Fingerprint Matching and Classification using an Onion Layer algorithm of Computational Geometry

First Hamzeh Khazaei, Second Ali Mohades

***Abstract*—** In this paper, we present a new approach to fingerprint matching and classification using an onion layer algorithm of computational geometry. In order to extract valid minutiae we apply some image processing steps on input fingerprint. Using an Onion layer algorithm we construct nested convex polygons of minutiae, then based on polygons property, we perform matching and classification of fingerprints; we use the most interior polygon in order to calculate the rigid transformations parameters and perform local matching, consequently, global matching applied. This method rejects non matching fingerprint in local matching and avoid time consuming global matching steps.

We develop new classification scheme of fingerprints based on this approach. Unlike classic classification of fingerprints, this novel approach distributed fingerprints in classes equally, and none of image processing techniques are required for this classification. This normal distribution of fingerprints in different classes has great effect on identification time.

***Keywords*—** Fingerprint matching, Fingerprint Classification, Onion layer algorithms, Computational Geometry, Nested Convex Polygons

I. INTRODUCTION

Fingerprint matching is one of the most popular and reliable biometric techniques used in automatic personal identification. There are two main applications involving fingerprints: *fingerprint verification* and *fingerprint identification*. While the goal of fingerprint verification is to verify the identity of a person, the goal of fingerprint identification is to establish the identity of a person. Specifically, fingerprint identification involves matching a query fingerprint against a fingerprint database to establish the identity for an individual. To reduce search time and computational complexity, *fingerprint classification* is usually employed to reduce the search space by splitting the database into smaller parts (fingerprint classes) [3].

The approaches to fingerprint matching can be coarsely classified into three classes:

- Correlation-based matching.
- Minutiae-based matching
- Ridge-feature-based matching.

In correlation-based matching, two fingerprint images are superimposed and the correlation between corresponding pixels is computed for different alignments. During minutiae-based matching, the set of minutiae are extracted from the two fingerprints and stored as a sets of points in the two dimensional plane. Ridge-feature-based matching is based on such feature as orientation map, ridge lines and ridge geometry using the above techniques, *minutiae-matching* is most widely used approach that yields best matching results.

On the perpetual quest for perfection, a number of techniques devised for reducing *FAR* (False Acceptance Rate) and *FRR* (False Rejection Rate) were developed; computational geometry being one of such techniques[2].

Matching is usually based on lower-level features determined by singularities in the finger ridge pattern known as *minutia*. Given the minutiae representation of fingerprints, fingerprint matching can simply be seen as a point matching problem. In this context, matching two fingerprints implies finding a subset of minutiae in the first fingerprint that best match to a subset of minutiae in the second fingerprint through a geometric transformation in an optimal sense. Besides matching two fingerprints together, the main issue when dealing with large fingerprint databases is how to select the most similar fingerprints to the query fingerprint forms the database. Both of these problems appear very often in computer vision, particularly, in object recognition. As a result, methods for object recognition have much in common with fingerprint identification methods [1].

Two kinds of minutiae are adopted in matching: ridge ending and ridge bifurcation. For each minutia usually extract three features: type, the coordinates and the orientation.

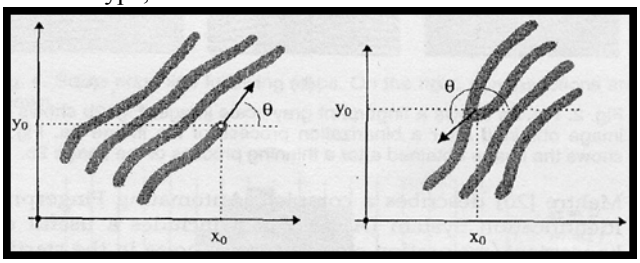

Figure 1, two types of minutiae, ridge ending and ridge bifurcation with their orientations.

Where ɵ is the orientation and (x_0,y_0) is the coordinate of minutiae.

Hamzeh Khazaei, M.Sc. student of computer science, Math and Computer Science Department, Amirkabir University Of Technology, Tehran, Iran (E-mail: hamzeh.khazaei@aut.ac.ir)

Ali Mohades, Assistant professor of computer science, Math and Computer Science Department, Amirkabir University Of Technology, Tehran, Iran. (E-mail: mohades@aut.ac.ir)

M. Poulos et al. develop an approach that construct nested polygons based on pixels brightness; this method need some image processing techniques[5].

Chen Feng and Marina Gavrilova present new results on fingerprint alignment and matching scheme based on *Delaunay Triangulation* approach. In their approach Delaunay triangles construct from minutiae, then with using some properties like edge length, interior angles and edge alignment perform the local matching, and calculate the rigid motion parameter (translation, rotation). Apply these parameters to the fingerprint template at database; finally, perform global matching [2].

Bebis et al. used the Delaunay triangle as the comparing index in fingerprint matching. The method works under assumption that at least one corresponding triangle pair can be found between the input and template fingerprint images. They use some invariants that are independent from translation, rotation and *scaling* [3].

Deng et al. use some graph concepts and proposed similar method like Bebis approach for local and global matching [4].

In this paper we use another geometric topologic structure, *Nested Convex Polygons* (NCP) [7,8], and establish a matching and classification schema. This novel approach is invariant from translation and rotation. We also have a local matching with using of the most interior polygon (*Reference Polygon*) and then apply global matching. In our approach unlike above techniques, we use *reference polygon* that is unique for every fingerprint; this uniqueness helps us to reject non matching fingerprints with minimum process and time. But in all above methods, they don't have such unique structure, so for rejection they should test all possible states, for local matching. We also based on our approach for matching, propose new and novel technique for classify fingerprints in classes with very better distribution than classic classification; this normal distribution reduces verification and identification time.

II. Nested Convex Polygons

Let S= $\{x_1,x_2, \ldots, x_n\}$ denote n points in two dimensional space. We use Quick Hall algorithm iteratively to construct nested polygons [8].

Algorithm1: *Construct Nested Convex Polygons(CNCP)*
Input: S= $\{x_1,x_2, \ldots, x_n\}$, K={},depth=0
Output: *Nested Convex polygons with their depth*

```
While(N(S)>0){
  K={};
  K=quickHall(S);
  S= S-K;
  StorePolygonProperties (K,depth);
  depth ++;
}//end of while
```

Where N(S) is the number of minutia in S, quickHall() method find the convex layer of given point set and StorePolygonProperties is a method that stores the reference polygon properties and it's depth finally.

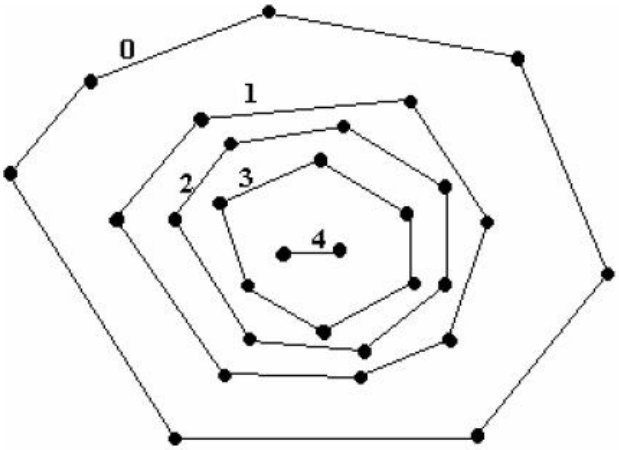

Fig. 2, Nested polygons constructed from point set.

III. Fingerprint Matching

The purpose of fingerprint matching is to determine whether two fingerprints are from the same finger or not. In order to this, the input fingerprint needs to align with the template fingerprint represented by its minutia pattern [10]. The following rigid transformation can be performed:

$$F_{s,\Delta\theta,\Delta x,\Delta y}\begin{pmatrix} x_{temp} \\ y_{temp} \end{pmatrix} = s\begin{pmatrix} \cos\Delta\theta & -\sin\Delta\theta \\ \sin\Delta\theta & \cos\Delta\theta \end{pmatrix}\begin{pmatrix} x_{input} \\ y_{input} \end{pmatrix} + \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \quad (1)$$

Where (s,ΔΘ,Δx,Δy) represent a set of rigid transformation parameters: (scale, rotation, translation). In our research, we assume that the scaling factor between input and template image is identical since both images are captured with the same device.

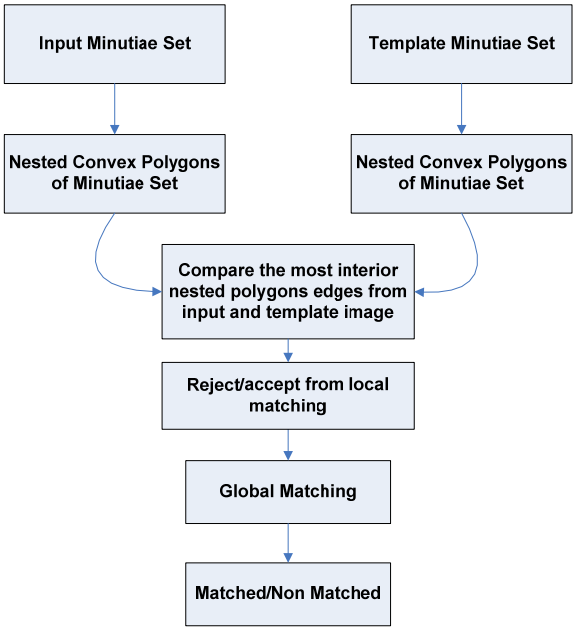

Fig. 3 Flow chart of generic fingerprint identification system.

IV. Construction Nested Polygons

Let Q=((x_1^Q, y_1^Q,Θ_1^Q,t_1^Q)...(x_n^Q,y_n^Q,Θ_n^Q,t_n^Q)) denote the set of n minutiae in the input image((x,y): location of minutiae; Θ: orientation field of minutiae; *t*: minutiae type, end or bifurcation;) an P=((x_1^P,y_1^P, Θ_1^P,t_1^P)...(x_m^P,y_m^P,Θ_m^P,t_m^P)) denote the set of m minutiae in template image.

Table 1, data structure used for comparing fingerprint images: Y: Dependent on fingerprint transformation; N: independent of it.

Feature	Fields				
Minutiae Point	x (Y)	y (Y)	Θ (Y)	Type (N)	
Polygon edges	Length (N)	Θ_1 (N)	Θ_2 (N)	$Type_1$ (N)	$Type_2$ (N)

Table 1 shows features that we use in local and global matching. In the table 1, Length is the length of edge; Θ_1 is the angel between the edge and the orientation field at the first minutiae point; $Type_1$ denote minutiae type of the first minutiae [2]. Using onion layer algorithm we construct nested polygons; for every fingerprint we store edge properties that mentioned in table1 of the *reference polygon*, plus its depth, and Minutiae points features that mentioned in first row of table 1 in database as template (*fingerprint Registration*). At the figure 4 the polygon at depth 6 is the reference polygon that used for local matching in order to calculate rigid transformation parameters; these parameters apply to the whole remain minutiae of input fingerprint in order to align with template fingerprint, then global matching is employed, and if the score of matching is higher than predefined threshold, two fingerprints are identical, otherwise they are from different fingers.

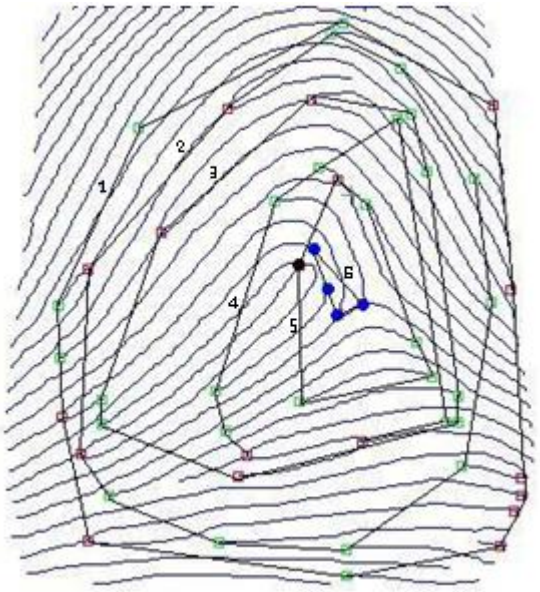

Fig. 4 Fingerprint with minutiae and its nested polygons.

V. Classification Step

Usually fingerprint classifies into six classes based on location of *center and delta points*; we must first begin by defining two terms, *core* and *delta*. The exact definitions of these terms are rather complicated, so we will content ourselves with informal definitions and a sample illustration. We will consider only the Loop pattern. The definitions for Whorl are similar; Arches and Tented Arches don't have cores or deltas. A Loop is defined by having at least one ridge that enters the print and recurves back exiting the print on the same side. The top of the innermost recurving ridge is defined as the *core.* The other side of a Loop contains ridges that enter the print and meet the recurving ridges. Some of these rise above, and some fall below the loop. The point where they diverge that is closest to the recurving ridges is the *delta*, (there is often a small island at this point) [11].

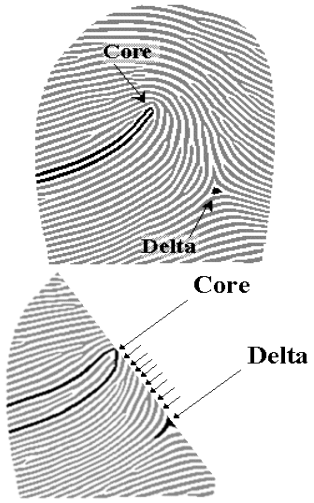

Fig. 5 Core and Delta of loop fingerprint.

Based on location of core and delta usually fingerprints divided in to 6 classes (Fig 6).

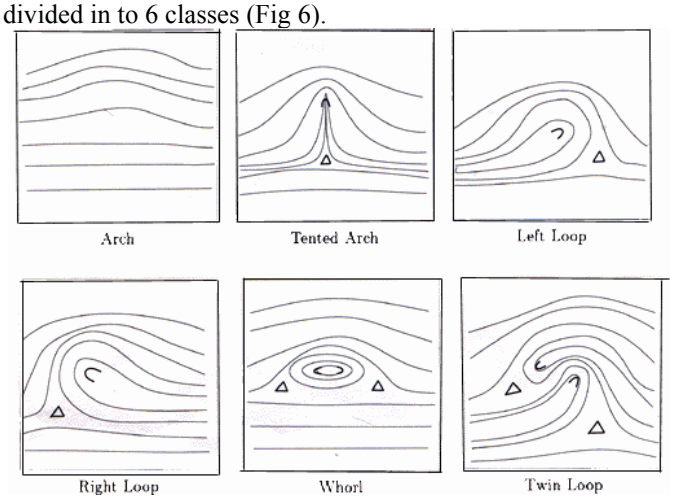

Fig 6 Representation of 6 Class in classic classification.

Figure 7 show 6 sample fingerprints in this classes:

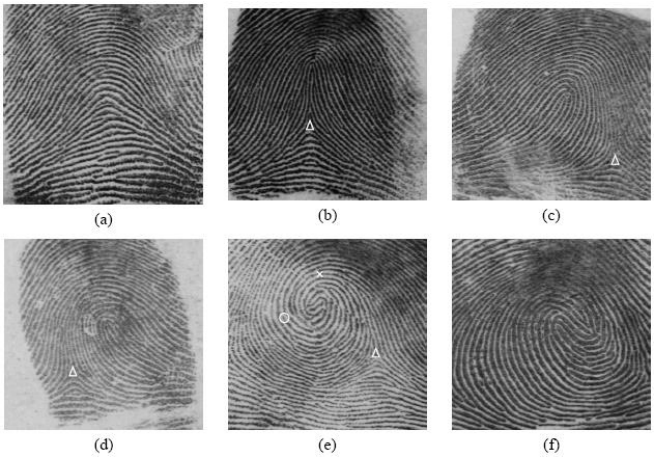

Fig. 7 Representation of 6 sample fingerprints in classic classification.

If we consider three types of loops (right, left and twin) as a one class; mix arch and tented arch as on a class, and whorl as separate class, then distribution of human being fingerprints in these class as below:

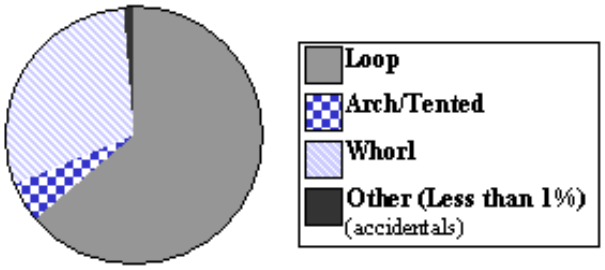

Fig. 8 Distribution of fingerprints in classic classification.

As figure 8 illustrates, the most *historic* classification does not have suitable distribution in order to reduce registration, identification and verification time [6,7].

Base on our algorithm we propose novel approach to classify fingerprints with better distribution in different classes.

We use following formula to identify classes for fingerprint:

$$C(F) = (N(p_0) + 2N(p_1) + \ldots + kN(p_k)) \% M \qquad (1)$$

Where $N(p_0)$ is the number of the most exterior polygon minutiae, $N(p_1)$ is the number of second polygon minutiae and $N(p_k)$ is the number of *reference polygon* minutiae; M is number of classes that we want to have; C(F) is an integer between 0 and M-1.

Experimental results show that proper value for M is a number between 4 and 8. It's a tradeoff between reducing registration and identification time and accurate classification and matching.

The most advantage of new classification is its normal distribution of fingerprints among different classes. We bring a test case of this new classification run on FVC 2000 fingerprint database, in section 8.

VI. Fingerprint Registration

We divided registration in three steps: firstly, in pre processing step, some image processing techniques that customize for fingerprint, like segmentation, normalization, Gabor filter and etc apply on fingerprint input image [6]. Next binarization and thinning are employed and valid minutiae are extracted from thin image:

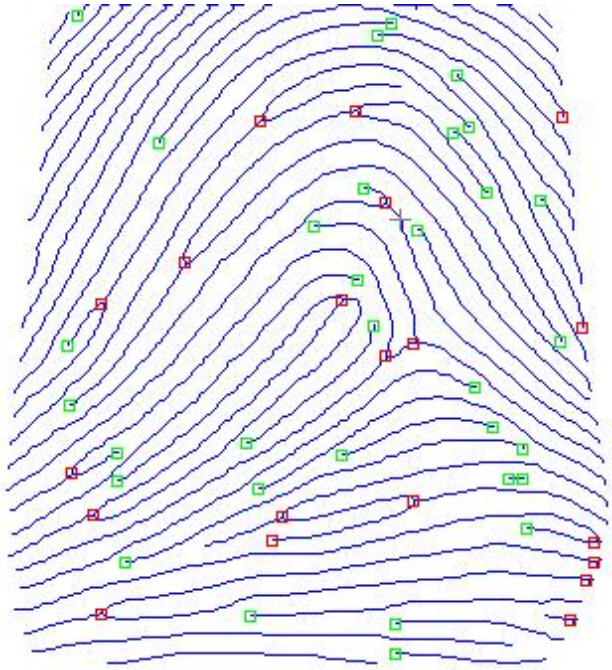

Figure 9, Final results of pre processing steps, fingerprint minutiae

Secondly, we apply onion layer algorithm and construct NCPs. We store invariant feature (Table 1) for *reference polygon* plus its depth and variant features for other polygons in the database as a template. At last Classification are employ and fingerprint store in its own class in database.

VII. Fingerprint Matching Based on NCP

The purpose of fingerprint matching is to determine whether two fingerprints are from the same finger or not. At this step input fingerprint image goes through pre processing, NCPs construction and determining its class like registration step. Afterward depend on *Identification* (1→n matching) or *verification* (1→1 matching) we perform matching. In *verification mode* we do not have to determine the class of fingerprint; retrieve fingerprint from database template and perform matching. But the purpose of identification is to identify unknown person, so in this mode, the class of input fingerprint is detect and matching of unknown fingerprint with templates at that class continue until happen a matching or rejection at the end. Following steps elaborate our algorithm in identification mode:

Some abbreviations that we use in algorithm:

- **RP**: *Reference Polygon*
- **RT**: *Reference Triangle*
- $\mathbf{P_i}$: *RP of input fingerprint image*
- $\mathbf{P_t}$: *RP of template fingerprint image*
- $\boldsymbol{\beta_i}$: angle between two adjacent edges of RP in RT of input fingerprint image
- $\boldsymbol{\beta_t}$: angle between two adjacent edges of RP in RT of template fingerprint image

- Other *abbreviations* are interpreted base on Table 1.

Algorithm2:

1. Compare input and template RP depth:

$$D=|Dept(P_i) - Dept(P_t)| \qquad (2)$$

2. If D>= 2, then two fingerprints are not from same finger, so matching reject at this step.

3. Otherwise select one of the P_i edges (E_i) and find corresponding edge in P_t; two edges are correspond (equal) if satisfy four following conditions:

$$Len(E_i)\text{-}Len(E_t) < T_1 \qquad (3)$$
$$Type_1(E_i) = Type_1(E_t) \qquad (4)$$
$$Type_2(E_i) = Type_2(E_t) \qquad (5)$$
$$\Theta_{i1}\text{-}\Theta_{t1} < T_2 \text{ and } \Theta_{i2}\text{-}\Theta_{t2} < T_2 \qquad (6)$$

Where, T_1 and T_2 are respectively thresholds of length of edges, and minutia angle and orientation of edge, respectively.

4. Repeat step 3, until find two adjacent edges in P_i that have two adjacent edges corresponding in P_t. If such couple adjacent edges don't exist in two RPs, matching reject at this step.

5. Using of such couple adjacent edges, a triangle constructed as Reference Triangle (RT). One more step needed to ensure that two triangle are exactly corresponded, that's satisfy with following condition:

$$|\beta_i - \beta_i| < T_3 \qquad (7)$$

Where, T_3 is threshold of angle between two adjacent edges in two RT.

6. Using of RT, calculate the rigid transformation parameters ($\Delta\theta,\Delta x,\Delta y$) .

7. Employ formula (1) and apply rigid transformation parameters on input fingerprint image for alignment.

8. Achieve the number of corresponding minutiae pairs. Given an alignment ($\Delta\theta,\Delta x,\Delta y$), minutiae of input fingerprint are mapped into template fingerprint. Judging they are matched or not according to equations (8);

$$m = \begin{cases} yes, if \sqrt{(X_i - X_{temp})^2 + (Y_i - Y_{temp})^2} \le r_0 \text{ and} \\ Type_i = Type_t \text{ and} \\ |\theta_i - \theta_t| < \theta_0 \\ no, Otherwise \end{cases} \qquad (8)$$

Where, r_0 and Θ_0 are thresholds of distance and orientation respectively.

9. Similarity calculate according to equation (9);

$$p = \frac{2n}{m+q} * 100 \qquad (9)$$

Where, m and q are the number of minutiae in two fingerprints and *n* is the number of matched minutiae.

If p be greater than predefined value, so two fingerprint are the same, otherwise go back to step 3. This iteration continues until either no candidate at step 4 exists, or accepts at step 10.

VIII. Experimental Results

We perform experiments using the fingerprint database of FVC 2000 to evaluate the correctness of algorithm in this paper and show the results of experiments.

A. Experiment Condition

The experiment uses four databases: DB1_a, DB2_a, DB3_a and DB4_a in database FVC 2000 [12]. Each database contained 800 fingerprints from 100 different fingers and in each database dry, wet, scratched, distorted and markedly rotated fingerprints were also adequately represented.

The test is performed under *Linux Fedora Core 6* O.S. on *Intel® centerino® 1.86 GHz and 1 G RAM* machine.

B. Results on the experiment

We compare our results with *Cspn* algorithm in FVC 2000 in terms of FMR and FNMR; this comparison shows the accuracy of new algorithm.

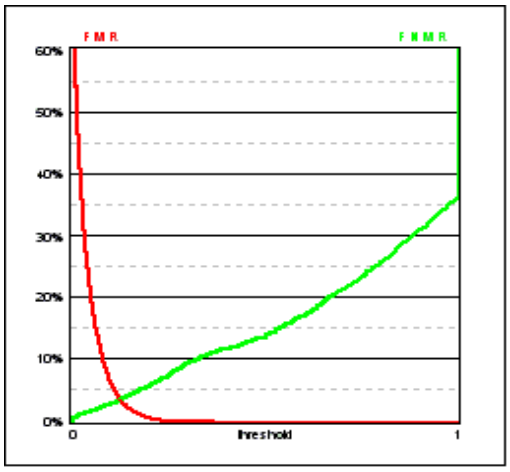

Fig. 9 FMR and FNMR of new algorithm.

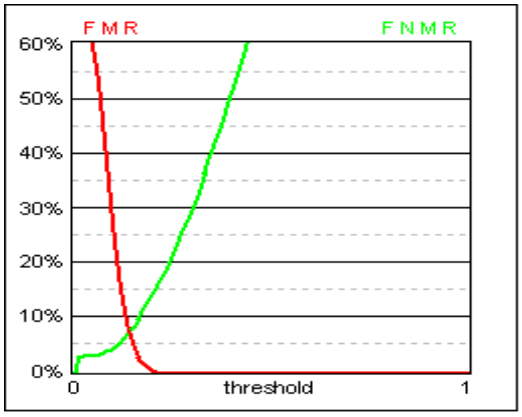

Fig. 10 FMR and FNMR of Cspn algorithm.

The best value for threshold is cross point of two curves. Our algorithm has less error than Cspn at this point. Moreover

comparison between ROC diagrams of two algorithms shows the excellence of our algorithm.

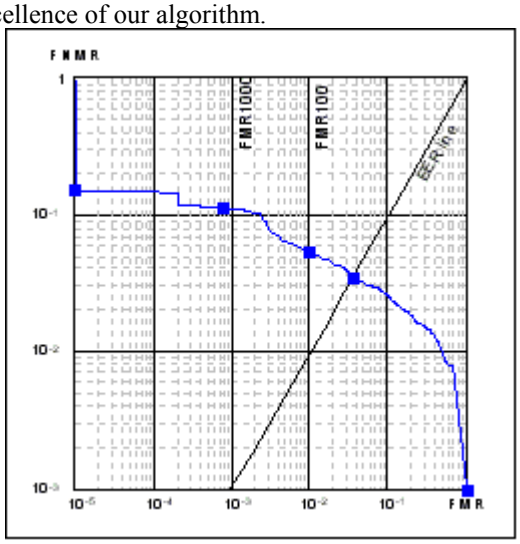

Fig. 11 ROC of new algorithm.

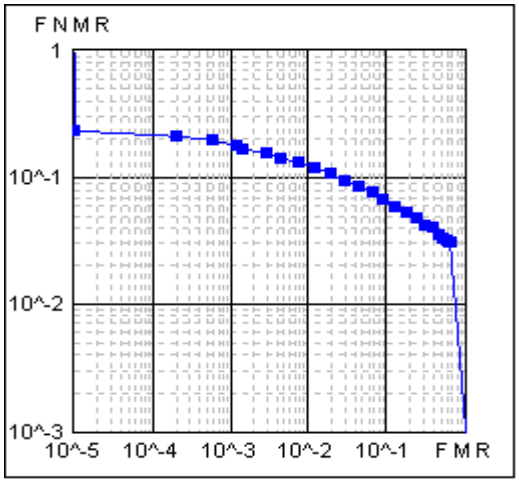

Fig. 12 ROC of Cspn algorithm.

The results of Cspn's algorithm are downloaded from FVC 2000 web page.

We also test our algorithm for classification, on FVC 2000 databases for evaluate its distribution of fingerprints among different classes.

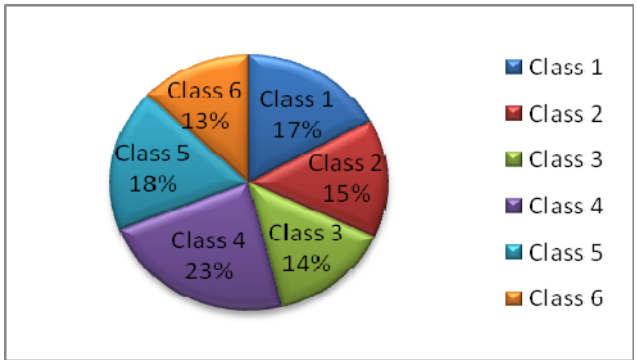

Fig. 12 Distribution of fingerprints among 6 classes based on new algorithm.

IX. Conclusion

We have developed a novel minutiae matching and classification algorithm based on nested convex polygons. The theory analysis of computational complexity shows that the NCP approach for fingerprint matching is more efficient than the standard minutiae based matching algorithms.

The most important characteristics of the proposed algorithm are: fast identification, very fast in rejection, more accurate than classic minutiae matching, and outclass classification than former classification approach, because distribute fingerprints in classes more equally. Moreover this approach for classification is completely different from known classification method. Another advantage of proposed algorithm is that non of image processing techniques are require for matching and classification, especially this classification technique is the first attempts for classify fingerprints without using *core* and *delta* points in fingerprints.

Our future objectives are: (1) considering new computational geometry structure for matching and classification in order to more resistant against noise and poor quality fingerprints, (2) more work on classification formula and find suitable parameters in order to have a better distribution of fingerprints through different classes.

Acknowledgment

The authors gratefully acknowledge the *Algorithms and Computational geometry* group at *Mathematic and Computer Science Department* of *Amirkabir University of technology* for their sympathy. We are also very thankful to Bahram Sadeghi B. for his helps and comments.

References

[1] Liu, Ning, Yin., Yilong and Zhang., Hongwei. A Fingerprint Matching Algorithm Based On Delaunay Triangulation Net, The Fifth IEEE International Conference on Computer and Information Technology, 2005.

[2] Wang, Chengfeng. and Gavrilova, Marina L. Delaunay Triangulation Algorithm for Fingerprint Matching, The 3rd IEEE International Symposium of Voronoi Diagrams in Science and Engineering, 2006.

[3] Bebis, George, Deaconu, Taisa. and Georgiopoulos, Michael., Fingerprint Identification Using Delaunay Triangulation, IEEE International Conference on Information Intelligence and Systems, 1999.

[4] Deng, Huimin. and Qiang, Huo. Minutiae Matching Based Fingerprint Verification Using Delaunay Triangulation And Aligned-Edge-Guided Triangle Matching, Audio- and Video-based Biometric Person Authentication, Springer Berlin / Hedilberg , 2005.

[5] Poulos, M., Magkos, E., Chrissikopoulos, V., Alexandris, N., Secure Fingerprint Verification Based on Image Processing Segmentation Using Computational Geometry Algorithms, Proceedings of the IASTED International Conference on Signal Processing, Pattern Recognition, and Apllications, Rhodes Island, Greece, June 30- July 2, 2003, ACTA Press, 308-312.

[6] Maltoni, Davide., Maio, Dario., Jain, Anil K. and Prabhakar, Salil., Handbook of Fingerprint Recognition, Springer, 2003.

[7] Rourke, Joseph O. Computational Geometry In C, Cambridge University Press, 1995

[8] Sack, J.-R. and Urrutia, J. Handbook of Computational Geometry, Elsevier, 2000.

[9] Jain, A., Hong, L. and Bolle, R. On-line Fingerprint Verification, IEEE TPAMI, vol.4, pp.302-313, 1997.

[10] X. D. Jiang and W. Y. Yau, " Fingerprint minutiae matching based on the local and Global Structures", Proceedings of the 15th International Conference on Pattern Recognition, 2000, pp.1042-1045.
[11] http://www.geradts.com/anil/ij/vol_002_no_001/papers/paper005.html
[12] http://bias.csr.unibo.it/fvc2000/