Optica Applicata, Vol. XLIII, No. 3, 2013
DOI: 10.5277/oa130311

An improved algorithm for feature extraction from a fingerprint fuzzy image

KAMIL SURMACZ*, KHALID SAEED, PIOTR RAPTA

AGH University of Science and Technology, Faculty of Physics and Applied Computer Science, al. Mickiewicza 30, PL-30059 Cracow, Poland

*Corresponding author: surmacz@fis.agh.edu.pl

Proper fingerprint feature extraction is crucial in fingerprint-matching algorithms. For good results, different pieces of information about a fingerprint image, such as ridge orientation and frequency, must be considered. It is often necessary to improve the quality of a fingerprint image in order for the feature extraction process to work correctly. In this paper we present a complete (fully implemented) improved algorithm for fingerprint feature extraction, based on numerous papers on this topic. The paper describes a fingerprint recognition system consisting of image preprocessing, filtration, feature extraction and matching for recognition. The image preprocessing includes normalization based on mean value and variation. The orientation field is extracted and Gabor filter is used to prepare the fingerprint image for further processing. For singular point detection, the Poincaré index with a partitioning method is used. The ridgeline thinning is presented and so is the minutia extraction by CN algorithm. The paper contains the comparison of obtained results to the other algorithms.

Keywords: biometrics, feature extraction, fingerprints, minutiae, Gabor filter, feature vector, image processing.

1. Introduction

Although each human has many unique characteristics, only a few are useful for biometric analysis. The main criterion in selection of biometric features is its usefulness rating in the process of identification or verification of a human being. For example, fingerprints, retina, iris or voice are very useful characteristics [1]. They are universal, unique, fixed and can be easily measured. These properties decide about usefulness of a biometric characteristic. Fingerprints fulfill the mentioned above requirements and are commonly used in biometrics.

In the last 20 years, huge progress has been made in the area of fingerprint recognition. The increase in popularity of biometric systems is driven by the constantly expanded capabilities of personal computers as well as the trend in biometric devices towards decreasing costs and increasing data quality. Because of the mentioned reason,

more and more biometric systems are used everywhere where personal identification or verification is needed. The algorithm presented in this paper is based on many publications about this topic. It contains implementation of each stage of feature extraction that best meets the authors criteria of selection. For the purpose of this paper, the full algorithm has been implemented, from the initial segmentation to the feature vector generation.

The paper is organized as follows. The fingerprint feature extraction algorithm is presented in Section 2. In Section 3, the results of parameter testing are shown. Section 4 contains the analysis of the results of the algorithm and the comparison with other algorithms. Lastly, the summary follows in Section 5.

2. Proposed algorithm

Fingerprints can be described by minutia-type, location and orientation. The most often applied minutiae are ridge endings, ridge bifurcation, core and delta (focal points).

To localize and identify minutiae on a fingerprint pattern, the acquisition of large amounts of information about an image is needed. Often algorithms use image preprocessing (for example: improving contrast, lighting or histogram alignment) to improve image quality, although the algorithm presented in this paper does not use any of these techniques. Instead of image preprocessing, we use an initial image analysis, improving the image quality using only filtration. The conclusion drawn from studying literature [2–9] is that the best results in image analysis are obtained using filtration based on image direction and frequency. The presented algorithm is therefore based on a Gabor filter, which greatly improves fingerprint image quality. The general scheme of the algorithm is presented in Fig. 1.

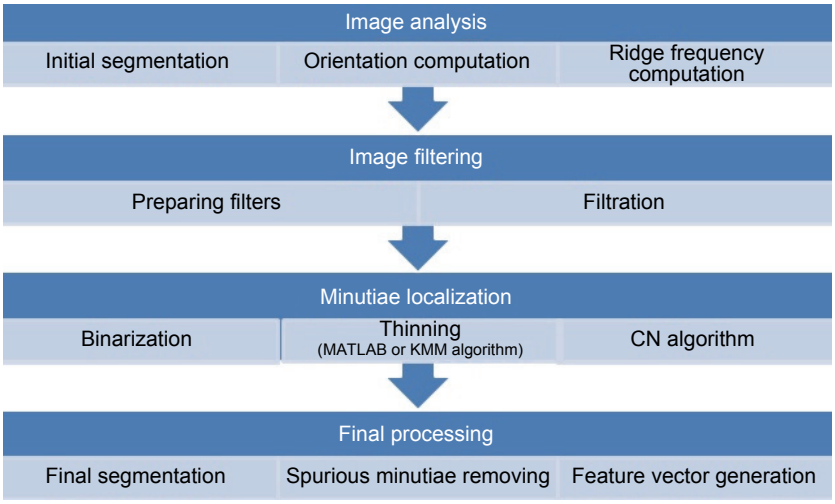

Fig. 1. The general structure of the proposed algorithm.

2.1. Initial image analysis

The image is directly analyzed after acquisition from an optical scanner. This stage is meant to provide information needed for proper filtration.

An initial segmentation is performed to localize a fingerprint area on the input image. Afterwards, the orientation of each pixel and frequency of ridges is calculated.

2.1.1. Initial segmentation

Initial segmentation separates the region of actual fingerprint from the background. The latter one causes distortion in statistical data in further processing. An initial segmentation method is very simple and is based on elementary statistical functions: mean value and variation. By calculating the mean intensity of pixels and variation in intensity for each block, the information needed for image segmentation is acquired. Because the fingerprint is a major part of the image, we can set parameters describing a range of mean intensity and mean intensity variance to decide whether or not each block belongs to the fingerprint or to the background.

2.1.2. Orientation computation

The next step in our algorithm is to compute an orientation matrix for an input image. This is a critical step, because the filters that will be used correspond to the calculated orientation. For calculation of an orientation matrix we use a Sobel operator, a gradient filter. The Sobel operator consists of two matrixes (Eqs. (1)). Convolution of this matrixes with an input image gives gradients in directions x and y, respectively. We can also compute the direction of a gradient using these matrixes.

Let an input image be matrix A. Matrixes G_x and G_y contain gradients in vertical and horizontal directions, respectively:

$$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * A, \qquad G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A \tag{1}$$

where $*$ to mean two-dimensional convolution operator.

We can use these matrixes for gradient magnitude computation:

$$G = \sqrt{G_x^2 + G_y^2} \tag{2}$$

We can also compute the direction of the gradient:

$$\theta = \operatorname{atan}\left(\frac{G_y}{G_x}\right) \tag{3}$$

However, more accurate results may be computed by taking into account the pixel neighborhood:

$$V_x(i,j) = \sum_{u=i-\frac{W}{2}}^{i+\frac{W}{2}} \sum_{v=j-\frac{W}{2}}^{j+\frac{W}{2}} 2G_x(u,v)G_y(u,v) \tag{4}$$

$$V_y(i,j) = \sum_{u=i-\frac{W}{2}}^{i+\frac{W}{2}} \sum_{v=j-\frac{W}{2}}^{j+\frac{W}{2}} \left[G_x^2(u,v) - G_y^2(u,v)\right] \tag{5}$$

$$\theta(i,j) = \frac{1}{2}\tan^{-1}\left[\frac{V_x(i,j)}{V_y(i,j)}\right] \tag{6}$$

where W is the window size, G_x and G_y are the matrixes of the gradient in vertical and horizontal directions, respectively.

For the computation of the directional map, we use an unfiltered image. As a result, some error values may occur in some regions (especially between the ridges, where distortion is very high). To avoid errors in these kinds of regions, we propose the use of Gaussian filtering (Fig. 2).

2.1.3. Core and delta detection

Core and delta detection is very important, because these points are the reference points for minutiae detection. Comparison of fingerprints in very large databases takes a significant amount of time. To reduce computation time, classification of fingerprints at the registration stage is needed. When cores and deltas are found, a fingerprint can be easily classified. There are many algorithms for fingerprint classification (for example: crossing number [10], or line search based method, widely described in [11]).

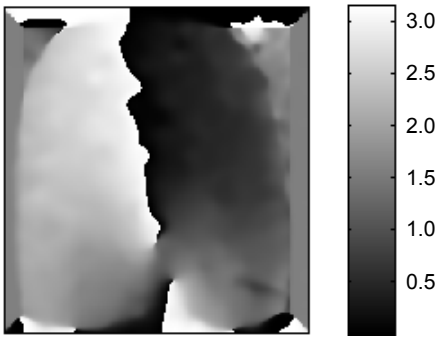

Fig. 2. The direction map of fingerprint ridges. Range is from 0 to Π (values in radians). Line made from transition between white and black color is perpendicular to ridges and it points to singularity (core or delta).

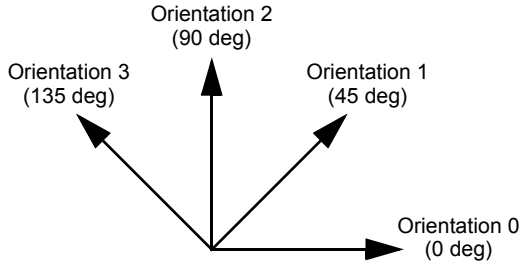

Fig. 3. Quantized directions for directional map [5].

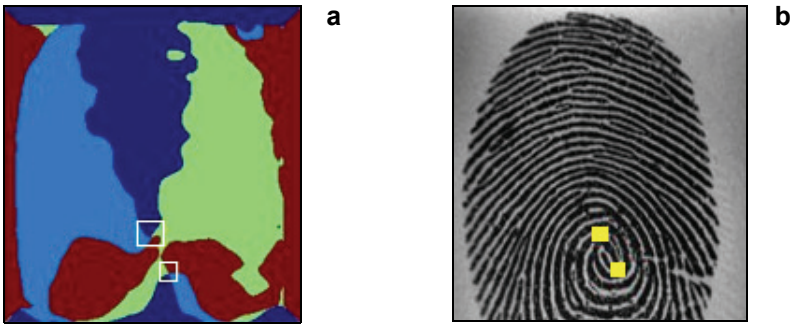

Fig. 4. Quantized directional map with regions containing delta or core candidates (white squares) (**a**). Core locations mark on original image (**b**).

The proposed algorithm uses a simplified Poincaré method, characterized by high precision and short calculation time. During the first stage, a directional map is quantized in four directions, as shown in Fig. 3.

In a quantized directional map, we look for pixels which have at least two neighboring pixels with another direction. This produces a list of delta and core candidates.

Whether an analyzed pixel is a delta or a core candidate is determined by the value distribution of the neighboring pixels. The localization of focal points is determined by mean localization of delta and core candidates in this region (Fig. 4).

2.1.4. Ridge frequency computation

For ridge frequency computation, the input image is divided into blocks with some size $W \times W$ pixels ($W = 16$ was selected). For each region, the mean direction is calculated and local maxima in the orthogonal direction are found. In this way we obtain a number of ridges in each block. Ridge frequency is defined as a rate of ridges per unit of block edge length. This method is very sensitive to errors in direction computation, so we only take the region of interest into account. After computing the ridge frequency for each block, we can present an image as a frequency matrix.

2.2. Image filtering

Single image filtering using a Gabor filter keeps precisely selected frequency band and ridge direction. A fingerprint image consists of many ridge frequencies and ridge directions, therefore we need many iterations of the Gabor filter (filters bank). Gabor filters improve image quality and fill discontinuities in a ridge flow. This helps in reduction of the list of false minutiae detected in the next step of the proposed algorithm. The Gabor filter is based on spatial frequency and ridge direction. A ridge on a fingerprint image always has those values well defined, therefore Gabor filtering gives very good results. The Gabor filter is defined as the harmonic function multiplied by the Gaussian function. Thanks to this modulation, the value of points localized near the middle of a ridge are amplified and the value of points localized further from the middle are reduced.

Mathematical representation of the Gabor filter is described by:

$$G(x, y; \theta, f) = \exp\left[-\frac{1}{2}\left(\frac{x_\theta^2}{\sigma_x^2} + \frac{y_\theta^2}{\sigma_y^2}\right)\right]\cos(2\pi f x_0) \tag{7}$$

$$x_0 = x\cos(\theta) + y\sin(\theta) \tag{8}$$

$$y_0 = -x\sin(\theta) + y\cos(\theta) \tag{9}$$

where θ is the Gabor filter orientation, f is the sine wave frequency, σ_x and σ_y are the standard deviation of the Gabor filter in horizontal and vertical directions, respectively, x_0 and y_0 define coordinates system of a filter frame.

The Gabor filter is convoluted with an original image. Computation of the convolution of pixel (i, j) requires calculation of pixel orientation and ridge frequency in this pixel's neighborhood. Gabor filtering binds function G and N using

$$E(i,j) = \sum_{u=-\frac{w_x}{2}}^{\frac{w_x}{2}} \sum_{v=-\frac{w_y}{2}}^{\frac{w_y}{2}} G\Big(u, v; O(i,j), F(i,j)\Big)\, N(i-u, j-v) \tag{10}$$

where O is the image orientation, F is the ridge frequency, N is the input image, E is the output image, w_x and w_y are the width and length of a filter mask, respectively.

2.3. Minutiae extraction

To apply the minutiae search algorithm, the image must be filtered in two stages: binarization and thinning. Binarization is a transformation of an image from a grayscale to a binary image (black and white). Binarization facilitates further image analysis, because it allows to distinguish between the epidermal ridges and the background. There are a few binarization methods. The simplest one is based on thresholding. In our algorithm we set the binarization threshold to 0.

To localize epidermal ridge bifurcations and terminals, image filtering must include thinning. Thinning is a morphological transformation, and thinned images

consist of one pixel-wide lines. To thin an image, a structural element is needed to act as a sort of template in which the middle point and 8 neighboring points are defined and transformed. For each point of a processed image, neighboring point values are compared with points of the structural element. If a point's neighborhood is identical to the structural element, then the value of the point being analyzed remains unmodified. Where the neighboring points are different than a structural element scheme, the point value is changed to 0. In this paper, a thinning algorithm implemented in MATLAB environment was used.

On the basis of the number of non-zero neighbors, each pixel potentially can be a source of information about existing minutiae. The crossing number algorithm is a widely-used algorithm for minutiae detection. In this algorithm, the eight closest neighbors of each pixel are taken into account using masks of dimension 3×3.

By calculating the CN number

$$\mathrm{CN} = 0.5\sum_{i=1}^{8}\left|P_i - P_{i-1}\right| \tag{11}$$

the information whether a considered pixel is a center of minutiae and determines the kind of minutiae is acquired. Usually most pixels in thinned fingerprint images will be classified as background (CN = 0), and a smaller number will be ridge continuations (CN = 2). The algorithm should classify a considerable number of pixels as ridge terminals (CN = 1) or bifurcations (CN = 3). Ridge crossings (CN = 4) are very rare and their detection is often mistaken with bifurcation detection. Therefore, the presented algorithm does not take into account this type of minutiae in the feature vector. Results are presented in Fig. 11**d**. For every found minutiae, beside its type, we save localization and direction.

2.4. Final processing

Final processing consists of three steps: segmentation, spurious minutiae removal and feature vector generation.

Final segmentation identifies the ridge flow area. This area is often termed the region of interest (ROI). Identifying and cutting the ROI helps remove minutiae found at ridge terminals.

There are two main reasons for the occurrence of spurious minutiae. The first one is noise caused by dirt on fingers or scanners. The second one is deformation caused by filtration and thinning operations. All types of minutiae can be represented as a compound of two basic minutiae: bifurcation and termination. Most algorithms used for removing false minutiae are based on calculating the distance between particular minutiae, allowing valid and false minutiae to be distinguished.

We can formulate rules used for detection of false minutiae. For example, if the distance between a bifurcation and a terminal is lower than some value D and both minutiae belong to the same ridge, then the minutiae should be considered false. The parameter D represents the mean distance between two parallel ridges. If the distance between two bifurcations is lower than D and both belong to the same ridge, then

we can confidently discard the minutiae. If two neighboring terminations are separated by *D* and their directions form a small angle, then both terminations should be classified as false minutiae. If two terminations close to each other are on the same ridge, both form an island and should be deleted. If a bifurcation point consists of at least two neighboring bifurcations belonging to the same ridge (caused by dirt or a dermal scar), the minutiae is also classified as false.

The final step of our algorithm is feature vector generation. Feature vector consists of three elements. First one corresponds to localization and orientation of ridge terminals. The second one corresponds to bifurcation in the same way. The last element (optionally) contains referred points localization (core or delta). Feature vectors can be directly compared.

3. Parameter optimization

This section contains the results from testing the algorithm across various parameters. Graphical presentation of the results will justify our choice of parameters.

3.1. Direction map low-pass filtering

For correct image filtration it is necessary to find the direction of every pixel. The resulting directional map should have smoothly changing values for neighboring

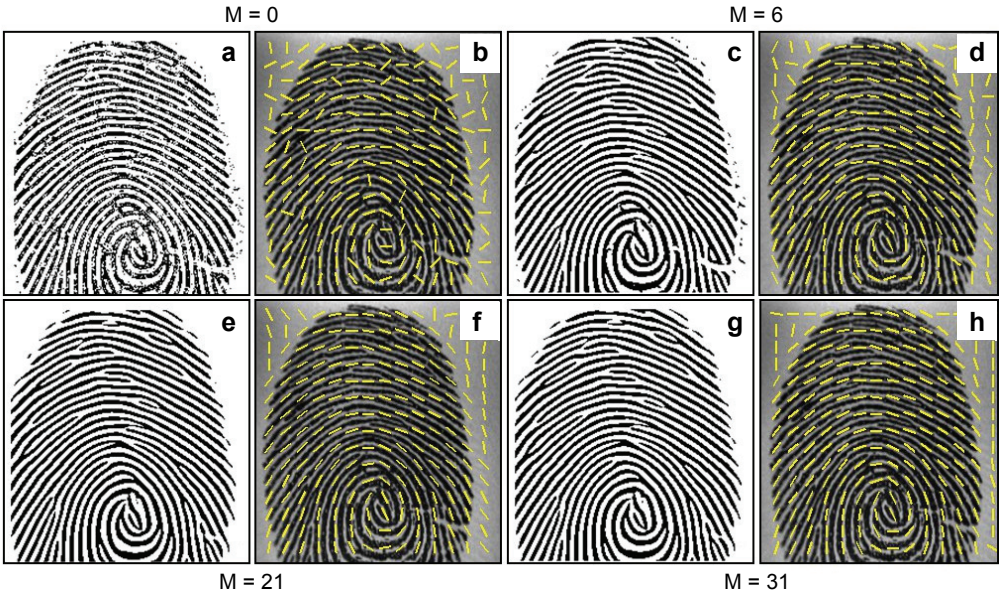

Fig. 5. Filtered image (**a**, **c**, **e**, **g**) and visualization of image orientation (**b**, **d**, **f**, **h**) for various Gaussian filter sizes *M*.

pixels. To obtain this effect, we use a Gaussian filter with a standard deviation of 5 pixels. The independent variable was the filter size M.

Analyzing Fig. 5, we see that for $M = 0$ (without Gaussian filtering) filtration was not performed correctly. For $M = 6$ we obtained much better results, although slight errors can be observed near the edges of ridge flow areas and the regions with wide spaces between ridges. The best results for the whole image correspond to $M = 31$, for which even the spaces between ridges have the correct orientation.

3.2 Parameter *k* of Gabor filters

The parameter k has a huge impact on image filtration. This parameter determines σ parameters calculated from:

$$\sigma_x = k_x F(i, j) \tag{12}$$

$$\sigma_y = k_y F(i, j) \tag{13}$$

In proposed algorithm $k_x = k_y = k$. Figure 6 presents the results of image filtration for selected values of k parameter.

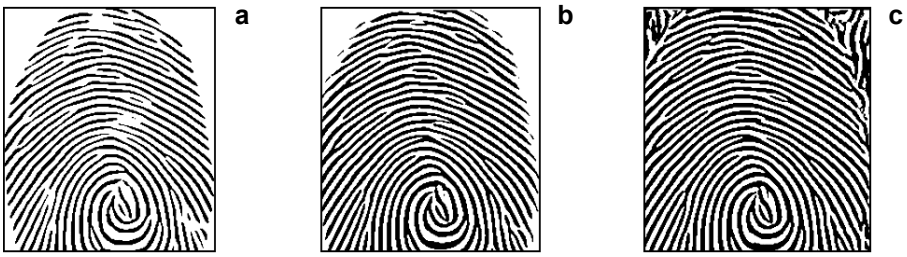

Fig. 6. Results of image filtration with parameters: $k = 0.4$ (**a**), $k = 0.5$ (**b**) and $k = 0.6$ (**c**).

After analyzing Fig. 6, the conclusion may be drawn that for $k = 0.4$ the filtered image is not complete. For $k = 0.6$, errors in image filtration – especially in discontinuous areas and near edges – can be observed. However, for $k = 0.5$ the image is correctly filtered.

3.3. Gabor filter size and its influence on filtering quality

The size of the Gabor filter mask has a large influence on the effect of image filtration. The filter size is determined by σ, where $w_x = w_y = n\sigma_x$. In Figure 7, the results of filtration for selected values of n are presented.

After analyzing Fig. 7, the conclusion can be reached that the filter size is too small (for example Fig. 7**a**) and creates errors in filtration. As the filter size is gradually increased, the improvement in filtration is observed. For $n = 3$, the results are satis-

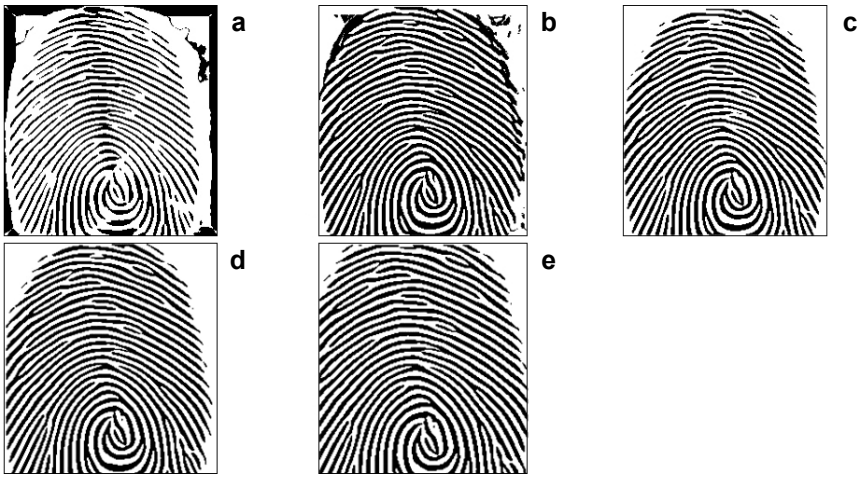

Fig. 7. Results of filtration for various n parameter values: $n = 1$ (**a**), $n = 2$ (**b**), $n = 2.5$ (**c**), $n = 3$ (**d**) and $n = 6$ (**e**).

factory. Further increase in n does not lead to better filtering results, however, and furthermore creates an increase in computation time. Another advantage of $n = 3$ is the larger size of an output image.

4. Analysis of the results in comparison to other algorithms

The results of our algorithm were compared with results of Fourier filtration, histogram alignment and the Hong algorithm (presented in [12]). The algorithms were tested on images of various quality. The image used for tests has many ridge discontinuities, caused by a careless scanning process. A comparison of different algorithms applied to this image is presented in Figs. 8–11.

A comparison of the aforementioned algorithms leads to the conclusion that algorithms based on Fourier filtration and histogram alignment produce unsatisfactory results, since images without correct filtration cannot be thinned and analyzed with the CN algorithm. Therefore, images filtered with these algorithms should be

Fig. 8. Results of algorithm based on Fourier filtration [12]. Filtered image (**a**), binarized form (**b**) and thinned form (**c**) with marked minutiae inside ROI (**d**).

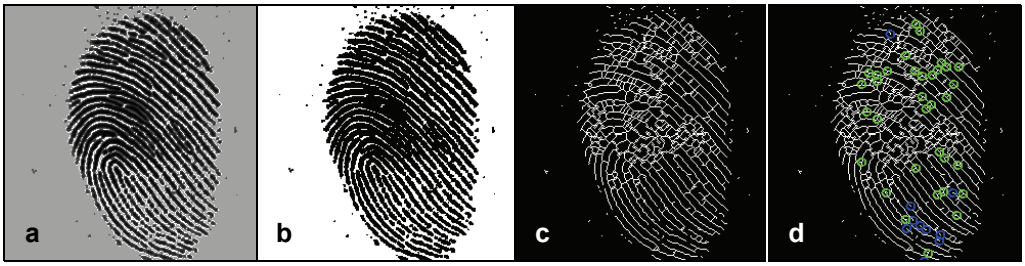

Fig. 9. Results of algorithm based on histogram alignment [12]. Filtered image (**a**), binarized form (**b**) and thinned form (**c**) with marked minutiae inside ROI (**d**).

Fig. 10. Results of algorithm based on Hong filtration [12]. Filtered image (**a**), binarized form (**b**) and thinned form (**c**) with marked minutiae inside ROI (**d**).

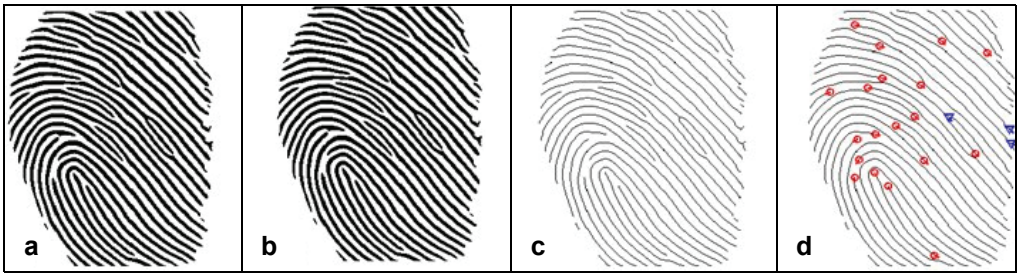

Fig. 11. Results of the algorithm presented in this paper. Filtered image (**a**), binarized form (**b**) and thinned form (**c**) with marked minutiae inside ROI (**d**).

analyzed using different methods (for example described previously Maio and Maltoni algorithm). Results obtained using the Hong algorithm are very similar to the results of the algorithm proposed in this paper because both are based on Gabor filtration. Differences are caused by different parameter values and methods of generating direction maps. The number of detected spurious minutiae is determined by parameters' values. In Table 1, the comparison of algorithms for finding minutiae is presented.

After analyzing Table 1, the conclusion can be drawn that applying the CN algorithm to a incorrectly filtered image gives very poor results, generating problems with localization and counting the true minutiae. The Fourier transform amplifying algorithm and the histogram alignment method lead to a great number of spurious minutiae. The false

T a b l e 1. Results of comparison of algorithms for finding minutiae.

	Algorithm minutiae finding (terminals/bifurcations)			
	Fourier	Histogram alignment	Hong	Proposed
After segmentation	52/56	84/46	16/4	19/3
After spurious minutiae removal	12/18	30/14	16/4	19/3

minutiae detection algorithm resulted in a greater number of correct minutiae being found, but the absolute value of false minutiae is still not satisfactory. Hong's method, as well as the presented algorithm, give comparable results. Furthermore, in both cases, erasing faulty minutiae was not necessary.

5. Conclusions

In this paper we presented a new algorithm for fingerprint feature extraction which does not need image preprocessing. Our algorithm was implemented in a MATLAB environment and was adjusted for educational purposes. Each stage of image processing can be displayed by the software user. The final result of the algorithm is a feature vector (with focal points), which can be used by any chosen fingerprint comparison method. Our tests of the algorithm for various images of fingerprints give good results. Fingerprint processing results obtained by this algorithm are comparable with the results of the algorithms presented in Section 4. Filtered image quality can be improved by increasing the precision of directional and frequency maps.

Acknowledgements – This work was supported by grant No. 11.11.220.01/prof. Kulakowski within the grants of AGH University of Science and Technology in Kraków. The authors are truly grateful to Mr. Jonathan Aguilar for his help with the English language check.

References

[1] SAEED K., NAGASHIMA T., *Biometrics and Kansei Engineering*, Springer, NY, 2012.

[2] BHANU B., BOSHRA M., XUEJUN TAN, *Logical templates for feature extraction in fingerprint images*, [In] *15th International Conference on Pattern Recognition, Proceedings*, Vol. 2, 2000, pp. 846–850.

[3] CHEN Y., JAIN A.K., *Beyond minutiae: a fingerprint individuality model with pattern, ridge and pore features*, International Conference on Biometrics, 2009.

[4] ESPINOSA-DURO V., *Minutiae detection algorithm for fingerprint recognition*, IEEE Aerospace and Electronic Systems Magazine **17**(3), 2002, pp. 7–10.

[5] HONG L., WAN Y., JAIN A., *Fingerprint image enhancement: algorithm and performance evaluation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(8), 1998, pp. 777–789.

[6] PRABHAKAR S., *Fingerprint Classification and Matching Using a Filterbank*, Ph.D. Thesis, Michigan State University, 2001.

[7] RATHA N.K., SHAOYUN CHEN, JAIN A.K., *Adaptive flow orientation-based feature extraction in fingerprint images*, Pattern Recognition **28**(11), 1995, pp. 1657–1672.

[8] THAI R., *Fingerprint Image Enhancement and Minutiae Extraction*, The University of Western Australia, 2003.
[9] PORWIK P., WIECLAW L., *A new fingerprint ridges frequency determination method*, IEICE International Journal Electronics Express **6**(3), 2009, pp. 154–160.
[10] ARCELLI C., DI BAJA G.S., *A width independent fast thinning algorithm*, IEEE Transactions on Pattern Analysis and Machine Intelligence **PAMI-7**(4), 1985, pp. 463–474.
[11] OHTSUKA T., WATANABE D., AOKI H., *Fingerprint core and delta detection by candidate analysis*, MVA2007 IAPR Conference on Machine Vision Applications, Tokyo, Japan, 2007, pp. 130–133.
[12] BUCKO L., *Fingerprint Identification*, M.Sc. Thesis, Białystok University of Technology, Poland, 2009, (in Polish: *Identyfikator linii papilarnych* – Praca magisterska).

Received December 12, 2012
in revised form February 3, 2013