Fuzzy Logic Method for Partial Fingerpint Recognition

Radu Miron, Tiberiu Leția
Department of Automation
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
radu.miron@aut.utcluj.ro, tiberiu.letia@aut.utcluj.ro

Abstract—**Minutia-based algorithms are the preferred methods for matching fingerprints, because of their temporal performances. Although the correlation techniques are considered to be more reliable, they are much slower due to the fact that a large number of alignments (rotations and displacements) need to be taken into account. On the other hand, minutiae-based methods perform poorly when applied to low quality images, and might be impossible to employ for partial fingerprint recognition. This paper proposes a novel fuzzy logic method for matching partial fingerprints. The inputs of the proposed fuzzy logic algorithm are the correlation degree and the relative surface of the input fingerprint.**

Keywords-biometrics; fingerprint matching; partial fingerprint; fuzzy logic.

I. INTRODUCTION

Biometric recognition systems are the most advanced automated techniques used to establish someone's identity. They are based on physiological and/or behavioral characteristics, identifying or verifying the identity of the individuals by "who they are", rather than by "what they possess" or by "what they remember" [1].

The most common biometric traits used by recognition applications are: fingerprints, retina, iris, hand geometry, face, veins, handwriting, voice, etc. Among all the possible biometric characteristics, fingerprint is the most widely spread, due to its distinctiveness and persistence [2]. Fingerprints never change, and the probability of two fingerprints being the same is 1 in 1.9×10^{15} [3]. These aspects helped the fingerprint based technology in becoming the undisputed biometric leader, having a market share of almost 67% in 2009. International Biometric Group forecasts that the annual biometric industry sales will triple by 2014 [4].

The most obvious structural feature of a fingerprint is its banded pattern of ridges and valleys (or furrows). These two types of structures run parallel, but they can also end or split, thus creating the two main types of minutiae (*level 2 features*): *terminations* and *bifurcations*, respectively (refer to Fig. 1). There are other types of detail features (e.g. pores, short ridges, cores, and deltas), but the FBI minutiae-coordinate model relies only on terminations and bifurcations [5].

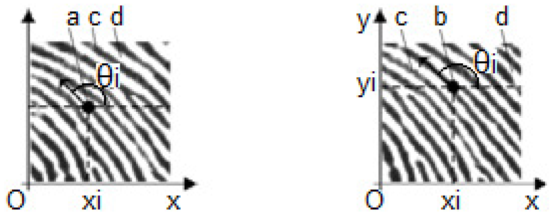

Figure 1. Minutiae types: a) termination; b) bifurcation; c) ridge; d) furrow.

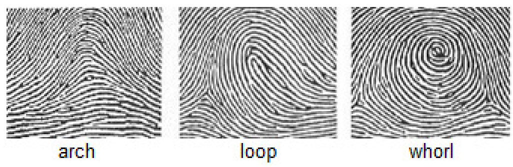

Figure 2. Fingerprint main topologies

A minutia is defined by the $\{x_i, y_i, \theta_i\}$ triplet. The coordinates of the minutiae are denoted by x_i and y_i, and θ_i is the angle between the tangent to the ridge line drawn in the minutia location, and the horizontal axis (refer to Fig. 1).

The other major feature class of the fingerprints is represented by the *singularities* (*level 1 features*) or the centers of the topologies (distinctive patterns that the ridges form in a fingerprint). For reducing the computational cost of recognizing a fingerprint, the database is structured according to the four main fingerprint topologies (the Henry classification system [6]): arch, left-loop, right-loop and whorl (refer to Fig. 2). These four main classes can be further divide into subclasses (e.g., arch class can be divided into plain arch and tented arch). While the Henry system has many classes, only 4, 5 or 7 of them are used in the automatic classification process.

Fingerprint recognition methods can be grouped into three major classes: (i) correlation-based matching, (ii) minutiae-based matching, and (iii) ridge feature-based matching [7]. While minutiae-based methods are the most popular because of their temporal performances, the correlation-based matching is considered to be more reliable, but very time consuming at the same time [8]. Minutiae-based recognition performs poorly on very well on low quality and partial input fingerprints [9]. The loss of singularity points makes ridge feature-based recognition and indexing techniques impossible to apply [10].

This paper proposes a novel method for matching partial fingerprints. The method is based on fuzzy logic and on a ridge-correlation algorithm. It also uses a set of minutiae for aligning the input image with the pre-stored template. The contribution of the paper is presented in Chapter III. The proposed method relies on the classical image processing and feature extraction algorithms presented in Chapter II. Some of these algorithms were also used by the authors in previous related work (refer to [24] and [25]).

II. State of the Art

In order to apply the proposed matching algorithm, the features of the fingerprint must be extracted first. The major steps involved in automatic fingerprint recognition: (1) the fingerprint acquisition, (2) the fingerprint image pre-processing (enhancement), (3) the feature extraction, (4) the fingerprint classification, and (5) the fingerprint matching.

A. Image enhancement

Image enhancement is a very important step performed before the fingerprint features extraction. Gabor filter-banks are the most well known technique employed for this purpose. Gabor filters are both frequency and orientation-selective. These properties make them very well suited for capturing the periodic pattern (with different local orientations) of a fingerprint [11, 12].

Histogram equalization is the first step used for the image enhancement process. It is used to adjust the contrast of the image. The lower contrast regions of the image gain a higher one. In order to obtain satisfying results, the histogram equalization has to be performed on blocks (e.g. an image of 256×320 pixel is divided into 16×16 pixel blocks).

The *local ridge orientation* determination requires dividing the image into $n \times n$ pixel blocks. According to [13], the local orientation of a block θ_{ij} can be calculated as in (1).

$$\theta_{ij} = 90^0 + \frac{1}{2} arctg\left(\frac{2G_{xy}}{G_{xx}+G_{yy}}\right), \tag{1}$$

$$G_{xy} = \sum_{u=i-\frac{n}{2}}^{i+\frac{n}{2}} \sum_{v=j-\frac{n}{2}}^{j+\frac{n}{2}} \nabla_x(u,v)\nabla_y(u,v), \tag{2}$$

$$G_{xx} = \sum_{u=i-\frac{n}{2}}^{i+\frac{n}{2}} \sum_{v=j-\frac{n}{2}}^{j+\frac{n}{2}} \nabla_x(u,v)^2, \tag{3}$$

$$G_{yy} = \sum_{u=i-\frac{n}{2}}^{i+\frac{n}{2}} \sum_{v=j-\frac{n}{2}}^{j+\frac{n}{2}} \nabla_y(u,v)^2. \tag{4}$$

where n is the block size and $\nabla_x(i,j)$ and $\nabla_y(i,j)$ are the gradient magnitudes of the pixel located at *(i,j)* coordinates, on x and y directions respectively, and can be calculated by using the Sobel operators.

By smoothing the three components (*Gxx, Gyy* and *Gxy*) with a Gaussian filter, presented in equation (5), better results are obtained (refer to Fig. 3) [14].

$$F(x,y) = \frac{1}{2\Pi\sigma} e^{-\frac{(x-x_0)^2+(y-y_0)^2}{2\sigma^2}}, \tag{5}$$

where x_0, y_0 are the coordinates of the Gaussian kernel's center, and σ is the standard deviation ($\sigma = 3$ for a 500 dpi fingerprint image [15]).

The *local ridge frequency* is determined for each of the $n \times n$ pixel blocks. Each block is traversed pixel by pixel on an imaginary line that passes through the center of the block and it is orthogonal to its orientation. The values of the pixels that are on this line are represented in a two dimensional plane. The resulted signal should have a sinusoidal shape, but usually it is highly affected by noise. In order to obtain a shape as close to a sinusoid as possible, the image is smoothened with a Gaussian filter. The local frequency is determined by counting the sinusoid's negative peaks (refer to Fig. 4) [16].

If S_{ij} is the distance between two negative peaks of the block centered at *(i,j)*, then the ridge frequency is calculated as:

$$f_{ij}=1/S_{ij} \tag{6}$$

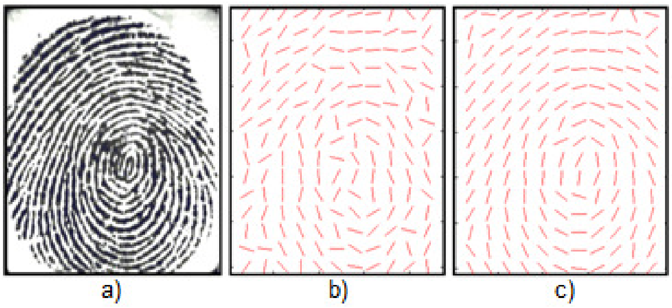

Figure 3. Fingerprint local ridge orientation: a) initial image, b) orientation field without Gaussian filtering, c) orientation field with Gaussian filtering

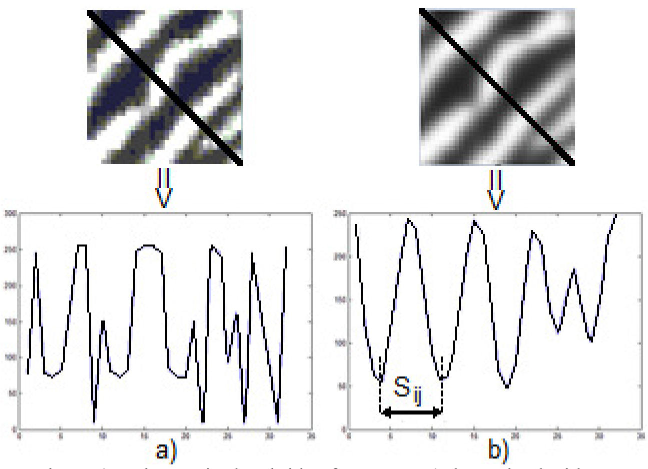

Figure 4. Fingerprint local ridge frequency: a) determined without Gaussian filtering, b) determined with Gaussian filtering

The next step of the fingerprint enhancement stage is the *image segmentation*. This step is performed in order to separate the fingerprint from the background, thus avoiding feature extraction from noisy areas. The background exhibits a very low grey-scale variance. This feature enables the use of a global threshold for image segmentation. The fingerprint image is divided into blocks of $n \times n$ *pixels*. If the variance of a block is lower than the global threshold, then the block is deleted. The grey-level variance of each block is calculated as:

$$V(k) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(I(i,j) - M(k))^2, \quad (7)$$

where *V(k)* is the grey-level variance of block *k, I(i, j)* is the grey-level of the pixel of coordinates *(i, j)*, and *M(k)* is mean value of the pixels' grey-level from block *k* [14].

According to [17], better segmentation results are obtained if the grey-level variance of a block is calculated along the orthogonal direction to its orientation (refer to Fig. 4). This is because a noisy image block has no directional dependence, whereas the grey-level variance of a block of interest (true fingerprint area) is very high.

Next, the image is filtered by using symmetric Gabor filters. The image is convoluted with the filter calculated as:

$$\begin{aligned} &g(x,y;\theta,f) = \exp\left(-\frac{1}{2}\left[\frac{x_0^2}{\sigma_x^2}+\frac{y_0^2}{\sigma_y^2}\right]\right)\cos(2\Pi f x_0), \\ &x_0 = x\cos\theta + y\sin\theta, \\ &y_0 = -x\sin\theta + y\cos\theta. \end{aligned} \quad (8)$$

where μ is the orientation of the Gabor filter, f is the ridge frequency, σ_x and σ_y are the standard deviations of the Gaussian envelope along the x and y axes, respectively, and θ is the local ridge orientation.

Reference [18] empirically determined the values σ_x=σ_y=4. This will result in a constant bandwidth filter. Because of the fact that the ridge frequency varies across the fingerprint image, the use of such a filter could lead to false artifacts construction and non-uniform enhancement. The two variables can be calculated as:

$$\sigma_x = k_x * f(i,j)^{-1}, \qquad \sigma_y = k_y * f(i,j)^{-1}. \quad (9)$$

where f is the ridge frequency, k_x=k_y=0.5 are constant variables determined for 500 dpi images [14].

The last two steps of the enhancement process are the *binarization* and the *thinning* of the image. Image binarization is performed in order to transform the grayscale image (the filtered fingerprint image) into a binary image (black ('0') and white ('1') only). The thinning of the image is used to reduce the ridges' thickness to one pixel; it helps minutiae extraction.

Fig. 5 presents the results obtained by applying the presented steps to three input fingerprint images. The images were scanned with an optic sensor (Wison Corp OR100) at the resolution of 256×320 pixels.

After enhancing the image, the features of the fingerprint are extracted: minutiae and singularities.

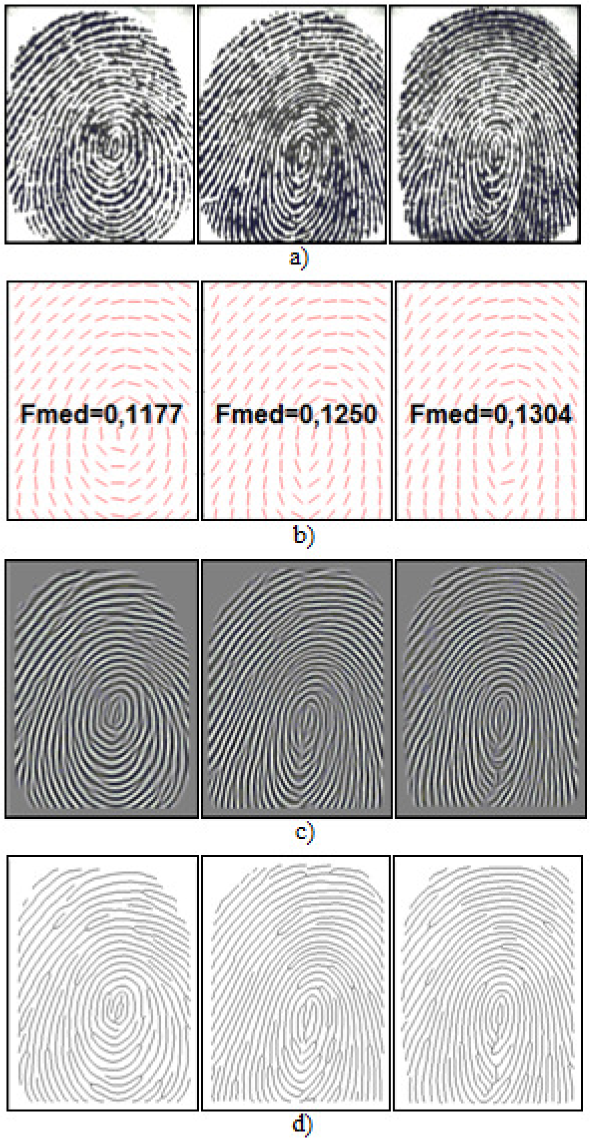

Figure 5. The results of the main enhancement process steps: a) initial scanned image, b) ridge orientation field and the medium ridge frequency, c)filtered image (Gobor filters), d)binarized and thinned image.

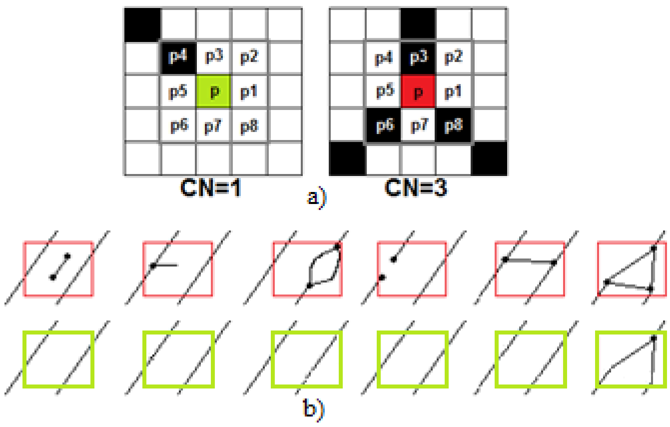

Figure 6. Minutiae: a) determining the terminations(CN=1) and the bifurcations (CN=3), b) false minutiae reduction.

B. *Feature extraction*

The *minutiae extraction* is done by examining the *8-neighborhood* of each ridge skeleton pixel (see Fig. 6 a)). There are a number of false minutiae that have to be removed from the fingerprint image. This step is necessary in order to eliminate noisy formations like those presented in Fig. 6 b).

The most common method to detect the fingerprint singularities is based on the Poincare index calculated from the local orientation field [19]. This method fails to detect most arch-type singularities. Another method, better suited for arch detection consists of shifting a rectangular window across the fingerprint image. When the upper region of the window contains the most ridges having an orientation close to 0^{o} and the orientation of the lower region changes abruptly a singularity is marked (refer to Fig. 7) [20].

C. *Partial fingerpint recognition*

Reference [21] proposes an algorithm that utilizes ridge information by choosing representative points. The chosen points together with the existing minutiae are used along with classical minutiae matching algorithms.

Typical one ridge stretches over a large distance and their width varies from 100µm to 300µm. Shorter and thinner ridges are actually dots and incipient ridges, level 3 fingerprint features (along with pores and ridge shape). Reference [22] obtained good results in matching dots and incipient ridges extracted from latent partial fingerprints.

Reference [23] proposes a minutiae-based algorithm for matching partial fingerprints. The method uses a two-hidden-layer fully connected neural network for determining the final matching score.

The approach presented in [24] is based on a fuzzy logic algorithm for matching partial fingerprints. The algorithm involves the correlation of the regions enclosed by the thinned ridges of the input with those from the template. The input and the template images are aligned by using a set of two bifurcations, thus reducing the computation time.

The partial fingerprint matching method presented by [25] differs from the existing studies that focus on one-to-one matching. This method retrieves candidate lists for each input partial fingerprint. The goal is to reconstruct (predict) the global topology pattern from incomplete information, thus finding the candidates.

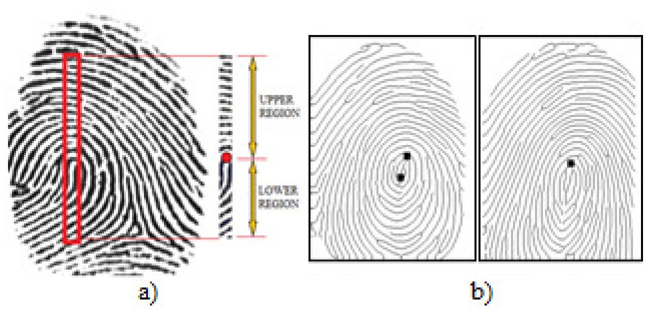

Figure 7. Singularities detection: a) ridge distribution in the neighborhood of a singularity, b) singularities detected by using the presented method.

III. Partial Fingerprint Recognition Algorithm

A. *Aligning the two images*

Applying correlation-based algorithms directly is very time consuming process, because a large number of rotations and translations are needed. In order to reduce the computational time, the proposed algorithm tries to correlate a set of minutiae from the input with another one from the template.

Let *I* and *T* be the minutiae sets of the input fingerprint and of the template, respectively. Each minutia *m* is represented by a triplet $\{x, y, \theta\}$, where *x* and *y* are the location's coordinates and θ is the angle of the minutia.

$$\begin{aligned} I &= \{m_1, m_2, ..., m_k\},\ m_i=\{x_i, y_i, \theta_i\},\ i=1..k, \\ T &= \{m'_1, m'_2, ..., m'_l\},\ m'_j=\{x_j, y_j, \theta_j\},\ j=1..l, \end{aligned} \tag{10}$$

where *k* and *l* are the numbers of minutiae in *I* and *T*, respectively.

Each set (*I* and *T*) are divided into two subsets. Each subset contains only the terminations and the bifurcations of the fingerprint:

$$\begin{aligned} I &= I_T \cup I_B, \\ T &= T_T \cup T_B, \end{aligned} \tag{11}$$

where I_T and I_B denote the input's termination and bifurcation subsets, and T_T and T_B denote the template's termination and bifurcation subsets, respectively [25].

The steps needed to align the input image with the template are presented in Algorithm 1 (also refer to Fig. 8):

Algorithm 1: Aligning the input and the template

1 Choose three minutiae from I, so that:
   - they are only from I_B (if possible);
   - they form the triangle with the largest sides possible (not equilateral, and not isosceles).

2 Find the corresponding minutiae from T, so that:
   - the sides of the formed triangle are equal with those of the triangle from step 1;
   - the orientations of the minutiae from T are rotated by the same angle in regard to those from I;

3 Shift the input image along Ox and Oy axes until one minutiae from I overlaps one minutiae from T.

4 Rotate the input image around the correlated minutiae (from step 3) until the other two minutiae from I are correlated.

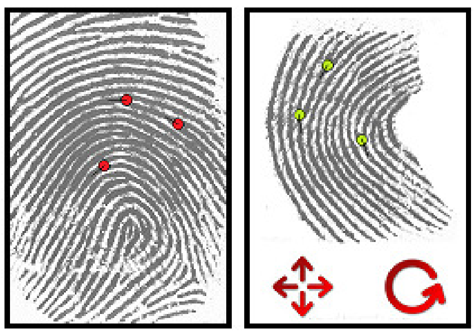

Figure 8. Minutiae correlation for image alignment

B. *Correlation degree and relative surface*

After aligning the input image with the template, the ridge correlation degree is calculated. At this point, the ridges of both the template and the input image are thinned. Due to non-linear distortions and image processing, the thinned ridges of two prints captured from the same finger are almost certain slightly different. In order to compensate this inconvenient, the ridges from the template are dilated (to 3 pixels).

The ridges correlation degree is calculated by using the next formula:

$$cd = N_O/N_T\ [\%]. \tag{11}$$

where *cd* is the correlation degree, N_O is the number of pixels that compose the input ridges overlapping a thickened ridge from the template, and N_T is the total number of the pixels of the input ridges (refer to Fig. 9).

The relative surface of the input fingerprint (*rs*) is given by the ratio between the partial fingerprint foreground surface (S_I) and the surface of the template's foreground (S_T):

$$rs = S_I/S_T\ [\%]. \tag{12}$$

C. *Fuzzification and fuzzy algorithm*

The correlation degree and the relative surface of the input fingerprint are fuzzified with the membership function presented in Fig. 10 a). Since the two variables take values in the same domain, the same membership function can be used to fuzzify both of them. The fuzzified variables denoted by *cdf* and *rsf*, respectively, take values in the domain {L, M, H} (i.e. low, medium and high).

Since any input that has a low correlation degree needs to be rejected, the scale of the *Ox* axis for *cd* starts at 50%.

The proposed fuzzy logic matching algorithm is based on the inference and it uses Mamdami rules with the following form:

$$\textit{IF (cdf is X) AND (rsf is Y) THEN (matchf is Z),} \tag{13}$$

where *X*, *Y* and *Z* take values in the set {L, M, H}.

The fuzzy logic rules described above are implemented as shown in Table I. This table provides the fuzzy logic values for the output variable *matchf*.

TABLE I. FUZZY LOGIC RULES

matchf		**cdf** is		
		L	**M**	**H**
rsf is	**L**	L	L	M
	M	L	M	M
	H	L	M	H

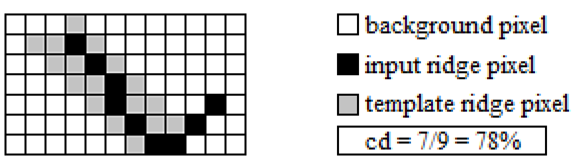

Figure 9. Correlation degree determination

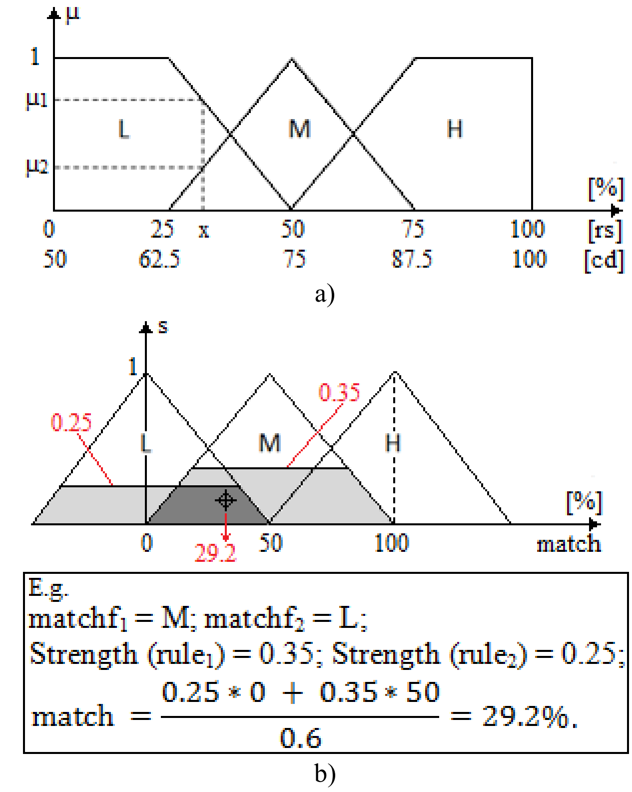

Figure 10. Membership function: a) for fuzzification, b) for defuzzification.

D. *Defuzzification*

Both *cdf* and *rsf* can take two fuzzy values (with different membership degrees μ_1, μ_2) at the same time. The consequence is that more than one rule can be activated for the same crisp input values (up to 4 rules can fire). The final output value is obtained by combining the outputs of all the rules that fired (inference of the rules).

The defuzzification process involves the use of the rule strengths. The strength of each rule is calculated as in (14). The weight of the *cdf* membership grade is twice more important than the weight of *rsf* membership grade.

$$Strength\,(rule_i) = s_i = \tfrac{2}{3}\,\mu_k(rshcdf) + \tfrac{1}{3}\,\mu_k(rsmcdf), \tag{14}$$

where $\mu_k(x)$ is the membership grade function and *i* is the number of the activated rule.

The output variable *match* (crisp value) is calculated by using the center-of-gravity method (refer to equation (15)).

$$match\,[\%] = \frac{\sum_{i=1}^{t} w_i * s_i}{\sum_{i=1}^{t} s_i}, \tag{15}$$

where s_i is the strength of the i^{th} rule, w_i is the center of gravity of the output fuzzy set of the i^{th} rule, and *t* is the number of the activated rules for the same input set.

The defuzzification membership function is presented in Fig. 10 b), along with an example of the defuzzification process.

IV. CONCLUSIONS

A fuzzy logic based algorithm that involves the correlation of the thinned ridges of an input with those from the template is proposed for matching partial fingerprints. This novel method combines the temporal performances of the minutiae-based algorithms with the reliability of the correlation-based ones.

The image processing modules were implemented in Matlab 7.8. Table II presents the temporal performance of the main algorithms used to implement the fingerprint image enhancement and feature extraction steps.

Although some promising results were obtained during tests, further testing of the method is needed. The partial fingerprints used for validating the proposed method were obtained by cropping images captured with an optical sensor (Wison Corp OR100) and prints from FCV 2002 DB2. Fig. 11 presents one example of a partial fingerprint matching.

Future work will include the establishment of a much needed partial fingerprint database for performance comparisons with other similar methods found in literature. A novel algorithm for correlating the ridges of the input with those of the template will be evaluated. This new algorithm will give a correlation score for each pixel of the input, based on the pixel's distance to the closest ridge.

TABLE II. TEMPORAL PERFORMANCES

Normalization	**Orientation**	**Frequency**	**Gabor filter**	**Minutiae**	**Singularities**
108 ms	78 ms	255 ms	810 ms	6 ms	220 ms

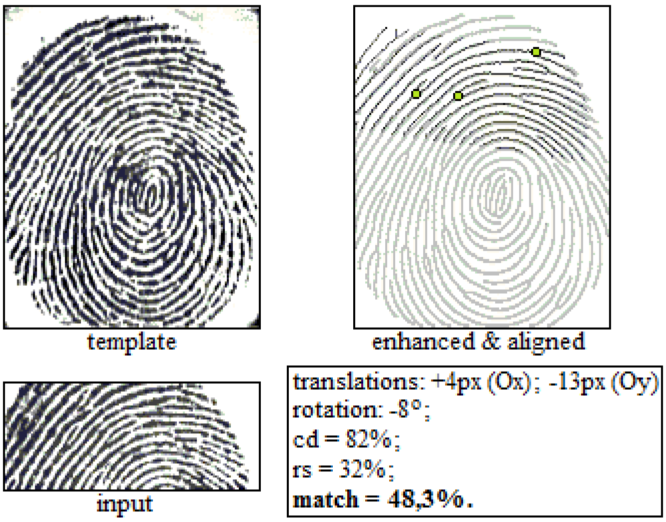

Figure 11. Experimental results.

REFERENCES

[1] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," Circuits and Systems for Video Technology, IEEE Transactions on Circuits and Systems for Video Technology. vol. 14, Jan. 2004, pp. 4-20.

[2] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, Handbook of Fingerprint Recognition, Springer-Verlag, New York, 2003, pp. 7-12.

[3] W. F. Leung, S. H. Leung, W. H. Lau, A. Luk, "Fingerprint Recognition Using Neural Network, Neural Networks For Signal Processing", Proceedings Of The 1991 IEEE Workshop.

[4] Biometrics Market and Industry Report 2009-2014. International Biometric Group, New York, NY (PRWEB), Nov. 12, 2008.

[5] J. H. Wegstein "An Automated Fingerprint Identification System", U.S. Government Publication, Washington, DC: U.S. Dept. of Commerce, National Bureau of Standards, 1982.

[6] E. R. Henry, "Classification and Uses of Fingerprints", London: Routledge, 1900.

[7] D. Maltoni, "A Tutorial on Fingerprint Recognition", Biometric Systems Laboratory - DEIS - University of Bologna, 2005.

[8] K. Abbad, N. Assem, H. Tairi and A. Aarab, "Fingerprint Matching Relying on Minutiae Hough Clusters", ICGST - GVIP Journal, ISSN: 1687-398X, Volume 10, Issue 1, February 2010, pp. 39-44.

[9] A. N. Marana, A. K. Jain, "Ridge-Based Fingerprint Matching Using Hough Transform", Computer Graphics and Image Processing, SIBGRAPI, 2005, p. 112-119.

[10] H. Le, D. Bui, "Online fingerprint identification with a fast and distortion tolerant hashing", Journal of Information Assurance and Security 4, 2009, p. 117-123.

[11] A. Ross, A. K. Jain, and J. Reisman, "A Hybrid Fingerprint Matcher", Pattern Recognition, Vol. 36, No. 7, pp. 1661-1673, 2003.

[12] M. U. Munir and M. Y. Javed, "Fingerprint Matching using Gabor Filters", National Conference on Emerging Technologies (ACM-IEEE), 2004.

[13] N.K. Ratha, K. Karu, S. Chen, A.K. Jain, "A Real-Time Matching System for Large Fingerprint Databases", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 8, 1996, p799−813.

[14] R. Thai, "Fingerprint image ehancement and minutiae extraction", Technical Report for the Honours Programme of the School of Computer Science and Software Engineering, The University of Western Australia, 2003, pp. 8-10.

[15] Z. M. Kovacs-Vajna, R. Rovatti, M. Frazzoni, "Fingerprint Ridge Distance Computation Methodologies", Pattern Recognition, vol. 33, no. 1, 2000, pp. 69−80.

[16] W. Zhang, Y. Wang, "Core-Based Structure Matching Algorithm of Fingerprint Verification", Pattern Recognition, 2002. Proceedings. 16th International Conference on, IEEE, 2002, p. 70-74.

[17] N. K. Ratha, S.Y. Chen, A. K. Jain, "Adaptive Flow Orientation-Based Feature Extraction in Fingerprint Images", Pattern Recognition, vol. 28, no. 11 (1995) pp. 1657−1672.

[18] L. Hong, Y. Wan, A. K. Jain, "Fingerprint Image Enhancement: Algorithms and Performance Evaluation", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 8, 1998, pp. 777−789.

[19] L. Wei, "Fingerprint Classification Using Singularities Detection", International Journal of Mathematics and Computers in Simulation, Issue 2, Vol. 2, 2008, pp.158-162.

[20] C.H. Park, S.K. Oh, D.M. Kwak, B.S. Kim. "A New Reference Point Detection Algorithm Based on Orientation Pattern Labeling in Fingerprint Images", Pattern Recognition and Image Analysis, IbPRIA, 2003.

[21] G. Fang, S. N. Srihari, H. Srinivasan, and P. Phatak. "Use of ridge points in partial fingerprint matching", Biometric Technology for Human Identification IV, 2007. pp. 65390D1–65390D9.

[22] Y. Chen and A. K. Jain, "Dots and incipients: Extended features for partial fingerprint matching", Biometric Symposium, BCC, Baltimore, September, 2007.

[23] T. S. Jea, "Minutiae-based Partial Fingerprint Recognition", PhD Thesis submitted to Graduate School of the University at Buffalo, the State University of New York. 2005. pp. 33-66.

[24] R. Miron, and T. Leţia, "Improved Personal Identification Method Based on Partial Fingerprints", Control Engineering and Applied Informatics (CEAI - SRAIT). vol. 12, no. 4, December 2010, pp. 24-29.

[25] R. Miron, and T. Leţia, "Fuzzy Logic Decision in Partial Fingerprint Recognition", IEEE AQTR-2010. Cluj-Napoca, Romania, Tome III pp. 439-444.

[26] Y. Wang, and J. Hu, "Global Ridge Orientation Modeling for Partial Fingerprint Identification", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 33, Issue 1, Jan. 2011. pp. 72-87