Songklanakarin J. Sci. Technol.
32 (1), 71-79, Jan. - Feb. 2010

Songklanakarin Journal of Science and Technology
http://www.sjst.psu.ac.th

*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.

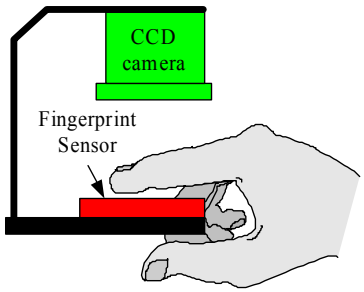


Figure 1. Top-view image and fingerprint recognition system.

*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 greyscale 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 greyscale 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 ±30° 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:

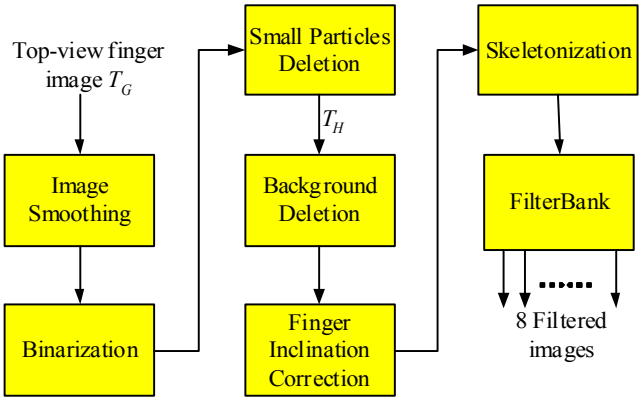


Figure 2. Flowchart of the preprocessing algorithm.

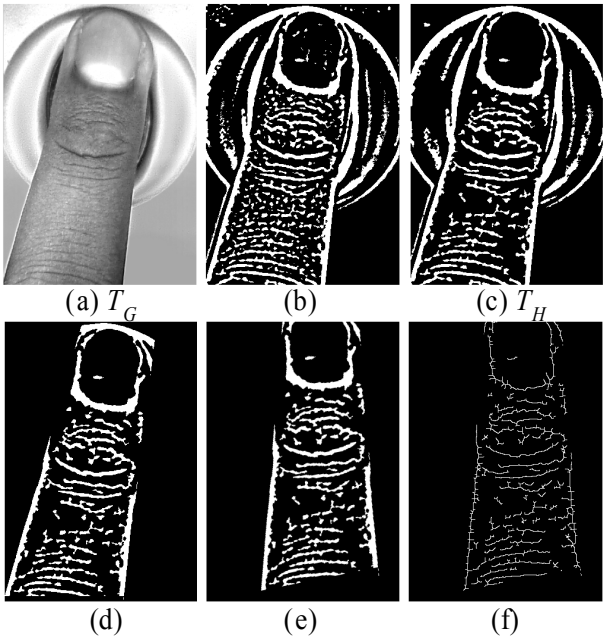


Figure 3. Images obtained in each preprocessing step.

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

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

2) The parameters for the left-edge of the finger are obtained by letting $C$ be the set of contours in $T_L$, where $C = \{C_1, C_2, C_3,..., C_k\}$ and $k$ = 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:

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

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:

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

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 $T_G$, is created using the condition:

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

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

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

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

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

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 $\mu$ and $\lambda$ be the angles of inclination of the left and right edges of the finger respectively, defined by:

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

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

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

### 3.5 Filterbank

The skeletonized top-view finger image is manipulated using a bank of oriented filters, with eight different $\theta$ 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 $\theta$ 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) $\theta$ is the degree setting on the oriented filter. The filtered image $Q_\theta$ 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 $Q_\theta$. The variance for each square cell at location $(h,v)$ is calculated using (Chaikan and Karnjanadecha, 2007):

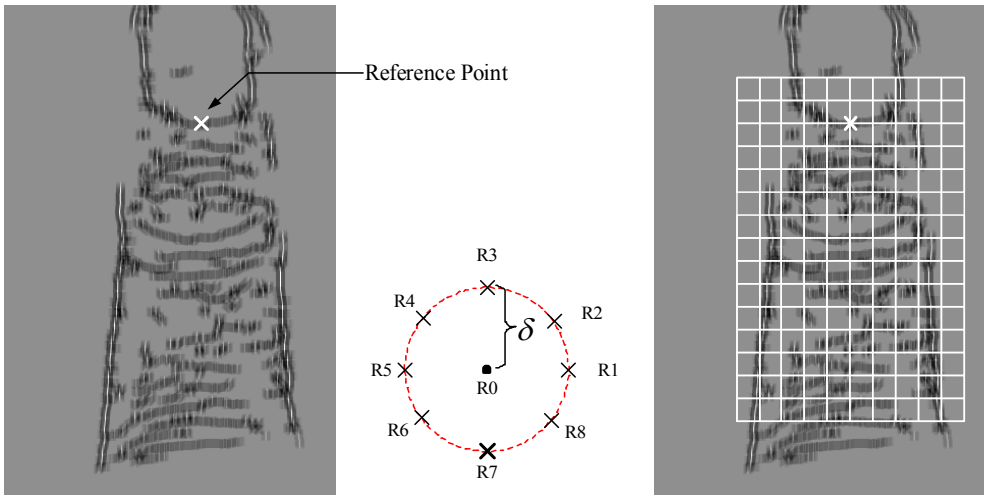


Figure 4. (a) Finger image filtered at 90° (b) Eight locations for reference point error compensation (c) square tessellation on the filtered image.

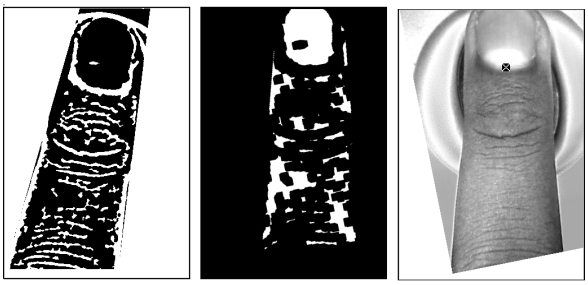


Figure 5. Top-view finger images for each step of the reference point detection algorithm.

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

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

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,..., $H$-1 and $v$ to range from -2,-1,0,..., $V$-3.

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

$$\Psi^i = \left\{\Psi^i_0, \Psi^i_{22.5}, \Psi^i_{45}, \Psi^i_{67.5}, \Psi^i_{90}, \Psi^i_{112.5}, \Psi^i_{135}, \Psi^i_{157.5}\right\} \quad (11)$$

where

$$\Psi_j = \left\{\sigma_1^2, \sigma_2^2, \sigma_3^2, ...., \sigma_{H\times V}^2\right\}.$$

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 $\delta$ (10) pixels from $R_0$, as shown in Figure 4(b). In the enrollment module, only $\Psi^0$ is extracted from the input top-view finger image. In the authentication module, the NailCode $\Psi = \left\{\Psi^0, \Psi^1, \Psi^2, ..., \Psi^8\right\}$ 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 $T_P$:

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

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 ℒ are deleted, giving ℒ = {}
        find contours from image T_P, putting all the results in ℒ
        for each contour L_i
            if (N_i>threshold and ratio_i>r_th)
                ref_pt_found = true
            else
                remove L_i from ℒ

        if (ref_pt_found =true)
            select contour L_j from 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:

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

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.

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

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:

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

### 3.8 Matching

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

$$matching\_score_{top} = \min(E_0, E_1, ..., E_8) \quad (14)$$

## 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,

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

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

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 ($\Delta\theta$ = -Φ ; $\Delta\theta \le \Phi$; $\Delta\theta$ += λ)
   $R^*$ 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 $Z$

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

  }

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

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, $\omega_1$ and $\omega_2$, where the first is the imposter and the other is the genuine class.

Define $\boldsymbol{X} = \{ x_1, x_2, .., x_n \}$ 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:

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

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 $\beta$ 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)

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

## 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

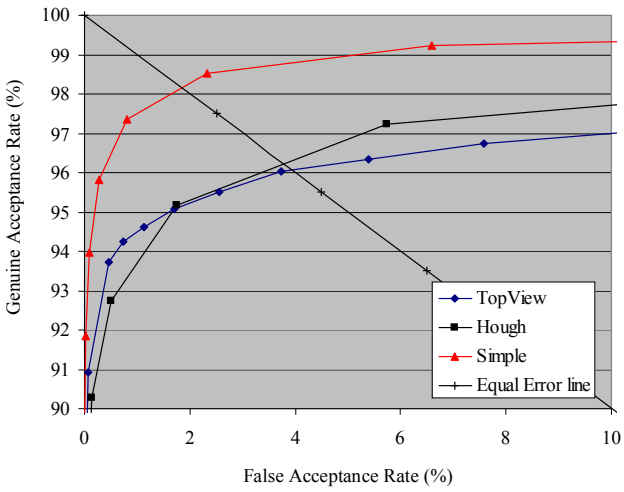


Figure 8. Verification performance of individual matchers.

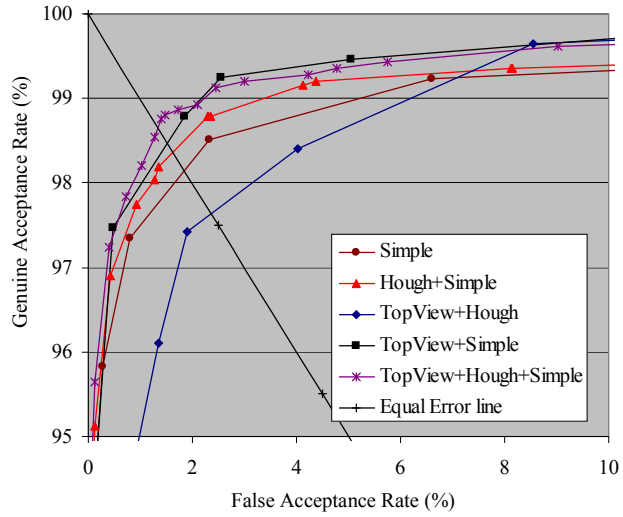


Figure 9. Verification performance of all combinations.

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 preprocessing 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*7*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 *Hough*) | 3.13 |
| | Matching (Algorithm *Simple*) | 135.80 |

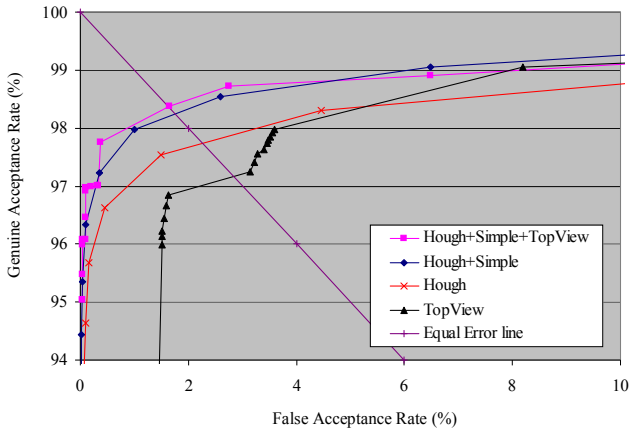


Figure 10. Performance in the identification mode.

(a) (b)

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

Table 2. Equal Error Rate of the tested configurations.

| Mode of Operation | Test Configuration | EER(%) |
|---|---|---|
| Verification | *TopView* | 3.91 |
| | *Hough* | 3.80 |
| | *Simple* | 1.86 |
| | *Hough+Simple* | 1.64 |
| | *TopView+Simple* | 1.52 |
| | *TopView+Hough* | 2.35 |
| | *TopView+Hough+Simple* | 1.35 |
| Identification | *Hough* | 2.27 |
| | *TopView* | 2.84 |
| | *Hough+Simple* | 1.76 |
| | *TopView+Hough+Simple* | 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 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 multichannel 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 verifcation. Pattern Recognition. 35, 861-874.