A Fingerprint Minutiae Recognition System Based on Genetic Algorithms

Jihad Jaam[1], Mohamed Rebaiaia[2], and Ahmad Hasnah[1]
[1]Computer Science and Engineering Department, University of Qatar, Qatar
[2]Computer Science Department, University of Batna, Algeria

Abstract: *In this paper, we propose a fingerprint verification system using two different modules: the automatic classification of fingerprints which is based on the minutiae-matching algorithms and the verification-search technique which is based on genetic algorithms. Our experiments on a large set of fingerprint images show that our approach is highly promising. Its performance shows a great flexibility in recognizing a person very quickly with an error-prone of at most 1%.*

Keywords: *Image processing, pattern recognition, fingerprint's minutiae detection, genetic algorithms.*

Received July 13, 2004; accepted April 18, 2005

1. Introduction

Fingerprint matching technique is the most popular biometric approach used to identify people. The biometric features are known as "Galton features" due to Francis Galton (1872) who developed a probabilistic model of fingerprint individuality in term of minutiae distribution. These Galton features called also *minutia*e, are defined as the notion of branch and end points of epidermal ridges [1, 2, 3, 12, 13, 14, 15]; they contain particular information that enables their use in identification analyses. Each feature has in fact, a specific type, a unique location on the fingerprint, and a specific orientation. The orientation can be defined for an end point, for example, as the approximate tangent angle to the ridge ending as shown in Figure 1. Verification-based minutiae techniques called also biometric identification have been generalized to cover all the fields where high security must be imposed as in law enforcement for criminal identification. In practice, police agencies tend to focus solely on two fingerprint identification techniques. The first one is concerned with the uniqueness of a configuration of fingerprint feature and the second one deals with processing and feature matching parameters [3].

The task of comparing an input fingerprint's minutiae with only one target is relatively easy. The problem becomes tedious when the matching is concerned with comparing one fingerprint's minutiae to many others, for instance the input fingerprint with those ones from a specific database. The existing techniques used in pattern recognition as the relaxation and simulated annealing which are in general ineffective due to time consuming and the exponential computation of the problem. In this work, we adopt the technique of direct scale minutiae detection which has been proposed by many different researchers [1, 2, 3, 7, 8]. We have applied two different techniques: the first one is concerned with the programming of the well-known Hough transform procedure [13], and the second one is a singular one based on the application of genetic algorithms. Different experiments have been carried out successfully and the obtained results show that genetic algorithms are highly efficient in fingerprint recognition than other classical techniques.

2. Fingerprint Characteristics

A fingerprint is a textual image containing a large number of ridges-valley patterns (the ridges are black lines and the valleys are depicted by white lines) that form groups of almost parallel curves, whorls, loops, deltas and bridge crossing as shown in Figures 1-a and 1-b. More technical details about the fingerprint characteristics are given in the Handbook of Maltoni *et al.* [9].

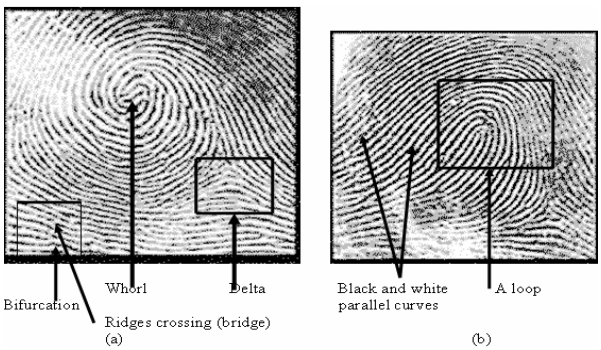


Figure 1. Basic fingerprint patterns.

It has been established and proven that fingerprint's ridges are individually unique and are unlikely to change during the life of a person. The structure of

ridges in a fingerprint is very complex; it is a combination of special features such as ridge endings, ridge bifurcation, short ridges and ridge enclosures as shown in Figure 2 and Figure 3. So the types, positions and orientations characterize the fingerprint matching.

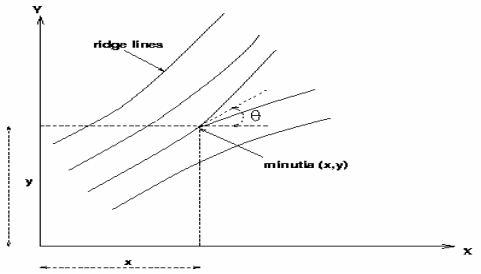


Figure 2. Minutiae orientation.

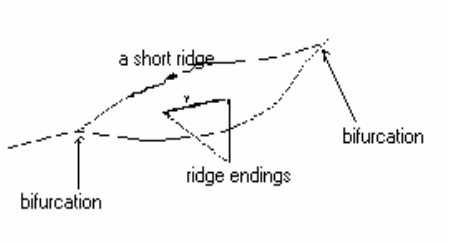


Figure 3. Ridge enclosures.

3. Automated Matching Principles

The fingerprint verification system proceeds according to the following two main steps:

1. The extraction minutiae from the fingerprint image.
2. The comparison of the minutiae set.

To extract minutiae from a grey-scale fingerprint image we proceed by performing a number of successive steps like the determination of the direction field, the smoothing and filtering, the binarization of the image, the extraction of the skeleton and the feature extraction and finally the post-processing. We note that the process of minutiae extraction could be simplified using the algorithm given in [1].

Generally, the minutiae extraction steps can be described as follows:

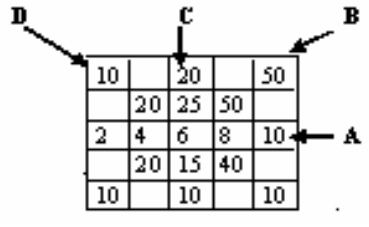
1. Image acquisition and data generation.
2. Block direction, computation and the smoothing.
3. Segmentation and binarization.
4. Skeleton post-processing (thinning).
5. Minutiae detection.

Details of the various operations are given according to the following points.

3.1. Block Direction Procedure

To compute the image direction we have used the method of Coetzee as presented in [3]. First of all, we divide the image of size 256x256 pixels into small blocks where the size of one block is 16x16 pixels. The directional image is obtained by computing the dominant direction of each block. The information contained in each block is represented by the intensity of each pixel and used as a numeric values to compute each sub-direction. We will explain the Coetzee method through the following example: Assume that the size of the considered block is 5x5 and the intensity values are represented by the following table:

Table 1. Intensity values of the greyscale block image.



Assume that the sub-direction labels are: *A, B, C, D*. The sub-direction *A* is considered as the row 3 with the intensity values (2, 4, 6, 8, 10). The arithmetic mean is equal to 6. The sub-direction *B* is considered as the second diagonal with the intensity value (10, 20, 6, 50, 50), so the corresponding mean is equal to 27.2. The sub-direction *C* corresponds to the value (20, 25, 6, 15, 10) and the mean average is 15.2. We continue to choose all the direction vectors and compute their averages.

In the second step, the algorithm computes for each direction vector the absolute difference between each value with the average and calculates the sum of the absolute differences. For example for the direction *A*, we have the following:

$|2 - 6| = 4$; $|4 - 6| = 2$; $|6 - 6| = 0$; $|8 - 6| = 2$; $|10 - 6| = 4$ and sum (A) = 12

In the third step, we choose the sub-direction corresponding to the little value obtained in the previous step. This sub-directed is assumed to be the dominant one. For this example, the sub-direction A is the dominant direction (sum (A) = 4 + 2 + 0 + 2 + 4 = 12). The result of a case study is shown in Figure 7-c.

3.2. Segmentation and Binarization Procedure

Image segmentation consists of the classification of the image pixels into classes or regions. The pixels are then labelled with a ranked number and grouped into classes or regions so that pixels belonging to regions with (more or less) homogeneous grey level have the same label pixels. Many techniques are available in the literature where some of them consider the problem of segmentation as a problem of classification and use for example neural networks techniques, especially when the number of regions is known in advance.

In fingerprint recognition, a series of operations are used to convert the grey-scale image "G", to binary. First, G is low pass filtered to reduce the noise by convolution with a 3x3 shaped kernel, w, defined as follows:

$$G_{lpf}(i, j) = G(i, j) \otimes w(i, j)$$

In the second stage the resulting image is sub-sampled in x-direction, by the coefficient α.

$$G_s(i, j) = G_{lpf}(\alpha i, j) \text{ for } i = 0, 1, \ldots, X/\alpha \text{ for } j = 0, 1, \ldots, Y$$

The third stage is to use the dynamic threshold procedure to convert the image to binary format. The procedure proceeds first by segmenting the improved image "G_s" into X / A by X / B blocks. Then the next step consists of computing the mean pixel value U given by the following equation:

$$U_{m\,n} = \frac{\left(\sum_{j=n\,B}^{(n+1)B-1} \sum_{i=m\,A}^{(m+1)A-1} G_S(i, j) \right)}{A \times B}$$

For $m = 0, 1, \ldots, X / A$ and $n = 0, 1, 2, \ldots, Y / B$. X and Y are the dimensions of G_s which is the improved image obtained by suppressing light noise using Gaussien filter or mid-point filter. A and B are the dimension of the local block. In our case we have chosen $A = B = 10$.

The matrix U_{mn} is computed as the average of the values of the pixels of each block. Inside the block, the pixels are subdivided into two sets. The pixels with value little than the mean U are transformed into zero value (white value). A pixel with value greater than or equal to U is transformed to the binary value 1 (black value). The resultant image is then used to analyse the morphological perspectives.

3.3. Skeleton Post-Processing Procedure

The process of skeleton post-processing is referred to as "cleaning" or "thinning" of the binary image. The cleaning routine includes the ability to detect and classify ridge structures. The skeleton image is then obtained by eroding (filtering) the objects contained in the binary image until they are of one pixel wide. A specific procedure is used in parallel to assure the connectivity and the topography of the ridges. In order, to perform the cleaning procedure we have used two well-known algorithms: the algorithm of Zhong and Suen and the algorithm of Marthon. For the final choice we have retained the first one, due to the unexpected results obtained by the second algorithm as shown in Figure 6.

Zhong and Suen algorithm, proceed as follows:

First of all, we assume that a boundary pixel, is a pixel with a greyscale equal to 1 and where at least eight neighbours points are equal to zero. The first step enforces a boundary point to zero satisfying the following conditions (thinning rules):

$$2 \le n(p) \le 6 \quad (1)$$
$$s(p1) = 1 \quad (2)$$
$$p2 \wedge p4 \wedge p6 = 0 \quad (3)$$
$$p4 \wedge p6 \wedge p8 = 0 \quad (4)$$

Where $n(p1)$ is the number of pixel set to 1 in the neighbours of $p1$; it is computed as follows:

$$n(p1) = p2 + p3 + \ldots + p8 + p9 \quad (5)$$

and $s(p1)$ is the number of transition 0-1 and to consider the ordering $p2, p3, \ldots, p9$.

For example, $n(p1) = 4$ and $s(p1) = 3$ taken from Figure 4-b.

P9	P2	P3
P8	P1	P4
P7	P6	P5

(a)

0	0	1
1	P1	0
1	0	1

(b)

Figure 4. The central pixel and its neighbours.

In the next step, the conditions (1) and (2) didn't change, but the conditions (3) and (4) become as follows:

$$p2 \wedge p4 \wedge p6 = 0 \quad (3')$$
$$p4 \wedge p6 \wedge p8 = 0 \quad (4')$$

Step 1 of the algorithm is applied to each point of the contour. If the conditions (1) or (4) are violated, the referred value of the point rest unchanged. If all the conditions are satisfied, the point is removed. The algorithm iterates the process until no point is subject to be suppressed. The result is the skeleton of the binary image as shown in Figure 5-b or 7-b.

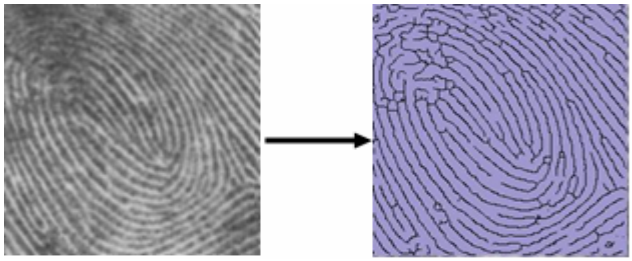


Figure 5. Zhang and Suen algorithm result.

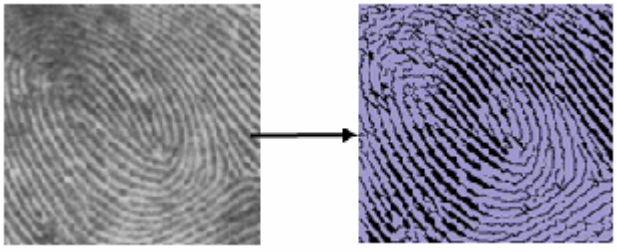


Figure 6. Marthon algorithm result.

3.4. Minutiae Detection Procedure

The first step used to find pores is to determine and to store the location of all the skeleton end and branch points. An end point is defined as any white pixel with either one or no neighbours. A branch point is a point that has exactly three neighbours. All the other skeleton components have exactly two neighbours and

are referred to as connecting points [13]. The detection of end and branch point locations are used to develop a minutiae map as shown in Figure 7-b.

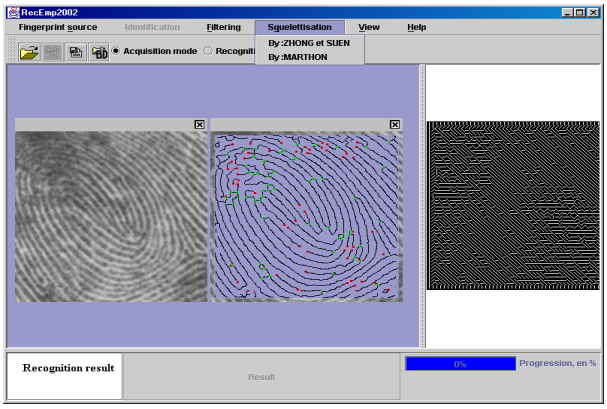


Figure 7. (a) Fingerprint image, (b) Skeleton after detecting minutiae, (c) Directions image.

4. Fingerprint Recognition Procedure

Consider the problem of comparing two different fingerprints selected from complete prints. The first segment is obtained from a known user at the time of enrolment, or registration, into the system as a collection of fingerprints in a database. The second segment results from a live-scan acquired for the purpose of verifying a user's identity. To decide if the live scan segment either matches or does not match the registered segment lead us to calculate the matching degree. This becomes effective after choosing a pre-specified acceptance level. For this case, we have used two algorithms. One based on the Hough transformation and the second and new one based on genetic algorithm. In the following discussion, we will detail these algorithms.

4.1. Hough Transformation Algorithm

The principal objective of this algorithm is to compute the function $F = (s, \theta, \delta x, \delta y)$ which transforms the set of minutiae P into the set Q, where s represents a scaling factor, θ an angle of rotation and $(\delta x, \delta y)$ a translation in the xy-plane as shown in Figure 8. P and Q are the two fingerprints to be compared which can be re-written as follows:

$P = \{p_1, K, p_m\}$ and $Q = \{q_1, K, q_n\}$, where $p_i = (x_i, y_i, \alpha_i)$ and

$$q_i = (u_j, v_j, \beta_j), \text{ for } 1 \le i \le m, \ 1 \le j \le n.$$

The Hough transform applied on the detection of the lines can be applied on the point correspondence as shown in Figure 8. Thus, Hough transform $F(p) = (x', y', \alpha')$ of a minutiae $p = (x, y, \alpha)$ is computed as follows:

$$\begin{vmatrix} x' \\ y' \\ \alpha' \end{vmatrix} = S \begin{vmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1/S \end{vmatrix} \begin{vmatrix} x \\ y \\ \alpha \end{vmatrix} + \begin{vmatrix} \delta x \\ \delta y \\ \theta \end{vmatrix}$$

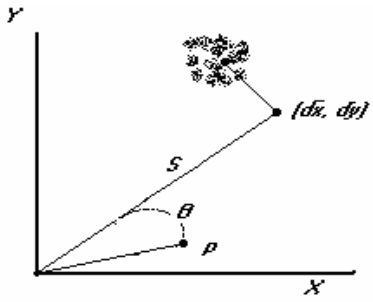


Figure 8. F (p) is the transform of p.

The computing results are grouped in a string table called $A(k, l, m, n)$. Hough algorithm is presented as follow:

Procedure Hough

$A(k, l, m, n) \leftarrow 0$; $k = 1, \ldots, K$; $l = 1, \ldots, L$; $m = 1, \ldots, M$; $n = 1, \ldots, N$
For $(p_r, p_p, \alpha) \in P$ *do*
  For $(q_r, q_p, \beta) \in Q$ *do*
    For $\theta \in \{\theta_1, \ldots, \theta_L\}$ *do*
      If $\alpha + \theta = \beta$ *then*
        For $s \in \{s_1, \ldots, s_L\}$ *do*

$$\begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} q_r \\ q_p \end{pmatrix} - s_k \begin{pmatrix} \cos\theta_l & \sin\theta_l \\ -\sin\theta_l & \cos\theta_l \end{pmatrix} \begin{pmatrix} p_r \\ p_p \end{pmatrix}$$

Add evidence for $F_{s_k \beta_l \Delta_x \Delta_y}$

        Endfor
      Endif
    Endfor
  Endfor
Endfor

Result $\leftarrow \arg\max_{k, l, m, n} A(k, l, m, n)$
End Procedure

4.2. The Genetic Algorithm Detection

Genetic Algorithms (GA's) are search and optimisation methods based on evolution in nature. They were first developed by John Holland from the University of Michigan in 1975 [6]. The power of GA's is, instead of working with a particular point in the solution space they proceed using more than one points and they are not necessarily concerned with finding the optimal solution, but with producing a "satisfactory'' one. The structure of a genetic algorithm is based on natural selection [4]. First, an initial population of feasible solutions is randomly generated. The initial population consists of *chromosomes*, encoded representations of solutions. The Selection takes place between members of the population, and a child is formed from a combination of the parent chromosomes. Whether or not the child becomes a member of the population depends on its *fitness* value. Each new child chromosome is compared against the worst member of the population, and the better one is kept in the population. By producing new generations in this

manner, the population improves and the best member of the final population is the solution returned by the algorithm [5].

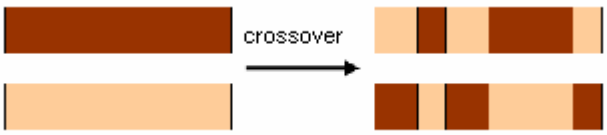


Figure 9. Uniform crossover transformation

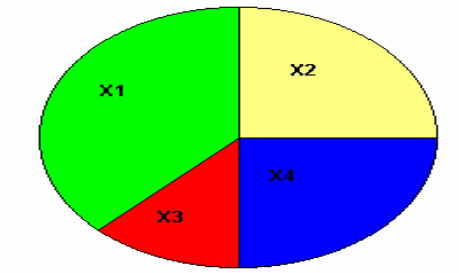


Figure 10. The roulette wheel.

The genetic algorithm proceeds as follows:

Genetic Algorithm

Begin
Initialisation: population size, number of generations;
Repeat
Select two individuals I_1 and I_2 in the population;
Apply the crossover operator on I_1 and I_2 to Produce a child I_3;
Replace one of the two individuals by I_3;
Delete Individuals in the population, which will be replaced by I_3;
Perform immigration;
Until the population converges;
Report results.
End

The above algorithm is summarized and applied for the detection and comparing the minutiae from two fingerprint's say A (target one) and B (memorized). It proceeds as follows.

4.2.1. Coding the Population Characteristics

The population is presented according to Figure 11.

Figure 11. Chromosome's population coding.

A gene in such coding corresponds to the value '0' or '1' (boolean values), where:

'1': There is a relation between the fingerprint minutiae of A and of those of B.
'0': The opposite case.

The size of the population is equal to the natural number 10 "chromosomes".

After the coding we create two classes called respectively "POP" and "CHROMO" representing the population and the chromosome.

4.2.2. Initial Population Generation Mechanism

The initial population is generated by the following Boolean procedure:

```
Public int myRand(int range) {
 If (range = = 0)
        return 0 ;
 Return (int) Math.round(Math.tandom( )/(1.0 / (range
        – 1))) ;
}
```

The fitness value expressing the quality of each solution is generated and used to select the candidate solutions that will contribute to the next generation. The fitness function is identified as the correspondence rate computed from the relation between both images.

The algorithm computes for each chromosome the evaluation function *EVAL (Vj)*, and then the complete evaluation is performed as follows: $F = \Sigma_j EVAL(Vj)$.

In the next step, the selection mechanism uses the following method (roulette 'casino') to choose randomly each individual chromosome.

```
Public double [] roulette (int U)
 {
   If (U = = 1)
    {
      For (int k = 0, k = 0 ; k < taille_pop ; k++)
         random [k] = getRandom(1) ;
      Return random;
    }
   Else
    {
      For (int k = 0, k < taille_pop*taille_chrom ;
           k++)
         randomMut [k] = getRandom (1) ;
      Return randomMut ;
    }
 }
```

The roulette method returns a Boolean vector of dimension 10 (number of chromosomes). We then compute the probability P_j to select the chromosome V_j as follows:

$$P_j = \frac{EVAL(V_j)}{F}$$

and the cumulative probability $Q_j = P_1 + P_2 + ... + P_j$.

The selection is performed by running the roulette ten times. At each time the system generates randomly the number r taken in the interval [0, 1].

If $r < Q_1$ (the cumulative probability $(Q_1 = P_1)$, the chromosome Vj is chosen as the best one. Otherwise

we select the chromosome V_j with $2 \leq j \leq population\text{-}size$ and $Q_{(j-1)} < r < Q_j$.

4.2.3. Operators to Diversify Population

The crossover operator: For each chromosome of the new population we generate the random number r in the interval [0, 1], if $r < Pc$, we select the chromosome for the crossover operation where Pc: The crossover probability is taken in the range 70% and 95%.

The mutation operator is used to guaranty the exploration of all the space. It consists of the generation of the random number r taken in [0, 1] (as much as the number of the size of the population). If r $\leq P_m$, the system pushes the bit to migrate with P_m the probability of mutation taken from the scale [0.5%; 1%].

5. Experiments

A large number of fingerprint's images have been tested (about 100). They have been collected from friends and family members to avoid data errors. We have tested and verified one to one the fingerprint's images using both the part of the system based on Hough procedure and the second part using verification algorithm based on genetic algorithm. The results of our experiments have shown that genetic procedure is more accurate than the Hough procedure and gives about 99% of precision where Hough procedure gives about 96%. We have also verified manually the conclusion of the system about minutiae features. A relevant part shows that about 6% of type feature (arch, loop, whorl, etc.) are false or missed.

6. Conclusion and Future Work

We have presented a fingerprint verification system based on the gray-scale ridges to recognise and classify minutiae of various types. In addition to that processing, the system is used to match two sets of minutiae, the traditional algorithm of Hough and the technique of evolutionary programming based on genetic algorithms. The experimental results on a database of 100 fingerprint's image have shown that the additional-based genetic algorithm is more effective in term of correctness and less time consuming than Hough procedure and Fourier transform. The implementation was written in Java JDK1.2 and is accessible as a web-based application. As a future work, our vision will be extended to the use of Genetic Programming (GP) (GP is different from GA-genetic algorithms, see [11]) to capture the information in the image and for image filtering, pixel classification, segmentation and the detection of minutiae.

Acknowledgments

We would like to thank the engineers Abdesselam Hamidi and Salim Moussaoui for their help in programming the fingerprint system.

References

[1] Asker M. B., Gerben T., Sabih H. G., Leo P., and Berend J., "A Correlation-Based Fingerprint Verification System," *in Proceedings of ProTisc Workshop on Circuits, Systems and Signal Processing*, Veldhoven, The Netherlands, November 2000.

[2] Asker M. B. and Sabih H. G., "Computational Intelligence in Fingerprint Identification," *in Proceedings of the 2nd IEEE Benelux Signal Processing Symposium (SPS'2000)*, The Netherlands, March 2000.

[3] CoertzeeL., " Fingerprint Recognition," *Master Thesis*, Faculty of Electronics and Computer Engineering, University of Pretoria, South Africa, 1992.

[4] Goldberg D. E., "Genetic Algorithms in Search," *Optimization, and Machine Learning* , Reading, Massachusetts, Addison Wesley, 1989.

[5] Goldberg D. E., Deb K., and Clark J. H., "Genetic Algorithms, Noise, and the Sizing of Populations," *Complex Systems*, vol. 6, pp. 333-362, 1992.

[6] Holland J., *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, 1975.

[7] Jain A. K., Hong S., Pankanti S., and Bolle R., "An Identity Authentification System Using Fingerprints," *in Proceedings of the IEEE*, vol. 85, pp. 1365-1388, 1997.

[8] Jonathan D., Stosz L., and Alyea A., "Automated System for Fingerprint Authentification Using Pores and Ridge Structure," Department of Defence, USA, 1994.

[9] Maltoni D., Maio D., Jain A.K., and Prabhakar S., *Handbook of Fingerprints Recognition*, Springer, 2003.

[10] Nalini K. R., Shaoyun C., Kalle K., and Jain A. K., "A Real-Time Matching System for Large Fingerprint Database," *IEEE Transaction on Patterns Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 799-813, 1996.

[11] Perter G., Han S., Asker M. B., and Sabih H. G., "PMDGP: A Distributed Object-Oriented Genetic Programming Environment," *in Proceedings of the 7th Annual Conference of the Advanced School for Computing and Imaging*, Heijen, The Netherlands, 2001.

[12] Rao K., "On Fingerprint Pattern Recognition," *Pattern Recognition*, vol. 10, no. 1, pp. 15-18, 1978.

[13] Stoney D. A., "Distribution of Epidermal Ridge Minutiae," *American Journal of Physical Anthropology*, vol. 77, no. 33, pp. 367-376, 1988.
[14] Stoney D. A. and Thornton J. I., "A Systematic Study of Epidermal Ridge Minutiae," *Journal of Forensic Sciences*, vol. 32, no. 5, pp. 1182-1203, 1987.
[15] Tu V. L., Young C. K., and Minh H. N., "A Fingerprint Recogniser using Fuzzy Evolutionary Programming," *in Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, Maui, Hawaii, January 2001.

Jihad Jaam obtained his BSc degree in 1988, Master degree in 1990 and his PhD in 1994 in computer science and mathematics of computing from France South Universities and The National Council of Scientific Research (CNRS), France. Currently, he is an associate professor of computer science and engineering at the College of Engineering, Department of Computer Science and Engineering, Qatar University. He is the author and co-author of 12 national textbooks in information technology and around 70 research papers published in different scientific journals and conferences proceedings. His research interests include stochastic algorithms, artificial intelligence, logic and mathematics of computing, satisfiability, problem solving, graph theory, combinatorics, and information retrieval.

Mohammed Rebaiaia is an assistant professor of computer science at the University of Batna, Algeria. He obtained a Diploma of engineering in computer science in 1983 and a Diploma equivalent to French 3rd cycle doctorate in computer science and operation research in 1986 from university of Annaba, Algeria. He teaches many courses as software engineering, formal methods, artificial intelligence, operation research, and algorithms. He is the author and co-author of more than 40 research articles and scientific reports published in international journals and pertinent conferences proceedings. His current research is in the field of pattern recognition, optimisation algorithms, software and hardware verification and specification, UML-RT and embedded real-time systems.

Ahmad Hasnah obtained his BSc in mathematics form Qatar University in 1990, his Master degree in 1992 and his PhD in computer science from the University of Illinois, USA, in 1996. Currently, he is an assistant professor of computer science and the head of the Computer Science and Engineering Department, College of Engineering, Qatar University. He is the author and co-author of more than 35 research papers published in international journals and scientific events. His research interests include information retrieval, cross-lingual retrieval, Arabic language processing, machine translation, algorithms, and artificial intelligence.