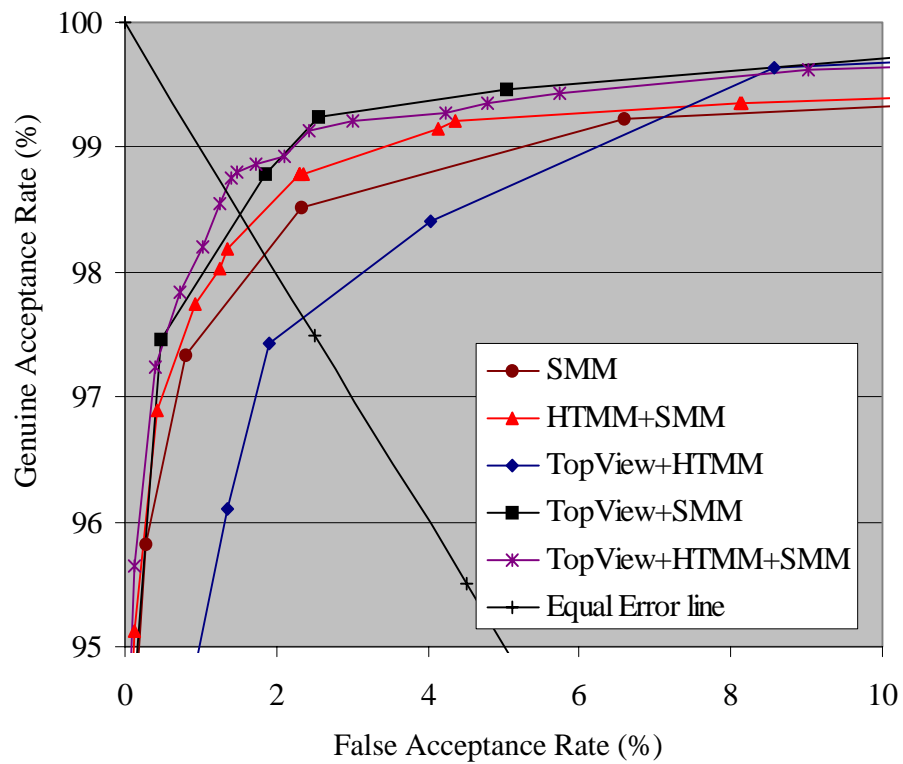


Figure 3.21. Verification performance of all combinations.

Finally, the matching scores of all three matchers are combined, *TopView+HTMM+SMM*. This outperformed all the paired matcher combinations at low FAR values, but for FARs greater than 2%, the *TopView+SMM* combination had lower rejection rates.

The Equal Error Rate (EER) was used to measure the strength of performance gains. The *TopView*, *HTMM* and *SMM* matchers alone yield EERs of 3.91%, 3.80% and 1.86% respectively. The combinations of *TopView+SMM*, *HTMM+SMM*, *TopView+HTMM* yield EERs of 1.52%, 1.64% and 2.35%, respectively. The combination of all three matchers gives the best EER of 1.35%.

As shown in Table 3.8, most of the computing time for the top-view finger image processing is spent on the preprocessing. While the NailCode matching process requires considerably low computation time. Average computing time used to perform verification for NailCode matching and Hough transform-based minutiae matching were 20 μ s and 3.125 ms, respectively. The average time for performing verification using our *SMM* minutiae matching algorithm was 135.8 ms. This reveals that *SMM* is not suitable for directly searching the entire database because of its time-consuming behavior. However, due to its higher accuracy compared to *TopView* and *HTMM*, the *SMM* matcher will be used to improve personal identification accuracy.

Table 3.8. Average computing time for one test on a 2.4 GHz Pentium 4.

Source	Process	Computing time (ms)
Top-view finger image	Preprocessing	193.95
	Feature Extraction	4.01
	Matching	0.02
	Reference point detection	19.05
Fingerprint	Preprocessing	119.64
	Feature Extraction	1.68
	Post Processing	281.45
	Matching (Algorithm <i>HTMM</i>)	3.13
	Matching (Algorithm <i>SMM</i>)	135.80

3.8.2 Performance in the identification mode

To evaluate the performance of the identification mode, the 800 finger samples in our database were divided into four databases of 200 samples each. A total of 22,400 ($800 \times 7 \times 4$) identification operations were evaluated. When the system used a *HTMM* matcher alone its EER was 2.27%, while the *TopView* matcher's EER was 2.84%.

The *HTMM* and *SMM* matchers were combined, but the likelihood ratio was not utilized. Instead, the *HTMM* matcher was used to match the input features against all the templates in the database to find the best ten finger details. The *SMM* matcher was then employed to re-verify these ten details to find the best match. Figure 3.22 shows that this combination had an EER of 1.76%.

When the three matchers are combined, the *TopView* matcher is used to verify the extracted input feature against all the templates in the database. The five best finger details from the *TopView* matcher were obtained and added to a candidate list. The *HTMM* matcher was also utilized to search the database to find the five finger details with the highest matching scores, and they were also put into the candidate list. The *SMM* matcher re-verified all the finger details in the list, and the best match was found. This configuration had an EER of 1.64%. The Equal Error Rate of all experiments are summarized in Table 3.9.

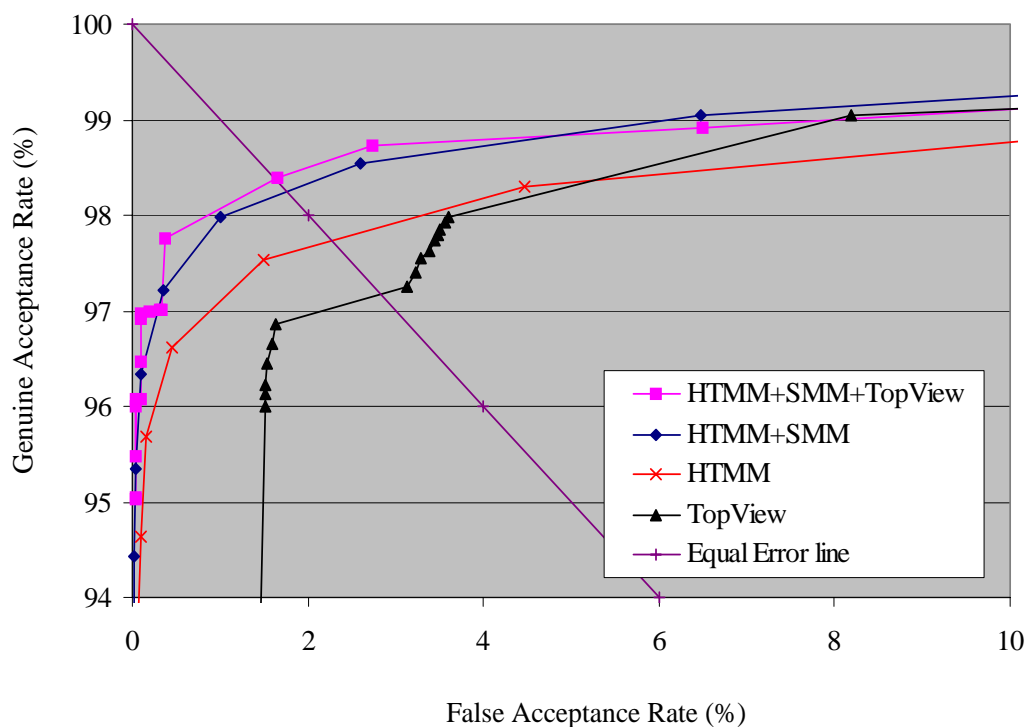


Figure 3.22 Performance in the identification mode.

Table 3.9. Equal Error Rate of the tested configurations.

Mode of Operation	Test Configuration	EER(%)
Verification	<i>TopView</i>	3.91
	<i>HTMM</i>	3.80
	<i>SMM</i>	1.86
	<i>HTMM+SMM</i>	1.64
	<i>TopView+SMM</i>	1.52
	<i>TopView+HTMM</i>	2.35
	<i>TopView+HTMM+SMM</i>	1.35
Identification	<i>HTMM</i>	2.27
	<i>TopView</i>	2.84
	<i>HTMM+SMM</i>	1.76
	<i>TopView+HTMM+SMM</i>	1.64

The average computation time to perform 1:200 matches in the identification operation for the *TopView* and *HTMM* matchers was 4 ms and 625 ms, respectively. The *SMM* matcher required 1.358 seconds to perform 1:10 verifications in both the *SMM+HTMM* and the *SMM+HTMM+TopView* configurations.

3.9 Summary

This chapter begins with the experiments to find the best values of the top-view preprocessing and feature extraction parameters. Then the effect of a reference point location error to the system accuracy was revealed. The results of the reference point error compensation are reported. Three matchers were implemented and the performance of each matcher was evaluated. These three matchers were combined together and the performance of each combination was evaluated. The average computing time of the tested systems are also included in this chapter.

CHAPTER 4

CONCLUSIONS AND FUTURE WORK

4.1 Discussions

System accuracy is highly dependent on the precision of the reference point location: if it cannot be detected by the autonomous detection algorithm, then the input top-view image is rejected. The rejection rate of the proposed algorithm is 0.75 percent. During the enrolment process, if the top-view finger image is rejected, then the user must re-enter the input feature. When the image is rejected during the authentication process, the next step depends on the particular system's requirements. Re-entry is compulsory when both features are simultaneously required. On the other hand, if the top-view finger image is just an additional feature used to augment system accuracy, then a decision can be made when the bottom-view matcher gives a strong enough match.

The average reference-point location error of the proposed algorithm is about 10 pixels. This error can be reduced by our compensation method, but the NailCode matching time is increased by about 8 times compared to when only one reference point is utilized. However, the computation time required for NailCode matching is considerably less than for minutiae matching.

Since the ROC curve gives the overall accuracy of the system at different thresholds, the selection of an appropriate threshold value for a specific system depends on its application. In very high security applications, a system with very low FAR is required at the expense of user convenience. Some genuine users with low matching scores may be rejected by the system, and re-authentication is mandatory. On the other hand, forensic applications require a low FRR to detect the criminal at the expense of investigating large numbers of falsely accepted users. Civilian applications need both low FAR and low FRR. The most appropriate operating point for civilian applications is the EER [56]. The ROC curves shown in figures 3.20-3.22 reveal that the combination of two or more matchers are more suitable for civilian and high security applications than for forensic applications because all the operating points of the combined system under the equal-error-line give better accuracy than a system before combination.

After combination, the NailCode feature reduces the verification error rate of the system by 17.68%. This value is obtained by comparing the results at the EER point between the *HTMM+SMM* and the *HTMM+SMM+TopView* configurations. In identification mode, the system error is reduced by 6.82%. NailCode improves the accuracy of the fingerprint matching system, while requiring very low computation times, and being able to operate in both identification and verification modes. We recommend that a NailCode matcher should be used to increase the accuracy of fingerprint recognition systems.

Since the system using NailCode alone gives lower accuracy than fingerprinting, we suggest that top-view finger imaging should not be used alone to verify or identify individuals, especially when high security is required. Imaging should be employed in conjunction with fingerprinting to improve overall recognition accuracy. These two features can be easily utilized together as parts of one user operation.

Cuts or incised wounds on the finger might affect system accuracy. However, it is difficult to test this circumstance. Skin wrinkles on a finger will increase over time, but at a slow rate. For example, it has been demonstrated that the same finger captured 990 days after its previous snapshot can still be correctly identified [55]. However, the number of the tested fingers is too low for statistical analysis because the author did not plan to test this issue during the earlier stages of the experiments. In order to keep the top-view feature up-to-date, biometric updating should be used to overcome any time variances. The simplest approach is to update the top-view feature when the matching score of both top-view and bottom-view are greater than the pre-specified thresholds. The update should be done at run-time to avoid user inconvenience.

The top-view finger image can be used to reduce the possibility of fake fingerprints. In other words, it can be used to detect the liveness of fingerprint. To do this, a combination strategy between the top-view and bottom-view must be employed at the abstract level. When combined with one fingerprint matcher, the top-view and bottom-view matchers simply report “yes” or “no”, and an AND operation is employed to make a final decision. On the other hand, when NailCode is combined with more than one fingerprint matcher, a majority vote rule can be utilized.

Since we do not have the library to get a fingerprint image directly from a digital Persona UareU4000B fingerprint sensor, modifications were carried out to demonstration code from ITWORKS (<http://www.itworksolutions.com>, 2010) to get the fingerprint image from the sensor. The code uses DLLs to interface with the fingerprint sensor hardware, and the acquired image seems a little degraded compared to the sample fingerprints from DB2 of FVC2004 [41], where the images are obtained from the same sensor. It is believed that the accuracy of the implemented system would be improved if the proper sensor interfacing library was used.

The finger inclination angle can be detected during top-view finger preprocessing, and used to reduce the search space for fingerprint matching. However, to ensure that the top-view and bottom-view features are statistically independent of each other, the inclination angle is not utilized in the fingerprint matching module.

4.2 Research Contributions

This thesis proposes the use of top-view finger imaging to increase the accuracy of fingerprint recognition systems without requiring extra work by the user. Methods for preprocessing, feature extraction, and matching were invented for the top-view finger image. The proposed method works well in both verification and identification modes. Since top-view finger imaging needs very low computation time compared to fingerprinting, it can be used as an indexing method for reducing the time for large database searching. It also reduces the possibility of fraud by having recognition rely on more than one feature.

4.3 Future Work

The skin wrinkles on a finger will increase over time, so it is recommended that the systematic process of keeping the top-view feature up-to-date should be further studied and developed.

Although NailCode matching can tolerate with the translation and rotation of the finger, it cannot tolerate scaling of the top-view finger image. This means that NailCode matching is a sensor-dependent feature, so the same sensor is required for both the enrolment and authentication processes. A scaling ability must be developed to overcome this limitation.

A top-view finger image might be useful for reducing the use of fake fingerprints. Fingerprint liveness detection using the top-view finger image should be further developed.

Other feature extraction techniques should be investigated, so the need for a reference point can be dropped.

REFERENCES

- [1] Maltoni, D., Maio, D., Jain, A.K. and Prabhakar, S., *Handbook of Fingerprint Recognition*, Springer, New York, U.S.A., 2003, pp. 15, 25-26, 146-147, 258-280.
- [2] Hong, L. and Jain, A.K. "Integrating faces and fingerprints for personal identification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1295-1307, Dec. 1998.
- [3] Jain, A.K., Hong, L. and Kulkarni, Y., "A Multimodal Biometric System using Fingerprint, face and Speec," *Proceedings of International Conference on Audio-and-Video-Based Biometric Person Authentication*, pp. 182-187, 1999.
- [4] Marcialis, G.L. and Roli, F., "Fingerprint verification by fusion of optical and capacitive sensors," *Pattern Recognition Letters*, vol. 25, no. 11, pp. 1315-1322, Jun. 2004.
- [5] Prabhakar, S. and Jain, A.K., "Decision-level fusion in Fingerprint verification," *Pattern Recognition*, vol. 35, no. 4, pp. 861-874. Apr. 2002.
- [6] Wang, X., Sushita, K., and Shimizu, K., "Evaluation of personal identification system by transillumination imaging of a finger," *IEEE EMBS Asian-Pacific Conference on Biomedical Engineering*, pp. 306-307, Oct. 2003.
- [7] Xueyan, L., Xhuxu, G., Fengli, G., and Ye, L., "Vein Pattern Recognitions by Moment Invariants," *The 1st International conference on ICBBE*, pp. 612-615, Jul. 2007.
- [8] Kumar, A., and Zhou, Y., "Personal identification using finger knuckle orientation features," *Electronics Letters*, vol. 45, no. 20, pp. 1023-1025, Sep. 2009.
- [9] Takeda, M., Uchida, S., Hiramatsu, K., Matsunami, T., "Finger image identification method for personal verification," *International Conference on Pattern Recognition*, pp. 761-766, Jun. 1990.

- [10] Liu, Ming, L., Kunlun, L., Tianshu, Z., Baozong, Y., Zhenjiang, M., "PDE-based Finger Image Denoising," *International Conference on Artificial Intelligence and Computational Intelligence*, pp. 540-543, Nov. 2009.
- [11] Kittler, J., Hatef, M., Duin, R.P.W., Matas, J., "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226-239, Mar. 1998.
- [12] Hong, L. Wan, Y., Jain, A., "Fingerprint image enhancement: algorithm and performance evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 777 – 789, Aug. 1998.
- [13] Farina, A., Kovacs-Vajna, Z. M., Leone, A., "Fingerprint minutiae extraction from skeletonized binary images," *Pattern Recognition*, vol. 32, no. 5, pp. 877-889. May 1999.
- [14] Gonzalez, R. and Woods, R., *Digital Image Processing*(2nd edition), Prentice-Hall, New Jersey. U.S.A., 2002, pp. 600-607.
- [15] Canny, J., "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679-698, Nov. 1986.
- [16] Green, B., "Canny Edge Detection Tutorial," http://www.pages.drexel.edu/~weg22/can_tut.html. (accessed. April 15, 2010).
- [17] Brunelli, R., Falavigna, D., "Person identification using multiple cues," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 10, pp. 955-966, Oct. 1995.
- [18] Duc B., Bigun E.S., Bigun J., Maitre G., and Fischer S., "Fusion of Audio and Video Information for Multi Modal Person Authentication," *Pattern Recognition Letters*, vol. 18, no. 9, pp. 835-843, Sep. 1997.
- [19] Ross, A., Jain, A.K., and Qian, J., "Information Fusion in Biometrics," *Proc. Int. Conf. on Audio and Video-Based Biometric Person Authentication*, pp. 354-359, 2001.

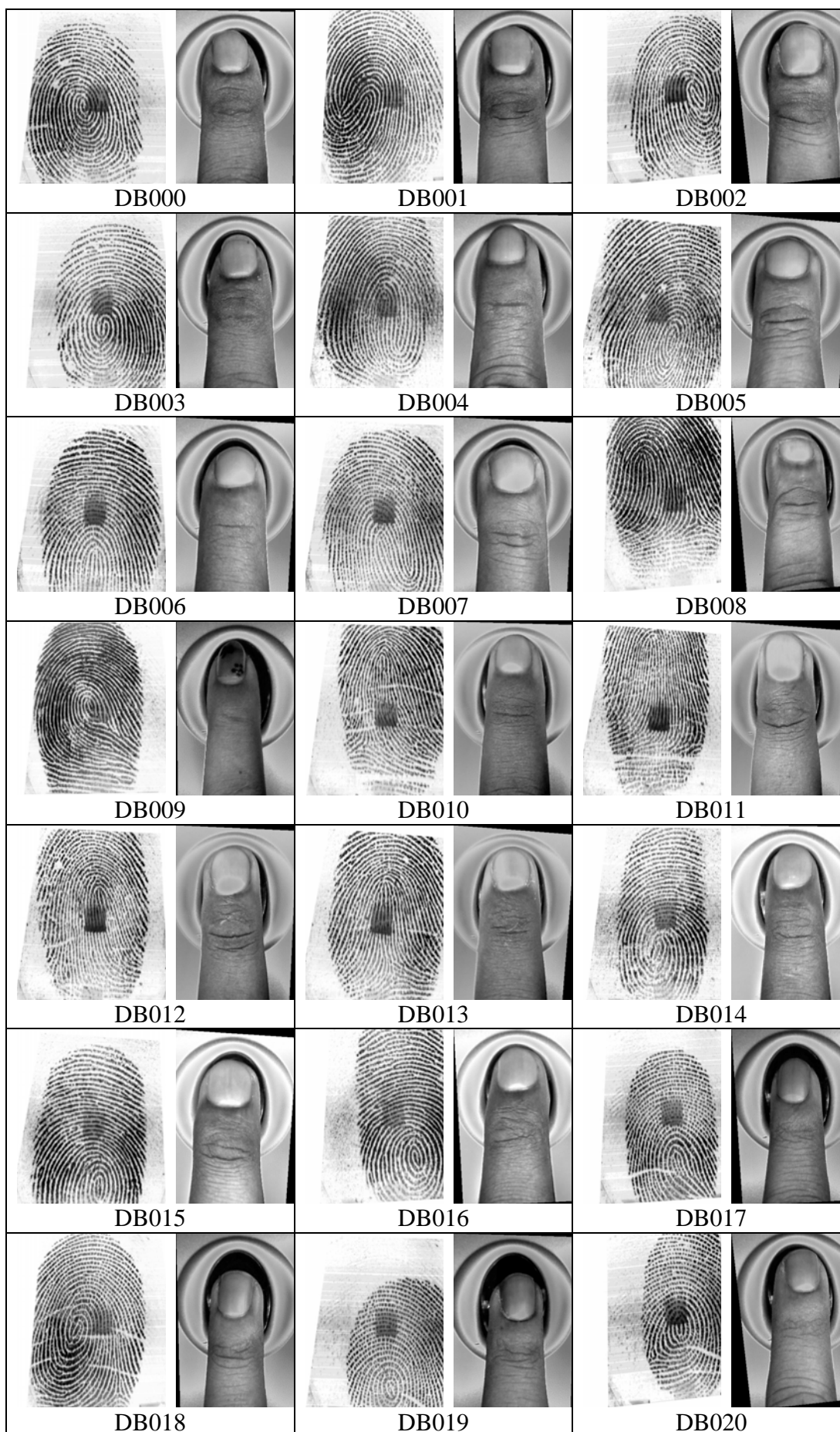
- [20] Choudhury, T., Clarkson, B., Jebara, T., Pentland, A., "Multimodal Person Recognition Using Unconstrained Audio and Video," *Proc. 2nd Conf. Audio- and Video-Based Biometric Person Authentication*, Univ. of Maryland, pp. 176-181, 1999.
- [21] Verlinde, P., Chollet, G., Acheroy, M., "Multimodal identity verification using expert fusion," *Information Fusion*, vol. 1, no. 1, pp. 17-33, Jul. 2000.
- [22] Ben-Yacoub, S., Abdeljaoued, Y., Mayoraz, E., "Fusion of Face and Speech Data for Person Identity Verification," *IDIAP research report*. January 1999. IDIAP-RR-99-03.
- [23] Babler, W.J., "Embryological Development of Epidermal Ridges and Their Configurations," *Birth Defects Original Article Series*, pp. 199-208, vol. 27, no. 2, 1991.
- [24] Senior, A., "A hidden Markov model fingerprint classifier," *Conference Record of the Thirty-First Asilomar Conference on Signals, Systems & Computers*, pp. 306-310, Nov. 1997.
- [25] Hao, G., Zong, Y.O., "Automatic Fingerprint Classification based on Embedded Hidden Markov Models," *International Conference on Machine Learning and Cybernetics*, pp. 3033-3038, Nov. 2003.
- [26] Munir, M.U., Javed, M.Y., "Fingerprint Matching using Ridge Patterns," *First International Conference on Information and Communication Technologies*, pp. 116-120, Aug. 2005.
- [27] Cappelli, R., Maio, D., Maltoni, D., "Indexing Fingerprint Databases for Efficient 1:N Matching," *Proceedings Sixth International Conference on Control, Automation, Robotics and Vision (ICARCV2000)*, Singapore, Dec. 2000.
- [28] Jain, A.K., Prabhakar, S., Hong, L., Pankanti, S., "Filterbank-Based Fingerprint Matching," *IEEE Transactions on Image Processing*, vol. 9, no. 5, pp. 846-859, May 2000.

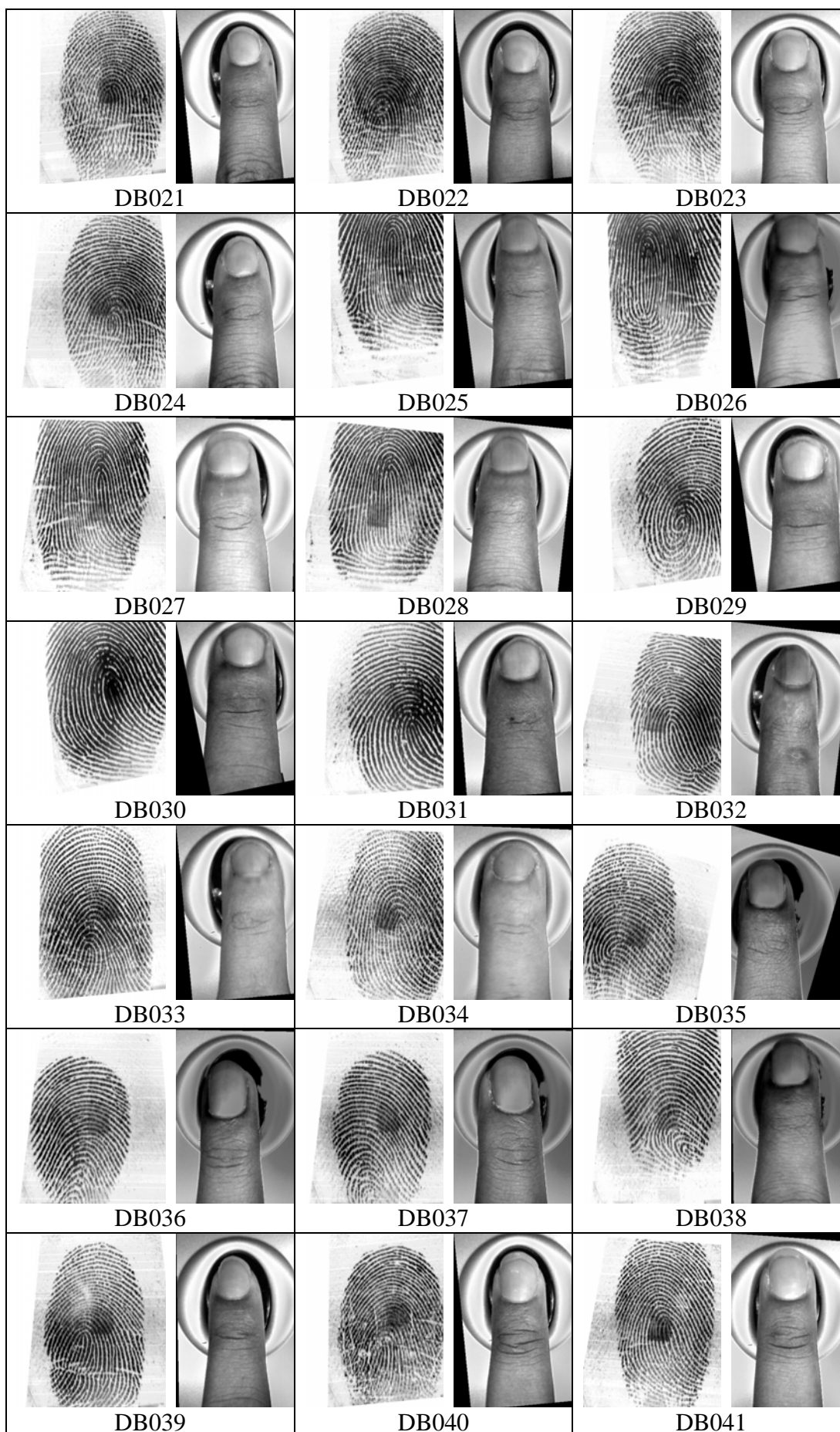
- [29] Lifeng, S., Zhao, F., Tang, X., “Improved fingercode for filterbank-based fingerprint matching,” *International Conference on ICIP 2003*, pp. 895-898, Sep. 2003.
- [30] Shigematsu, S., Morimura, H., Tanabe, Y., Adachi, T., Machida, K., “A single-chip fingerprint sensor and identifier,” *IEEE Journal of Solid-State Circuits*, vol. 34, no. 12, pp. 1852 – 1859, Dec. 1999.
- [31] Xiao, Q., Raafat, H., “A combined statistical and structural approach for fingerprint image postprocessing,” *IEEE International Conference on Systems, Man and Cybernetics*, pp. 331-335, Nov. 1990.
- [32] Maio, D., Maltoni, D., “Direct gray-scale minutiae detection in fingerprints,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 1, pp. 27 – 40, Jan. 1997.
- [33] Jiang, X., Yau, W.Y., Ser, W., “Detecting the fingerprint minutiae by adaptive tracing the gray-level ridge,” *Pattern Recognition*, vol. 34, no. 5, pp. 999-1013, May 2001.
- [34] Beleznai, C., Ramoser, H., Wachmann, B., Birchbauer, J., Bischof, H., Kropatsch, W., “Memory efficient fingerprint verification,” *International Conference on Image Processing*, pp. 463-466, Oct, 2001.
- [35] Suethanoowong, A., and Karnjanadecha, M., “Embedded Fingerprint Verification System,” *EECON28*, pp. 1005-1008, Oct. 2005.
- [36] Ratha, N.K., Karu, K., Shaoyun Chen, Jain, A.K., “A real-time matching system for large fingerprint databases,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 799 – 813, Aug. 1996.
- [37] Lindoso, A., Entrena, L., Lopez-Ongil, C., Liu, J., “Correlation-based fingerprint matching using FPGAs,” *IEEE International Conference on Field-Programmable Technology*, pp. 87-94, Dec. 2005.
- [38] Watson, C.I., Wilson, C.L. 1992. NIST Special Database 4, Fingerprint Database, U.S. National Institute of Standards and Technology. Reading, MA.

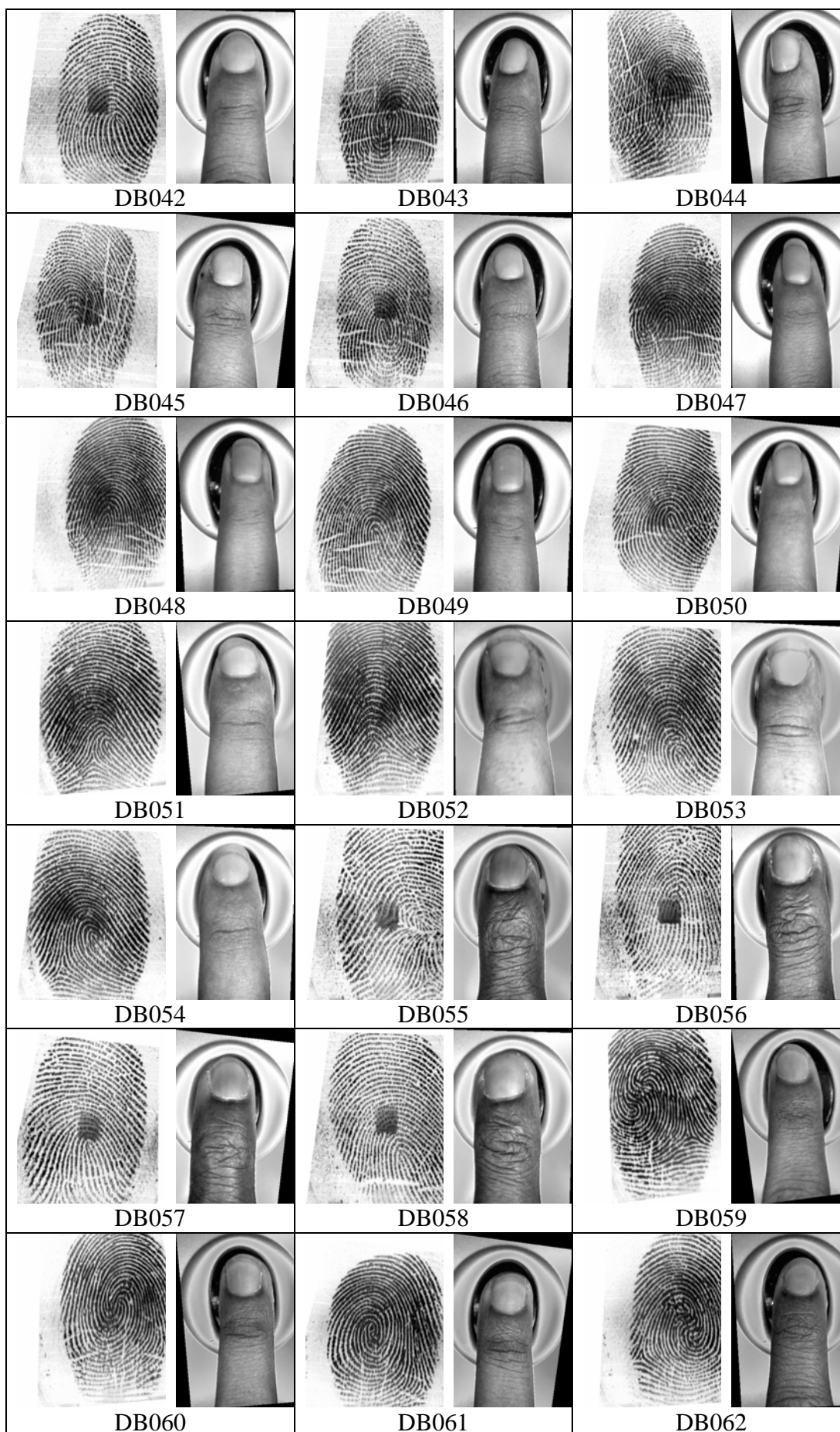
- [39] Maio, D., Maltoni, D., Cappelli, R., Wayman, J.L., Jain, A.K., “FVC2000: fingerprint verification competition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 402-412, Mar. 2002.
- [40] Maio, D., Maltoni, D., Cappelli, R., Wayman, J.L., Jain, A.K., “FVC2002: Second Fingerprint Verification Competition,” *16th International Conference on Pattern Recognition*, pp. 811-814, Aug. 2002.
- [41] Biometric System Lab - University of Bologna, “FVC2004: Fingerprint Verification Competition,” [http://bias.csr.unibo.it /fvc2004](http://bias.csr.unibo.it/fvc2004). (accessed. April 15, 2010).
- [42] Jain, A.; Lin Hong; Bolle, R., “On-line fingerprint verification,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 302-314, Apr. 1997.
- [43] Bazen, A.M., Gerez, S.H., “Systematic methods for the computation of the directional fields and singular points of fingerprints,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 905–919, Jul. 2002.
- [44] Senior A., Bolle, R., “Improved Fingerprint Matching by Distortion Removal,” *IEICE Transactions on Information and Systems*(Special Issue on Biometrics), vol. E84-D, no. 7, pp. 825-832, Jul. 2001.
- [45] Lee, D., Choi, K., and Kim, J., “A robust fingerprint matching algorithm using local alignment,” *Proc. Int. Conf. on Pattern Recognition*, vol. 3, pp. 803-806, 2002.
- [46] Bradski, G. and Kaehler, A., *Learning OpenCV*, O’Reilly, U.S.A, 2008, pp. 248-250, 455-457.
- [47] Wahab, A., Chin, S.H., Tan, E.C., “Novel approach to automated fingerprint recognition,” *IEE Proceedings of Vision, Image and Signal Processing*, pp. 160-166, Jun. 1998.

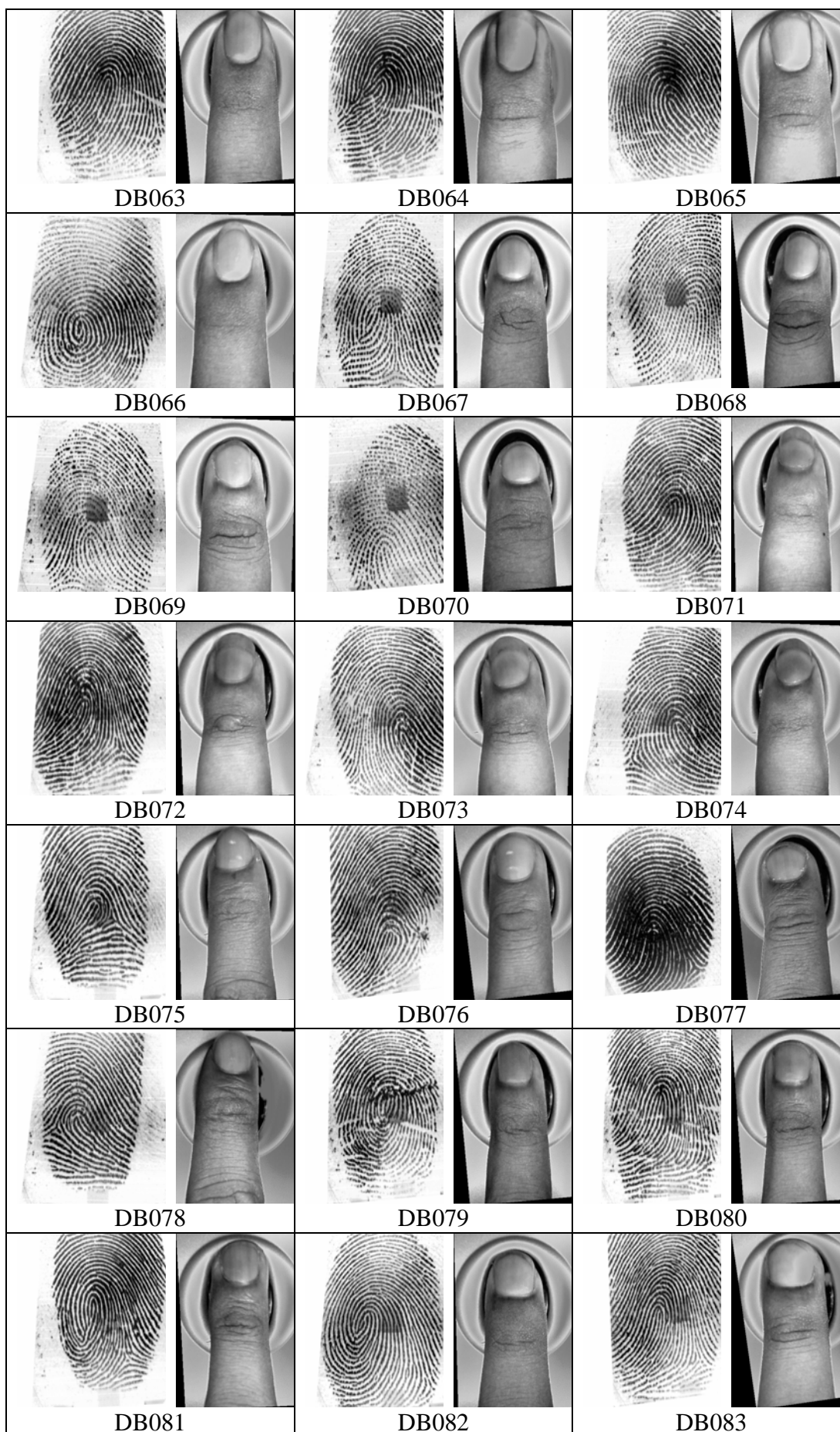
- [48] Jiang, X., Yau, W.Y., "Fingerprint minutiae matching based on the local and global structures," *15th International Conference on Pattern Recognition*, pp. 1038-1041, Sep. 2000.
- [49] Hao, Y., Tan, T., Wan, Y., "Fingerprint matching based on error propagation," *International Conference on Image Processing*, pp. 273-276, Sep. 2002.
- [50] Nagaty, K.A., "An energy-based fingerprint matching system," *Consumer Communications and Networking Conference*, pp. 706-709, Jan. 2004.
- [51] Guo, H., "A Hidden Markov Model Fingerprint Matching Approach," *Proceedings of 2005 International Conference on Machine Learning and Cybernetics*, pp. 5055-5059, Aug. 2005.
- [52] Seow, B.C., Yeoh, S.K., Lai, S.L., Abu, N.A., "Image based fingerprint verification," *Student Conference on Research and Development : SCOReD*, pp. 58-61, Jul. 2002.
- [53] Jiang, X., "A study of fingerprint image filtering," *International Conference on Image Processing*, pp. 238-241, Oct. 2001.
- [54] Duda, R.O., Hart, P.E. and Stork, D.G., *Pattern Classification*(2nd Edition), Wiley, New York, U.S.A., 2000, pp. 27, 52, 616.
- [55] Chaikan, P., Karnjanadecha, M., "Person Recognition using Fingerprints and Top-View Finger Images," *Songklanakarin Journal of Science and Technology*, vol. 32, no. 1, pp. 71-79, Jan.-Feb. 2010.
- [56] Jain, A., and Pankanti, S., "Automated Fingerprint Identification and Imaging Systems," <http://www.research.ibm.com/ecvg/pubs/sharat-forensic.pdf>. (accessed. April 15, 2010).

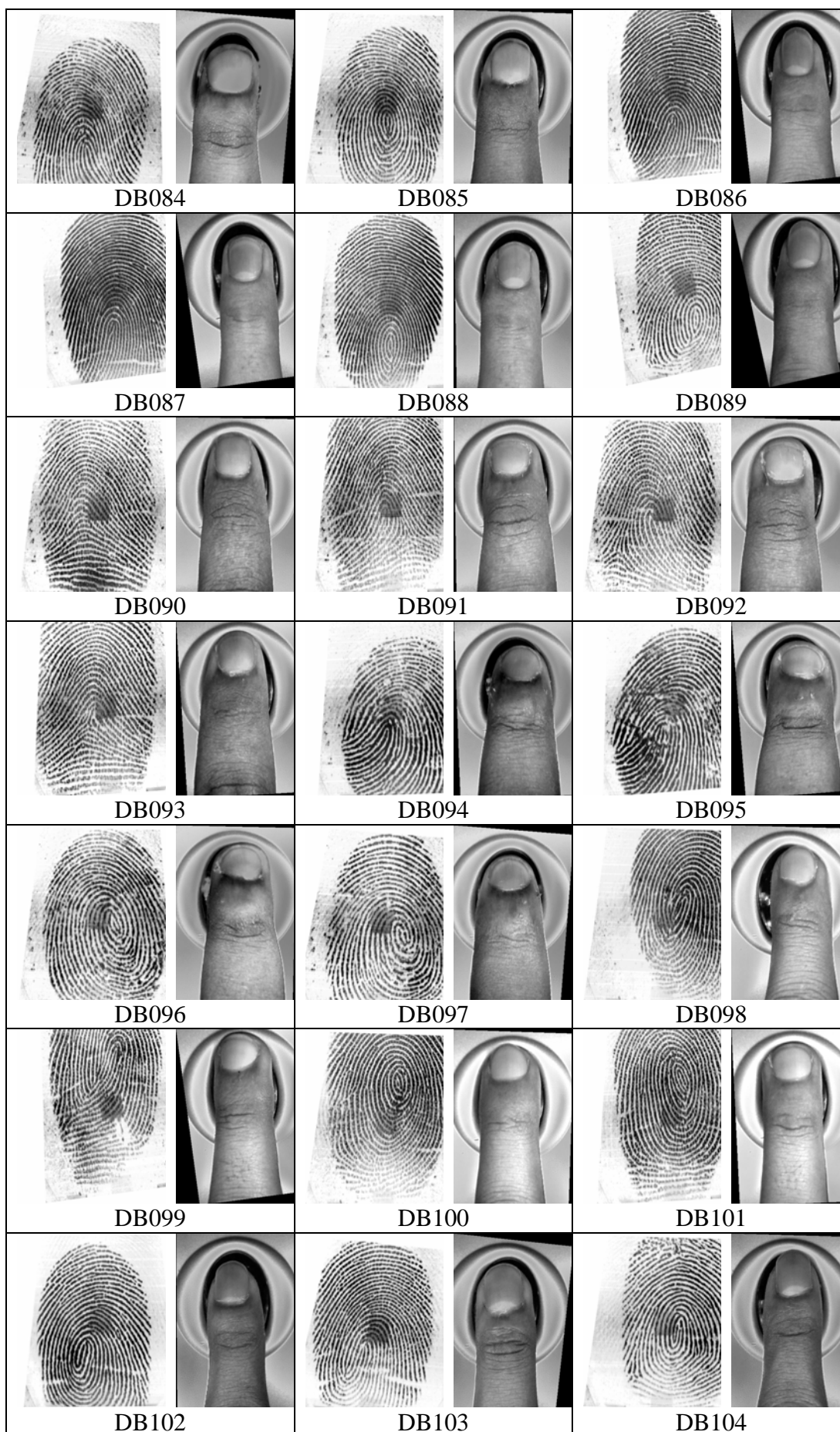
APPENDIX A. TEST DATABASE

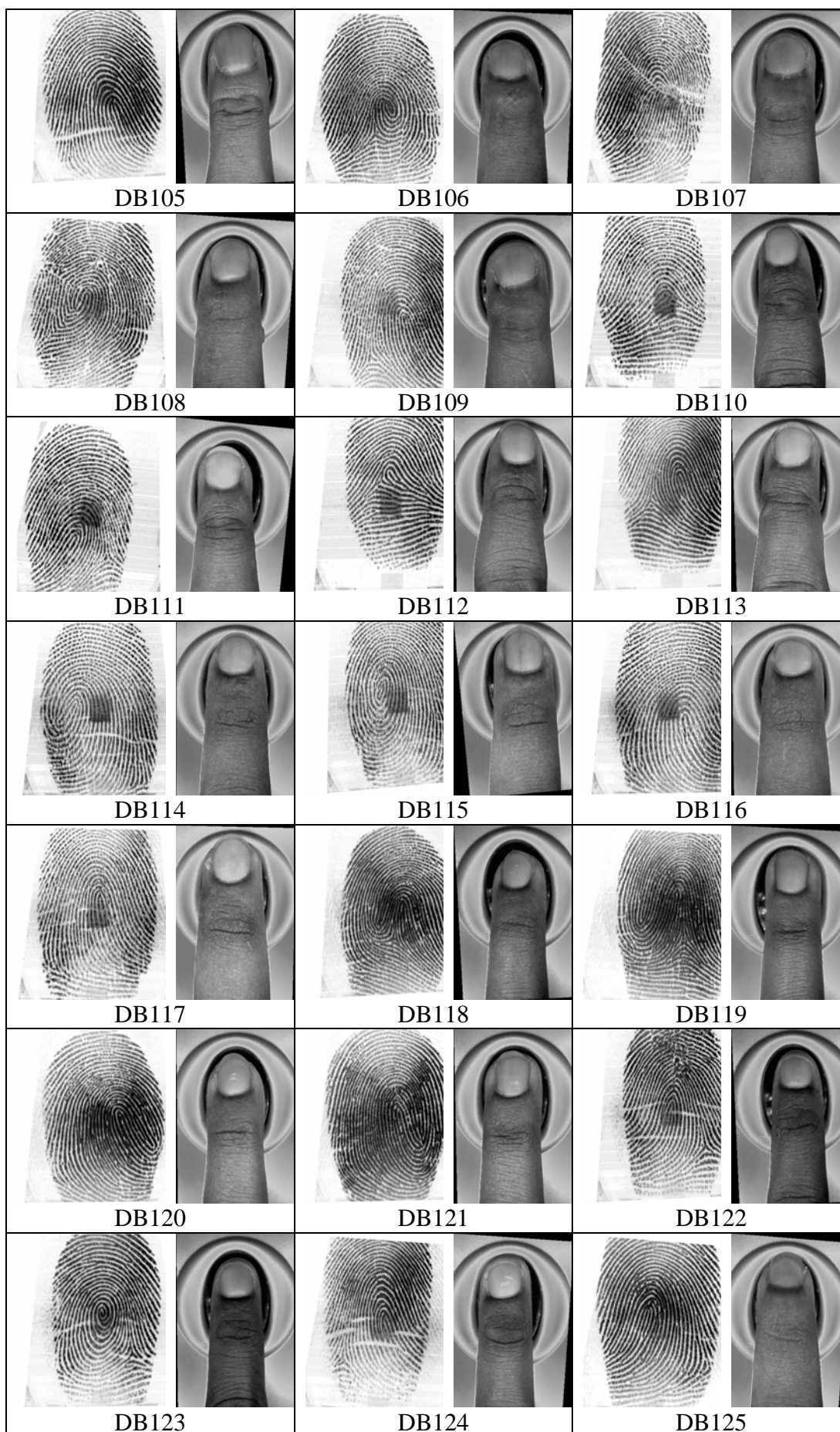


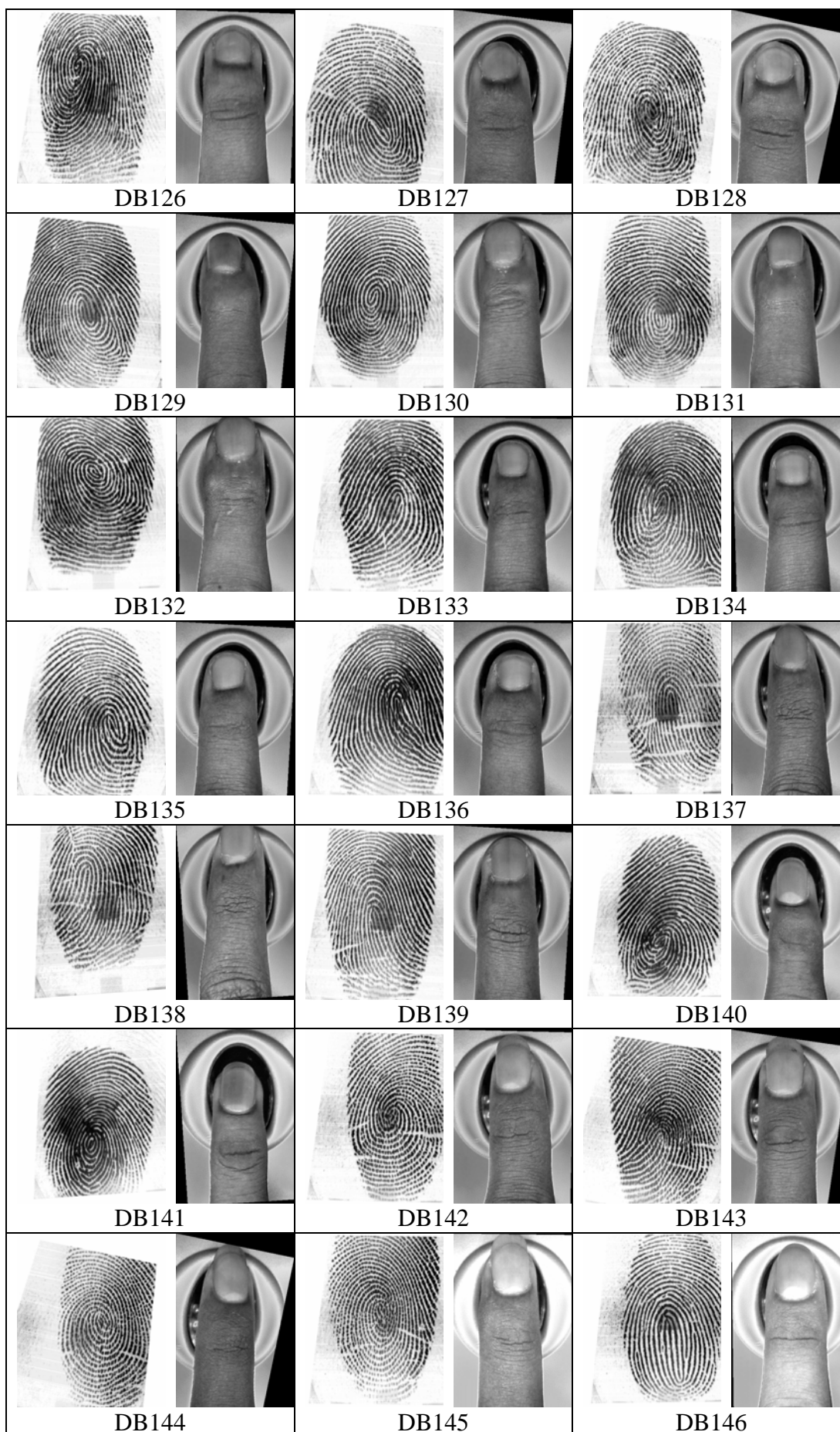


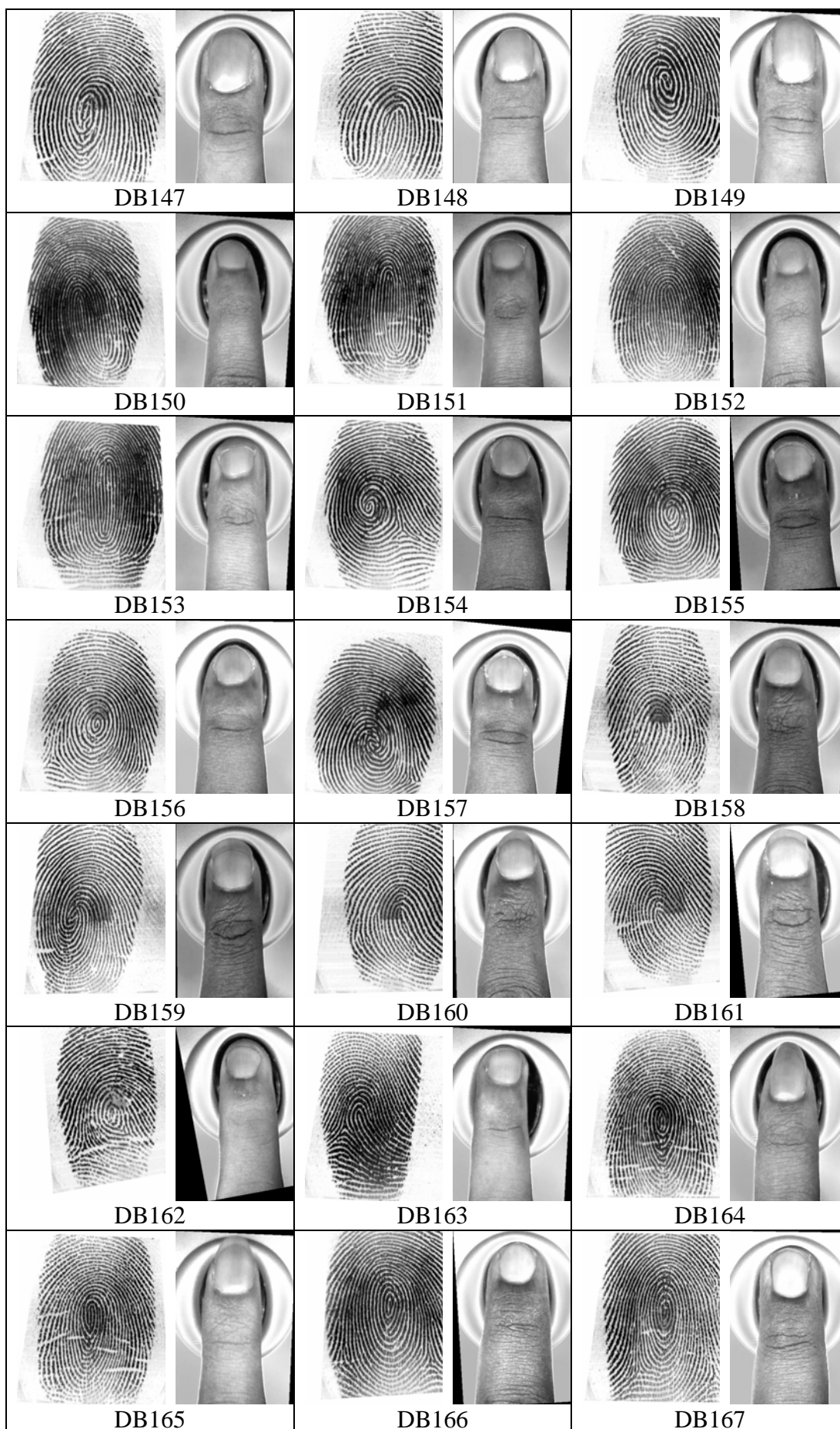


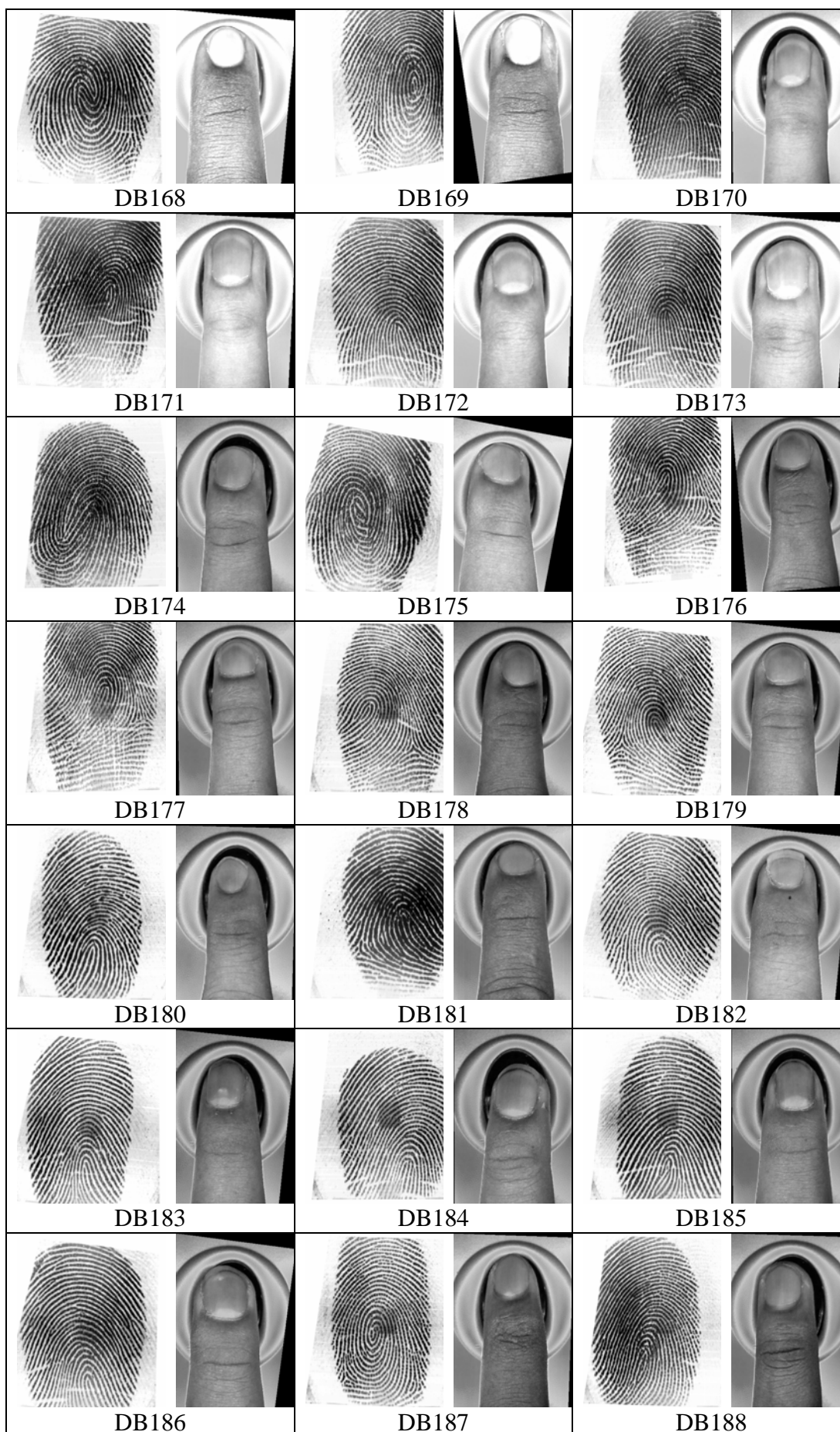


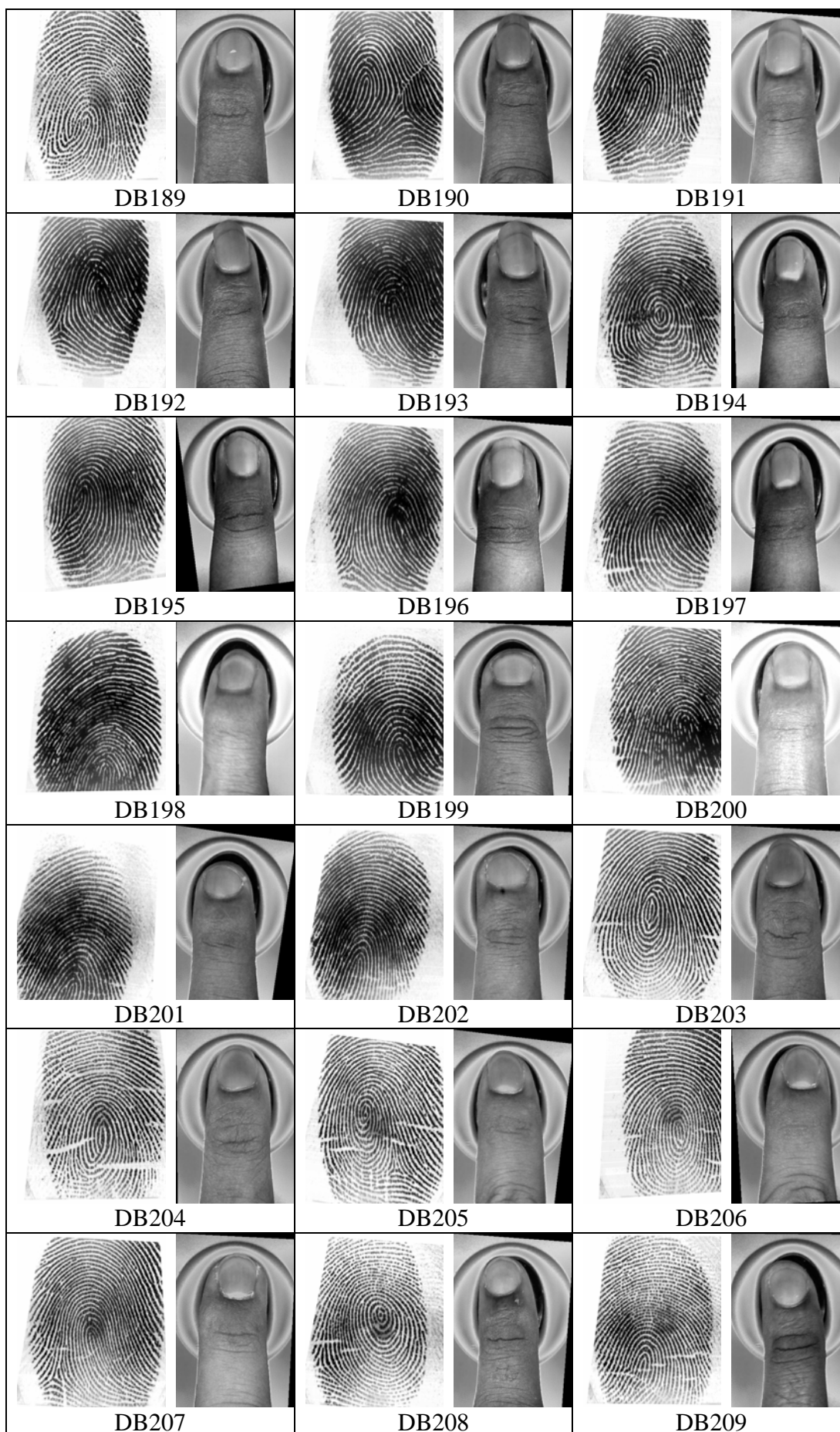


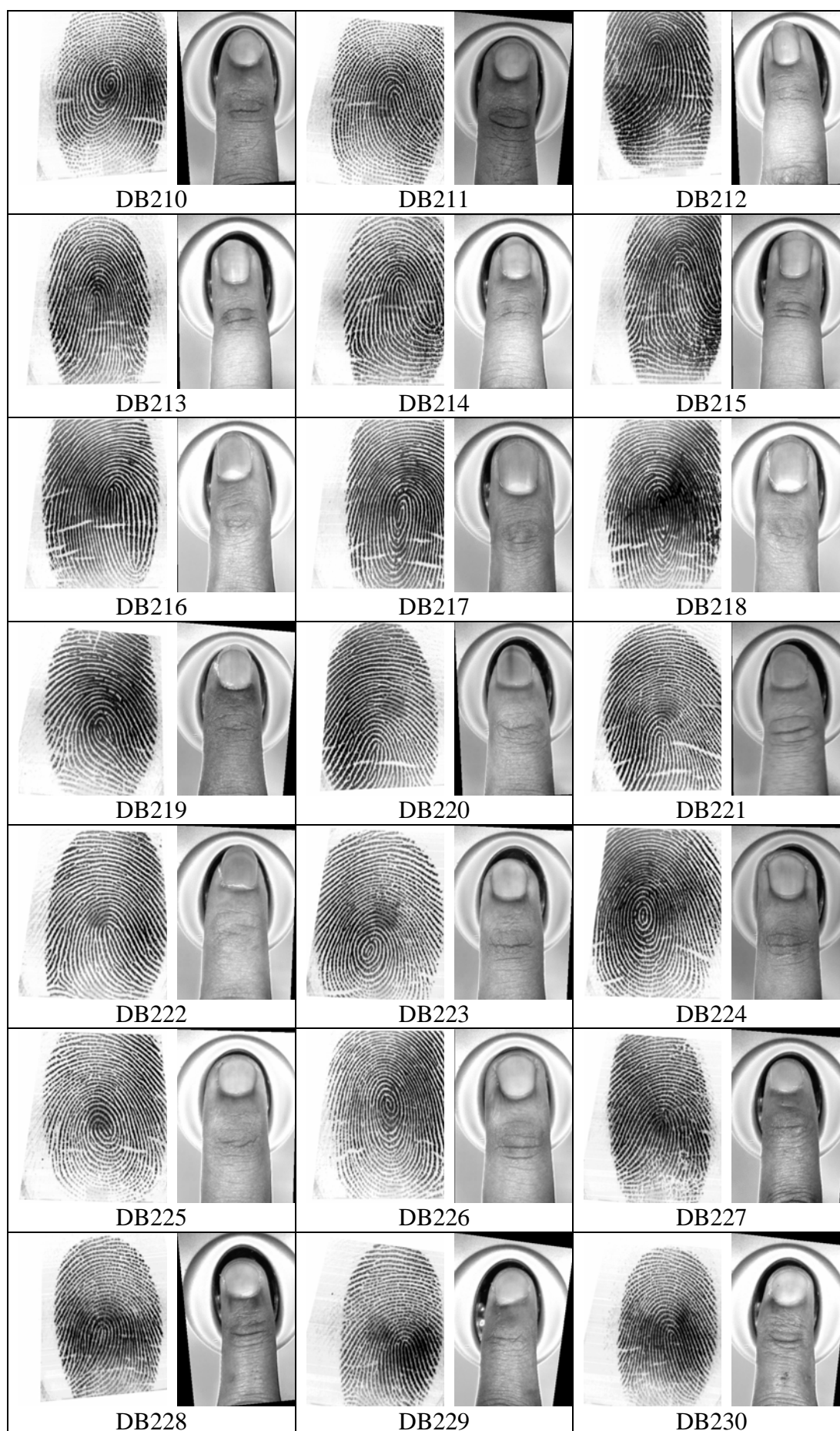


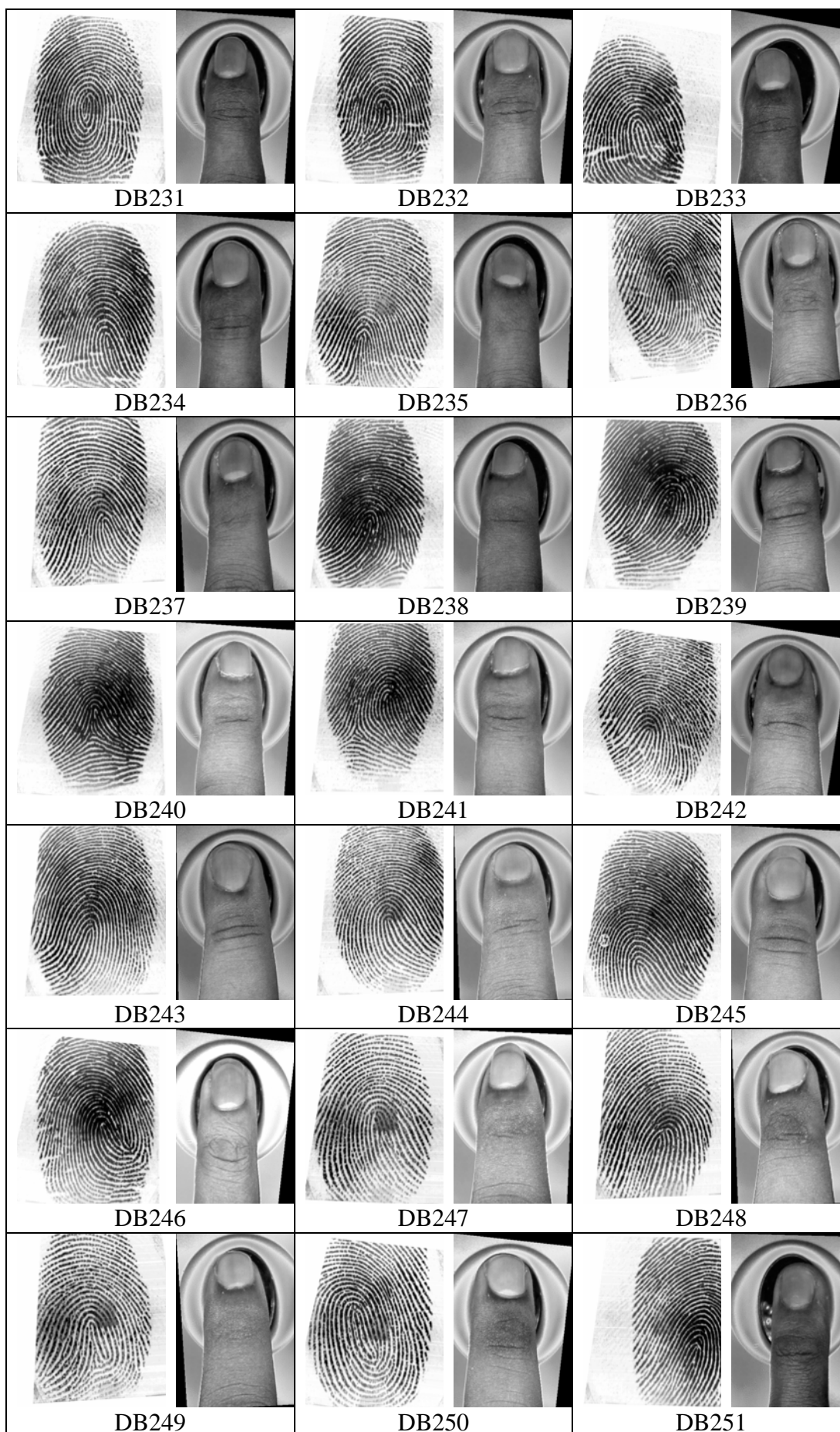


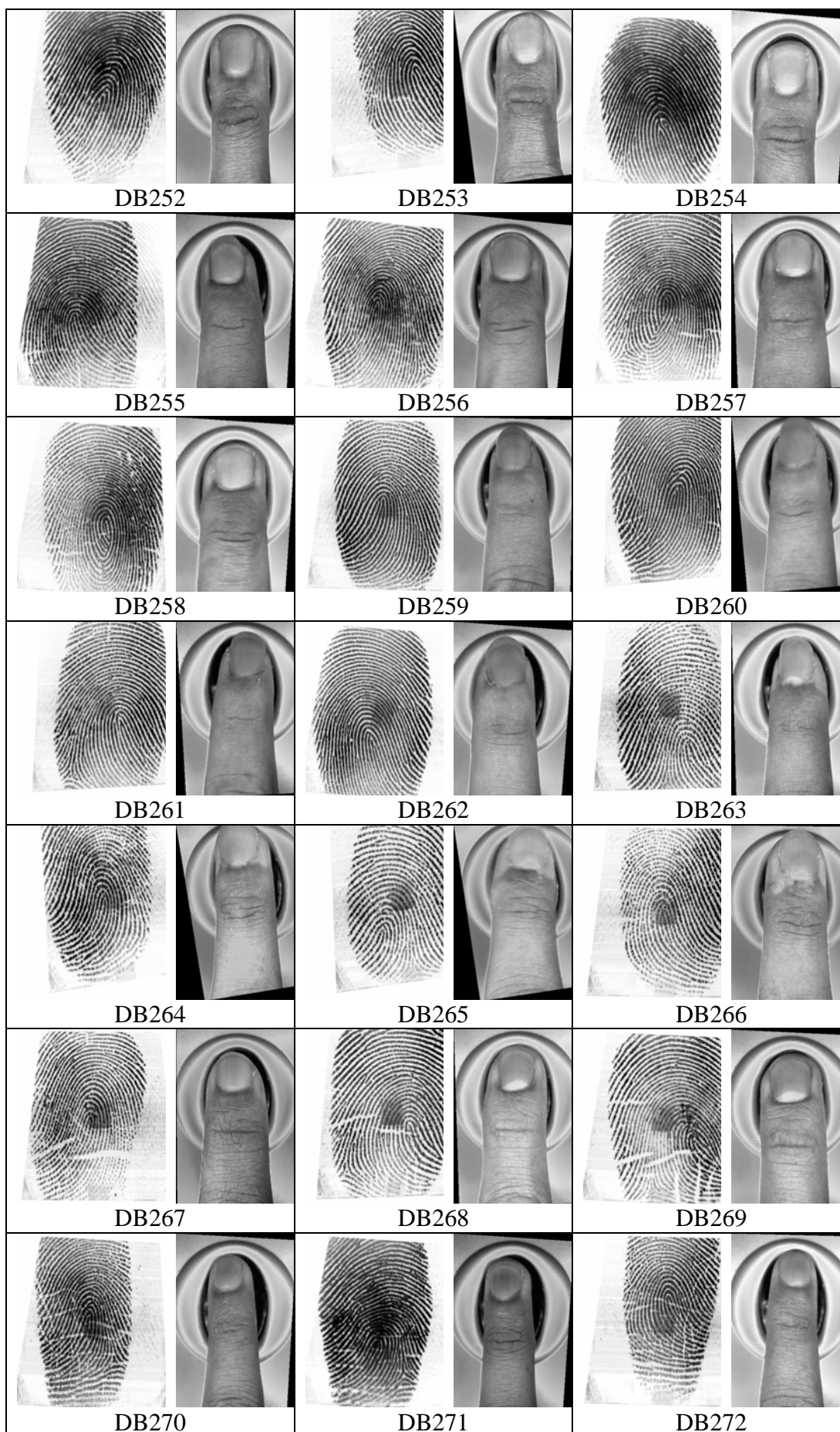


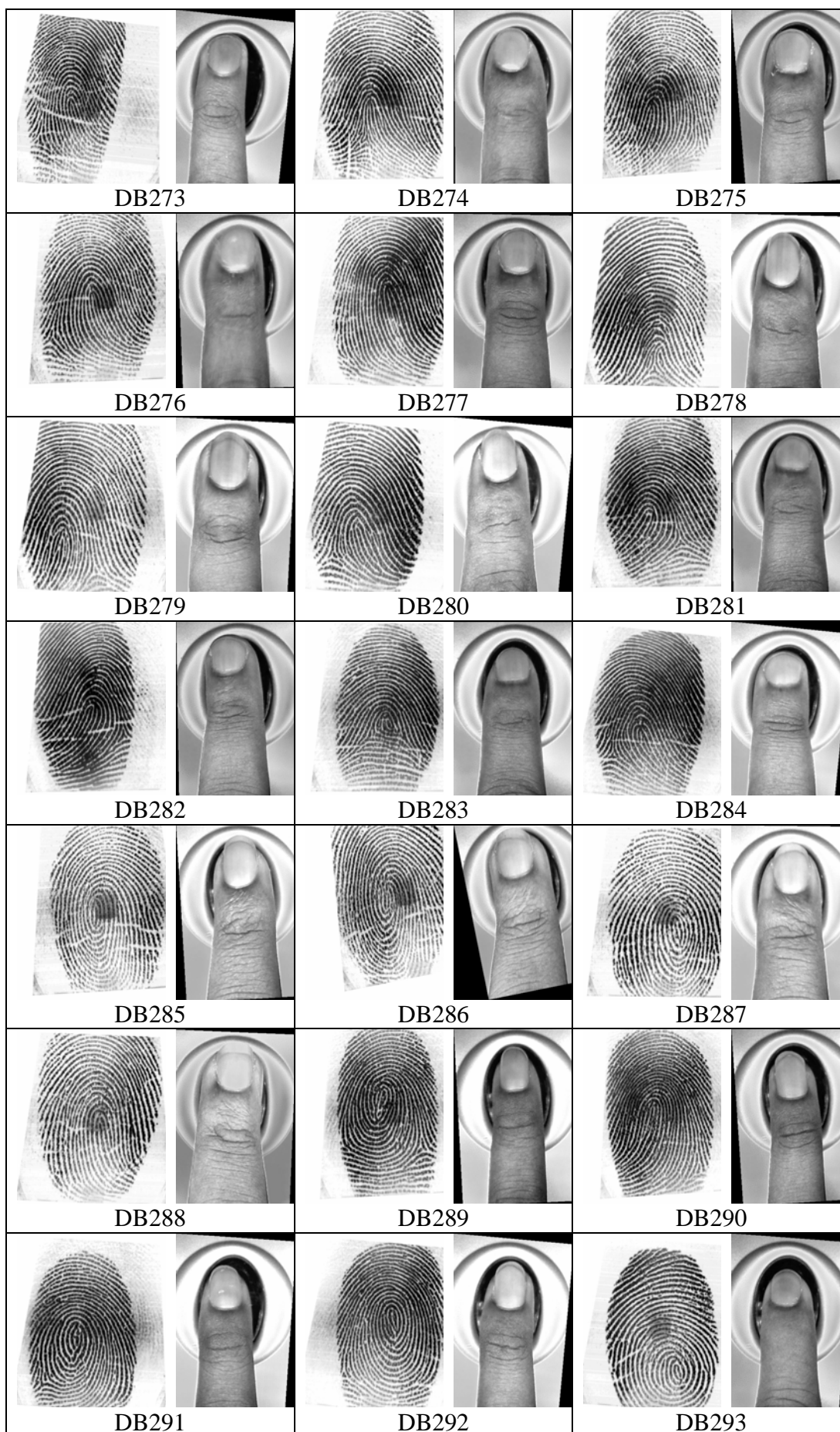


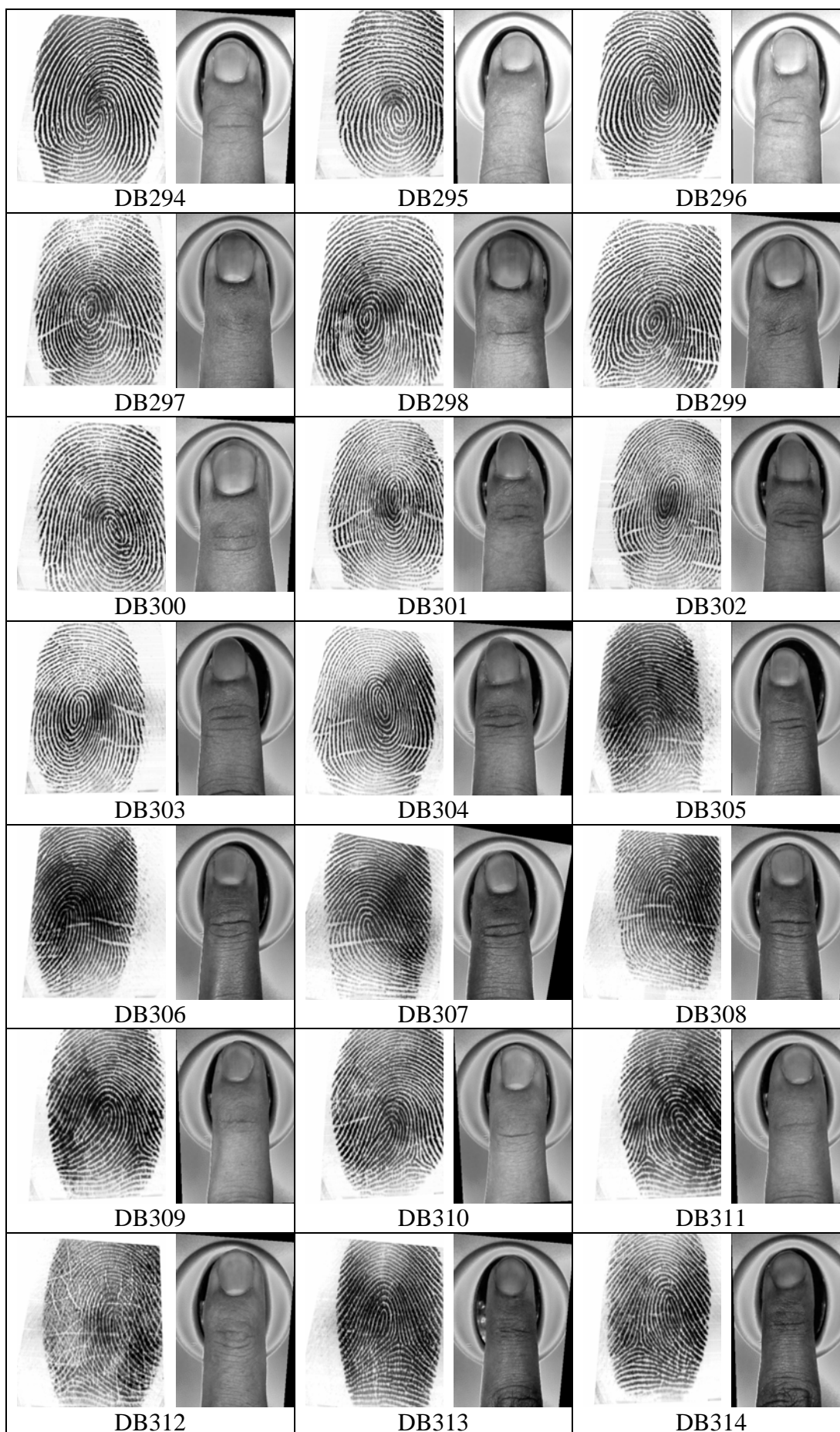


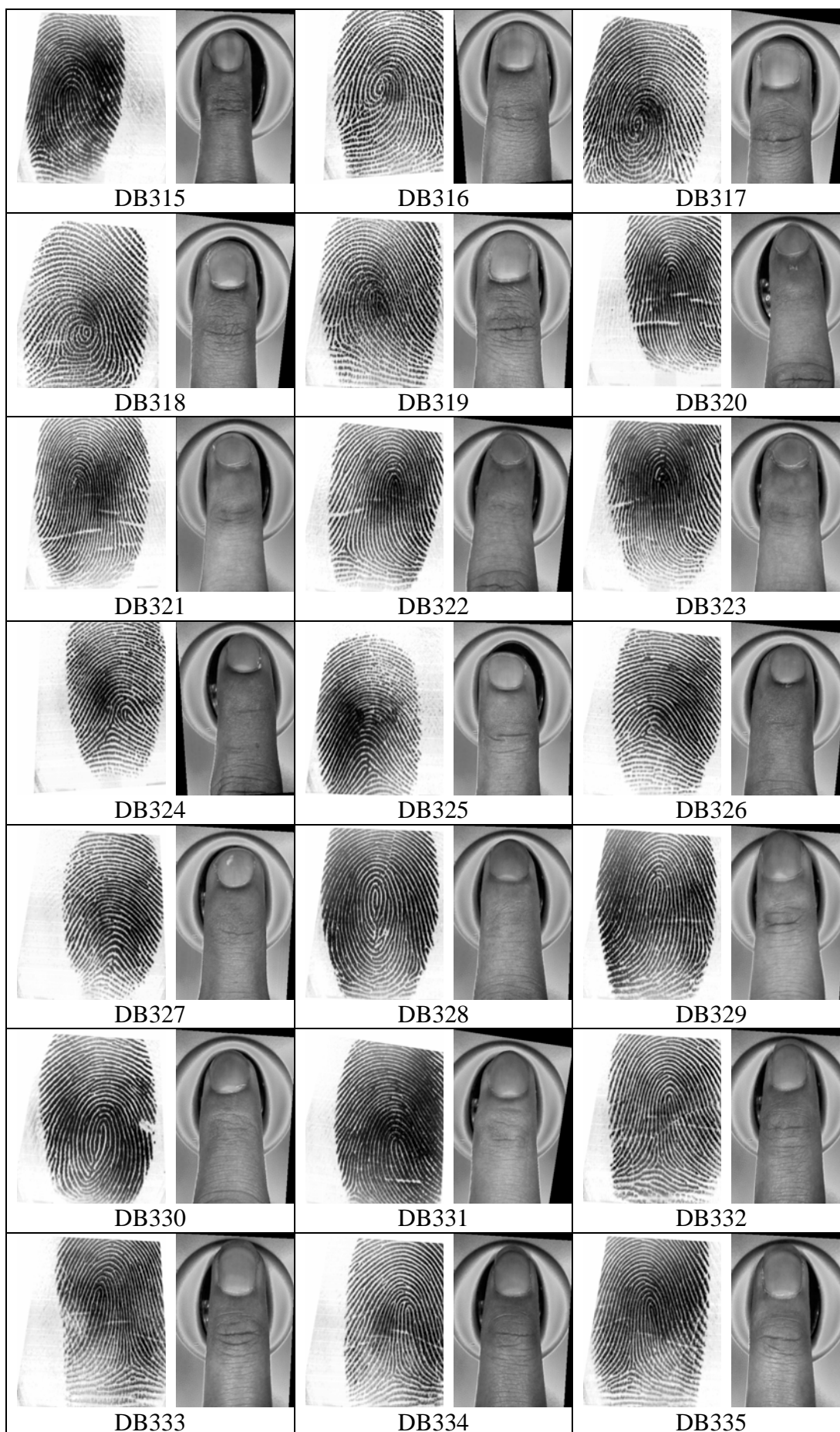


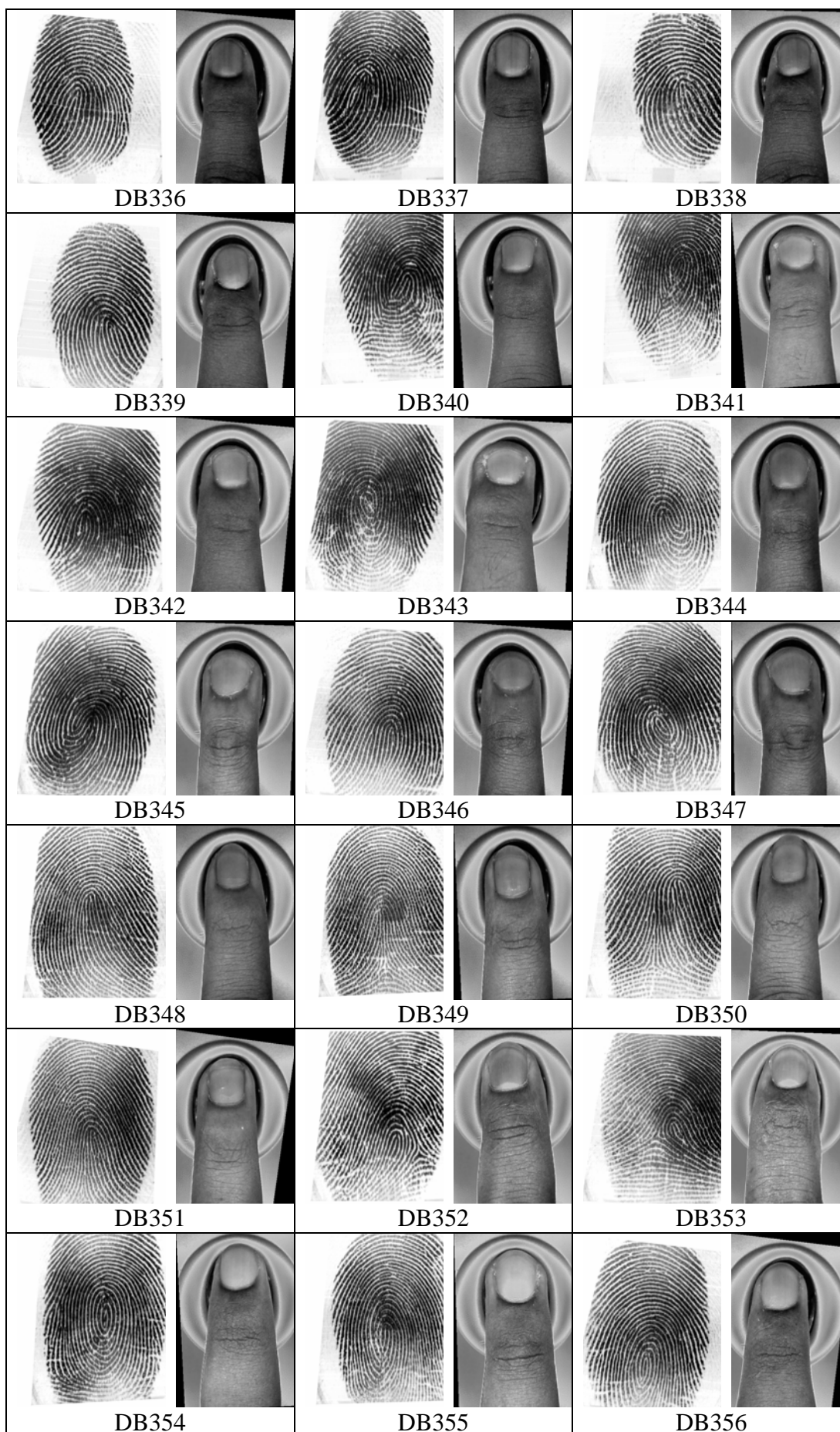


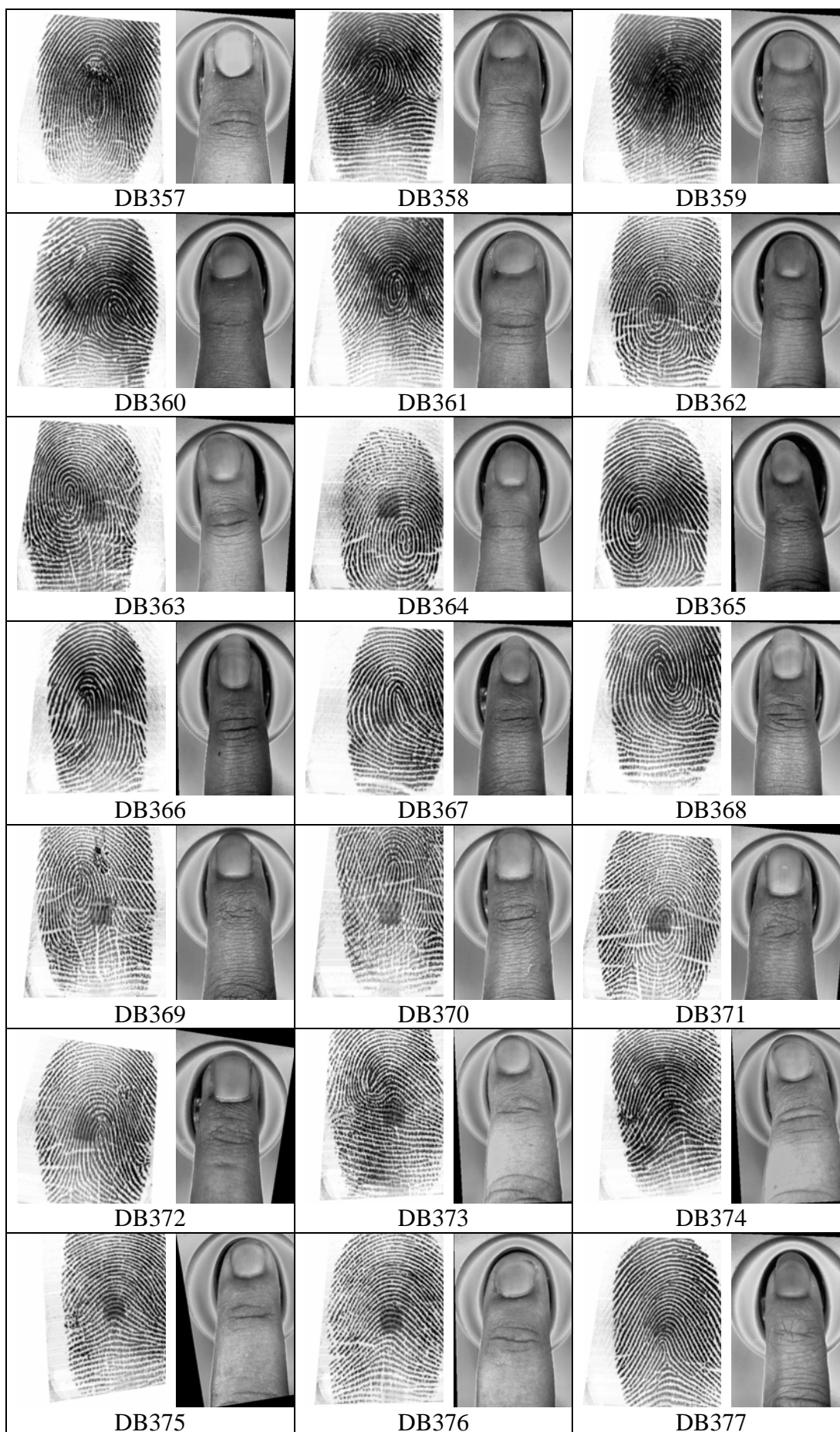


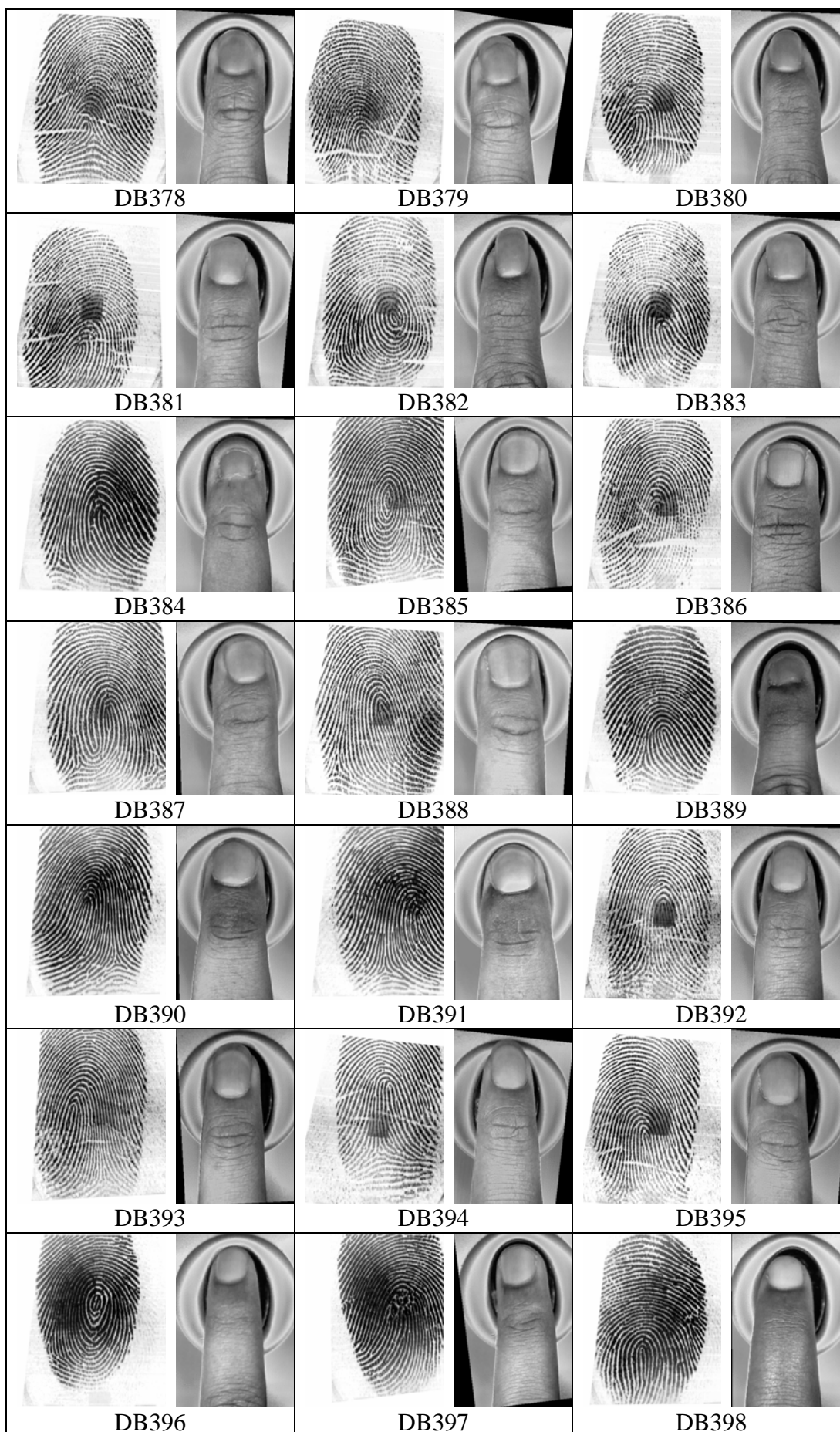


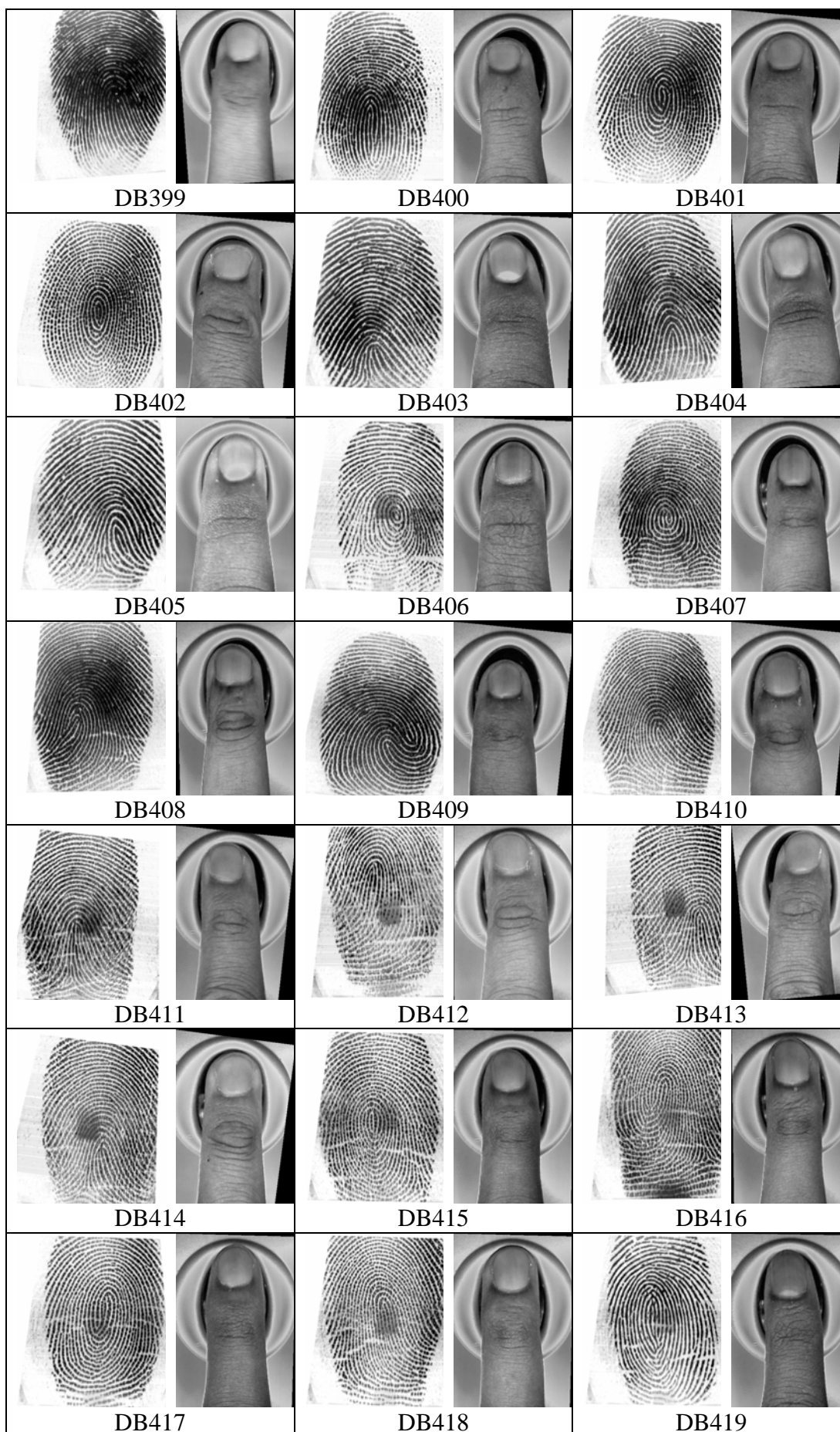


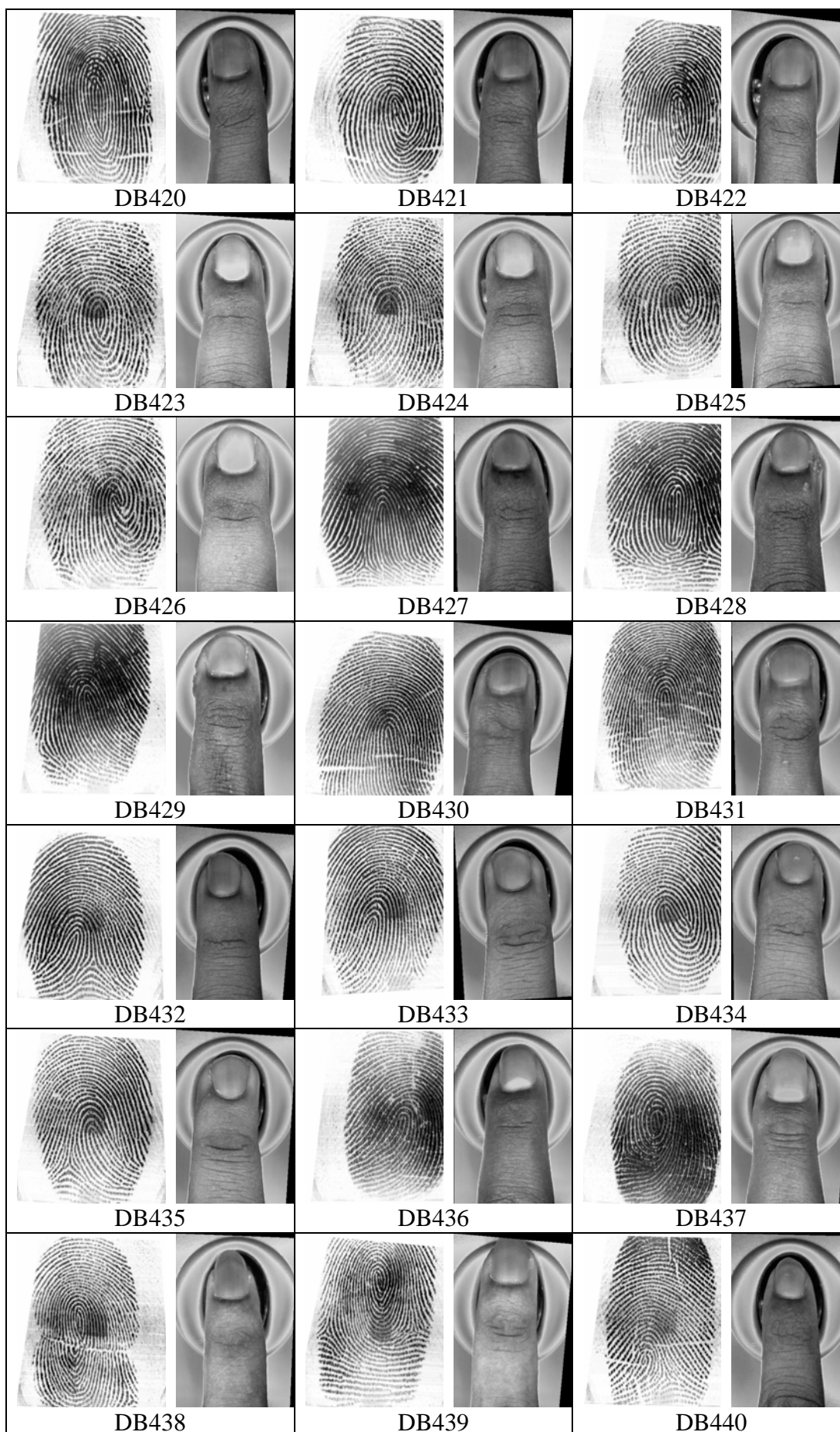


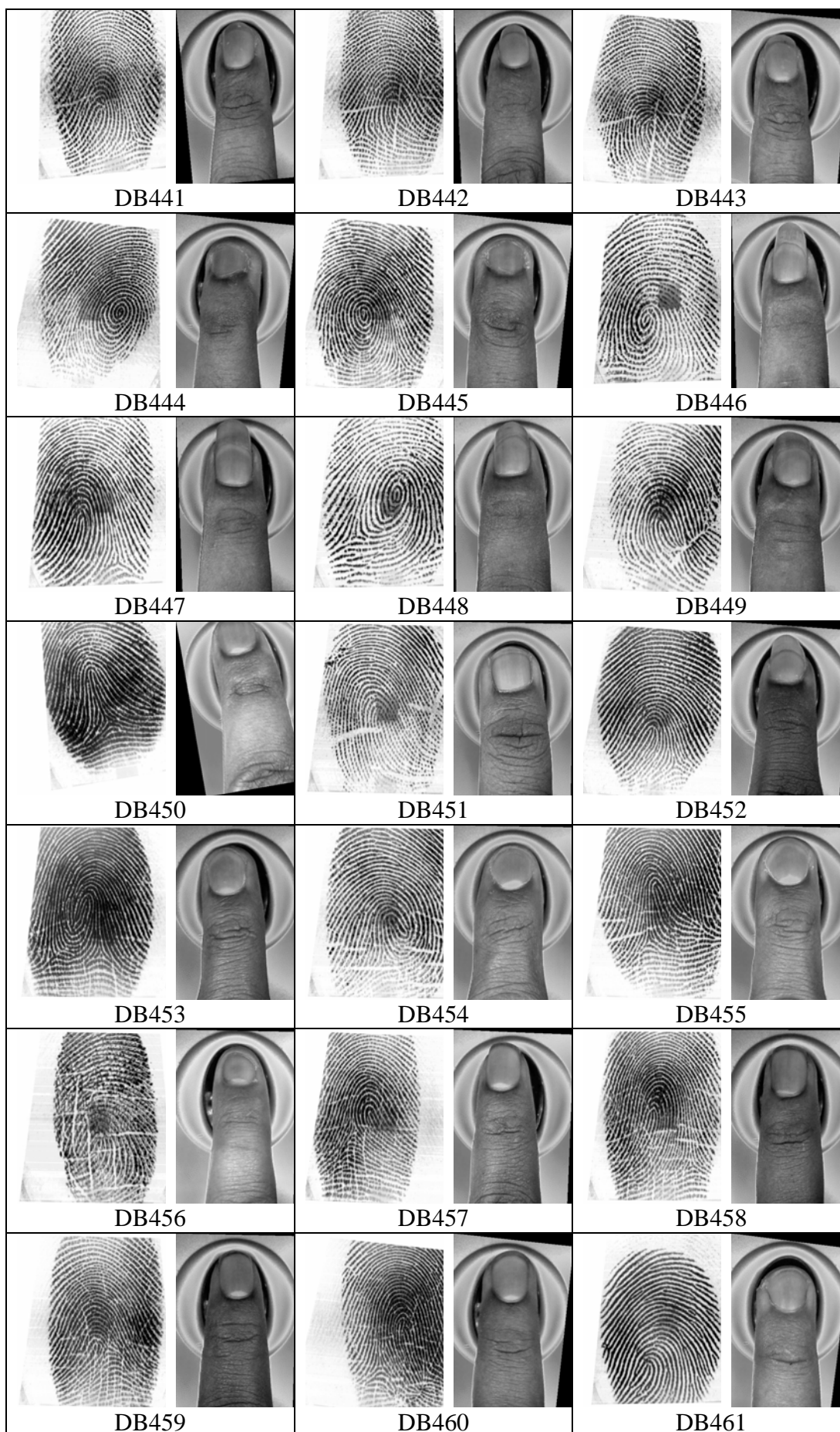


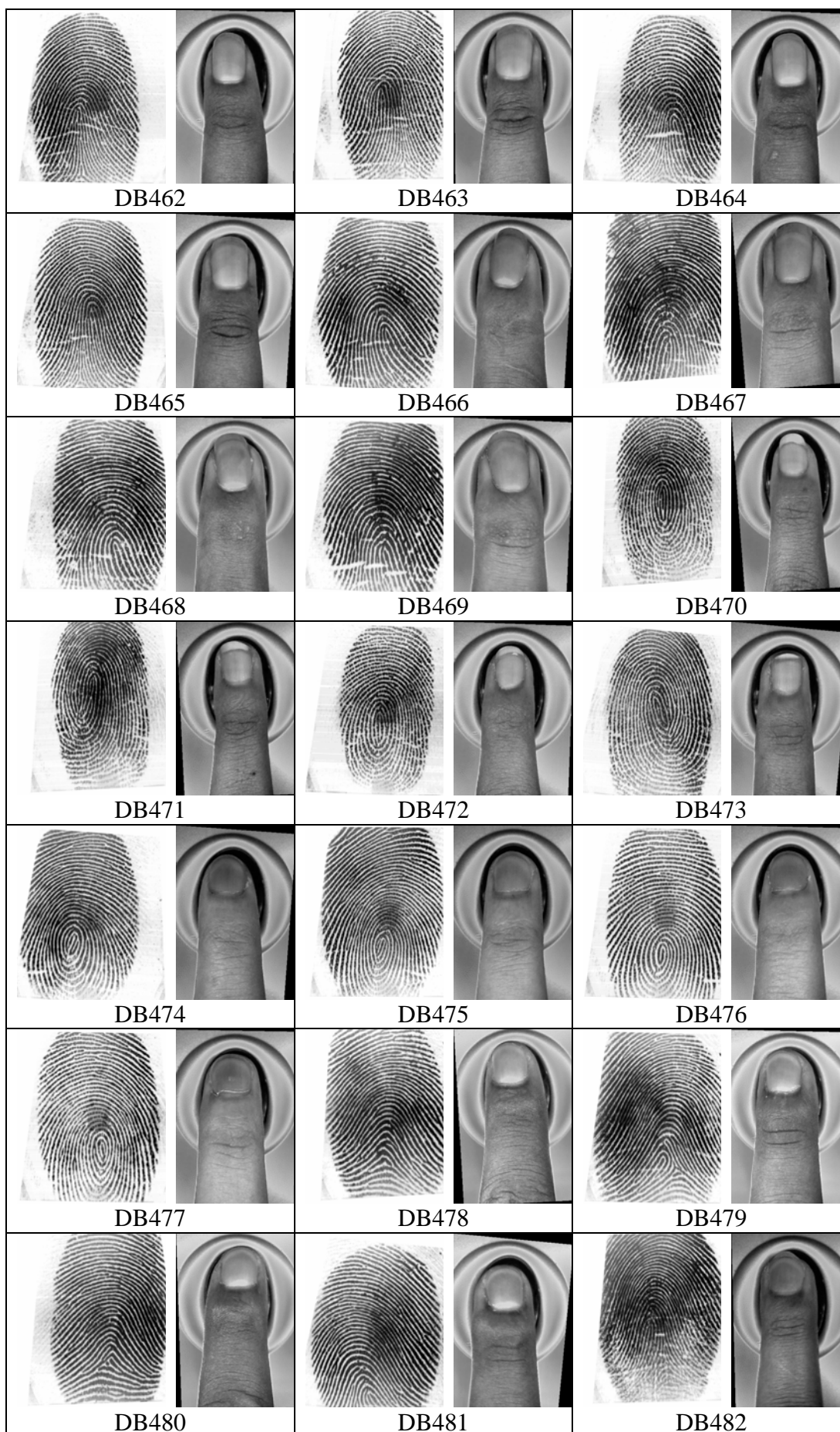


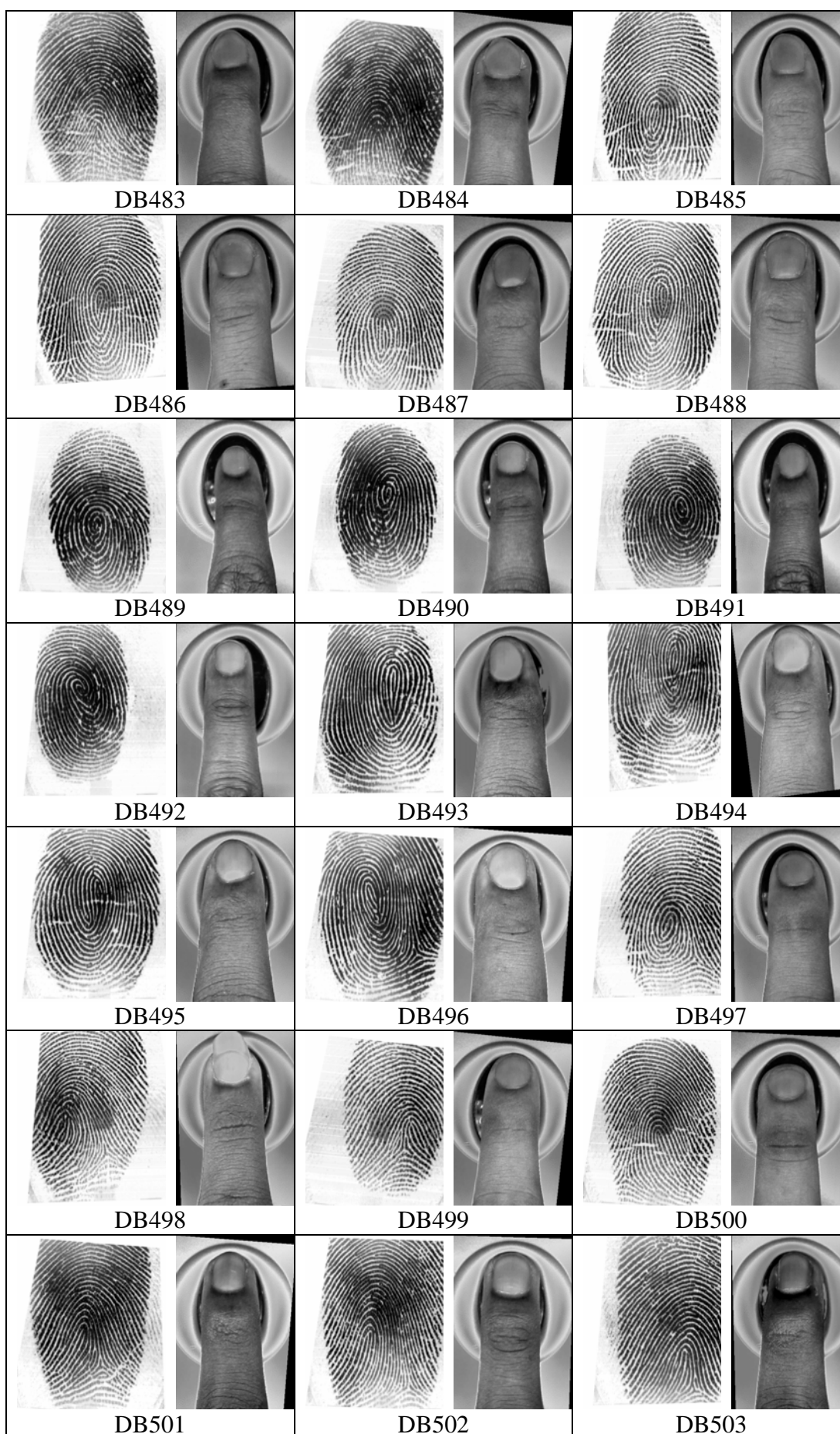


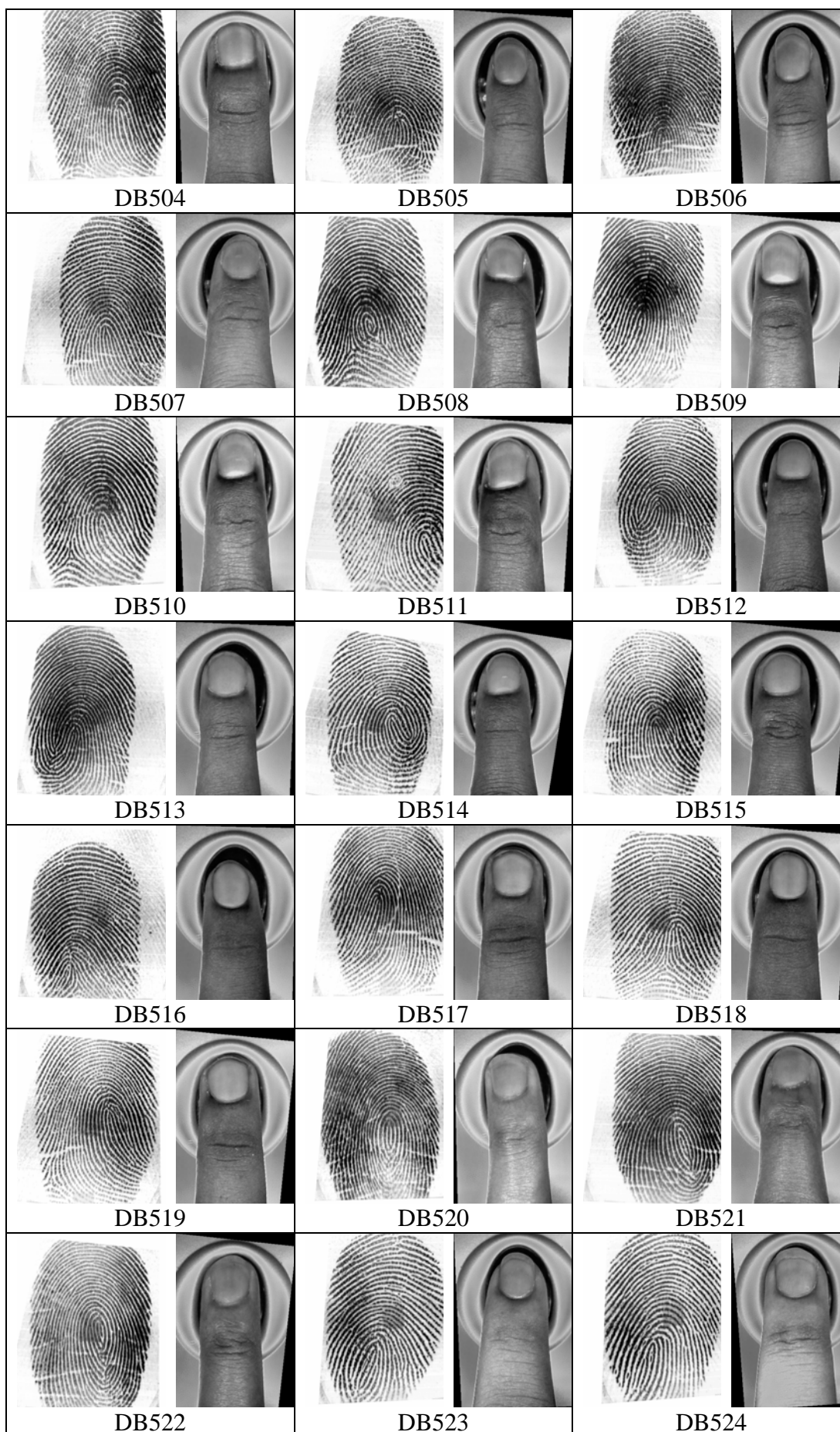


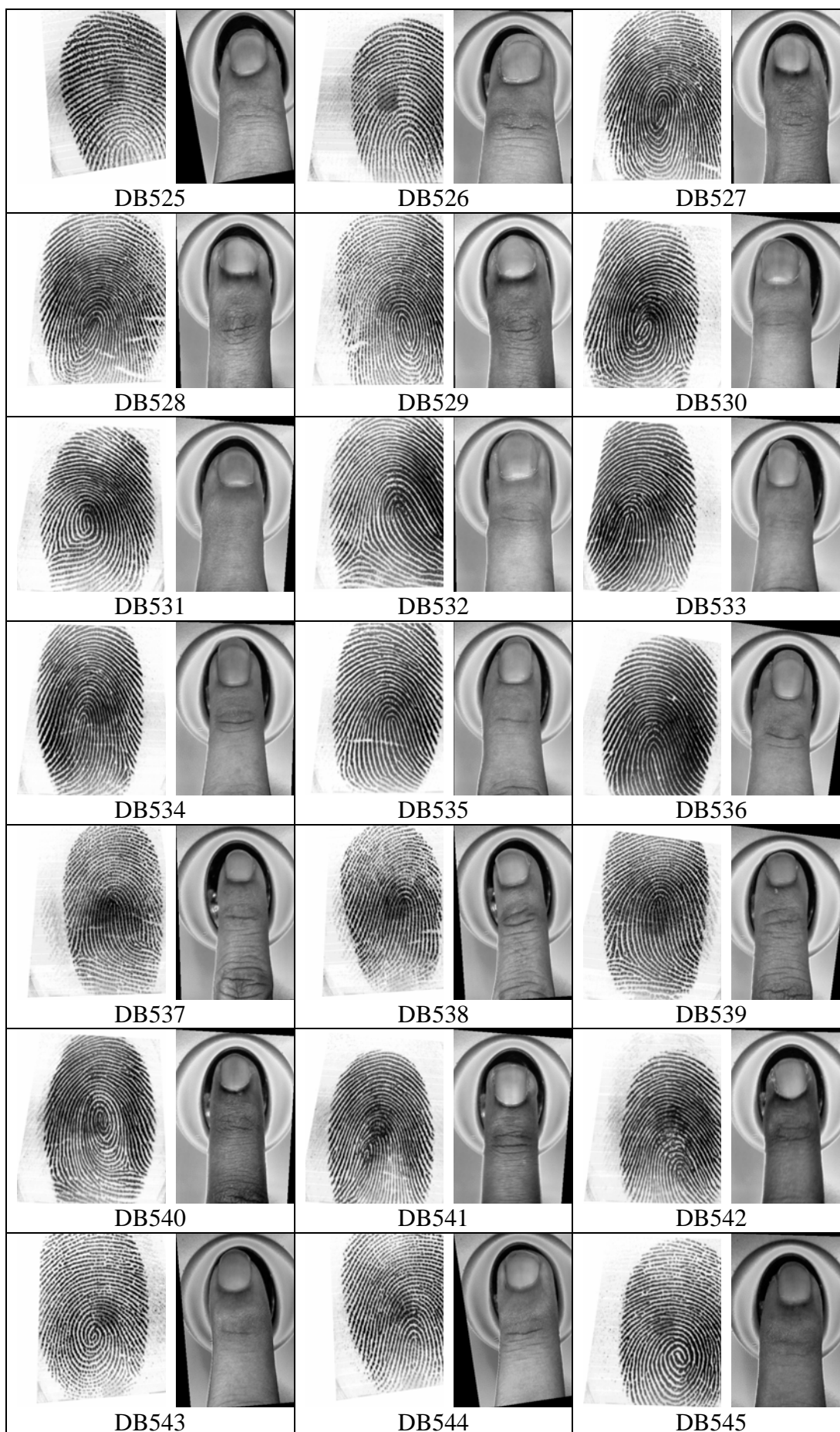


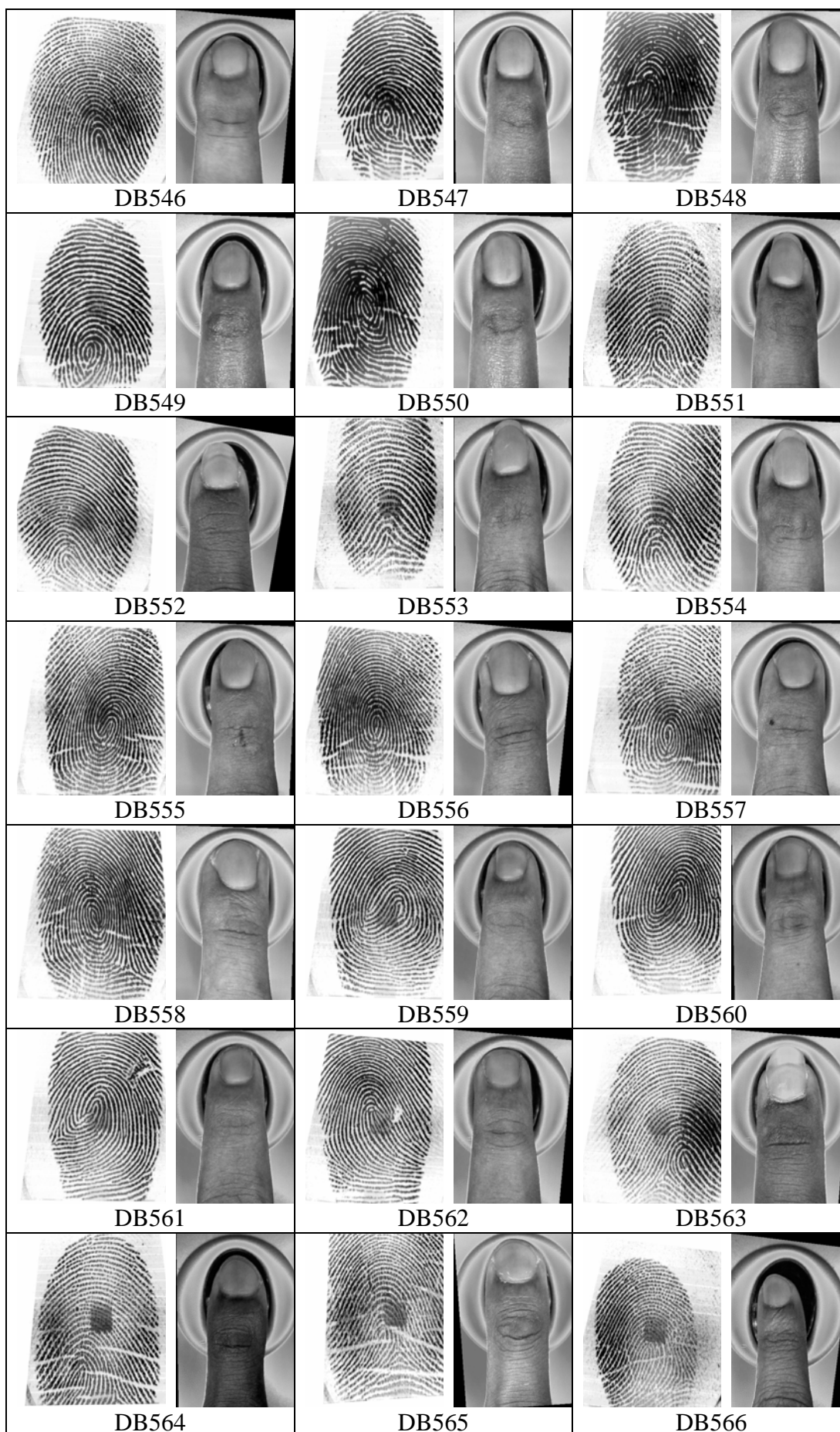


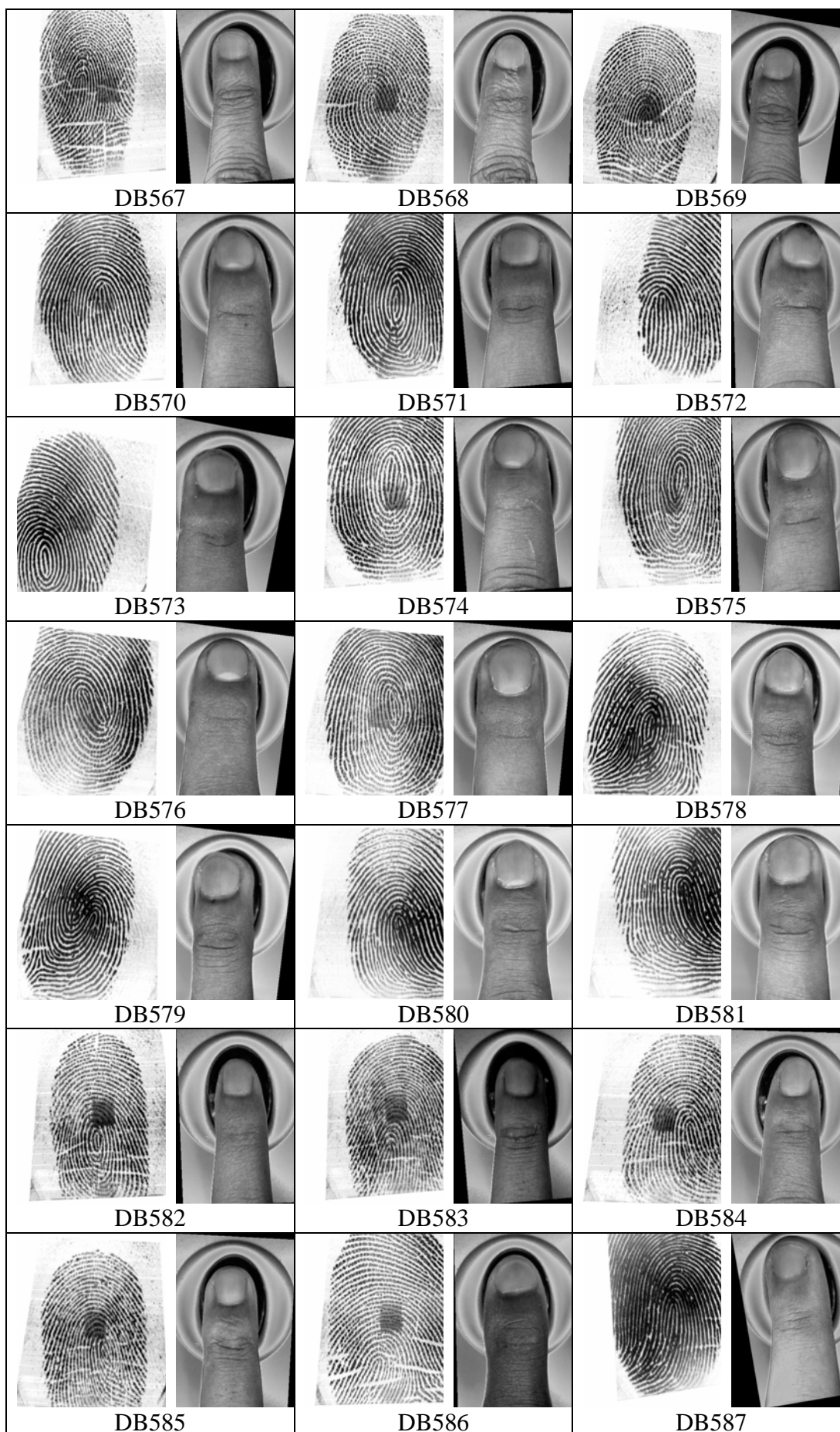


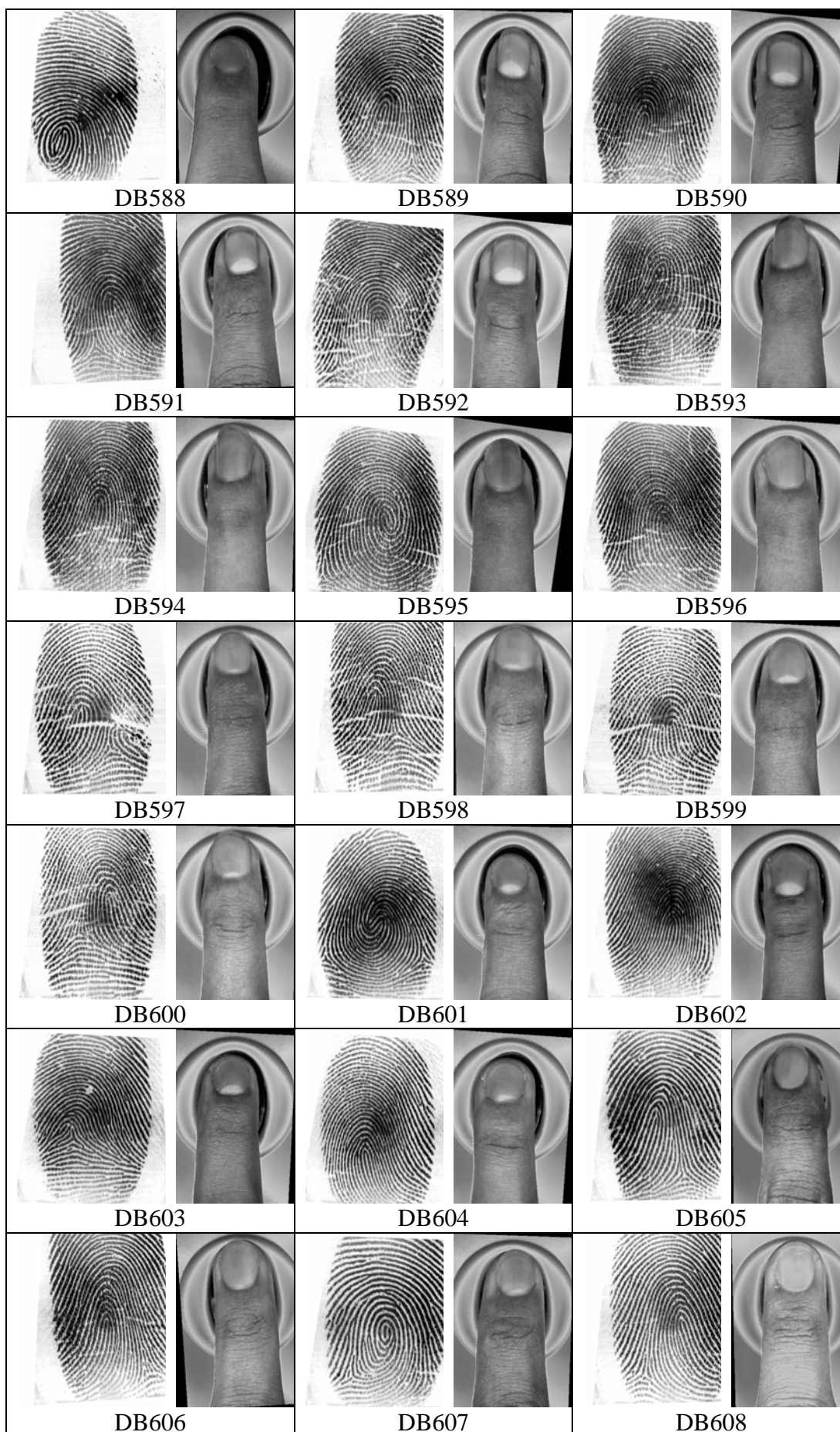


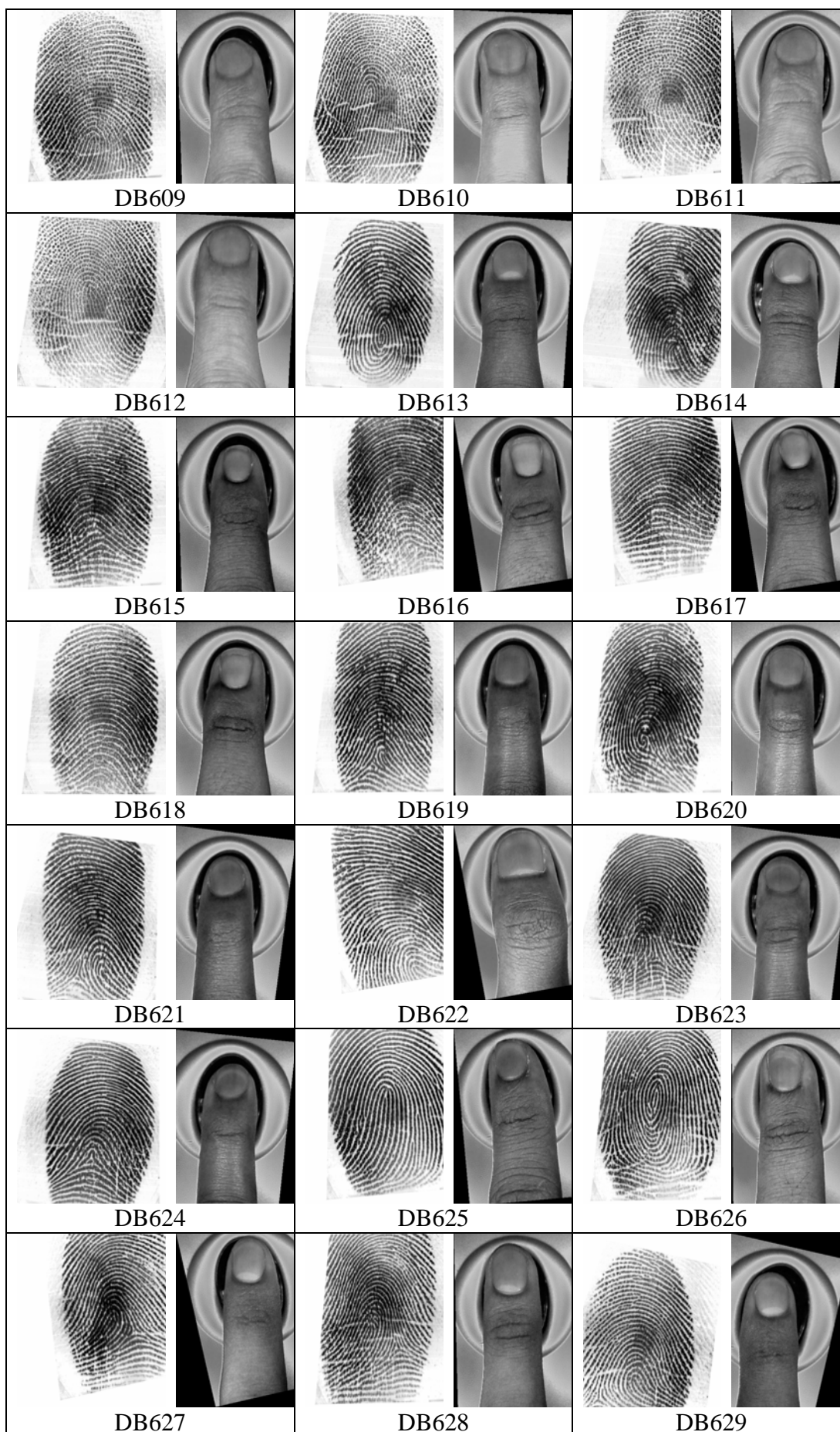


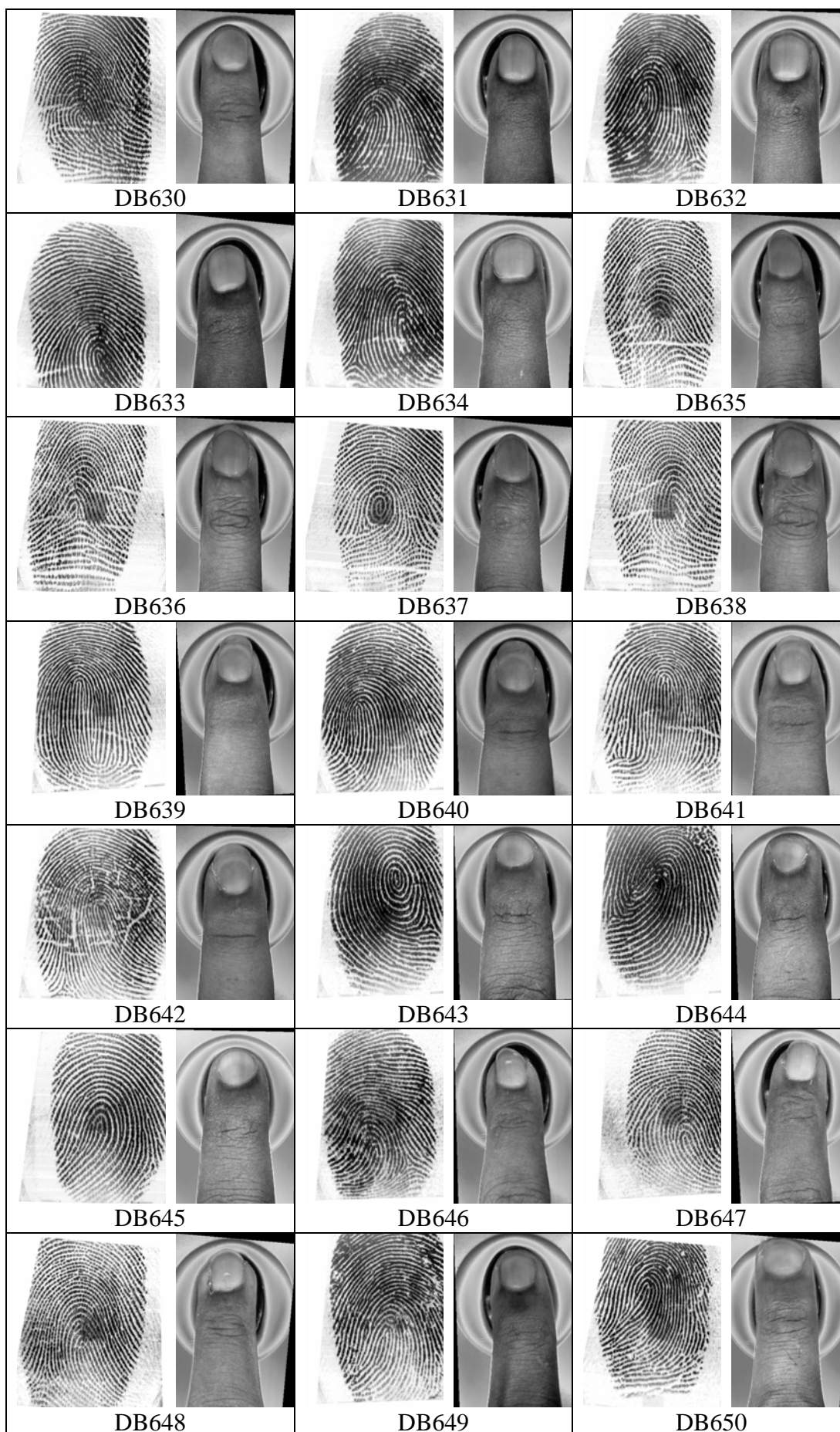


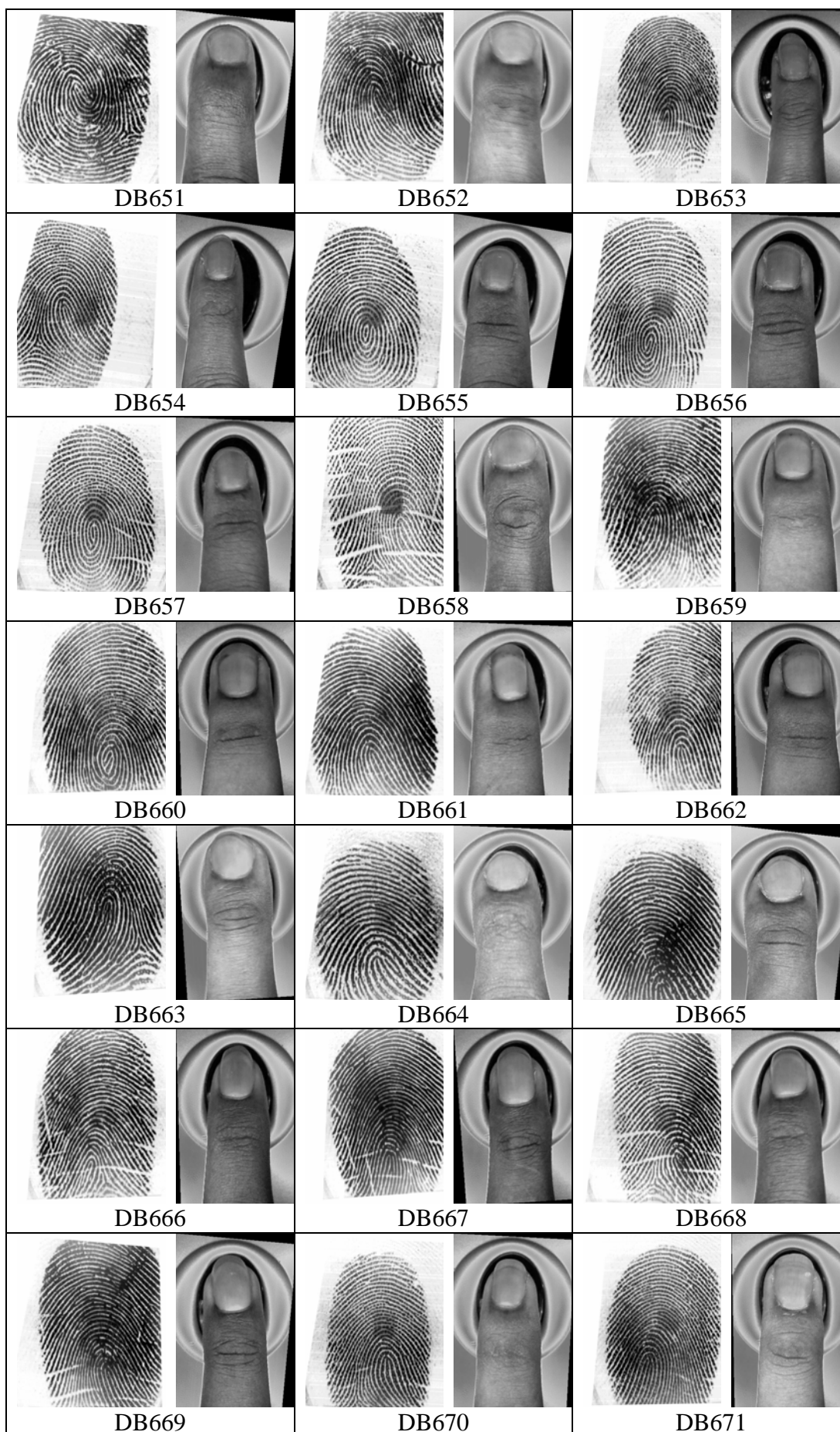


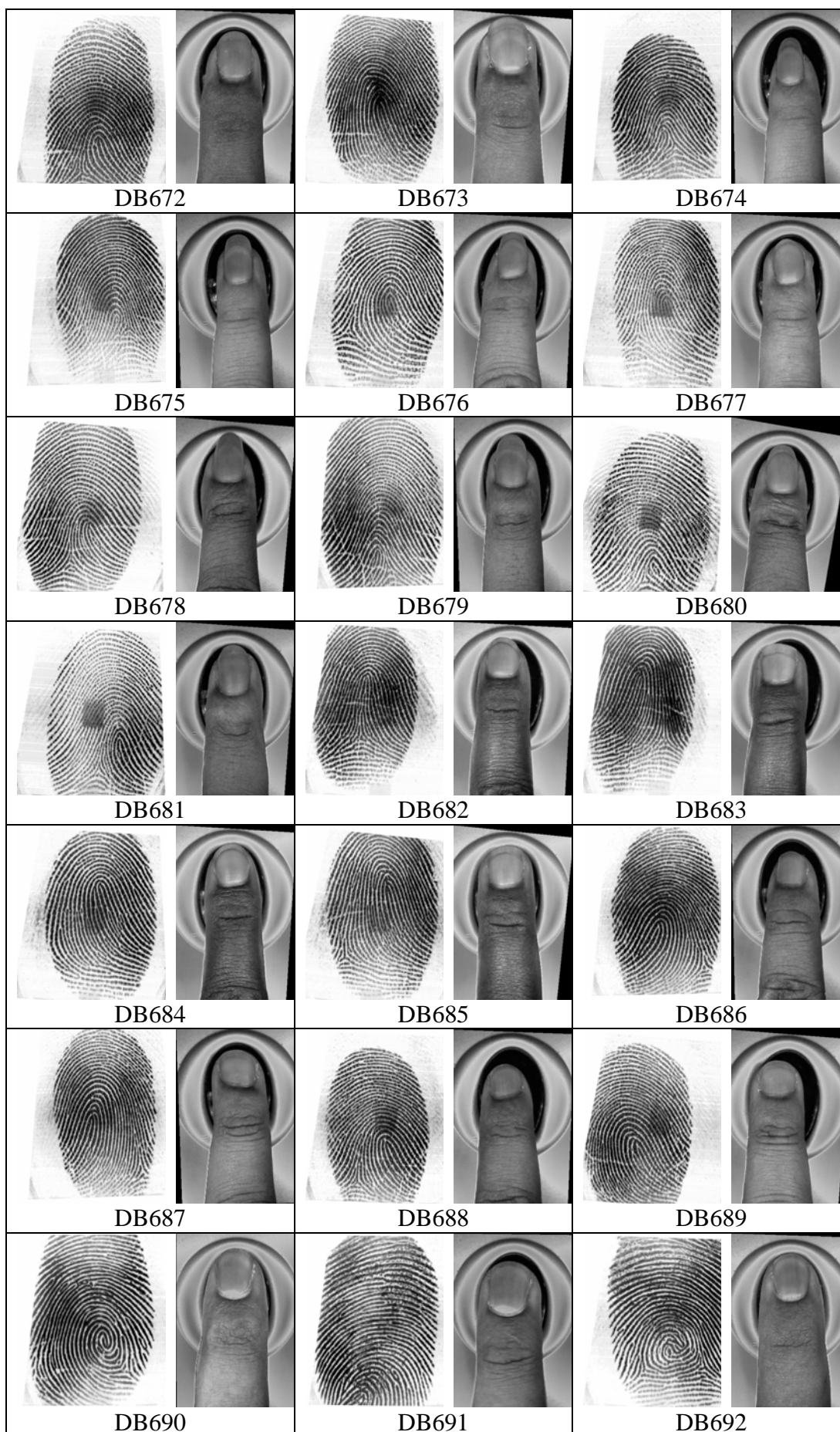


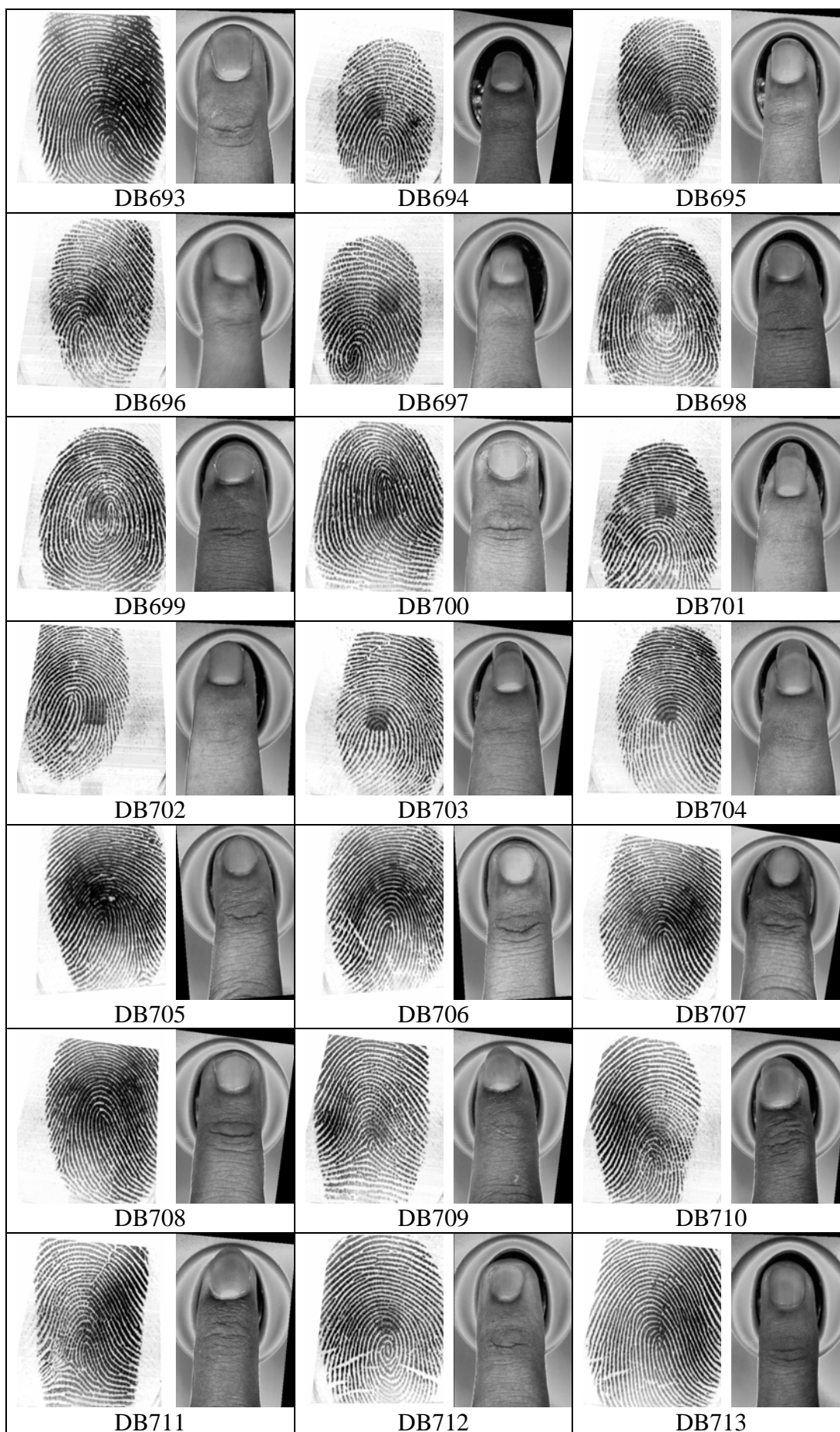


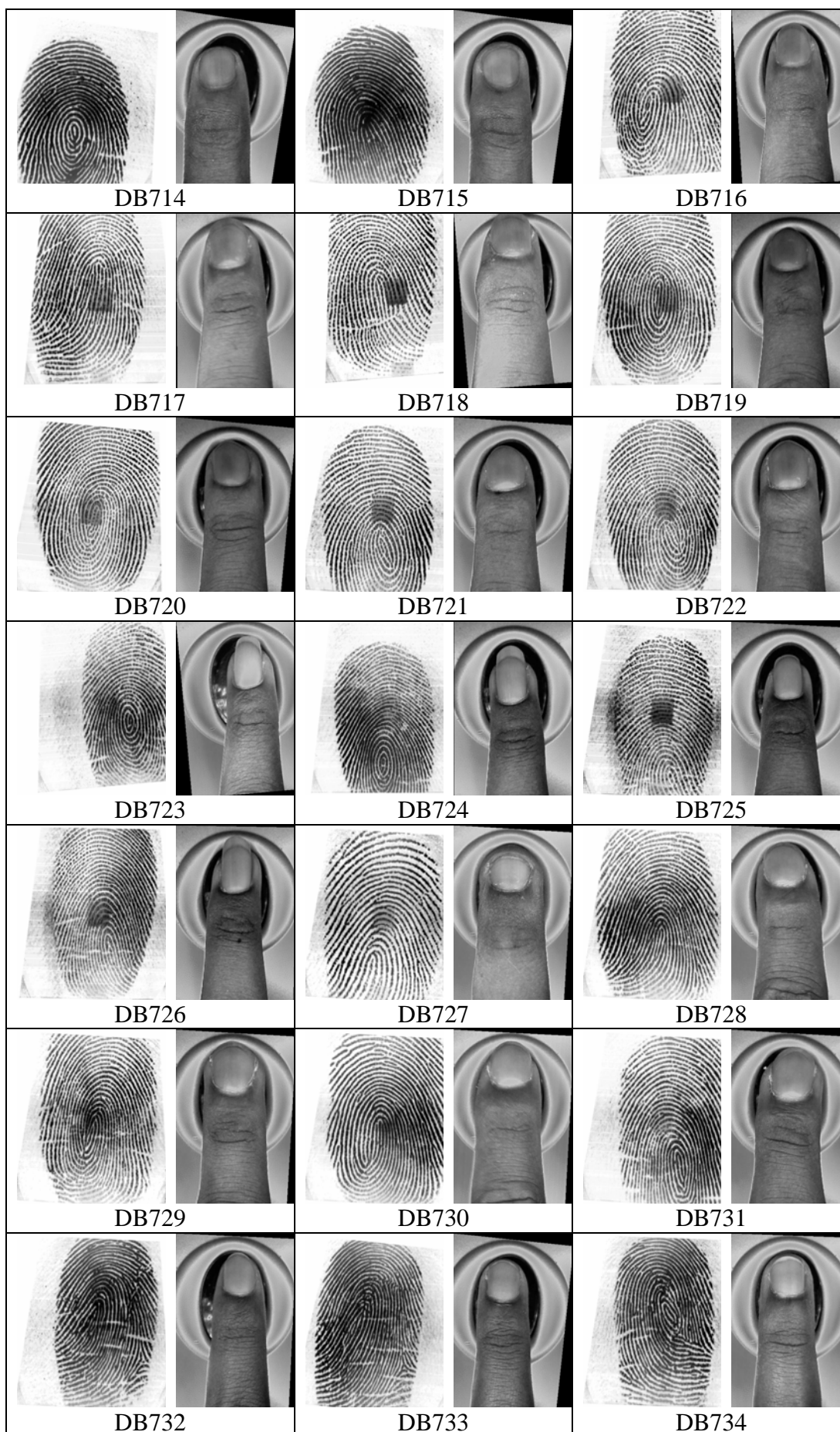


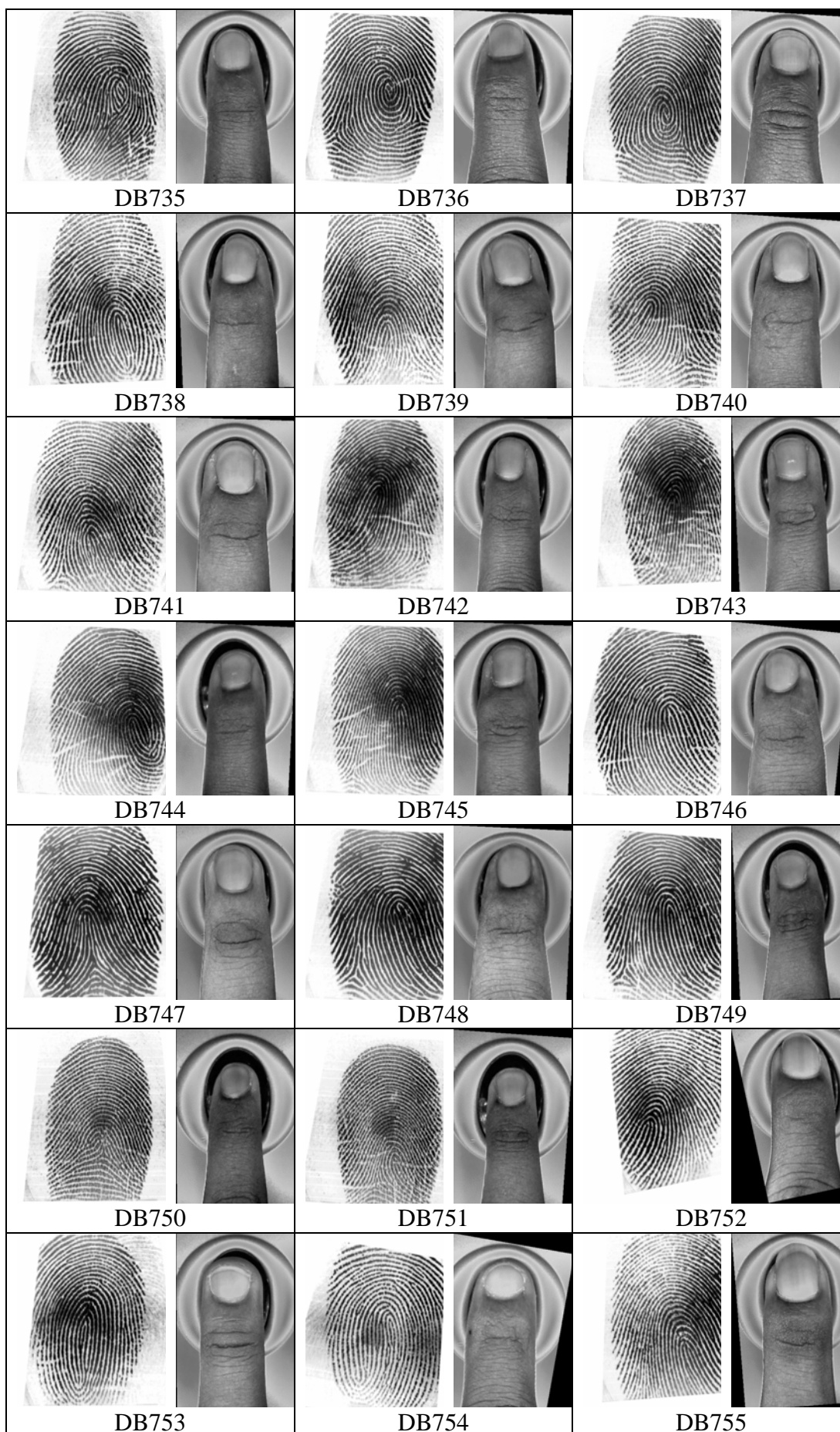


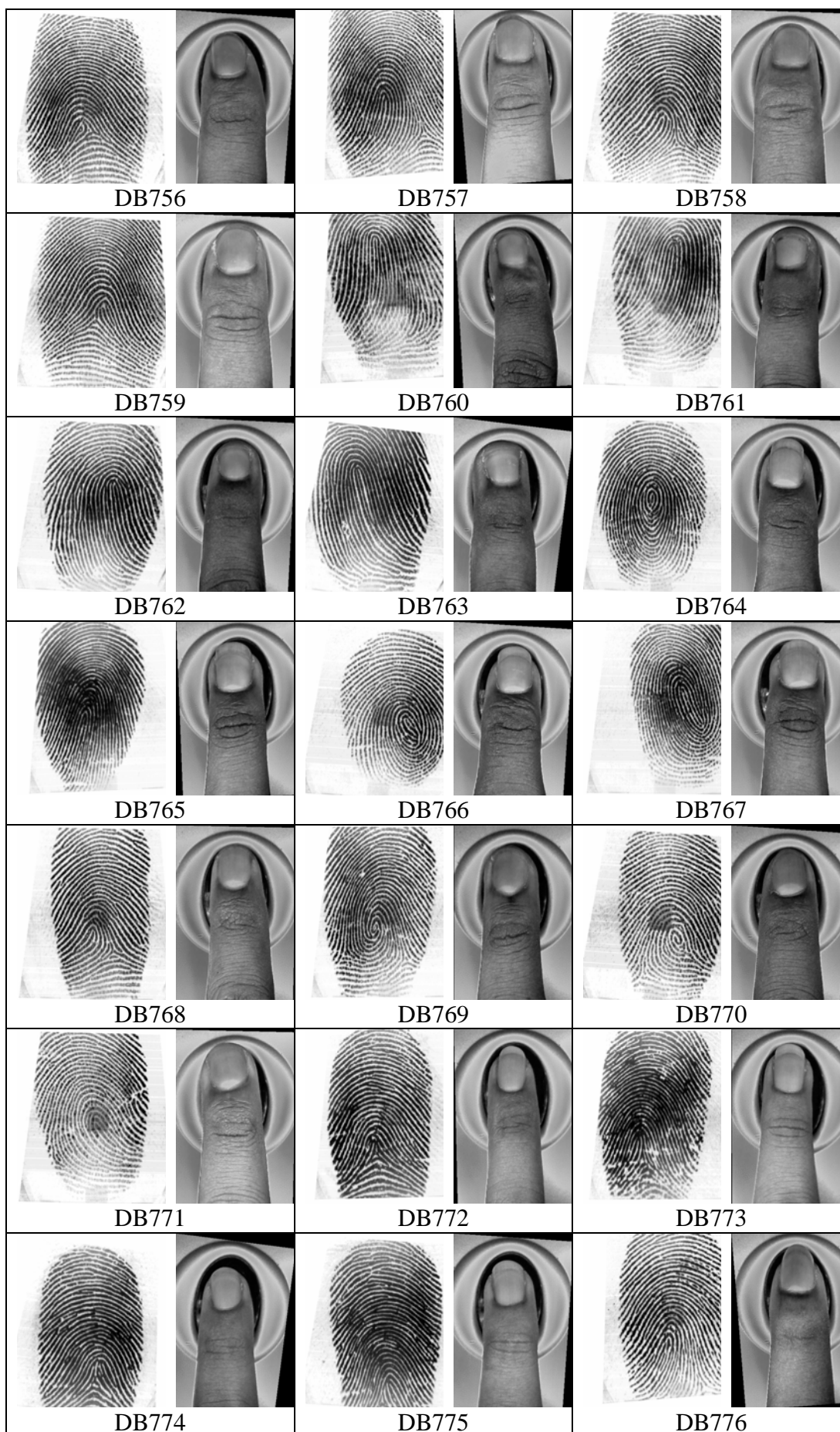


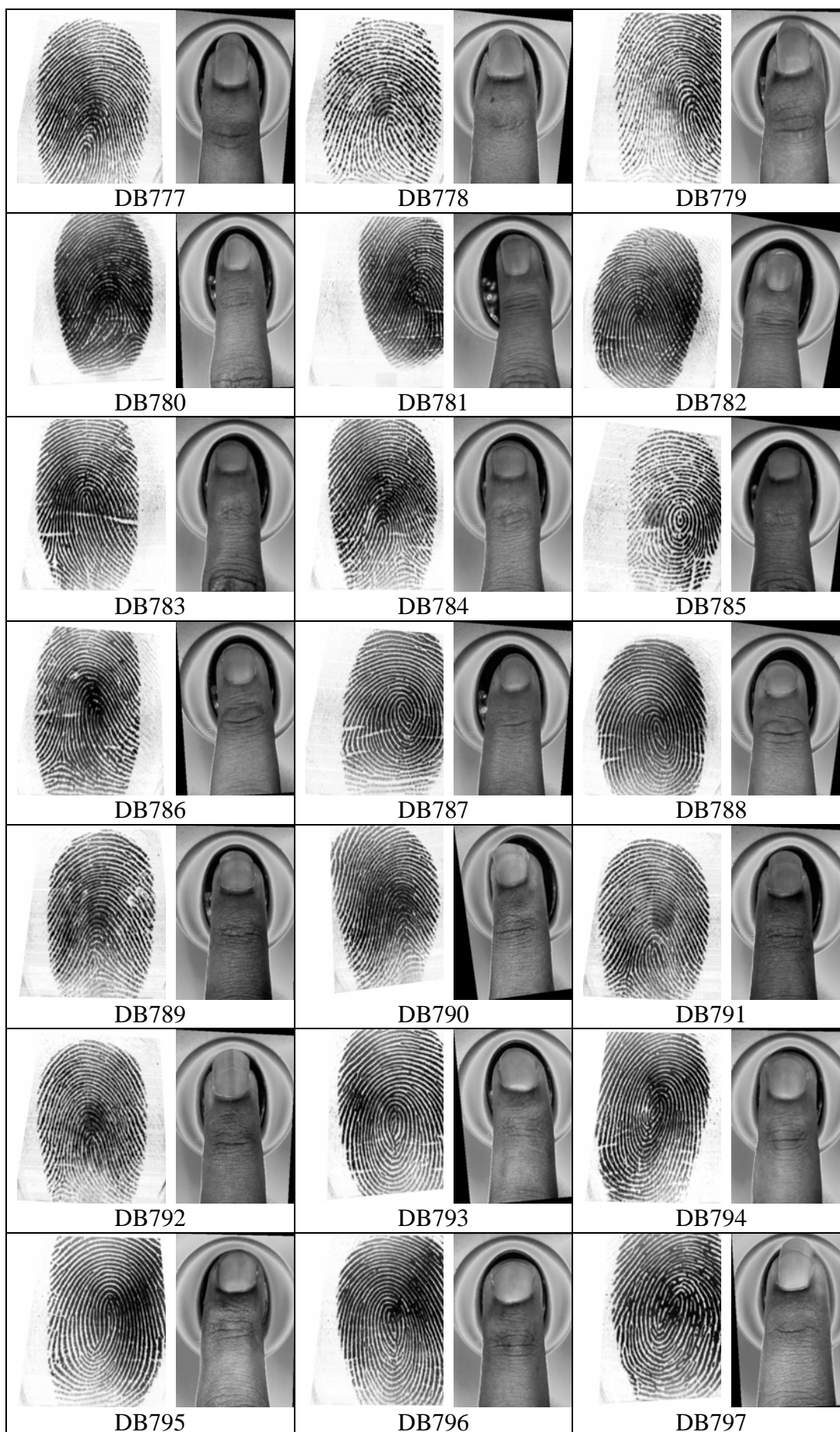


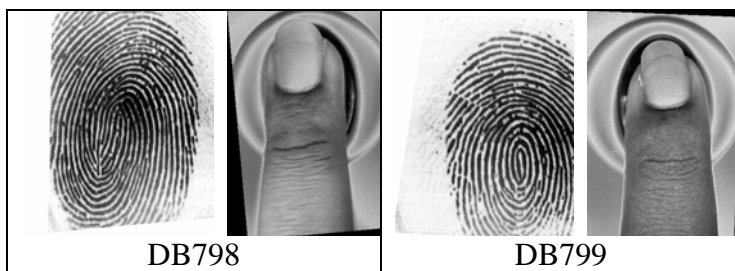












DB798

DB799

APPENDIX B

LIST OF PUBLICATIONS

1. Chaikan, P. and Karnjanadecha, M., "The use of Top-View Finger Image for Personal Identification," Proc. Of the International Symposium on Image and Signal Processing and Analysis: ISPA 2007. Istanbul, Turkey, pp. 343-346, September 27-29, 2007. (Impact Factor:2007:NA)
2. Chaikan P., and Karnjanaecha M., "A Reference Point Detection Algorithm for Top-View Finger Image Recognition," Proc. of the 5th International Symposium of Image and Signal Processing and Analysis: ISPA2007," Istanbul, Turkey, pp. 347-350, September 27-29, 2007. (Impact factor:2007:NA)
3. Chaikan, P., Karnjanadecha, M., "Integrating Fingerprint and Top-View Finger Image for Personal Verification," Proceedings of the 7th PSU Engineering Conference - PEC7, Faculty of Engineering PSU, Hatyai, Songkhla, Thailand, pp. 157-160, May 21-22, 2009. (Impact Factor:2009:NA)
4. Chaikan, P., Karnjanadecha, M., 2010, "Person Recognition using Fingerprints and Top-View Finger Images," *Songklanakarinn Journal of Science and Technology*, Vol. 32, No. 1, pp.71-79, January-February 2010. (Impact Factor:2010:1.0)



ISPA 2007

CD version

Proceedings of the 5th International Symposium on

Image and Signal Processing and Analysis

Istanbul, Turkey, September 27-29, 2007

M. Petrou, T. Saramäki, A. Erçil, S. Lončarić (Eds.)

IEEE 07EX1763C
ISBN 953184117-7
ISSN 1845-5956

Technical support :
secretariat@isispa.org
tel: +385 1 6129911
fax: +385 1 6129652



The Use of Top-View Finger Image for Personal Identification

Panyayot Chaikan, Montri Karnjanadecha

Department of Computer Engineering, Faculty of Engineering,
Prince of Songkhla University, Thailand
{panyayot, montri}@coe.psu.ac.th

Abstract

This paper describes a feasibility study for using a top-view finger image to increase the accuracy of fingerprint recognition without adding any new user operations. A CCD camera captures a top-view finger image while the user is touching a fingerprint sensor, and the acquired gray scale image is preprocessed to enhance the edges, the skin furrows, and the nail shape before the image is filtered by a bank of Oriented-Filters. A square tessellation is applied to the filtered image to create a feature map, called a NailCode. The NailCode is employed in the matching process by employing a Euclidean distance computation. The experiment reveals that personal identification accuracy using NailCode feature is 96.57%. It is recommended that NailCode is employed in conjunction with fingerprint for multimodal biometric [1] systems will increase the identification accuracy.

1. Introduction

Personal identification using fingerprint is ubiquitous because fingerprints of each person are unique and time invariant [2]. However, fingerprint recognition remains a complex, challenging problem, with the accuracy of fingerprint recognition having reached a limit which is difficult to improve. One approach is multimodal biometrics which combines more than one human feature for recognition purposes. For example, Hong [3] employs the face in conjunction with fingerprints, Jain [4] uses speech, face and fingerprints, and Marcialis [5] utilizes two different types of fingerprint sensor. All of these methods have the same purpose: augmenting recognition accuracy which is otherwise limited by using only fingerprint detection. Their main drawback is that the additional features increase the complexity of the user interaction.

Our approach nests on the idea that the skin wrinkles and furrows on the top of a person's fingers are different, and that this information could be easily captured with a small camera above the fingerprint

sensor, as shown in Figure 1. The additional image information should increase the accuracy of the system without adding any extra tasks to the user. The question is "How well a top-view fingers image can identify a person?". The contribution of this paper centers on a feasibility study of using a top-view finger image only for personal identification. We will combine it with fingerprint detection in a multimodal biometric system in the future.

The rest of the paper is organized as follows: section 2 starts with a brief overview of personal identification using a top-view finger image, followed by the detail explanation of each step. Section 3 gives the detail of feature extraction. Section 4 presents the matching process. In section 5, the experimental results are given, and section 6 concludes the paper.

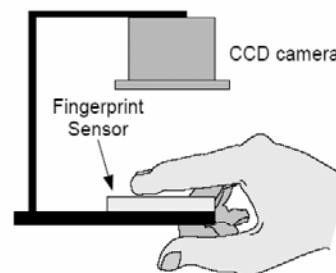


Figure 1. Top-view Finger Image Identification.

2. Personal Identification using a Top-View Finger Image

Our NailCode generation and matching process can be summarized in the following steps:

- 1) The acquired grey-scale top-view finger image, called F_G , of size 326×480 pixels is smoothed using a Gaussian filter.
- 2) Adaptive thresholding [6] is performed on the resulting image from step 1 to get a binary image, F_T . The block size of the adaptive threshold is set to 25 pixels.

- 3) The color of image F_T is inverted using the following condition:

$$F_T(x, y) = \begin{cases} 0 & \text{if } F_T(x, y) = 255 \\ 255 & \text{otherwise.} \end{cases} \quad (1)$$

- 4) Small particles made up of white pixels less than the threshold value are deleted. The resulting image, F_D , is shown in Figure 2(c).
 5) The background of the finger image is deleted using the algorithm described in section 2.2.
 6) The image is rotated to ensure that it is aligned vertically with the x-axis of the image coordinate. This is done using the φ value derived using the algorithm described in section 2.3.
 7) The rotated image is skeletonized [7], features extracted using the algorithm described in section 3.
 8) The derived features are matched against a database to find the most likely matching finger using the algorithm described in section 4.

2.1 Finger Alignment Parameter

When a finger is pressed on the fingerprint sensor, it may be up to $\pm 45^\circ$ away from the assumed vertical orientation. The inclination is detected, and the image rotated. The algorithm works as follows:

- 1) The Canny algorithm [8] is applied to the F_D image, resulting in F_E .
 2) D_L and D_R are the bottom left and right edge images (326*480 pixels) of the finger. They are obtained by copying some part of F_E using the following conditions:

$$D_L(x, y) = \begin{cases} F_E(x, y) & , \begin{pmatrix} 0 \leq x < 180, \\ 340 < y < 480 \end{pmatrix} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

$$D_R(x, y) = \begin{cases} F_E(x, y) & , \begin{pmatrix} 146 \leq x < 326, \\ 340 < y < 480 \end{pmatrix} \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

- 3) The parameter of the bottom-part left edge of the finger are obtained by letting C be the set of contours in D_L , where $C = \{C_1, C_2, C_3, \dots, C_k\}$ and k = number of contours. A line-fitting algorithm [9] is applied to each contour C_i to find the straight-line parameter S_i of that contour. For each S_i we get:

$$S_i = (V_x^i, V_y^i, X_0^i, Y_0^i) \quad , \quad i = 1..k \quad (4)$$

where (V_x^i, V_y^i) is a normalized vector collinear to the line and (X_0^i, Y_0^i) is some point on the line.

- 4) The parameter S_{left} of the left-edge finger is selected from the set of S_i using the following condition:

$$S_{left} = S_j \text{ where } \left(\begin{array}{l} \text{abs} \left(\tan^{-1} \left(\frac{V_y^j}{V_x^j} \right) \right) \leq \frac{\pi}{4} \quad \text{and} \\ N_j > N_i \text{ for all } j \neq i, \quad i = 1..k \end{array} \right) \quad (5)$$

where N_i is the number of white pixels in each contour C_i .

- 5) The parameter S_{right} of the right-edge finger can be derived by applying steps 3-4 to D_R . The following left and right edge finger parameters can be used for finger alignment correction.

$$S_{left} = (V_{xl}, V_{yl}, X_{0l}, Y_{0l}) \quad (6)$$

$$S_{right} = (V_{xr}, V_{yr}, X_{0r}, Y_{0r}) \quad (7)$$

2.2 Background Subtraction

The image from the CCD camera includes the fingerprint sensor covered by the finger. The background must be removed from the image to ensure that only the finger image is processed. Background deletion is done as follows:

- 1) Image M_L , which has the same size as F_G , is created using the following condition:

$$M_L(x, y) = \begin{cases} 255 & \text{if } \begin{pmatrix} (m_l < 0 \text{ and } y \geq m_l x + c_l) \\ \text{or } (m_l > 0 \text{ and } y \leq m_l x + c_l) \end{pmatrix} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

$$\text{where } m_l = \frac{V_{yl}}{V_{xl}} \quad \text{and} \quad c_l = Y_{0l} - \frac{V_{yl} X_{0l}}{V_{xl}}$$

- 2) Image M_R , which has the same size as M_L , is created using the following condition:

$$M_R(x, y) = \begin{cases} 255 & \text{if } \begin{pmatrix} (m_r < 0 \text{ and } y \leq m_r x + c_r) \\ \text{or } (m_r > 0 \text{ and } y \geq m_r x + c_r) \end{pmatrix} \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

$$\text{where } m_r = \frac{V_{yr}}{V_{xr}} \quad \text{and} \quad c_r = Y_{0r} - \frac{V_{yr} X_{0r}}{V_{xr}}$$

- 3) F_F is the background-deleted image. It is created from the operation:

$$F_F = M_R \cap M_L \cap F_D. \quad (10)$$

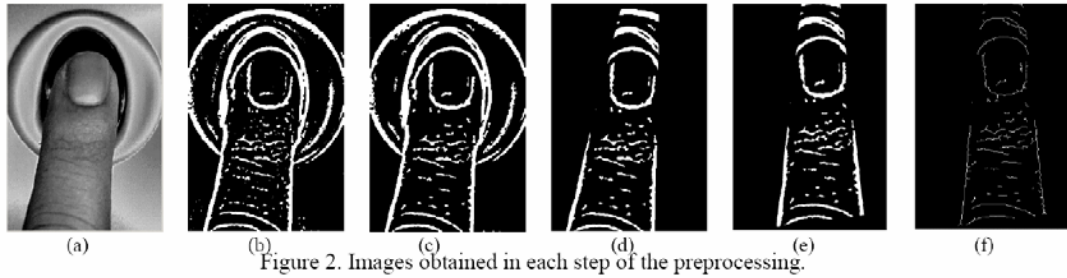


Figure 2. Images obtained in each step of the preprocessing.

2.3 Finger Alignment Correction

φ is the angle to rotate the finger image around the origin:

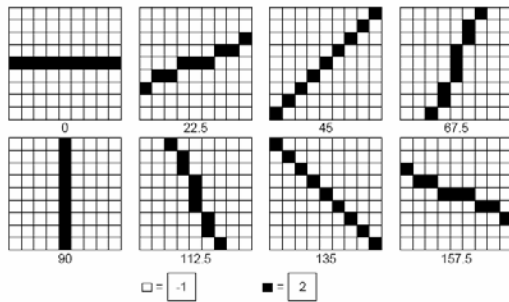
$$\varphi = \begin{cases} 0.5(\mu + \lambda) & \text{if } \mu\lambda < 0 \\ 0.5(\mu + \lambda - \pi) & \text{else if } \mu \geq 0 \text{ and } \lambda \geq 0 \\ 0.5(\mu + \lambda + \pi) & \text{otherwise} \end{cases} \quad (11)$$

μ and λ are the angle of inclination of the left and right edges of the finger, defined as:

$$\mu = \tan^{-1}\left(\frac{V_{yl}}{V_{xl}}\right), \quad \lambda = \tan^{-1}\left(\frac{V_{yr}}{V_{xr}}\right) \quad (12)$$

3. Feature Extraction

To extract features from the finger image, the skeletonized image is filtered using a bank of Oriented Filters, which use different θ values ($0^\circ, 22.5^\circ, 45^\circ, 67.5^\circ, 90^\circ, 112.5^\circ, 135^\circ$ and 157.5°) with respect to the x-axis. Figure 3 shows the kernel Oriented Filters for different θ values.

Figure 3. Kernel of Oriented Filters with different θ values.

The Oriented filters enhance the ridge lines along the θ angles while blurring the lines that lie in other directions. Feature extraction is carried out as follows:

- 1) An image reference point is located at the middle of the nail-base is defined, as shown in Figure 4(a). To maximize the finger-image identification, the reference point was manually defined in this paper.
- 2) Q_θ is the image filtered at θ° . Q_θ is tessellated into $H \times V$ square cells of size $W \times W$ using the reference point $R(x_r, y_r)$, as shown in Figure 4(b).
- 3) The variance of the pixel intensity of each tessellated cell is computed using:

$$\sigma_{hv}^2 = \frac{1}{W^2} \sum_{x=c}^d \sum_{y=a}^b p(x, y)^2 - \left(\frac{1}{W^2} \sum_{x=c}^d \sum_{y=a}^b p(x, y) \right)^2 \quad (13)$$

where

σ_{hv}^2 = The variance of the pixel intensity for each tessellated cell. h and v define the column and row number of that cell,
 $h = 0..9, v = -2, -1, 0, \dots, 12$;

$p(x, y)$ = The pixel intensity of image Q_θ at location (x, y) ;

H = The number of cells in each column;

V = The number of cells in each row;

$a = y_r + Wv$

$b = y_r + W(v + 1)$

$c = x_r + W(h - 0.5H)$

$d = x_r + W(h - 0.5H + 1)$

The appropriate value of W , H and V were determined empirically to be 15, 10 and 15 respectively.

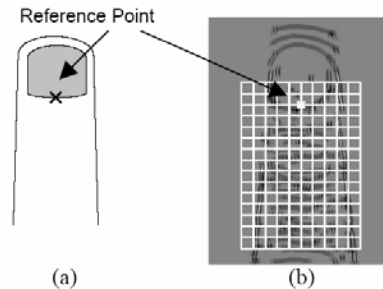


Figure 4. Reference point location and square tessellation on the filtered image.

4) After applying step 3 to every filtered image, we get:

$$\text{NailCode} = \{ V_0, V_{22.5}, V_{45}, V_{67.5}, V_{90}, V_{112.5}, V_{135}, V_{157.5} \} \quad (14)$$

where

$$V_\theta = \begin{bmatrix} \sigma_{(-2,0)}^2, \sigma_{(-2,1)}^2, \dots, \sigma_{(-2,H-1)}^2 \\ \sigma_{(-1,0)}^2, \sigma_{(-1,1)}^2, \dots, \sigma_{(-1,H-1)}^2 \\ \sigma_{(0,0)}^2, \sigma_{(0,1)}^2, \dots, \sigma_{(0,H-1)}^2 \\ \sigma_{(1,0)}^2, \sigma_{(1,1)}^2, \dots, \sigma_{(1,H-1)}^2 \\ \vdots \\ \sigma_{(P-3,0)}^2, \sigma_{(P-3,1)}^2, \dots, \sigma_{(P-3,H-1)}^2 \end{bmatrix}$$

4. The Matching Process

The Euclidean distance is computed in order to match the extracted features from the input image with those kept in our enrolled database. E_n is the Euclidean distance between the input image and the n^{th} finger image stored in the database, while the database consists of k enrolled fingers. By calculating the Euclidean distance between the input finger and each enrolled finger image, we get:

$$E = \{E_1, E_2, E_3, \dots, E_k\}. \quad (15)$$

The input finger is said to match with the i^{th} database finger if and only if:

$$E_i < E_j \text{ for all } i \neq j, 1 \leq i \leq k, 1 \leq j \leq k. \quad (16)$$

5. Experimental Results

The testing system consists of a Creative VF0080 CCD camera equipped with a Digital Persona UareU4000B fingerprint sensor in a light controlled environment. The system code was written in Visual C++ using OpenCV to capture the finger image whenever the fingerprint sensor was pressed. Our finger image database consists of 800 finger images from 100 different fingers with 8 images per individual. One image of each individual was employed to enroll the system, while the other 7 images were used to test the system. The reference point of each image was set manually. When assessing the accuracy of the system, we found that 676 finger images were correctly recognized while 24 finger images were misinterpreted. In other words, the accuracy of the system is 96.57%.

6. Conclusions

This paper describes a feasibility study for a new technique which increases the accuracy of fingerprint

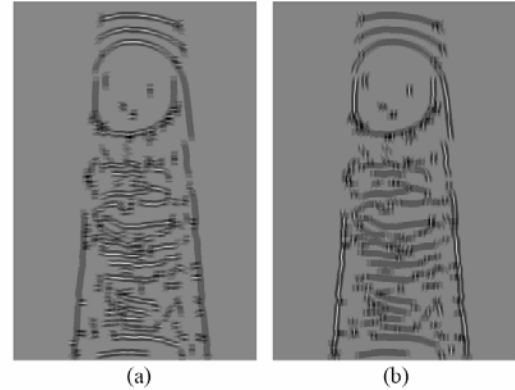


Figure 5. Finger images filtered at (a) 0° and (b) 90° .

identification without requiring the user to do additional tasks. Our results show that a top-view finger image will enhance a fingerprint recognition system. One drawback is that finger image is not time invariant, but we can use runtime biometric updating [10] to overcome this problem. The reference point detection algorithm is described in another paper [11].

Acknowledgements

The authors would like to thank Dr. Andrew Davison for the English correction of the first draft of this paper.

References

- [1] D. Maltoni, D. Maio, A.K. Jain, S. Prabhakar, *Handbook of Fingerprint Recognition*, (Springer-Verlag New York, Inc., Secaucus, NJ, 2003).
- [2] A. Jain, L. Hong; R. Bolle, On-line fingerprint verification, *IEEE PAMI*, 19(4), 1997, pp.302-314.
- [3] L. Hong, A.K. Jain, Integrating faces and fingerprints for personal identification, *IEEE PAMI*, 20(12), 1998, pp. 1295- 1307.
- [4] A.K. Jain, L. Hong, Y. Kulkarni., A Multimodal Biometric System using Fingerprint,Face and Speech, *Proc. Int. Conf. On Audio-and-Video-Based Biometric Person Authentication (2nd)*, 1999, pp. 182-187.
- [5] G. L. Marcialis, F. Roli, Fingerprint verification by fusion of optical and capacitive sensors, *Pattern Recognition Letters*, 25(11), 2004, pp. 1315 – 1322.
- [6] R. Gonzalez; R Woods, *Digital Image Processing*, (2nd edition, Prentice-Hall, Inc., 2002).
- [7] The MathWorks, Inc., www.mathworks.com
- [8] J. Canny, A Computational Approach to Edge detection, *IEEE PAMI*, 8(6), 1986, pp. 679-698.
- [9] Intel Corporation, *Open Source Computer Vision Reference Manual*, (Version 004, Dec 2001)
- [10] X. Jiang, W. Ser, "Online Fingerprint Template Improvement, *IEEE PAMI*, 24(8), 2002, pp. 1121-1126.
- [11] P. Chaikan, M. Karnjanadecha, A Reference Point Detection Algorithm for Top-View Finger Image Recognition, *To appear in Proc. Int. Conf. On ISPA2007*.

ISPA 2007

CD version

Proceedings of the 5th International Symposium on

Image and Signal Processing and Analysis

Istanbul, Turkey, September 27-29, 2007

M. Petrou, T. Saramäki, A. Erçil, S. Lončarić (Eds.)

IEEE 07EX1763C
ISBN 953184117-7
ISSN 1845-5956

Technical support :
secretariat@isispa.org
tel: +385 1 6129911
fax: +385 1 6129652



A Reference Point Detection Algorithm for Top-View Finger Image Recognition

Panyayot Chaikan, Montri Karnjanadecha
 Department of Computer Engineering, Faculty of Engineering,
 Prince of Songkhla University, Thailand
 {panyayot, montri}@coe.psu.ac.th

Abstract

This paper describes an algorithm for automatic reference point detection in a top-view finger image recognition system. In tests of 700 finger images, only 6 images were rejected by our algorithm. A reference point location error correction technique was developed to improve the recognition accuracy. When using the proposed algorithm, the accuracy of the top-view finger image identification system was only reduced to 93.80% compared to 96.57% when using a manually defined reference point. This shows the feasibility of using top-view finger images to increase the recognition accuracy of fingerprint identification.

1. Introduction

Fingerprint recognition is one of the most prominent biometric identification methods, partly due to its cost benefits when compared to other biometric systems (e.g. iris, retina, DNA). There have been many attempts to combine other features with fingerprint system to increase the identification accuracy. [1] These multimodal biometrics, include Jain [2] which uses speech and face features in conjunction with fingerprints, Hong [3] which employs both fingerprints and face features, and Marcialis [4] which utilizes two fingerprint images from different types of sensor. The drawback of these systems is that they require the user to carry out additional tasks.

Our paper [5] shows that top-view finger image is very possible to be used for improving the accuracy of fingerprint identification, but the system nests on a good reference point location for extracting features. A manually determined reference point is an extra user task and so has the same problem as described before.

This paper proposes an autonomous reference-point detection methodology for the top-view finger image identification system, so no additional tasks need to be carried out by the user.

The rest of the paper is organized as follows: section 2 starts with a brief overview of top-view finger image recognition. Section 3 gives details on our proposed reference-point detection algorithm. Section 4 presents the reference point location error compensation. Experimental results appear in section 5, and section 6 concludes the paper.

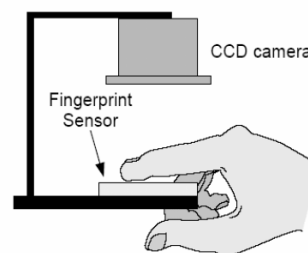


Figure 1. The Multimodal biometric system using finger image and fingerprint

2. Personal Identification using a Top-View Finger Image

A gray scale top-view finger image captured from a CCD camera is smoothed, binarized, and inverted, as shown in Figure 2(a). The image is rotated so it is vertically oriented. The image is skeletonized and filtered with a bank of Oriented Filters, the resulting images were extracted for features using a reference point during NailCode [5] generation. The NailCode is employed in the matching process by employing a Euclidean distance computation.

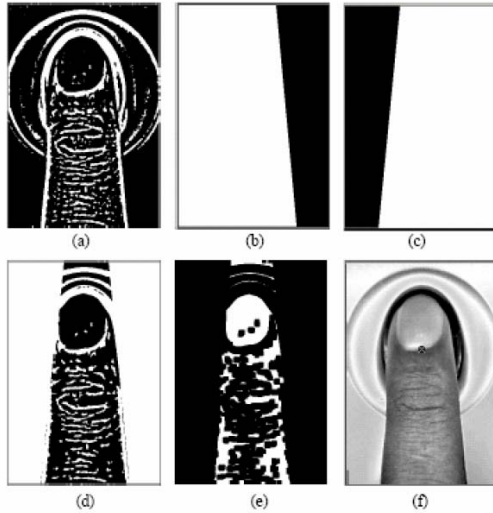


Figure 2. Top-view finger images for each step of the reference point detection algorithm.

2. Reference Point Detection Algorithm

The reference point of a top-view finger image is located at the midpoint of the finger's nail-base, as shown in Figure 3. The steps for reference point detection can be summarized as follows:

- 1) Let S_{left} and S_{right} are the inclination parameters of the left and right edge of the finger:

$$S_{left} = (V_{xl}, V_{yl}, X_{0l}, Y_{0l})$$

$$S_{right} = (V_{xr}, V_{yr}, X_{0r}, Y_{0r}).$$

(V_{xi}, V_{yi}) is a normalized vector collinear to the line, and (X_{0i}, Y_{0i}) is some point on the line.

- 2) Image M_L , as shown in Figure 2(c), is the same size as the gray-scale top-view input image. It is created using the following condition:

$$M_L(x, y) = \begin{cases} 255 & \text{if } \left(\begin{array}{l} (m_l < 0 \text{ and } y \geq m_l x + c_l) \\ \text{or } (m_l > 0 \text{ and } y \leq m_l x + c_l) \end{array} \right) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$\text{where } m_l = \frac{V_{yl}}{V_{xl}} \quad \text{and} \quad c_l = Y_{0l} - \frac{V_{yl} X_{0l}}{V_{xl}}$$

- 3) Image M_R , as shown in Figure 2(b), is the same size as M_L . It is created using the following condition:

$$M_R(x, y) = \begin{cases} 255 & \text{if } \left(\begin{array}{l} (m_r < 0 \text{ and } y \leq m_r x + c_r) \\ \text{or } (m_r > 0 \text{ and } y \geq m_r x + c_r) \end{array} \right) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

$$\text{where } m_r = \frac{V_{yr}}{V_{xr}} \quad \text{and} \quad c_r = Y_{0r} - \frac{V_{yr} X_{0r}}{V_{xr}}.$$

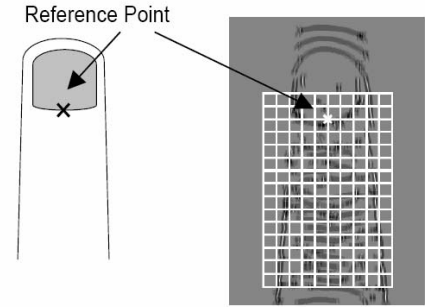


Figure 3. Reference point and tessellated top-view finger image.

- 4) Image F_D , as shown in Figure 2(a), is the top-view finger image after being thresholded and inverted using the algorithm in [5]. It is employed to create image F_K , using the following condition:

$$F_K = (M_R \cap M_L \cap F_D) \cup \sim(M_R \cap M_L). \quad (3)$$

- 5) The F_K , as shown in Figure 2(d), is dilated using a square-shaped structure element of size 3*3 [6].
- 6) The color of the dilated image is inverted using:

$$F_T(x, y) = \begin{cases} 0 & \text{if } F_T(x, y) = 255 \\ 255 & \text{otherwise.} \end{cases} \quad (4)$$

- 7) The image is rotated to ensure that it is exactly vertically oriented, resulting in F_L .
- 8) L is the set of contours found in the image F_L :

$$L = \{L_1, L_2, L_3, \dots, L_f\}$$

where f is the number of contours detected in the image. The reference point location can be derived by applying the algorithm described in Figure 4 to F_L , using the following parameters in each iteration:

- N_i = The number of white pixels in each contour L_i
- BR_i = The bounding rectangle [7] of each contour L_i . Each rectangle contains the parameters:

$$\{x_i^{BR}, y_i^{BR}, w_i^{BR}, h_i^{BR}\}$$

where

y_i^{BR} = y coordinate of top most rectangle corner

x_i^{BR} = x coordinate of left most rectangle corner

w_i^{BR} = width of rectangle

h_i^{BR} = height of rectangle

- $ratio_i = \frac{N_i}{w_i^{BR} * h_i^{BR}}$

- 9) L_j is the selected contour derived from the algorithm shown in Figure 4. The reference point can be computed on the chosen contour L_j by using the following equation:

$$R(x, y) = (x_j^{BR} + 0.5w_j^{BR}, y_j^{BR} + h_j^{BR}) \quad (5)$$

```

no_loop=0
found_ref_pt = false
do
{
    find contours from image  $F_L$ 
    for each contour  $L_i$ 
        if ( $N_i > \text{threshold}$  and  $\text{ratio}_i > 0.6$ )
            found_ref_pt = true
            add  $L_i$  in the list
        else
            erode image  $F_L$ 
            no_loop++
    if (found_ref_pt=true)
        select contour  $L_j$  from the list with the largest ratio
}
while (no_loop < MAX_LOOPS and found_ref_pt=false)

if (no_loop < MAX_LOOPS)
    extract reference point from contour  $L_j$ 
else
    report that reference point can not be found

```

Figure 4. Automatic reference point detection algorithm.

4. Reference Point Location Error Compensation

If the derived reference point location is incorrect, then the identification accuracy will be affected. Table 1 shows that an increasing reference point location error reduces identification accuracy. The table was obtained by trying to tessellate the filtered image with a translated version of the reference point moved in eight directions with a distance value of δ .

The effect of reference point location errors are reduced in our algorithm by translating the reference point into eight directions around its original point. Nine NailCodes are generated for nine reference points.

Table1. Identification accuracy of personal identification with different reference-point location errors.

Reference point error (in pixels)	Identification accuracy (percent)
0	96.57
5	89.10
8	78.00
10	65.90
12	51.90
15	32.00

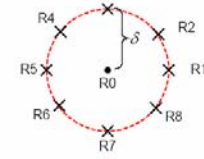


Figure 5. Eight points for reference point error compensation

5. Modified NailCode Matching

The effects of reference point location errors are handled by a modified version of our NailCode matching process [5]:

- 1) Let $R_0(x, y)$ is the reference point obtained by the algorithm described in section 3. The 8 translated versions are R_1 - R_8 , each of which is a distance δ away from R_0 , as shown in figure 5. The NailCode values for nine reference points are

$$\text{Extracted Features} = \{N_{R_0}, N_{R_1}, N_{R_2}, \dots, N_{R_8}\}$$

N_{R_i} is the NailCode for the reference point R_i .

- 2) The Euclidean distance between the NailCode N_{R_i} and each finger in the database is represented by:

$$E = \begin{Bmatrix} E_1^{R_0}, E_2^{R_0}, E_3^{R_0}, \dots, E_q^{R_0} \\ E_1^{R_1}, E_2^{R_1}, E_3^{R_1}, \dots, E_q^{R_1} \\ \vdots \\ E_1^{R_8}, E_2^{R_8}, E_3^{R_8}, \dots, E_q^{R_8} \end{Bmatrix}$$

where

q = The number of fingers in the database

$E_r^{R_i}$ = The Euclidean distance between the r^{th} finger in the database and the input finger tessellated using reference point R_i .

- 3) For each $E_r^{R_i}$, the two smallest Euclidean distances $E_m^{R_i}$ and $E_n^{R_i}$ are found such that :

$$E_m^{R_i} < E_n^{R_i} < E_p^{R_i}, \left(\forall (m \neq n, m \neq p \text{ and } n \neq p) \right) \left(1 \leq m \leq q, 1 \leq n \leq q, 1 \leq p \leq q \right) \quad (6)$$

- 4) The distance between E_m^{Ri} and E_n^{Ri} is calculated for every E_n^{Ri} , resulting in D_m^{Ri} :

$$D_m^{Ri} = E_n^{Ri} - E_m^{Ri} \quad (7)$$

where

$$D^{Ri} \in (D^{R0}, D^{R1}, D^{R2}, \dots, D^{R8}).$$

- 5) The input finger matches with the v^{th} finger in the database finger if and only if:

$$D_v^{Ri} > D_r^{Rj} \text{ for all } i \neq j, 0 \leq i \leq 8, 0 \leq j \leq 8. \quad (8)$$

6. Experimental Results

A gray scale finger image was captured from a Creative VF0080 CCD camera whenever the user touched the fingerprint sensor. Our top-view finger image database consists of 800 finger image from 100 different fingers, with eight images per individual. One image from each individual was employed to enroll the system, while the other seven images were used to test. This means that there were 700 test images in the database. Only 6 test images were rejected by our reference point detection algorithm. When manual reference point were utilized, the identification accuracy was 96.57%, which dropped to 73.78% when only a single automatic reference point, R_0 , was used. Table 2 shows that additional points R_1 - R_8 , improved the accuracy of the system dramatically, especially when δ was 9 or 10 pixels. In those cases, the identification accuracy was 93.80%.

Table 2. Identification accuracy with different reference point markings.

Reference point marking method	Reject	Accuracy
Manual	0	96.57 %
Automatic using R0 only	6	73.78 %
Automatic using R0-R8 ($\delta=5$)	6	87.03 %
Automatic using R0-R8 ($\delta=8$)	6	92.51 %
Automatic using R0-R8 ($\delta=9$)	6	93.80 %
Automatic using R0-R8 ($\delta=10$)	6	93.80 %

7. Conclusions

This paper shows the feasibility of using finger images for increasing the recognition accuracy of fingerprint identification without the need for additional user tasks. In particular, the user does not need to manually find the finger's reference point as required in [5]. We now plan to construct a personal identification system that uses finger images together with fingerprint data.




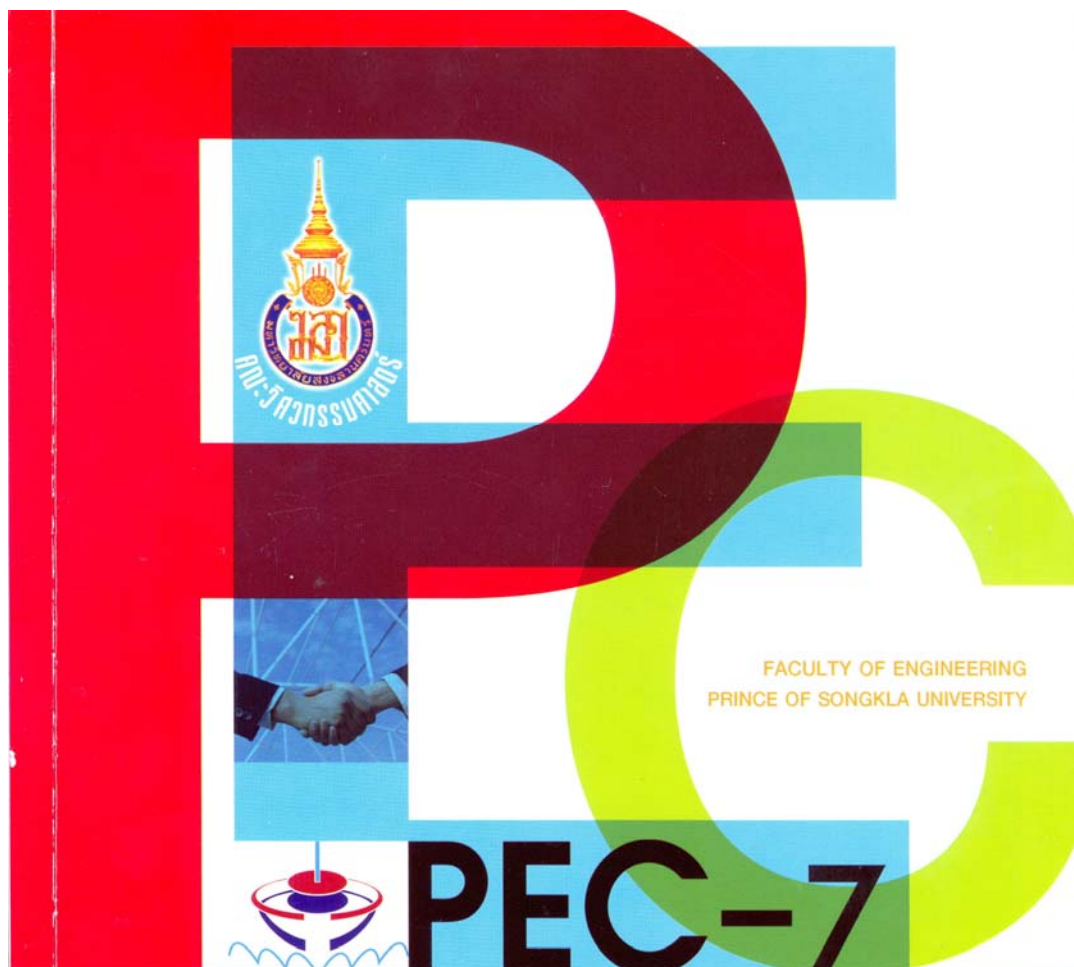
The average reference-point location error in our algorithm is about 10 pixels. This error is reduced by our compensation method, but processing time is increased by about 8 times compared to when only one reference point is utilized. This suggests that the algorithm is not suitable for an environment with limited CPU performance, such as embedded systems. In such an environment, reference point location accuracy must be improved to avoid the need for compensation.

Acknowledgements

The authors would like to thank Dr. Andrew Davison for the English correction of the first draft of this paper.

References

- [1] D. Maltoni, D. Maio, A.K. Jain, S. Prabhakar, *Handbook of Fingerprint Recognition*, (Springer-Verlag New York, Inc., Secaucus, NJ, 2003).
- [2] A.K. Jain, L. Hong, Y. Kulkarni., A Multimodal Biometric System using Fingerprint, Face and Speech, *Proc. Int. Conf. On Audio-and-Video-Based Biometric Person Authentication (2nd)*, 1999, pp.182-187.
- [3] L. Hong, A.K. Jain, Integrating faces and fingerprints for personal identification, *IEEE PAMI*, 20(12), 1998, pp. 1295–1307.
- [4] G. L. Marcialis, F. Roli, Fingerprint verification by fusion of optical and capacitive sensors, *Pattern Recognition Letters*, 25(11), 2004, pp. 1315 – 1322.
- [5] P. Chaikan, M. Kamjanadecha, The use of Top-View Finger Image for Personal Identification, *To appear on Int. Proc. Conf. On ISPA2007*.
- [6] R. Gonzalez; R Woods, *Digital Image Processing*, (2nd edition, Prentice-Hall, Inc., 2002).
- [7] Intel Corporation, *Open Source Computer Vision Reference Manual*, (Version 004, Dec 2001)



FACULTY OF ENGINEERING
PRINCE OF SONGKLA UNIVERSITY

PEC-7

การประชุมวิชาการทางวิศวกรรมศาสตร์ ครั้งที่ 7
The 7th PSU-Engineering Conference
21-22 พฤษภาคม 2552
ณ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

Integrating Fingerprint and Top-View Finger Image for Personal Verification

Panyayot Chaikan and Montri Karnjanadecha
Department of Computer Engineering, Faculty of Engineering, Prince of Songkla University,
Hat Yai, Songkhla, Thailand

E-mail: {panyayot,montri}@coe.psu.ac.th

Abstract

This paper describes the use of a top-view finger image to improve the accuracy of the fingerprint verification system. A CCD camera is used to capture the top-view finger image while the user press their finger on the fingerprint sensor without the need of additional interaction with the system. The feature map called NailCode is extracted from the top-view data and minutia features are extracted from the fingerprint image. By using the likelihood ratio that is calculated from the matching score from both top-view and fingerprint matcher, the improvement of the matching accuracy is obtained.

Keywords: Multimodal Biometric, NailCode, Fingerprint verification, Finger image.

1. Introduction

Personal verification using fingerprint is widely used because of it has been proved that fingerprint of each person is unique and time invariant. Many publication papers tried to improve the accuracy of fingerprint matching until the limitation have reached. To further improve the system accuracy, other biometric features have been used in conjunction with fingerprint to improve the matching accuracy, such as, Hong[1] employs the face in conjunction with fingerprints, Jain[2] uses speech, face and fingerprints, Marcialis[3] utilizes two different types of fingerprint sensors and Phrabhakar[4] uses 2 impressions of the same fingerprint. All of these approach give the improved matching accuracy compared to the one that used only fingerprint feature, but the drawback of these systems are that they require additional feature that need more user interaction with the system.

Our approach nests on the idea that the skin wrinkles and furrows on the top of a person's fingers are different, and that this information could be easily captured with a small camera above the fingerprint sensor. The additional image information should increase the accuracy of the system without adding any extra tasks to the user.

2. Feature extraction from Top-View finger image

The input top-view finger image is smoothed using a Gaussian filter before the inclination angle of the finger is detected using a Line-fitting algorithm [5], the obtained angle value is needed to rotate the image to be up right respect to the x-axis. Background deletion is done before applying the adaptive threshold to obtain a binary image. The filterbank with the setting value of 8 different degrees are applied to the image, each filtered image is tessellated respective to the reference point to create a feature map, called NailCode [6]. Figure 1 shows the example of the image preprocessed and filtered at 0° . A reference point is located at the middle-bottom of the nail of the finger image, as shown in figure 1, a technique for locating reference point was presented in [7]. In the matching process, a Euclidean distance is computed from the input and the template NailCode. The detail algorithm of the top-view finger image processing was presented in [6].

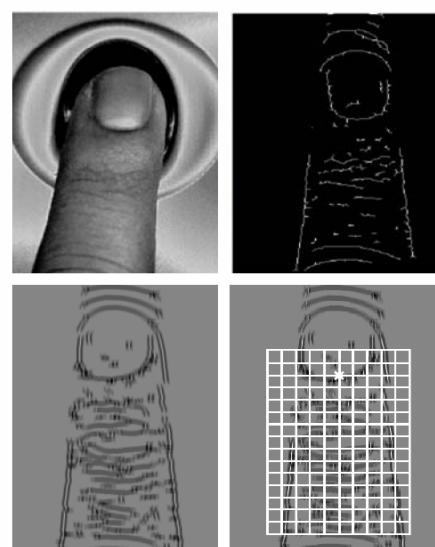


Figure 1. Preprocessing and feature extraction of the top-view finger image.

3. Fingerprint Feature extraction and matching system

Fingerprint preprocessing is done to the image using the techniques proposed by Hong[3]. Minutia features are extracted and post-processed by the algorithm proposed in [9]. Matching process can be summarized in the following steps:

- 1) Let P and Q are the minutia set of the template and the input fingerprint respectively,

$$P = \left\{ \left(x_1^p, y_1^p, \theta_1^p \right), \dots, \left(x_m^p, y_m^p, \theta_m^p \right) \right\}$$

$$Q = \left\{ \left(x_1^q, y_1^q, \theta_1^q \right), \dots, \left(x_n^q, y_n^q, \theta_n^q \right) \right\}$$

- 2) The score table of size $m \times n$ is created and all value are set to be zero, m and n stand for the number of minutia in P and Q respectively.
- 3) Perform the algorithm shown in Figure 2.

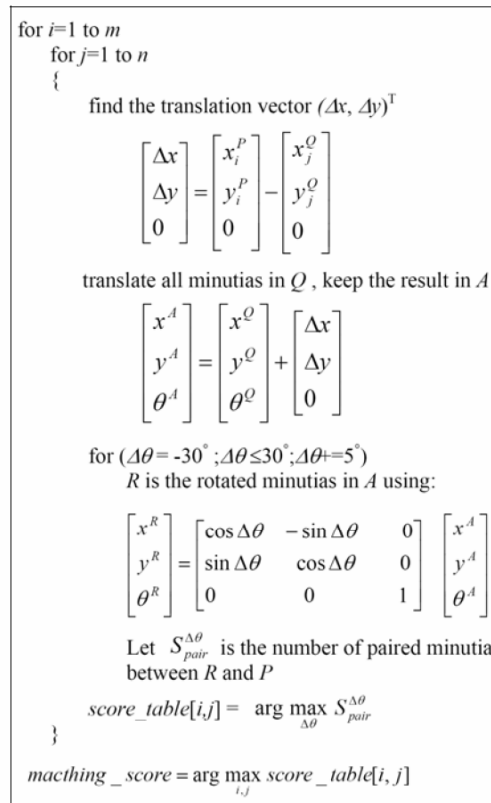


Figure 2. Matching algorithm for minutia feature.

The two minutias are said to be paired if and only if the spatial distance (sd) and the direction distance (dd) are less than the the threshold th_{sd} and th_{dd} respectively. The two distance values can be computed using:

$$sd = \sqrt{(x^R - x^P)^2 + (y^R - y^P)^2} \quad (1)$$

$$dd = \min(|\theta^R - \theta^P|, 2\pi - |\theta^R - \theta^P|) \quad (2)$$

The derived matching score is actually the number of matched minutias between the input fingerprint and the template. This value cannot be directly compared to the distance value from the top-view matcher because they are in the different domain.

4. Combination Scheme

To combine the 2 biometric features, we decided that the fusion is to be done at the confidence level. Figure 3 shows the block diagram of our implemented system, the decision is not perform at the 2 matchers, this means that the top-view and bottom-view matcher only send the matching score to the fusion module and the decision is made in this module at once.

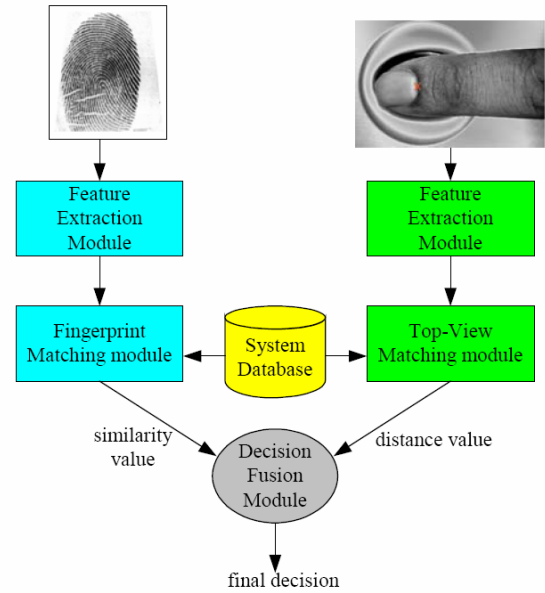


Figure3. Personal verification using top-view and bottom-view finger image.

The expected result of decision fusion is to derive the improvement of matching accuracy that is the top-view matcher gives the wrong result while the bottom-view matcher gives the right one, and vice versa. Let us suppose that the input top-view and bottom-view images are to be classified to the 2 possible classes, ω_1 and ω_2 , where the first one stands for the imposter class and the another stands for the genuine class. Let us define that x_1 is the distance value derived from the top-view matcher and x_2 is the matching score come from the bottom-view matcher.

The design goal of the biometric system depends on its application. In high security access applications, the low value of false acceptance rate (FAR) is required while the forensic applications need the low value of false rejection rate (FRR). Our design goal is to minimize both false acceptance rate (FAR) and false rejection rate (FRR), however, it is difficult to get an extremely low value of these 2 rates at the same time, so the approach that most designers

choose to implement the system is to minimize the equal error rate(ERR: the point that the system has FAR and FRR value at the same level) [10] as much as possible.

If we use top-view feature alone to verify person, the FAR and FRR value can be computed using:

$$FAR_{top} = \int_{-\infty}^{th_{top}} P(x_1 | \omega_1) dx_1 \quad (3)$$

$$FRR_{top} = \int_{th_{top}}^{\infty} P(x_1 | \omega_2) dx_1 \quad (4)$$

In contrast, if we use bottom-view feature alone to verify person, the FAR and FRR value can be calculated using:

$$FAR_{btm} = \int_{th_{btm}}^{\infty} P(x_2 | \omega_1) dx_2 \quad (5)$$

$$FRR_{btm} = \int_{-\infty}^{th_{btm}} P(x_2 | \omega_2) dx_2 \quad (6)$$

The value th_{btm} and th_{top} are the threshold value set at the top-view and bottom-view matcher respectively. Equations 3-6 state that the threshold value of both top-view and bottom-view matcher have the effect on the FAR and FRR values. If we adjust the threshold to increase the FAR value, the FRR value is decreased, and vice versa.

The final decision of the fusion module in figure 3 is made using:

$$\text{decision} = \begin{cases} \omega_2, & \text{if } (L > \beta \parallel x_1 < th_{top} \parallel x_2 > th_{btm}) \\ \omega_1, & \text{otherwise.} \end{cases} \quad (7)$$

The use of ORing operation in equation 7 increases the value of $FAR_{combined}$ but decreases the value of $FRR_{combined}$. By setting the value of th_{top} and th_{btm} to get very small value of FAR_{top} and FAR_{btm} respectively, the low value of $FAR_{combined}$ can be achieved. The likelihood ratio L can be calculated using:

$$L = P(x_1, x_2 | \omega_2) / P(x_1, x_2 | \omega_1) \quad (8)$$

The equation 8 tells us that if L is high, the input data is more likely to come from class ω_2 , the β value is the threshold value to decide that the input is the genuine person or not, this value is empirically determined. It is obvious that the joint probability in equation 8 is difficult to get directly from the training data set, by assuming that x_1 and x_2 are statistically independent, the joint probability density can be estimated using:

$$P(x_1, x_2 | \omega_k) = p(x_1 | \omega_k) p(x_2 | \omega_k) \quad (9)$$

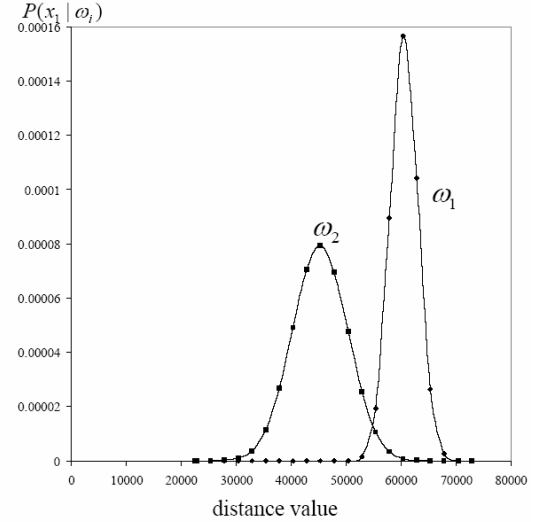


Figure 4. The estimated distribution for the top-view matcher.

5. Experimental Result

The test database was collected from 800 different fingers. One snapshot of each finger contains both top-view and bottom-view data. Seven snapshots were collected from each finger, each of which was used to enroll into the database while the other 6 snapshots were used to test the system. To approximate the top-view and bottom-view matching score distribution, 200 fingers were randomly selected and gaussian model was used. Figure 4 shows the example of the estimated top-view distribution.

Table 1 shows the false rejection rate of the system at the prespecified values of FAR. The fingerprint feature gives more performance than the top-view feature. At the FAR value from 0.01% to 1%, the combined system gives the improved FRR values, however, the use of ORing operation in equation 7 makes the FAR values of the combined system increasing, when we tested with our database, the FAR of the combined system could not be lower than 0.0107%.

Table1 FRR of the system at different values of FAR.

FAR	False Rejection Rate (FRR)		
	Using Top-View feature alone	Using fingerprint alone	Combined
1%	5.4%	2.6%	1.9%
0.1%	9.1%	5.9%	3.5%
0.01%	13.5%	10.5%	4.1%
0.001%	21.4%	17.5%	-

The receiver operating characteristic (ROC) curves [10] were plotted in Figure 5. The genuine acceptance rate in the ROC is actually the value of 1-FRR. When fingerprint feature was use in the

system alone, the ERR is at 1.75% while the top-view feature alone gave 3.9% of ERR. After combination, the accuracy of the system was improved so it had an ERR of 1.36%.

6. Conclusion

By using the additional top-view finger image, the performance of the personal verification system using fingerprint is improved. Our implemented system met the goal of making a system with the lowest ERR value, however, the combination mechanism has a limitation that it cannot give very small value of FAR (lower than 0.01%). This means that the combination mechanism must be further improved.

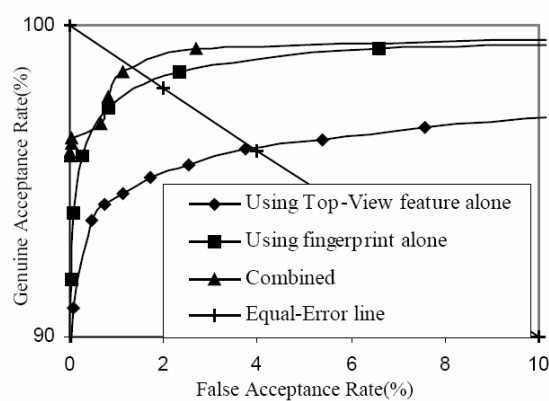


Figure 5. Performance of our personal verification system.

References

- [1] Hong, L. and Jain, A.K. 1998. Integrating faces and fingerprints for personal identification, *IEEE PAMI*, 20(12):1295-1307.
- [2] Jain, A.K., Hong, L. And Kulkarni, Y. A Multi-modal Biometric System using Fingerprint, Face and Speech, *Proc.Int. Conf. On Audio-and-Video-Based Biometric Person Authentication (2nd)*, 1999:182-187.
- [3] Marcialis, G. L. and Roli, F. 2004. Fingerprint verification by fusion of optical and capacitive sensors, *Pattern Recognition Letters*, 25(11):1315-1322.
- [4] Prabhakar, S. And Jain, A.K. 2002. Decision level fusion in Fingerprint verification. *Pattern Recognition*, 35: 861-874.
- [5] Intel Corporation, *Open Source Computer Vision Reference Manual*, (Version 004, Dec 2001).
- [6] Chaikan, P. and Karnjanadecha. 2007. The use of Top-View Finger Image for Personal identification. *Int. Proc. Conf. On ISPA2007*, September 27-29, 2007:343-346.
- [7] Chaikan, P. and Karnjanadecha, M. A Reference Point Detection Algorithm for Top-View Finger Image Recognition, *Int. Conf. On ISPA2007*, September 27-29, 2007:347-350.
- [8] Hong, L. and Jain, A.K. 1998. Fingerprint Image Enhancement : Algorithm and Performance evaluation, *IEEE PAMI*, 20(8):777-789.
- [9] Chaikan, P. and Karnjanadecha, M. 2008. A Systematic Method for Fingerprint Post-processing. *The 6th PSU Engineering Conference*, Thailand, May. 8-9, 2008.
- [10] Maltoni, D., Maio, D., Jain, A.K. and Prabhakar, S. 2003. *Handbook of Fingerprint Recognition*, Springer-Verlag New York, Secaucus, NJ.



ISSN 0125-3395

Songklanakarin

Journal of Science and Technology

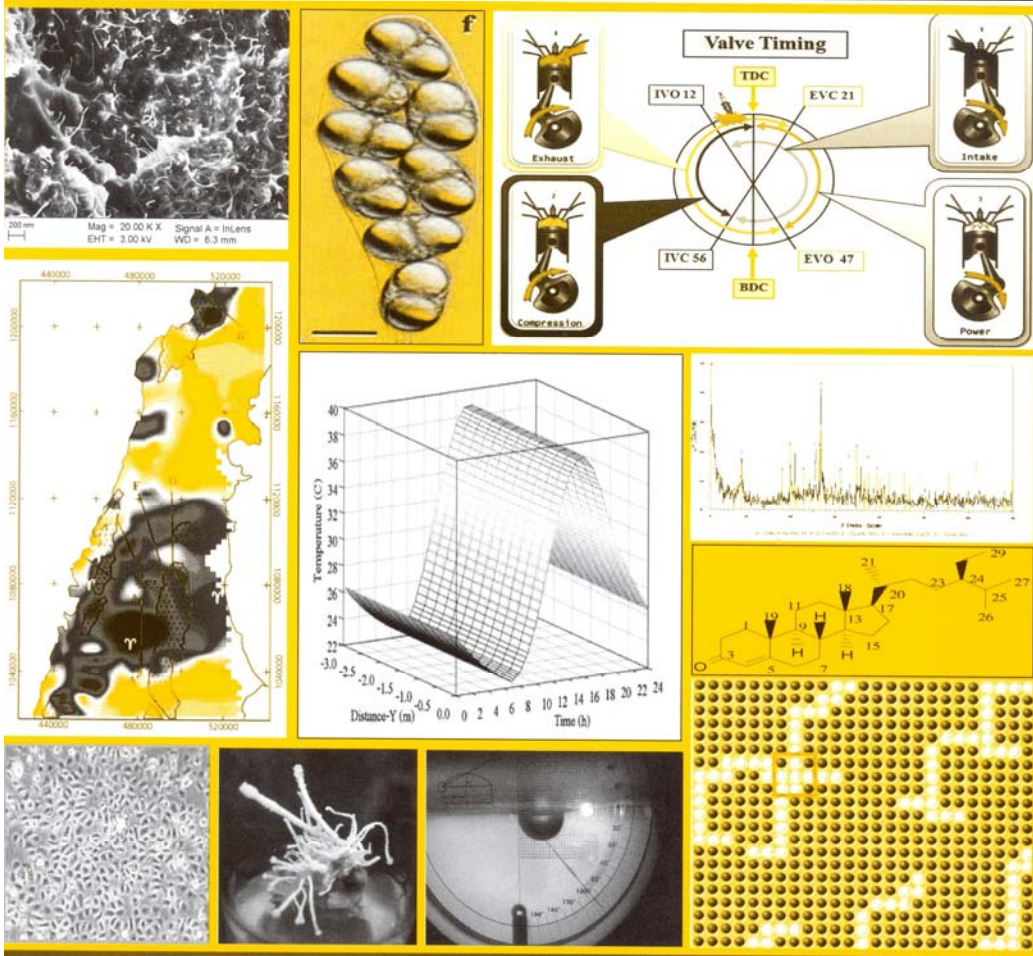
E-Journal



**R
D
O**

We Serve Research

Vol.32 No.1 January - February 2010 <http://www.sjst.psu.ac.th>



Published by the Research and Development Office, RDO
 Prince of Songkla University, PSU
 With the financial support from
 The Office of the Higher Education Commission : OHEC, Thailand



Original Article

Person recognition using fingerprints and top-view finger images

Panyayot Chaikan* and Montri Karnjanadecha

Department of Computer Engineering, Faculty of Engineering,
Prince of Songkla University, Hat Yai, Songkhla, 90112 Thailand.

Received 9 September 2009; Accepted 25 December 2009

Abstract

Our multimodal biometric system combines fingerprinting with a top-view finger image captured by a CCD camera without user intervention. The greyscale image is preprocessed to enhance its edges, skin furrows, and the nail shape before being manipulated by a bank of oriented filters. A square tessellation is applied to the filtered image to create a feature map, called a NailCode, which is employed in Euclidean distance computations. The NailCode reduces system errors by 17.68% in the verification mode, and by 6.82% in the identification mode.

Keywords: fingerprints, top-view finger images, multimodal biometrics, nailcode, image processing

1. Introduction

Person recognition by fingerprinting is ubiquitous because of its uniqueness and time invariance (Maltoni *et al.*, 2003). As a biometric feature, fingerprints offer high accuracy even when cheap sensors are utilized. However, fingerprint recognition accuracy has reached a limit which is difficult to surpass. One approach is multimodal biometrics, which combines multiple human features in the recognition process. For example, Hong and Jain (1998) employs the face in conjunction with fingerprints, Jain *et al.* (1999a) uses speech, face, and fingerprints, Marcialis and Roli (2004) utilize two different fingerprint sensors, while Prabhakar and Jain (2002) examine two fingers. All these methods augment recognition accuracy, with the drawback that the additional features increase the complexity of user interaction with the system.

Our approach rests on the idea that the skin wrinkles and furrows on top of each person's fingers are different, along with the size and shape of the fingers and finger nails. Utilizing these attributes will increase the accuracy of a multimodal biometric system without requiring extra work

by the user since the details can be captured with a small, inexpensive camera positioned above the fingerprint sensor, as shown in Figure 1. Top-view finger imaging also reduces the possibility of fraud by having recognition relying on more than one feature.

This paper is organized as follows: section 2 starts with an overview of biometric operation modes. Section 3 describes top-view finger image preprocessing, feature extraction, and matching. Section 4 outlines implementations for the fingerprint matching algorithms used by the top-view feature, while section 5 presents the decision fusion mechanism. Experimental results are given in section 6, and section 7 concludes the paper.

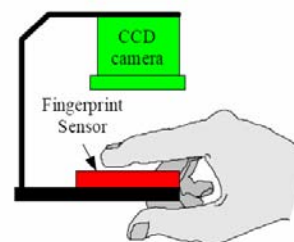


Figure 1. Top-view image and fingerprint recognition system.

*Corresponding author.

Email address: {panyayot, montri}@coe.psu.ac.th

2. Biometric Operation Modes

There are three main operational modes in biometric systems: classification, verification, and identification. Classification partitions the input pattern into n separated classes to reduce the search space in very large databases. For example, Jain *et al.* (1999b) classify fingerprints into six types: twin-loop, left-loop, right-loop, whorl, arch, and tented arch.

The input pattern is verified against templates to determine whether features come from the same individual or not. Computation time is not an issue because only 1:1 comparison is required.

Identification evaluates the input features to find the best matches with the templates in the database. The computation time must be as small as possible so that real time response can be achieved.

A good biometric system should ideally combine high accuracy with low computation time, though it is difficult to satisfy both demands. In a multimodal biometric classifier, the designer typically selects a low accuracy classifier with low computation time to identify the most n -probable match items from the database. Then the system switches to a high accuracy classifier, with a larger computation time, to check the n -candidate items to find the best match.

Our proposed multimodal biometric uses two features: fingerprinting and NailCodes. A NailCode is a feature map extracted from the top-view finger image using techniques described in section 3. The NailCode matcher performs well in both the verification and identification modes.

3. Top-view Finger Image Processing

3.1 Preprocessing

The grayscale top-view finger image obtained from the CCD camera is T_G , and has size $W \times L$ (see Figure 3(a)). Its preprocessing flowchart is shown in Figure 2. The main steps include:

1) *Smoothing*. Due to the presence of noise and non-uniform illumination in the image, a smoothing Gaussian filter is applied to T_G .

2) *Binarization*. The grayscale image is converted into two color image (black = 0 and white = 255) using an adaptive threshold (Gonzalez and Woods, 2002). The binarized image is inverted before the next step, as shown in Figure 3(b).

3) *Small Particle Deletion*. Small particles made up of white pixels less than the threshold value are deleted. The resulting image, T_H is shown in Figure 3(c).

4) *Background Deletion*. The background of the finger image is deleted using a parameter described in section 3.3. Figure 3(d) shows the resulting image.

5) *Finger Inclination Correction*. The image is rotated to align it vertically with the x-axis of the image, as shown in Figure 3(e).

6) *Skeletonization*. A thinning operation is applied to the image to create a skeleton for the remaining lines in the image, as shown in Figure 3(f).

7) *FilterBank*. The skeletonized image is manipulated by a filterbank holding eight different filtering directions. The result is eight images ready for feature extraction. Details are elaborated in section 3.5.

3.2 Finger image alignment parameter

When a finger is pressed on the fingerprint sensor, it may be up to $\pm 30^\circ$ away from the assumed vertical orientation. The inclination is detected, and the image is rotated as follows:

1) The Canny algorithm (Canny, 1986) is applied to the T_H image, producing T_K , which is copied into two images named T_L and T_R using the conditions:

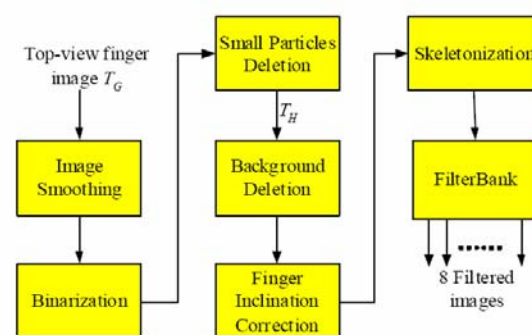


Figure 2. Flowchart of the preprocessing algorithm.

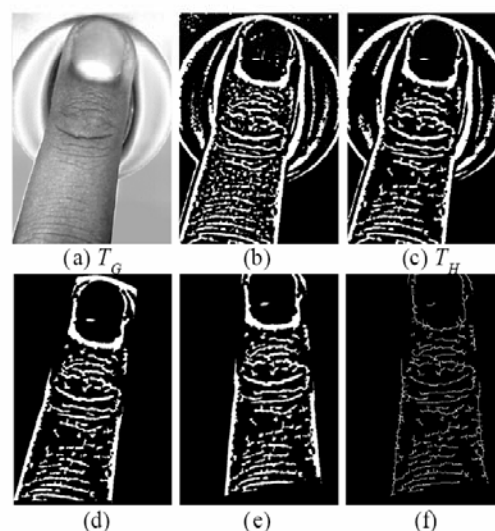


Figure 3. Images obtained in each preprocessing step.

$$T_L(x, y) = \begin{cases} T_k(x, y), & \begin{cases} 0 \leq x < W/2 \\ L/2 \leq y < L \end{cases} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

$$T_R(x, y) = \begin{cases} T_k(x, y), & \begin{cases} W/2 \leq x < W \\ L/2 \leq y < L \end{cases} \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

2) The parameters for the left-edge of the finger are obtained by letting C be the set of contours in T_L , where $C = \{C_1, C_2, C_3, \dots, C_k\}$ and $k = \text{number of contours}$. A line-fitting algorithm (Bradski and Kaehler, 2008) is applied to each contour C_i to find its straight-line parameter S_i . For each S_i we have:

$$S_i = (V_x^i, V_y^i, X_0^i, Y_0^i), \quad i = 1..k \quad (3)$$

where (V_x^i, V_y^i) is a normalized vector parallel to the fitted line and (X_0^i, Y_0^i) is a point on that line (Bradski and Kaehler, 2008).

3) The parameter S_{left} of the left-edge finger is selected from the set of S_i using the condition:

$$S_{left} = S_j \quad \text{where} \quad \begin{cases} \text{abs}(\tan^{-1}(V_y^j / V_x^j) \leq \pi/6 \quad \text{and} \\ N_j > N_i \text{ for all } j \neq i, \quad i = 1..k \end{cases} \quad (4)$$

where N_i is the number of white pixels in each contour C_i .

4) The parameter S_{right} of the right-edge finger can be derived by applying steps 3-4 to T_R .

$$S_{left} = (V_x^l, V_y^l, X_0^l, Y_0^l) \quad (5)$$

$$S_{right} = (V_x^r, V_y^r, X_0^r, Y_0^r) \quad (6)$$

3.3 Background deletion

The CCD camera image includes the background fingerprint sensor device which must be removed so that only the finger image is processed. Background deletion is achieved as follows:

1) Image M_L , which has the same size as T_L , is created using the condition:

$$M_L(x, y) = \begin{cases} 255 & \text{if} \begin{pmatrix} (m_l < 0 \text{ and } y \geq m_l x + c_l) \\ \text{or } (m_l > 0 \text{ and } y \leq m_l x + c_l) \end{pmatrix} \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

where $m_l = \frac{V_y^l}{V_x^l}$ and $c_l = Y_0^l - \frac{V_y^l X_0^l}{V_x^l}$.

2) Image M_R , which has the same size as M_L , is created using the condition:

$$M_R(x, y) = \begin{cases} 255 & \text{if} \begin{pmatrix} (m_r < 0 \text{ and } y \leq m_r x + c_r) \\ \text{or } (m_r > 0 \text{ and } y \geq m_r x + c_r) \end{pmatrix} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

where $m_r = \frac{V_y^r}{V_x^r}$ and $c_r = Y_0^r - \frac{V_y^r X_0^r}{V_x^r}$.

3) The background-deleted image, T_F , is created from the operation:

$$T_F = M_R \cap M_L \cap T_H \quad (9)$$

where \cap is a pixelwise-intersection operation, applied to equal-sized images.

3.4 Finger image inclination correction

1) Let μ and λ be the angles of inclination of the left and right edges of the finger respectively, defined by:

$$\mu = \tan^{-1} \left(\frac{V_y^l}{V_x^l} \right), \quad \lambda = \tan^{-1} \left(\frac{V_y^r}{V_x^r} \right).$$

2) The rotation of the finger around the origin is calculated using:

$$\varphi = \begin{cases} 0.5(\mu + \lambda) & \text{if } (\mu \times \lambda < 0) \\ 0.5(\mu + \lambda - \pi) & \text{else if } (\mu \geq 0 \text{ and } \lambda \geq 0) \\ 0.5(\mu + \lambda + \pi) & \text{otherwise.} \end{cases}$$

3.5 Filterbank

The skeletonized top-view finger image is manipulated using a bank of oriented filters, with eight different θ values ($0^\circ, 22.5^\circ, 45^\circ, 67.5^\circ, 90^\circ, 112.5^\circ, 135^\circ$, and 157.5°) with respect to the x -axis. The oriented filters enhance the ridge lines along the specified θ angles while blurring the lines that lie in other directions (see Figure 4(a)).

3.6 Feature Extraction

Feature extraction is carried out as follows:

1) The top-view finger image reference point, which is located in the middle of the nail base, is obtained using the algorithm described in section 3.7, and shown in action in Figure 4(a) and Figure 5(c).

2) θ is the degree setting on the oriented filter. The filtered image Q_θ is tessellated using the reference point (x_r, y_r) into $H \times V$ (10×15) square cells of size $w \times w$ (15×15), as shown in Figure 4(c). $p(x, y)$ denotes the pixel intensity at location (x, y) of Q_θ . The variance for each square cell at location (h, v) is calculated using (Chaikan and Karnjanadecha, 2007):

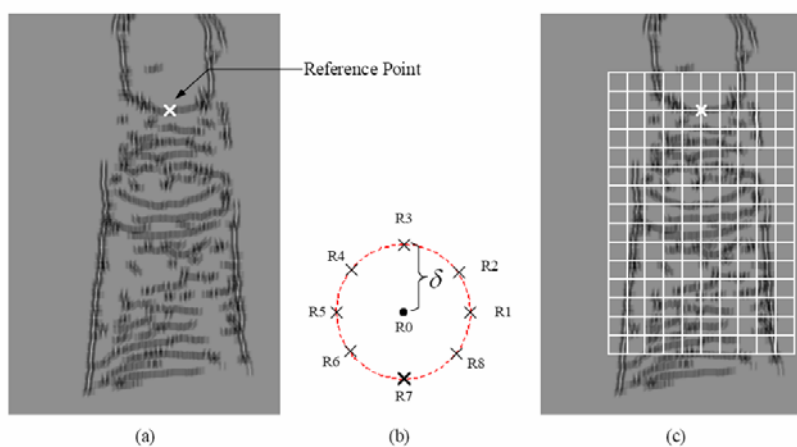


Figure 4. (a) Finger image filtered at 90° (b) Eight locations for reference point error compensation (c) square tessellation on the filtered image.

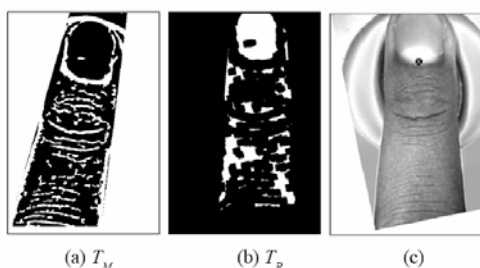


Figure 5. Top-view finger images for each step of the reference point detection algorithm.

$$\sigma^2(h, v) = \frac{1}{w^2} \sum_{x=c}^d \sum_{y=a}^b p(x, y)^2 - \left(\frac{1}{w^2} \sum_{x=c}^d \sum_{y=a}^b p(x, y) \right)^2 \quad (10)$$

where $a = y_r + vw$
 $b = y_r + w(v+1)$
 $c = x_r + w(h-0.5H)$
 $d = x_r + w(h-0.5H+1)$.

The tessellated area should cover the shape and the width of the nail base while avoiding problems with finger nail length variation. For that reason, we selected h to range from $0, 1, \dots, H-1$ and v to range from $-2, -1, 0, \dots, V-3$.

3) After applying step 2 to every filtered image, the extracted feature, called a NailCode, and denoted by Ψ^i , is calculated. Ψ^i for the images using reference point R_r , is defined by:

$$\Psi^i = \{ \Psi_{0}^i, \Psi_{22.5}^i, \Psi_{45}^i, \Psi_{67.5}^i, \Psi_{90}^i, \Psi_{112.5}^i, \Psi_{135}^i, \Psi_{157.5}^i \} \quad (11)$$

where

$$\Psi_j = \{ \sigma_1^2, \sigma_2^2, \sigma_3^2, \dots, \sigma_{H \cdot V}^2 \}.$$

4) Due to the possibility of a reference point detection error, a compensation technique is used. If R_0 is the reference point obtained by using the algorithm described in section 3.7, then there are eight translated versions, R_1 - R_8 , each δ (10) pixels from R_0 , as shown in Figure 4(b). In the enrollment module, only Ψ^0 is extracted from the input top-view finger image. In the authentication module, the NailCode $\Psi = \{ \Psi^0, \Psi^1, \Psi^2, \dots, \Psi^8 \}$ is extracted from the input top-view finger image.

3.7 Automatic detection of reference point location

The reference point of a top-view finger image is located at the midpoint of the finger's nail base. The steps for its detection are:

1) Let \cap , \cup and $'$ denote the intersection, union and inversion operations respectively. Image T_H , as shown in Figure 3(c), is employed to create image T_M using the condition:

$$T_M = (M_R \cap M_L \cap T_H) \cup (M_R \cap M_L)' \quad (12)$$

2) The image T_M as shown in Figure 5(a), is dilated and inverted before being rotated to be exactly vertical, resulting in T_p .

3) Let \mathcal{L} be the set of contours found in the image T_p :

$$\mathcal{L} = \{ L_1, L_2, L_3, \dots, L_f \}$$

where f is the number of contours detected in the image. The reference point location can be derived by applying the algo-

```

num_loop=0
ref_pt_found = false
do
{
  all values in  $\mathcal{L}$  are deleted, giving  $\mathcal{L} = \{\}$ 
  find contours from image  $T_p$ , putting all the results in  $\mathcal{L}$ 
  for each contour  $L_i$ 
    if ( $N_i > \text{threshold}$  and  $\text{ratio}_i > r_{th}$ )
      ref_pt_found = true
    else
      remove  $L_i$  from  $\mathcal{L}$ 

  if (ref_pt_found = true)
    select contour  $L_j$  from  $\mathcal{L}$  with the largest ratio
  else
    perform erosion operation on image  $T_p$ 
    num_loop++
}
while (num_loop < MAX_LOOPS and ref_pt_found = false)

if (num_loop < MAX_LOOPS)
  extract the reference point from the contour  $L_j$ 
else
  the reference point can not be found, and the image is rejected

```

Figure 6. Reference point detection algorithm.

rithm described in Figure 6 to image T_p , using the following parameters in each iteration:

N_i = The number of white pixels in each contour L_i

BR_i = The bounding rectangle (Bradski and Kaehler, 2008) of each contour L_i . Each rectangle contains the parameters:

$$\{x_i^{BR}, y_i^{BR}, w_i^{BR}, h_i^{BR}\}$$

where y_i^{BR} = y coordinate of top most rectangle corner

x_i^{BR} = x coordinate of left most rectangle corner

w_i^{BR} = width of rectangle

h_i^{BR} = height of rectangle.

$$\text{ratio}_i = \frac{N_i}{w_i^{BR} \times h_i^{BR}}$$

As shown in Figure 5(b), our algorithm tries to find the largest contours in the top-view finger image which are expected to be the nail. To avoid the contours that are larger than the nail, the ratio of the width and the length of the bounding rectangle is calculated, and contours with a ratio less than r_{th} (0.6) are thrown away.

4) L_j is the selected contour obtained from the algorithm in Figure 6. The reference point $R(x,y)$ is computed on L_j with:

$$R(x, y) = (x_j^{BR} + 0.5w_j^{BR}, y_j^{BR} + h_j^{BR}) \quad (13)$$

3.8 Matching

The Euclidean distance is computed as part of the matching operation. Let Ψ_T^0 be a NailCode template in the database and $\Psi = \{\Psi_{IP}^0, \Psi_{IP}^1, \Psi_{IP}^2, \dots, \Psi_{IP}^8\}$ be the NailCode extracted from the input top-view finger image. Each E_i in the Euclidean distance $E = \{E_0, E_1, \dots, E_8\}$ is the distance between Ψ_T^0 and Ψ_{IP}^i . The matching score between the input and the template is:

$$\text{matching_score}_{top} = \min(E_0, E_1, \dots, E_8) \quad (14)$$

4. Fingerprint Matching Algorithms

Fingerprint matching algorithms can be classified as minutiae-based and texture-based. We have developed two fingerprint matching systems based on minutiae matching: the first uses Hough transform-based matching while the other uses our own algorithm. They are combined with the top-view finger image matching system as described in section 5.

4.1 Hough transform-based minutiae matching (Algorithm Hough)

This algorithm tries to find the best transformation parameter (i.e. translation and rotation) between the input and the template minutiae. Each discretized transformation estimation is stored in an accumulator array, and the translation and rotation parameter are obtained by detecting the highest peak in the array. Since this algorithm uses an accumulator array in a similar way to a typical Hough transform, this algorithm is called Hough transform minutiae matching. The details of this algorithm can be found in Maltoni *et al.*, 2003.

Hough transform-based minutiae matching executes quickly but with low accuracy tolerances (compared to the other minutiae matching algorithms). Work by Prabhakar and Jain (2002) confirm these characteristics.

4.2 Our proposed minutiae-based fingerprint matching (Algorithm Simple)

Minutiae matching can be summarized by the following steps:

1) Let Z and R be the minutiae sets for the template and the input fingerprint,

$$Z = \{(x_1^Z, y_1^Z, \theta_1^Z), \dots, (x_z^Z, y_z^Z, \theta_z^Z)\}$$

$$R = \{(x_1^R, y_1^R, \theta_1^R), \dots, (x_r^R, y_r^R, \theta_r^R)\}$$

2) A score table of size $z \times r$ is created, with all its values set to zero. z and r stand for the number of minutiae in Z and R .

3) Execute the algorithm (shown in Figure 7).


```

for  $i=1$  to  $z$ 
  for  $j=1$  to  $r$ 
  {
    find the translation vector  $(\Delta x, \Delta y)^T$ 


$$\begin{bmatrix} \Delta x \\ \Delta y \\ 0 \end{bmatrix} = \begin{bmatrix} x_i^Z \\ y_i^Z \\ 0 \end{bmatrix} - \begin{bmatrix} x_j^R \\ y_j^R \\ 0 \end{bmatrix}$$


    translate all minutiae in  $R$ , storing the result in  $A$ 


$$\begin{bmatrix} x^A \\ y^A \\ \theta^A \end{bmatrix} = \begin{bmatrix} x^R \\ y^R \\ \theta^R \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ 0 \end{bmatrix}$$


    for  $(\Delta\theta = -\Phi; \Delta\theta \leq \Phi; \Delta\theta += \lambda)$ 
       $R^*$  is the rotated version of all minutiae in  $A$  using:


$$\begin{bmatrix} x^{R^*} \\ y^{R^*} \\ \theta^{R^*} \end{bmatrix} = \begin{bmatrix} \cos \Delta\theta & -\sin \Delta\theta & 0 \\ \sin \Delta\theta & \cos \Delta\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^A \\ y^A \\ \theta^A \end{bmatrix}$$


      Let  $S_{pair}^{\Delta\theta}$  be the number of paired minutiae between  $R^*$  and  $Z$ 


$$score\_table[i,j] = \arg \max_{\Delta\theta} S_{pair}^{\Delta\theta}$$

    }
  }
  matching\_scorebnm =  $\arg \max_{i,j} score\_table[i,j]$ 

```

Figure 7. Our Simple minutiae matching algorithm.

Two minutiae are paired if and only if their direction distance and spatial distance are less than the threshold values. The derived matching score is the number of matched minutiae between the input fingerprint and the templates. Because of its simplicity, this algorithm is called *Simple*.

5. Decision Fusion

Decision fusion derives an improvement in matching accuracy when the top-view matcher gives the wrong result while the bottom-view matcher gives the right one, or vice versa.

Suppose that an input feature can be a member of two possible classes, ω_1 and ω_2 , where the first is the imposter and the other is the genuine class.

Define $X = \{x_1, x_2, \dots, x_n\}$ as the matching score used in biometric verification. To make a final decision between the classes, the likelihood ratio L (Duda *et al.*, 2000; Prabhakar and Jain, 2002) is:

$$L = P(X|\omega_2) / P(X|\omega_1) \quad (15)$$

If L is high, then the input data is more likely to come from the genuine class. We decide that the input comes from the genuine class if $L > \beta$, where β is an empirically determined threshold value. The joint probability in equation 15 is difficult to obtain directly from training data, but by assuming that each x_i is statistically independent of each other, the joint probability density can be estimated using: (Duda *et al.*, 2000)

$$P(x_1, x_2, \dots, x_n | \omega_j) = \prod_{i=1}^n P(x_i | \omega_j) \quad (16)$$

6. Experimental Results

The system hardware is a Creative VF0080 CCD camera in a light controlled environment, combined with a Digital Persona UareU4000B fingerprint sensor. C++ software using the OpenCV library captures the top-view finger image whenever the fingerprint sensor is pressed. The test database holds details on 800 different fingers. A snapshot of a finger comprises both top and bottom-views. Eight snapshots were collected for each finger: one was added to the database while the other seven were used to test system performance.

Three matchers were implemented: (1) a *Hough* matcher using the Hough transform-based minutiae matching technique, (2) a *Simple* matcher utilizing our matching algorithm, and (3) a *TopView* matcher which employs the NailCode feature. The scores from these three matchers were combined to make a multimodal biometric system using the algorithm described in section 5.

6.1 Performance in the verification mode

The FAR (False Acceptance Rate) and FRR (False Rejection Rate) values were plotted on a Receiver Operating Characteristic (ROC) curve (Prabhakar and Jain, 2002) to judge the performance of the system. The genuine acceptance rate can be obtained from ROC as $1 - FRR$. For the FAR, 4,474,400 ($800 * 799 * 7$) matches were evaluated, and 5,600 ($800 * 7$) matches were examined to find the FRR.

We tested the verification performance against three conditions, with each condition using only one score from its respective matcher. There was no combination of these three systems. Figure 8 shows that the *Simple* matcher gives better accuracy than the other two matchers at every operating point. At low FARs, the *TopView* matcher gives higher accuracy than the *Hough* matcher, but the *Hough* matcher surpasses *TopView* at higher FAR values.

We combined the three matchers into pairs, and the likelihood ratio was used to perform decision fusion. System accuracy increased, as shown in Figure 9. The combination *TopView+Simple* gives the best accuracy. Also, the *Simple* matcher alone has better accuracy than a combination of *TopView+Hough* at all operating points with FARs lower than 7%. Since biometric system needs to operate at a low

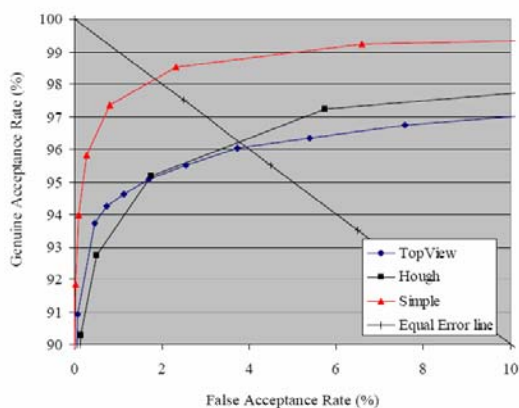


Figure 8. Verification performance of individual matchers.

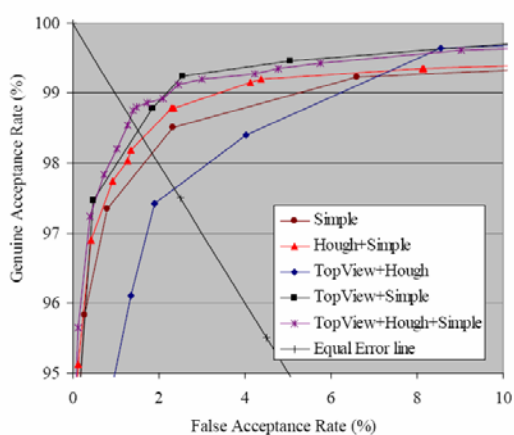


Figure 9. Verification performance of all combinations.

FAR value, this made us decide to use our minutiae matching algorithm to improve system accuracy in the identification mode.

Finally, we combined the matching scores of all three matchers, *TopView+Hough+Simple*. It outperformed all the paired matcher combinations at low FAR values, but for FARs greater than 2%, the *TopView+Simple* combination had lower rejection rates.

The Equal Error Rate (EER) (Maltoni *et al.*, 2003) was used to measure the strength of performance gains. The *TopView*, *Hough* and *Simple* matcher alone yield EERs of 3.91%, 3.80% and 1.86% respectively. The combinations of *TopView+Simple*, *Hough+Simple*, *TopView+Hough* yield EERs of 1.52%, 1.64% and 2.35% respectively. The combination of all three matchers gives the best EER of 1.35%.

As shown in Table 1, most of the computing time of the top-view finger image processing is spent on the pre-processing while the NailCode matching process requires considerably low computation time. The average computing time used to perform verification for NailCode matching and Hough transform-based minutiae matching were 20 ms and 3.125 ms respectively. The average time for performing verification using our *Simple* minutiae matching algorithm was 135.8 ms. This reveals that *Simple* is not suitable for directly searching the entire database because of its time-consuming behavior. However, due to its higher accuracy compared to *TopView* and *Hough*, we do use the *Simple* matcher to improve personal identification accuracy.

6.2 Performance in the identification mode

To evaluate the performance of the identification mode, the 800 finger details in our database were divided into four databases of 200 details each. A total of 22,400 ($800 \times 7 \times 4$) identification operations were evaluated. When the system used a *Hough* matcher alone its EER was 2.27%, while the *TopView* matcher's EER was 2.84%.

We combined the *Hough* and *Simple* matchers, but the likelihood ratio was not utilized. Instead, the *Hough* matcher was used to match the input feature against all the templates in the database to find the best ten finger details. The *Simple* matcher was then employed to re-verify these ten details to find the best match. Figure 10 shows that this combination had an EER of 1.76%.

Table 1. Average computing time for one test on a 2.4 GHz Pentium 4.

Source	Process	Computing time (ms)
Top-view finger image	Preprocessing	193.95
	Feature Extraction	4.01
	Matching	0.02
	Reference point detection	19.05
Fingerprint	Preprocessing	119.64
	Feature Extraction	1.68
	Post Processing	281.45
	Matching (Algorithm <i>Hough</i>)	3.13
	Matching (Algorithm <i>Simple</i>)	135.80

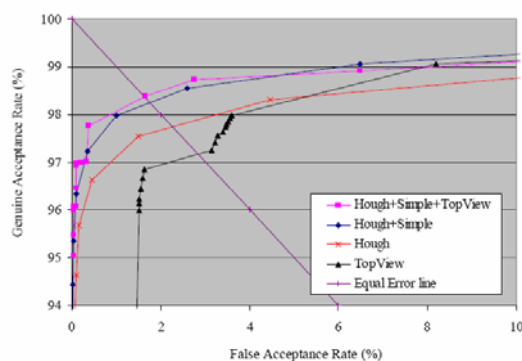


Figure 10. Performance in the identification mode.

Table 2. Equal Error Rate of the tested configurations.

Mode of Operation	Test Configuration	EER(%)
Verification	<i>TopView</i>	3.91
	<i>Hough</i>	3.80
	<i>Simple</i>	1.86
	<i>Hough+Simple</i>	1.64
	<i>TopView+Simple</i>	1.52
	<i>TopView+Hough</i>	2.35
	<i>TopView+Hough+Simple</i>	1.35
Identification	<i>Hough</i>	2.27
	<i>TopView</i>	2.84
	<i>Hough+Simple</i>	1.76
	<i>TopView+Hough+Simple</i>	1.64

When we combined the three matchers, we used the *TopView* matcher to verify the extracted input feature against all the templates in the database. The five best finger details from the *TopView* matcher were obtained and added to a candidate list. The *Hough* matcher was also utilized to search the database to find the five finger details with the highest matching scores, and they were also put into the candidate list. The *Simple* matcher re-verified all the finger details in the list, and the best match was found. This configuration had an EER of 1.64%. The Equal Error Rates of all experiments are summarized in Table 2.

The average computation time to perform 1:200 matches in the identification operation for the *TopView* and *Hough* matchers was 4 ms and 625 ms respectively. The *Simple* matcher required 1.358 seconds to perform 1:10 verifications in both the *Simple+Hough* and the *Simple+Hough+TopView* configurations.

7. Discussion and Conclusions

The NailCode feature reduces the verification error



Figure 11. Finger image captures at different times: (a) the initial image; (b) the same finger captured after 990 days had passed.

rate of the system by 17.68%. This value is obtained by comparing the results between the *Hough+Simple* and *Hough+Simple+TopView* configurations. In the identification mode, the system error is reduced by 6.82%. NailCode improves the accuracy of the fingerprint matching system, while requiring very low computation times, and being able to operate in both the identification and verification modes. We recommend that the NailCode matcher be used to increase the accuracy of fingerprint recognition systems.

Skin wrinkles on a finger will increase over time, but at a slow rate. For example, we have demonstrated that the same finger captured 990 days after its previous snapshot (see Figure 11) can still be correctly identified.

Since NailCode has lower accuracy than fingerprinting, it is recommended that top-view finger imaging should not be used alone to verify or identify individuals: it should be employed in conjunction with fingerprinting to improve overall recognition accuracy. These two features can be easily utilized together as part of one user operation. To keep the top-view feature up-to-date, biometric updating is recommended to overcome any time variances.

Acknowledgements

The authors are grateful to Dr. Andrew Davison for his kind help in polishing the language of this paper.

References

- Bradski, G. and Kaehler, A. 2008. Learning OpenCV, O'Reilly, U.S.A. pp. 248-250, 455-457.
- Canny, J. 1986. A Computational Approach to Edge Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence. 8(6), 679-698.
- Chaikan, P. and Karnjanadecha, M. The use of Top-View Finger Image for Personal Identification. International Symposium on Image and Signal Processing and

- Analysis:ISPA 2007. Istanbul, Turkey, September 27-29, 2007, 343-346.
- Duda, R.O., Hart, P.E. and Stork, D.G. 2000. *Pattern Classification* (2nd Edition), Wiley, New York, U.S.A., pp. 27, 52, 616.
- Gonzalez, R. and Woods, R. 2002. *Digital Image Processing* (2nd edition), Prentice-Hall, New Jersey, U.S.A., pp. 600-607.
- Hong, L. and Jain, A.K. 1998. Integrating faces and fingerprints for personal identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 20(12), 1295-1307.
- Jain, A.K., Hong, L. and Kulkarni, Y. 1999(a). A Multimodal Biometric System using Fingerprint, face and Speech. *Proceedings of International Conference on Audio-and-Video-Based Biometric Person Authentication*. 1999. 182-187.
- Jain, A.K., Prabhakar, S. and Hong, L. 1999(b). A multi-channel approach to fingerprint classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 21(4), 348-359.
- Maltoni, D., Maio, D., Jain, A.K. and Prabhakar, S. 2003. *Handbook of Fingerprint Recognition*. Springer, New York, U.S.A., pp. 15, 25-26, 146-147, 258-280.
- Marcialis, G.L. and Roli, F. 2004. Fingerprint verification by fusion of optical and capacitive sensors. *Pattern Recognition Letters*. 25(11), 1315-1322.
- Prabhakar, S. and Jain, A.K. 2002. Decision-level fusion in Fingerprint verification. *Pattern Recognition*. 35, 861-874.

VITAE**Name** Panyayot Chaikan**Student ID** 4713004**Educational Attainment**

Degree	Name of Institution	Year of Graduation
B.Eng (Computer Engineering)	King Mongkhut's Institute of Technology Ladkrabang	1999
M.Eng (Electrical Engineering)	King Mongkhut's Institute of Technology Ladkrabang	2002

Work

Appointed lecturer at the department of computer engineering, faculty of engineering, Prince of Songkla University, Thailand, since 2000 to present.