Biometric authentication system using fingerprints based on self-organizing map neural network classifier

M.A. El-Iskandarani [a] and H.M. Abdul-Kader [b]
[a] *Institute of Graduate Studies and Research, University of Alexandria, Alexandria, Egypt*
[b] *Faculty of Computers and Information, Monyfia University, Monyfia, Egypt*
meskand@yahoo. com, hatem6803@yahoo. com

Biometric recognition refers to the use of distinctive physiological and behavioral characteristics (e.g., fingerprint, face, and signature) for automatically recognizing or authentication of a person**.** A number of biometric technologies have been developed and several of them are being used in variety of fields. Among all biometric technologies, fingerprint based identification is the most common method which has been successfully used in numerous applications. The aim of this research is the development of a biometric authentication system based on individual fingerprint and PIN code. This paper briefly presents the developed system architecture, demonstrating the elements, and techniques adopted for fingerprint image enhancement, minutiae extraction, and matching. The classification technique is based on the self-organizing map neural network. A computer program written in MATLAB programming language is designed using several tool boxes. Experimental results show acceptable classification/identification accuracy based on the five clustering system.

يشير التعرف Biometric إلى استعمال الخصائص الفسيولوجية والسلوكية المتميزة (مثل بصمة الأصابع، الوجه، التوقيع) للتعرف الآلى أو التحقيق الشخصى. عدد من تقنيات Biometric قد طورت وعديد يستعمل فى كثير من المجالات، من أهمها التعرف بإستخدام بصمات الأصابع وهى الطريقة الأكثر شيوعاً التى استعملت بنجاح فى تطبيقات عديدة. الهدف من البحث تطوير نظام تحقيق مستند على بصمة الأصابع ورقم شفرة خاصمة بالفرد. تقدم هذه الورقة تطوير معمارية النظام مع عرض للعناصر، والتقنيات التى اختيرت لتحسين صورة الإصبع ثم استخراج وتطابق التفاصيل الدقيقة للبصمة، كما إن تقنية التصنيف تعتمد على الشبكة العصبية ذات خريطة التنظيم الذاتى. تم تصميم برنامج حاسوب بلغة برمجة MATLAB مع استخدام عدد من صناديق الأدوات وقد أعطت التجارب نتائج مقبولة مبنية على نظام الخمسة تجمعات.

Keywords: Biometric, Fingerprints, Enhancement, Minutiae extraction, Self organizing map

1. Introduction

Computerized automated security systems are getting more important nowadays. The key task of an automated security system is to verify that the users are in fact who they claim to be [1], [2]. The security system could ask the user to provide some information or to provide something only the user has access to, or it could identify some sort of trait that is unique for the user. Of course, some sort of combination of these methodologies is also possible [2], [3]. Fingerprint identification is commonly employed to support criminal investigations, and in biometric systems such as civilian and commercial identification devices.

The fingerprint of an individual is unique and remains unchanged over a lifetime. A fingerprint is composed of many ridges and furrows and is recognized from an impression of the pattern of ridges on a finger. A ridge is defined as a single curved segment, and a valley is the region between two adjacent ridges. The minutiae, which are the local discontinuities in the ridge flow pattern, provide the features that are used for identification. Details such as the type, orientation, and location of minutiae are taken into account when performing minutiae extraction [3], [4].

Fingerprint images are rarely of perfect quality; they may be degraded and corrupted with elements of noise due to many factors including variations in skin and impression conditions. This degradation can result in a significant number of spurious minutiae being created and genuine minutiae being ignored. A critical step in studying the statistics of fingerprint minutiae is to reliably extract Minu-

tiae from fingerprint images. Thus, it is necessary to employ image enhancement techniques prior to minutiae extraction to obtain a more reliable estimate of minutiae locations [5], [6].

Experiments using both synthetic test images and real fingerprint images are used to assess the performance of the developed system. This paper is organized into three main topics. Section 2 gives the architecture of the proposed authentication system, where the methodology of a series of techniques for fingerprint image enhancement is discussed. Then commonly used methods of minutiae extraction, fingerprint image post processing and fingerprint classification and identification. SOM neural network classifier has been adopted to classify fingerprint images into the five clusters: left loop, right loop, arch, tented arch and whorl. Section 3, presents system implementation using several MATLAB tool boxes, namely; Image processing toolbox, Neural network toolbox, Database toolbox and GUI tool box, and shows typical output screens. In section 4; we present our experimental results, followed by conclusions.

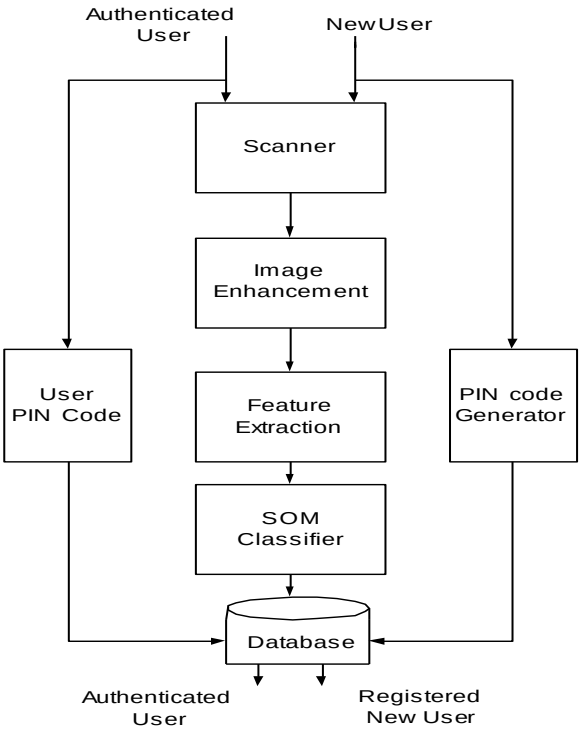


Fig. 1. System architecture.

2. Authentication system architecture

In this section a general overview for the proposed biometric authentication system is described. This system mainly depends on the person fingerprint and associated PIN code with sixteen numbers given to the authenticated person at the registration stage. Fingerprint authentication has many stages namely:
Image acquisition (capture the image by finger scanning).
Image enhancement techniques.
Features extraction processes.
Fingerprints pattern classification.

The proposed system architecture is shown in fig. 1; the details for each stage will be discussed in the following sections.

2.1. Fingerprint enhancement and processing

In order to ensure that the performance of an automatic fingerprint classification/ matching (identification/verification) system will be robust with respect to the quality of the fingerprint images, it is essential to incorporate a fingerprint enhancement algorithm. In an ideal fingerprint image, ridges and valleys flow in locally constant directions and minutiae can easily be detected by following the detected ridgelines to their endings and bifurcations. Actually, the fingerprint image can get corrupted due to a number of reasons [7]. This leads to the following problems:

- A number of spurious minutiae might be introduced
- Number of genuine minutiae might be lost.
- Large errors in minutiae orientation and position may be introduced.

To overcome these problems, a number of fingerprint image enhancement techniques have been proposed in the literature [8], [9]. In this paper the image enhancement processes which will be followed are shown in fig. 2. A Short overview for each process of the image enhancement is discussed blow.

2.1.1. Segmentation

Segmentation is the process of separating the foreground regions in the image from the background regions. In a fingerprint image,

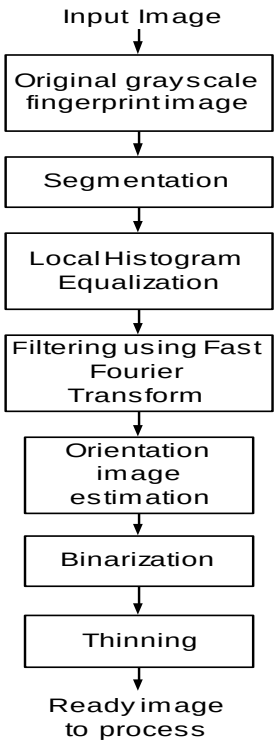


Fig. 2. Sequence of enhancement procedure.

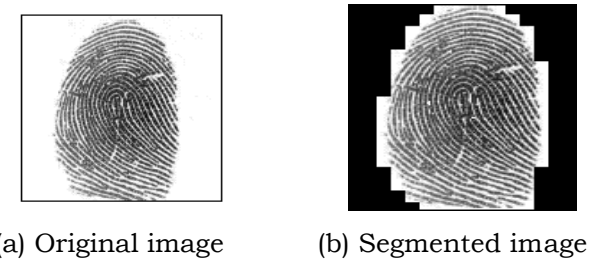
the background regions generally exhibit a very low grey-scale variance value, whereas the foreground regions have a very high variance. Hence, a method based on variance threshold can be used (fig. 3). To implement the segmentation process, firstly, the image is divided into blocks and the grey-scale variance is calculated for each block in the image. If the variance is less than the global threshold, then the block is assigned to be a background region; otherwise, it is assigned to be part of the foreground [10]. The gray-level variance for a block of size W is defined as:

$$V(k) = \frac{1}{W^2} \sum_{i=0}^{W-1} \sum_{j=0}^{W-1} (I(i,j) - M(k))^2 \quad . \qquad (1)$$

Where $V(k)$ is the variance for block k, $I(i, j)$ is the grey-level value at pixel (i, j), and M (k) is the mean grey-level value for the block k.

2.1.2. Contrast enhancement

Histogram equalization define a mapping of gray levels p into gray levels q. Such that



(a) Original image (b) Segmented image

Fig. 3. Result of segmentation using a variance threshold of 100.

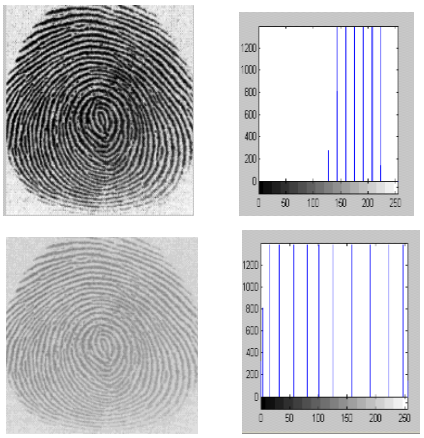


Fig. 4. Contrast enhancements : original image and histogram (up), enhanced image and histogram (down).

the distribution of gray level q is uniform this mapping stretches contrast (expand the range of gray levels) for gray levels near the histogram maxima. Since contrast is expanded for most of the image pixels, the transformation improve the detection ability of many image features. The highest and lowest value pixels are used in the transformation. The equation is:

New pixel=((*old pixel-low*) / (*high-low*))*255. (2)

The visualization effect, and the histogram before and after equalization are shown in fig. 4.

2.1.3. Fingerprint enhancement by fourier transform

In this algorithm, we divide the image into small processing blocks (32 by 32 pixels) and perform the Fourier transform according to the following equation:

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{Y=0}^{N-1} f(x,y)\times \exp\left\{-j2\pi\times\left(\frac{ux}{M}+\frac{vy}{N}\right)\right\}. \quad (3)$$

For u = 0, 1, 2, 31 and v = 0, 1, 2... 31.

In order to enhance a specific block by its dominant frequencies, we multiply the FFT of the block by its magnitude a set of times. Where the magnitude of the original FFT = $|F(u, v)|$.

Get the enhanced block according to:

$$g(x, y) = F^{-1}\left\{F(u,v)\times|F(u,v)|^{k}\right\}. \quad (4)$$

Where $F^{-1}(F(u, v))$ is done by:

$$f(x,y) = \frac{1}{MN}\sum_{x-0}^{M-1}\sum_{y-0}^{N-1} F(u,v)x\exp\left\{j2\pi x\left(\frac{ux}{M}+\frac{vy}{N}\right)\right\}. \quad (5)$$

For x = 0, 1, 2... 31 and y = 0, 1, 2, ..., 31.

The k in formula (4) is an experimentally determined constant. We found that k=0.45 is an optimal choice. While having a higher "k" improves the appearance of the ridges, filling up small holes in ridges, having too high a "k" can result in false joining of ridges. Thus a termination might become a bifurcation. Fig. 5 presents the image after FFT enhancement

2.1.4 Normalization

Normalization is used to standardize the intensity values in an image by adjusting the range of grey-level values so that it lies within a desired range of values [8]. Let $I(i, j)$ represent the grey-level value at pixel ($i. j$), and $N(i, j)$ represent the normalized grey-level value at pixel ($i; j$). The normalized image (fig. 6) is defined as:

$$N(i,j) = \begin{cases} M_0 + \sqrt{\dfrac{V_0(I(i,j)-M)^2}{V}} & if\ I(i,j) \geq M, \\ M_0 - \sqrt{\dfrac{V_0(I(i,j)-M)^2}{V}} & Otherwise, \end{cases} \quad (6)$$

Where M and V are the estimated mean and variance of I (i, j), respectively, and M_0 and V_0 are the desired mean and variance values, respectively. Normalization does not change the ridge structures in a fingerprint; it is performed to standardize the dynamic levels of variation in grey-level values, which facilitates the processing of subsequent image enhancement stages.

2.1.5. Orientation estimation

The orientation field of a fingerprint image defines the local orientation of the ridges contained in the fingerprint. The least mean square estimation method has been used to compute the orientation image [5]. The steps for calculating the orientation at pixel (i, j) are given as follows:

1. Firstly, a block of size W x W is centered at pixel (i, j) in the normalized fingerprint image.
2. For each pixel in the block, compute the gradients $\partial_x(i, j)$ and $\partial_y(i, j)$, which are the gradient magnitudes in the x and y directions, respectively. The horizontal and vertical Sobel operators are used to compute $\partial_x(i, j)$ and $\partial_y(i, j)$:

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad , \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

3. The local orientation at pixel (i, j) can then be estimated using the following equations:

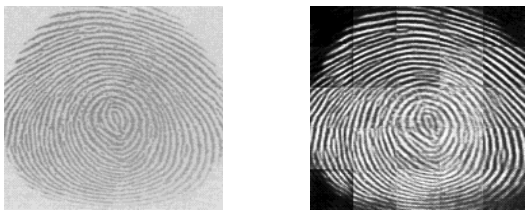


Fig. 5. Fingerprint enhancements by FFT: original image (left), enhanced image (right).

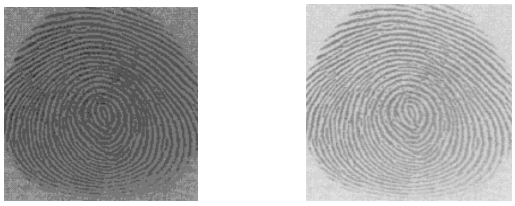


Fig. 6. Result of fingerprint normalization (right), original image (left).

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

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

$$\theta(i,j) = \frac{1}{2}\tan^{-1}\frac{v_y(i,j)}{v_x(i,j)}, \quad (7)$$

Where $\theta(i,j)$ is the least square estimate of the local orientation at the block centered at pixel (i,j).

4. Smooth the orientation field in a local neighborhood using a Gaussian filter. The orientation image is firstly converted into a continuous vector field, which is defined as:

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

$$\Phi_y(i,j) = \sin(2\theta(i,j)). \quad (8)$$

Where Φ_x and Φ_y are the x and y components of the vector field, respectively. After the vector field has been computed, Gaussian smoothing is then performed as follows:

$$\Phi'_x(i,j) = \sum_{u=\frac{-w\phi}{2}}^{u=\frac{w\phi}{2}} \sum_{v=\frac{-w\phi}{2}}^{\frac{w\phi}{2}} G(u,v)\Phi_x(i-uw, j-vw),$$

$$\Phi'_y(i,j) = \sum_{u=-\frac{w\phi}{2}}^{\frac{w\phi}{2}} \sum_{v=-\frac{w\phi}{2}}^{\frac{w\phi}{2}} G(u,v)\Phi_y(i-uw, j-vw), \quad (9)$$

Where G is a Gaussian low-pass filter of size (w_Φ x w_Φ).

5. The final smoothed orientation field O (fig. 7) at pixel (i,j) is defined as:

$$O(i,j) = \frac{1}{2}\tan^{-1}\frac{\Phi'_y(i,j)}{\Phi'_x(i,j)}. \quad (10)$$

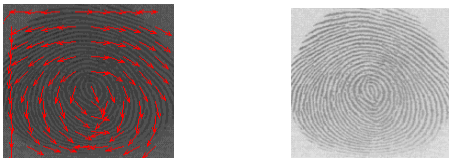


Fig. 7. Fingerprint orientation (right), original image (left).

2.1.6. Fingerprint image binarizations

Fingerprint Image Binarization is to transform the 8-bit gray fingerprint image to a 1-bit image with 0-value for ridges and 1-value for valleys [8]. After the operation, ridges in the fingerprint are highlighted with black color while valleys are white. A locally adaptive binarization method is performed to binarize the fingerprint image. Such a named method comes from the mechanism of transforming a pixel value to 1 if the value is larger than the mean intensity value of the current block (16x16) to which the pixel belongs (see fig. 8).

2.1.7. Thinning

The final image enhancement step typically performed prior to minutiae extraction is thinning. Thinning is a morphological operation that successively erodes away the foreground pixels until they are one pixel wide [8]. A standard thinning algorithm is employed, which performs the thinning operation using two sub-iterations. This algorithm is accessible in MATLAB via the `thin' operation under the *bwmorph* function. Each sub-iteration begins by examining the neighborhood of each pixel in the binary image, and based on a particular set of pixel-deletion criteria, it checks whether the pixel can be deleted or not. These sub iterations continue until no more pixels can be deleted (fig. 9). The application of the thinning algorithm to a fingerprint image preserves the connectivity of the ridge structures while forming a skeletons version of the binary image. This skeleton image is then used in the subsequent extraction of minutiae. The process involving the extraction of minutiae from a skeleton image will be discussed in the next section.

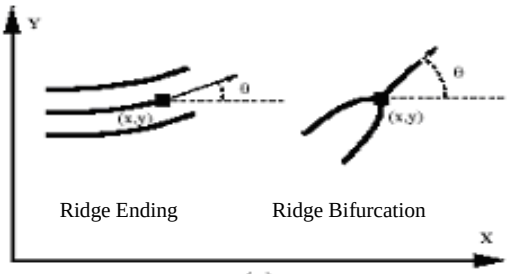
2.2. Fingerprint image feature extraction

After a fingerprint image has been enhanced, the next step is to extract the features from the enhanced image. These features as

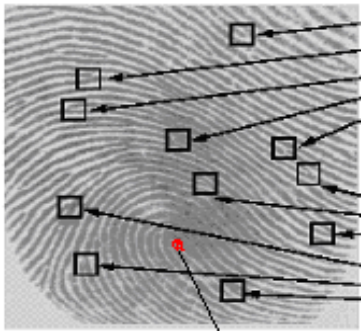
shown in fig. 10 are divided into two types: a) Features for classification and alignments (core and delta points). b) Features for matching called minutiae (bifurcation and termination) [12, 13]. This section provides discussion on the methodology and implementation of techniques for core point extraction, minutiae extraction and fingerprint image post processing.

Fig. 8 Image after adaptive binarization: Binarized image (left), Enhanced gray image (right)

Fig. 9. Fingerprint image after thinning: Binarized image (left), thinned image (right).



(a)



(b)

Fig. 10. fingerprint features.

2.2.1. Core point extraction

Most of the approaches proposed in the literature for singularity detection (core and delta points) operate on the fingerprint orientation image as shown in fig. 11.

These singularity points are highly stable and also scale and rotation invariant. Thus, the number and location of core and/or delta points are widely used by most of the classification methods. Table (1) shows the number and position of the core and delta points for the five pattern classes. There are many methods used for the core detection points such as Linear Phase Portraits, and Poincare algorithm [3, 13]. Poincare index which is derived from continuous curves is the most popular one, due its simplicity and considerable accuracy. For digital fingerprint images, a double core point has a Poincare index valued as 1, a core point 1/2 and a delta point -1/2.

The Poincare index at pixel (i,j) namely depends on the orientation O $(x_k\ ,\ y_k\ )$ which is obtained from eq. (10). This index is enclosed by a digital curve (with Np points) can be computed as follows:

$$Poincar(i,j) = 1/(2\pi)\sum_{k=0}^{N_{p-1}} \Delta(k). \qquad (11)$$

Where

$$\Delta(k) = \begin{cases} \delta(k) & if\ |\delta(k)| < \pi/2 \\ \pi + \delta(k) & if\ \delta(k) \le -\pi/2 \\ \pi - \delta(k) & otherwise \end{cases}$$

$$\delta(k) = O(x_{(k+1) \bmod Np},\ y_{(k+1) \bmod Np}) - O(x_k, y_k)$$

and it goes in a counter-clockwise direction from *0* to *Np-1*. For our method, *Np* is selected

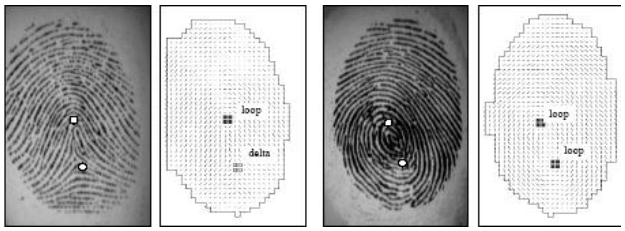


Fig. 11. fingerprint core points.

Table 1
Fingerprint pattern classes and the corresponding number of singular points

Pattern class	Core	Delta
Arch	0	0
Tented arch	1	1(middle)
Left loop	1	1(right)
Right loop	1	1(left)
Whorl	2	2

as 4. Then the nearest neighbor cluster method is used to remove the false ones. Any types of potential singular points within some distance of each other are regarded as one cluster. After the clustering, we analyze the numbers of cores *Nci* and deltas *Ndi* in cluster *i*.

1. If *Nci* = *Ndi*, no singular point exists;
2. If *Nci* - *Ndi=1*, the geometric center is regarded as a core point;
3. If *Nci* - *Ndi* =*-1*, the geometric center is regarded as a delta point;
4. If *Nci* - *Ndi* =*2*, the geometric center is regarded as a double core;
5. Otherwise, no singular points exist in the cluster.

After this, there will be verification of such singular points. Five windows centered on the singular point are used to calculate their Poincare indices and the window size increases by 3 pixels. The singular points for which most Poincare indices conform to its type are verified. The numbers of cores and deltas are *Nc* and *Nd*.

2.2.2. Minutiae extraction

The most commonly employed method of minutiae extraction is the Crossing Number (*CN*) concept [8, 14]. This method involves the use of the skeleton image where the ridge flow pattern is eight-connected. The minutiae are extracted by scanning the local neighborhood of each ridge pixel in the image using a 3x3 window. The *CN* value is then computed as in eq. (12), which is defined as half the sum of the differences between pairs of adjacent pixels in the eight-neighborhood. Using the properties of the *CN* as shown in fig. 12, the ridge pixel can then be classified as a ridge ending, bifurcation or non-minutiae point. For example, a ridge pixel with a *CN* of one corresponds to a ridge ending, and a *CN* of three corresponds to a bifurcation.

$$CN = 0.5\sum_{i=1}^{8}|Pi - Pi+1|, \qquad P9 = P1 \tag{12}$$

where P_i is the pixel value in the neighborhood of *P*. For a pixel *P*, its eight neighboring pixels are scanned in an anti-clockwise direction as follows:

After the *CN* for a ridge pixel has been computed, the pixel can then be classified according to the property of its *CN* value. As shown in fig.12, a ridge pixel with a *CN* of one corresponds to a ridge ending, and a *CN* of three corresponds to a bifurcation. For each extracted minutiae point both - *x* and *y* coordinates and the orientation of the associated ridge segment, and type of minutiae are recorded [8].

2.2.3 Fingerprint image post processing

False minutiae may be introduced into the image due to factors such as noisy images, and image artifacts created by the thinning process. Hence, after the minutiae are extracted, it is necessary to employ a post processing stage in order to validate the minutiae. Fig. 13 illustrates some examples of false minutiae structures, which include the spur, hole, triangle and spike structures [15].

The majority of the proposed approaches for image post processing in literatures are based on a series of structural rules used to

CN	Property
0	Isolated point
1	Ridge Ending point
2	Continuing ridge point
3	Bifurcation point
4	Crossing point

P_4	P_3	P_2
P_5	P	P_1
P_6	P_7	P_8

Fig. 12. Properties of the crossing number.

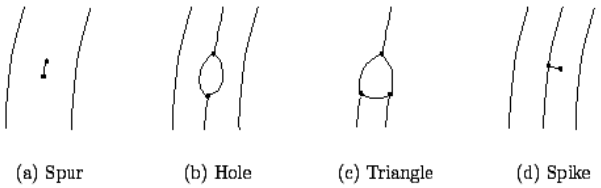


Fig. 13. Examples of typical false minutiae structures.

eliminate spurious minutiae [5]. A novel approach to the validation of minutiae is the post processing algorithm proposed by Tico and Kuosmanen [16]. However, rather than employing a different set of heuristics each time to eliminate a specific type of false minutiae, this approach incorporates the validation of different types of minutiae into a single algorithm. It tests the validity of each minutiae point by scanning the skeleton image and examining the local neighborhood around the minutiae. The algorithm is then able to cancel out false minutiae based on the configuration of the ridge pixels connected to the minutiae point. In this paper, we have implemented the minutiae validation algorithm which is given in [16]. This algorithm tests the validity of each minutiae point by scanning the skeleton image and examining the local neighborhood around the point. The first step in the algorithm is to create an image M of size *W x W*, where M corresponds to the *W x W* neighborhood centered on the candidate minutiae point in the skeleton image. The central pixel of M corresponds to the minutiae point in the skeleton image, and so this pixel is labeled with a value of -1. The rest of the pixels in M are initialized to values of zero. The subsequent steps of the algorithm depend on whether the candidate minutiae point is a ridge ending or a bifurcation.

2.2.3. (a) For a candidate ridge ending point

(1) Firstly, label with a value of 1 all the pixels in M, which are eight-connected with the ridge ending point (see fig. 14-b).

(2) The next step is to count in a clockwise direction, the number of 0 to 1 transitions (T01) along the border of image M. If T01 = 1, then the candidate minutiae point is validated as a true ridge ending.

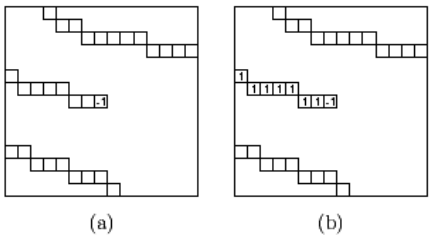


Fig. 14. Example of validating a candidate ridge ending point. T01 = 1.

2.2.3. (b) For a candidate bifurcation point

(1) Firstly, examine the eight neighboring pixels surrounding the bifurcation point in a clockwise direction. For the three pixels that are connected with the bifurcation point, label them with the values of 1, 2, and 3, respectively. An example of this initial labeling process is shown in fig.(15-b).

(2) The next step is to label the rest of the ridge pixels that are connected to these three connected pixels. This labeling is similar to the ridge ending approach, however, instead of labeling a single ridge branch, three ridge branches are now labeled. Let l = 1, 2 and 3 represent the label for each ridge branch. For each l, label with l all the ridge pixels that have a label of 0, and are connected to an l labeled pixel. Examples of the bifurcation labeling process are shown in figs. 15 (c, d and e).

(3) The last step is to count in a clockwise direction, the number of transitions from 0 to 1 (T01), 0 to 2 (T02), and 0 to 3 (T03) along the border of image M. If T01 = 1 ^ T02 = 1 ^ T03 = 1, then the candidate minutiae point is validated as a true bifurcation.

The algorithm is also able to cancel out these two types of false minutiae shown in fig. 16. Each bifurcation point in the hole structure can be eliminated due to the number of zero to two transitions (T02) along the border of the image window not being equal to zero. The spur structure contains only two ridge pixels in the centre of the image window, which means the ridge branch connected to the minutiae is not long enough to reach the window border. Hence, the ridge ending points cannot be validated as true minutiae, since the number of zero to one transitions (T01) along the border of the window is zero. To effectively eliminate the false minutiae the recommended window size is 23x23. If the window size is too small, results have shown that the algorithm is not effective in canceling out the false minutiae. Conversely, if the window size is too large, then the algorithm may incorrectly cancel out minutiae [16].

2.3. Classification techniques

Fingerprint classification, which classifies fingerprint images into a number of pre-

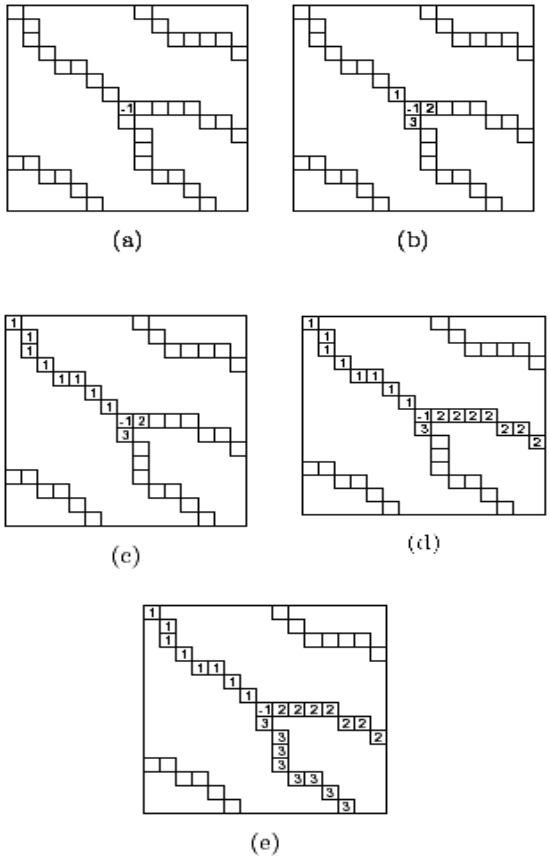


Fig. 15. Example of validating a candidate bifurcation point: T01=1^T02 =1 ^ T03 =1.

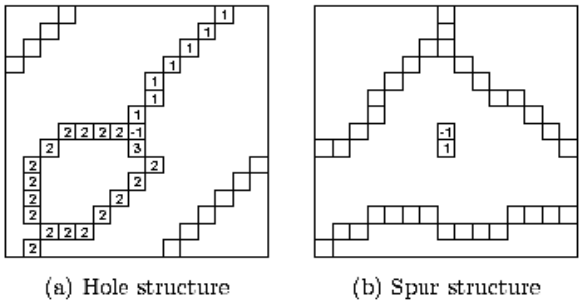


Fig. 16. An example of how the post processing algorithm cancels out the false minutiae. a window size of 23 x 23 is used, (a)T01 = 1 ^ T02 = 1 ^ T03 = 0, (b) T01 = 0.

defined categories, facilitates the matching task accordingly [12,13]. In classification stage, the features (e.g. singular points) are given to a self organizing map neural network to classify similar images into one cluster (left loop, right loop, arch, tented arch , and whorl) to reduce the search time, while in the matching stage the features (ridge endings and bifurcations) are given to a matching engine to follow the steps of identification/ verification, thus the input fingerprint is required to be matched only with its subset of fingerprints in the database [17].

2.3.1. Self organizing map neural network classifier

The Self-Organizing Map (SOM) is an unsupervised learning technique that reduces the dimensionality of data through the use of a self-organizing neural network**.** The SOM is typically applied to data in which specific classes or outcomes are not known in prior, and training is done in unsupervised mode [18]. In this case, SOM can be used to understand the structure of the input data, and in particular, to identify clusters of input records, that have similar characteristics in the high-dimensional input-space [19].

The most important practical applications of SOM are in exploratory data analysis, pattern recognition, speech analysis, robotics, industrial and medical diagnostics, instrumentation and control. The SOM (fig. 17) usually has an output layer of interconnected neurons that are fully connected to the input layer, so every neuron from the output layer is connected to every neuron in the input layer. Self-organizing feature maps SOM transform the input of arbitrary dimension into one or two dimensional discrete map subject to a topological (neighborhood preserving) constraint. The feature maps are computed using Kohonen unsupervised learning algorithm [19]. SOM with all nodes in input layer connected to all nodes in output layer, each input has a weight vector, this weight is initialized with random values. During the training, that weight adapts with each output change according to the algorithm used. In self-organizing maps, however, not only the weights of the connections to the winner neuron are adapted. Rather, there is a neighborhood relation defined on the competition layer which indicates which weights of other neurons should also be changed. This neighborhood relation is usually represented as a (usually two-dimensional) grid, the vertices of which are the neurons. This grid most often is rectangular or hexagonal.

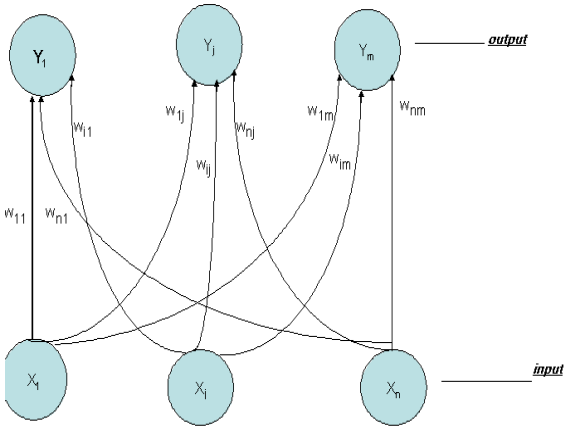


Fig. 17. Basic SOM structure.

2.3.1.(a) Algorithm for self organizing map

Assume output nodes are connected in an array (usually 1 or 2 dimensional) and the network is fully connected (all nodes in input layer are connected to all nodes in output layer), the learning algorithm [18, 19] of such network can be described as follows:

Step 0: Initialize weights *W*ij, set topological neighborhood parameters(R), set learning rate parameters (α).

Step 1: While stopping condition is false, do step 2-8.

Step 2: For each input vector *x*,do step 3-5.

Step 3: For each *j*, compute: $D(j) = \sum (W_{ij}-X_i)^2$

Step 4: Find index *J* such that *D*(*J*) is minimum.

Step 5: For all units *j* within a specified neighborhood of *J*, and for all *i*:

$$W_{ij}(new) = W_{ji}(old) + a[x_i - W_{ij}(old)]. \quad (13)$$

Step 6: Update learning rate.

Step 7: Reduce radius of topological neighborhood at specified times.

Step 8: Test stopping condition.

In this type of neural network, the learning rate is kept large at the beginning of training epochs and decreased gradually as learning proceeds. SOM has been used in this research as a basic classifier, where each fingerprint image is described by 256*256 pixels divided into 16*16 blocks. The orientation angle of each block is used as input to the neural network so the input layer consists of 256 nodes. Each image can fall into one of the five clusters (right loop, left loop, whorl, arch, tented arch), so the output layer consists of five nodes. The learning rate which we used is a = 0.5 and neighborhood radius R= 0. This process is designed to facilitate the identification process whenever the system is used, thus the searching time will be reduced as the system search only in one cluster.

2.4. Fingerprint minutiae match

The objective of the feature matching system is to determine whether or not the prints represent the same finger. There are a few issues to consider when trying matching two sets of minutiae [20]. Since skin is elastic, two corresponding minutia points might not be in exactly the same place in two sets. The fingerprints might have been rotated in different angles in the two images. One must also find some sort of common reference point for the two sets, this reference point in our research is the *core* point. The matching phase typically defines a metric of the similarity between two fingerprint representations. The matching stage also defines a threshold to decide whether a given pair of representations belongs to the same finger (matched pair) or not. To implement the matching stage we use the following algorithm [8], [20]:

1- Given the minutia position *(x, y)* and its orientation angle *θ*.

2- Calculate new *x* , *y* and *θ* $(x`,y`,\theta`)$ relative to core point position $(C_x, C_y, C_\theta)$ $x`=x- C_x$, $y`=y- C_y$ *and* $\theta`=\theta- C_\theta$

3- Transform the minutia position from Cartesian $(x`, y`)$ to polar *(e, r)* coordinates. $e= \tan^{-1}(y`/x`)$, and $r= (x`^2+y`^2)^{1/2}$

4- Assume that $(r`, e`, \theta`)$ of input image is (r_i, e_i, θ_i) and for template image (r_t, e_t, θ_t), then calculate:

r-difference = $r_i - r_t$, *e-difference* = $e_i - e_t$ *and* *θ-difference* = $\theta_i - \theta_t$

5- If (*r*-difference < *r*-threshold), (*e*-difference < *e*-threshold) and (*θ-difference* < *θ*-threshold) then the two minutia are regarded as a matched minutia pair

6- Calculate the final match ratio = (No. of matched pairs / No. of minutia of the template fingerprint) *100.

7- If the score is larger than a pre-specified threshold, then the two fingerprints are from

the same finger.

The matching algorithm needs to be elastic since the strict match requiring that all parameters (e, r, θ) are the same for two identical minutia which might be impossible, because fingerprints might have been rotated in different angels. In this paper the elastically match minutia was achieved by placing a bounding box around each template minutia [22], if the minutia to be matched is within the rectangle box and the direction discrepancy between them is very small [23]. In this paper, the algorithm achieved elastic match by using the threshold values (r-threshold= 8, e-threshold= 7, θ-threshold=15).

3. System implementation and typical output screens

The proposed system is implemented using MATLAB 6.5. Four MATLAB toolboxes are used, namely: Image processing toolbox, Neural network toolbox, Database toolbox and GUI tool box. The system is designed via separate M-files, where the GUI call each file whenever it is needed. Graphical user interface is built using GUI toolbox, where in all screens the user activate certain button. When the user press any button an M-file starts to run according to the function of the button. Also a database is created using SQL sever 2000 and linked to the system via MATLAB Database toolbox [21]. This database consists of five tables, each table corresponds to a class of fingerprints. An extra table is used to store the PIN code corresponding to each fingerprint registered in the system fig. 18.

The starting screen gives the user two choices. The first one, if the user wants to register a person in the system, while the second one if the candidate was already registered and we want to authenticate himself by the system. At the first case of new registration a unique PIN code will be automatically generated from 16 numbers. Then, it is requested from the candidate to enter his fingerprint via a special scanner. The system captures this fingerprint and starts operation following the steps according the flow chart shown in fig. 2. Then the candidate PIN code and its processed fingerprint are stored in the

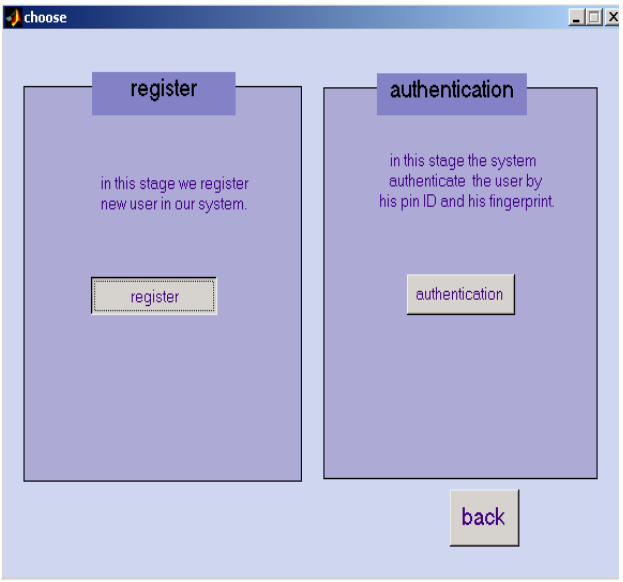


Fig. 18. Register and authentication screen.

database. The PIN code which is generated for the new candidate is unique, as the system generates a random PIN code and then check this number with the already registered PIN codes for other users. In the second case, there are two functions that can be executed, namely; verification and identification. First in the verification phase, the candidate gives the system its PIN code and its captured fingerprint. The system starts the image enhancement processes according to the steps shown in fig. 2, then starts the matching process according to the described algorithm in section II-4. Then the system check the PIN code with the registered one in the database. Finally the decision of the system that the user is authenticated if the PIN code is correct and the resulted matching ratio is greater than 65%.

The second phase is the identification one, where the identified image is given to the system. The system starts its image enhancement processes according to the steps shown in fig. 2. Then the SOM neural network classify this image to its respective cluster. Then the system starts to extract the minutiae of the given fingerprint and starts to search in the specified cluster, to determine the right candidate. Several output screens are displayed in figs. 19, 20, and 21.

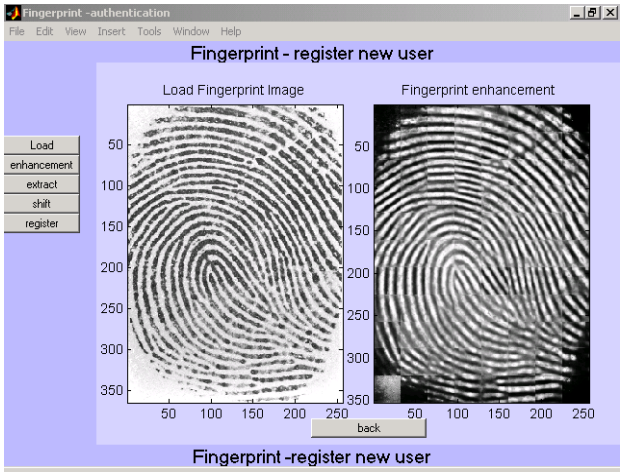


Fig. 19. Enhancement operation is displayed in this screen.

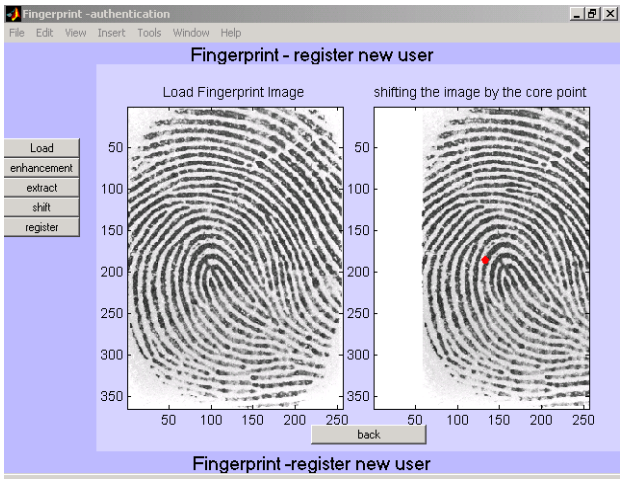


Fig. 20. Transfer the core point in the center of image.

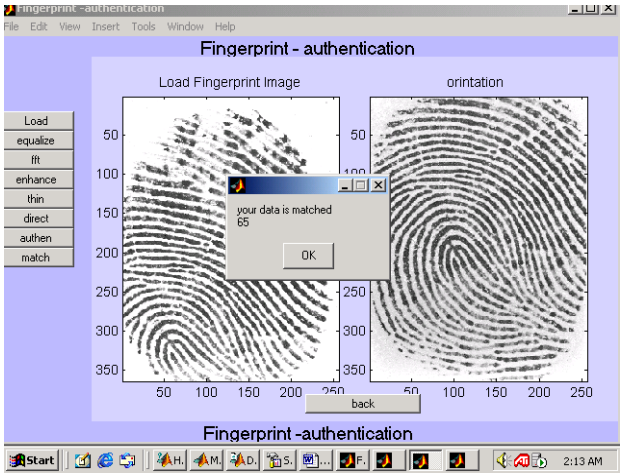


Fig. 21. Matching operation output.

3.1. Experimental results

A fingerprint database from FVC2000 (Fingerprint Verification Competition 2000) which can be freely accessed [22] is used to test the experiments performance. The developed system tests all the images without any fine tuning for the database. The experiments show that the system can differentiate false minutiae pairs from real minutia pairs with certain confidence level. Also a real 150 fingerprints were collected randomly to verify the system. Those real finger prints were collected via the ink that is used in the security console, and the images were captured via 500 dpi scanner. Classification accuracy obtained for the FVC2000 database experiments was 85%, while for the real fingerprint experiments it decreased to 74%, as the method we adopted to acquire the fingerprints, used a common ink which is not the perfect one. For sure, modern specialized sensors can be found, which can improve and enhance the fingerprints image acquisition stage.

4. Conclusions

In this paper the development of an authentication system using fingerprint and PIN code is presented. The system was completely designed and implemented using MATLAB software and connected to a SQL sever 2000 database via GUI. The main processes in the system are the image enhancement, high quality minutiae extraction, and the clustering using self organizing maps neural network. The system contains two main phases of jobs, namely new registration and identification /or verification. The experimental results have shown that a good and sufficient accuracy has been achieved.

References

[1] S. Prabhakar, S. Pankanti, A.K. Jain, "Biometric Recognition: Security and Privacy Concerns", IEEE Security and Privacy, pp. 33-42 (2003).

[2] A.K. Jain, A. Ross, S. Prabhakar, "An Introduction to Biometric Recognition", IEEE Trans. on Circuits and Systems for Video Technology, Vol. 14 (1), pp. 4-19

(2004).
[3] D. Maltoni, D. Maio, A.K. Jain, and S. Prabhakar, Handbook of Fingerprint Recognition, Springer, New York (2003).
[4] R.C. Gonzalez and R.E. Woods, Digital Image Processing, Prentice Hall, Upper Saddle River, N.J. (2002).
[5] L. Hong, Y. Wan, and A.K. Jain, "Fingerprint Image Enhancement: Algorithm and Performance Evaluation," IEEE Transactions on Pattern Analysis and Machine Intelligence Vol. 20 (8), pp. 777-789 (1998).
[6] U. Halici and G. Ongun "Fingerprint Classification through Self-Organizing Feature Maps Modified to Treat Uncertainties." Proceeding IEEE Vol. 84 (10), pp. 1497-1512 (1996).
[7] L. Coetzee and E.C. Botha, "Fingerprint Recognition in Low Quality Images", Pattern Recognition, Vol. 26 (10), pp. 1441-1460 (1993)
[8] R. Thai "Fingerprint Image Enhancement and Minutiae Extraction." http://www.csse.uwa.edu.au/~pk/studentprojects/raymondthai/RaymondThai.pdf. (2003).
[9] D.C. Douglas Hung, "Enhancement and Feature Purification of Fingerprint Images", Pattern Recognition, Vol. 26 (11), pp. 1661-1671 (1993).
[10] B.M. Mehtre and B. Chatterjee, "Segmentation of Fingerprint Images - A Composite Method", Pattern Recognition, Vol. 22 (4), pp. 381-385 (1989).
[11] L. O'Gorman and J.V. Nickerson "An Approach to Fingerprint Filter Design" Pattern Recognition, Vol. 22 (1), pp. 29-38 (1989).
[12] M. Kawagoe and A. Tojo, "Fingerprint Pattern Classification", Pattern Recognition, Vol. 17 (3), pp. 295-303 (1984).
[13] K. Karu and A.K. Jain, "Fingerprint Classification", Pattern Recognition, Vol. 29 (3), pp.389-404 (1996).
[14] D. Maio, D. Maltoni, "Direct Gray-Scale Minutiae Detection in Fingerprints," IEEE Trans.Pattern Anal. Machine Intell, Vol. 19 (1), pp. 27-40 (1997).
[15] Q. Xiao and H. Raafat " Fingerprint Image Post processing: a Combined Statistical and Structural Approach", Pattern Recognation, Vol. 24 (10), pp. 985-992 (1991).
[16] M. Tico, and P. Kuosmanen, "An Algorithm for Fingerprint Image Post processing", In Proceedings of the Thirty-Fourth Asilomar Conference on Signals, Systems and Computers, Vol. 2, pp. 1735-1739 (2000).
[17] N. Ratha, K. Karu, S. Chen and A.K. Jain, "A Real-time Matching System for Large Fingerprint Database", IEEE Trans. on Pattern Anal. Machine Intell., Vol. 18 (8), pp. 799-813 (1996).
[18] L. Fausett "Fundamentals of Neural Networks" Prentice Hall, New Jersey, 07632 (1994).
[19] T. Kohonen, Self-Organizing Maps, Second Extended Edition, Springer Series in Information Sciences, Vol. 30, Springer, Berlin, Heidelberg, New York, 1995 (1997).
[20] R. Germain, A Califano, and S. Colville, Fingerprint Matching using Transformation Parameter Clustering, IEEE Computational Science and Engineering, Vol. 4 (4), pp. 42-49 (1997).
[21] D.M. Kroenke "Database Processing Fundamentals, Design & implementation" Prentice Hall, New Jersey, 07458, 7th (2000).
[22] Fingerprint Verification Database FVC2000, http://bias.csr.unibo.it/fvc (2000).

Received August 28, 2005
Accepted September 29, 2005