Fingerprint image quick processing

Vladimir Gudkov, Chelyabinsk State University, Chelyabinsk, Russia
diana@sonda.ru
Maxim Bokov, South Ural State University, Chelyabinsk, Russia
guardianm@mail.ru

Abstract

The article briefly describes an approach of features extraction from fingerprint image with time restriction. Minutiae are saved in an image template. Templates are used for fingerprint identification.

Keywords: *Fingerprint, minutiae, image flow, image processing.*

1. INTRODUCTION

More than a century has passed since investigations in the field of biometrics started. Fingerprints have been used for over a century and are the most widely used form of biometric identification. Fingerprint identification is commonly employed in forensic science to support criminal investigations, and in biometric systems such as civilian and commercial identification devices. Criminal system must be very reliable whereas civilian and commercial systems must be fast and reliable [7]. It influences deeply on choice of fingerprint recognition methods, for example in the systems like Control and Access systems, Time and Attendance systems.

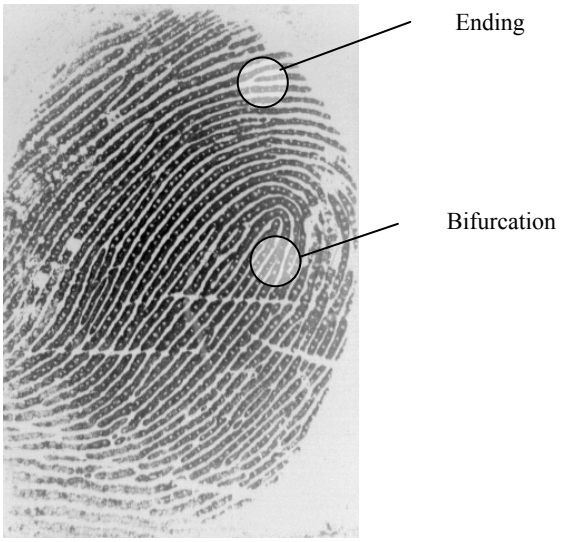


Figure 1: Fingerprint image.

The two most prominent ridge characteristics, called minutiae are: ridge endings and ridge bifurcations. Ridge endings are the points where the ridge curve terminates, and bifurcations are the ones where the ridge splits from a single path into two paths. Figure 1 illustrates an example of a ridge ending and a bifurcation. In the fingerprint image, ridges are dark whereas valleys are bright.

Quick processing is realized as a series of techniques, which include estimation of local ridge orientation, smoothing along ridges, estimation of local ridge period, segmentation, binarization and skeletonization. Although a minutiae-based approach is characterized by a high saliency, reliable automatic minutiae extraction can be problematic in extremely low-quality fingerprints devoid of any ridge structure. Time restriction limits a class of fingerprint images to medium and high quality fingerprint images.

The primary aim of this article is to implement a series of techniques for fingerprint image enhancement and minutiae extraction. Experiments with real fingerprint images were used to assess the performance of the implemented techniques.

Usually steps of preprocessing and enhancement are performed in order to simplify the fingerprint recognition. Therefore fingerprint image is presented in rectangular area G with a power $|G| = x_0 y_0$ as $F = \{f(x,y) \in 0..2^b - 1 \,|\, (x,y) \in X \times Y\}$, where b – represents the number of code bits allocated to each pixel (bit depths); $X = 0..x_0 - 1$ and $Y = 0..y_0 - 1$.

Consider two-dimensional images. An image represents an important class of data structures. Data objects may be taken as pixels, but it is more meaningful for image interpretation if we try, in some appropriate way, to take regions of the image as the data objects. Such regions may be approximate. One approach is to recursively divide the image into smaller regions. Such regions may be square or rectangular, to facilitate general implementation. Fingerprint image processing is represented structurally by pyramid $\Re$ of correlated hierarchies [1, 2, 6]. Image segmentation is performed for any layer of arbitrary hierarchy. For instance $l-th$ layer of $k-th$ hierarchy $F_k^{(l)}$ is segmented $x_h y_h$ square blocks $S_{hk}^{(l)}(x,y)$ with side length 2^{h-k} and vertices $(x,y) \in X_h \times Y_h$, where $k < h$, h is hierarchy number and $X_h = 0..x_h - 1$, $Y_h = 0..y_h - 1$.

Access to any point of the block $S_{hk}(x,y)$ is written in coordinates $(u,v) \in \overline{X}_{hk} \times \overline{Y}_{hk}$ as:

$$\begin{cases} \overline{X}_{hk} = & \{u + x2^{h-k} \,|\, x \in X_h \wedge u \in 0..2^{h-k} - 1\}, \\ \overline{Y}_{hk} = & \{v + y2^{h-k} \,|\, y \in Y_h \wedge v \in 0..2^{h-k} - 1\}. \end{cases} \quad (1)$$

Access to the central point of the block $S_{hk}(x,y)$ is written in coordinates $(u,v) \in \hat{X}_{hk} \times \hat{Y}_{hk}$ as:

$$\begin{cases} \hat{X}_{hk} = & \{2^{h-k-1} + x2^{h-k} \,|\, x \in X_h\}, \\ \hat{Y}_{hk} = & \{2^{h-k-1} + y2^{h-k} \,|\, y \in Y_h\}. \end{cases} \quad (2)$$

Block size of $h-th$ hierarchy is: $x_h = \left\lceil \frac{y_0}{2^h} \right\rceil$ and $y_h = \left\lceil \frac{y_0}{2^h} \right\rceil$, where $\lceil a \rceil$ – ceiling a.

With hierarchical segmentation, blocks of the layer F_k map to blocks' vertices of the layer F_h of pyramid $\Re$, where $k < h$. Correspondingly blocks' vertices map to blocks situated closer to base of pyramid [2]. Block size affects execution time and processing quality deeply. Assume that $S_h(x,y) = S_{h0}(x,y)$ and vertices $S_h(x,y) \in F_h$.

Pyramid's layers can be represented as a set of real numbers and initial image can be represented as a set of nonnegative real numbers [1, 6]. It gets rid of tiresome integer representation of a signal and simplifies expression, though image discretization (pyramid's layers) is kept.

Apertures are used widely for compact mathematical formalization. Meanwhile, slotted rectangular aperture $A_h(x,y,\alpha,w)$, $A_h^-(x,y,\alpha,w)$ and circular aperture $A_h(x,y,w)$ play a crucial part. These apertures are represented by set of the layer's points of $h-th$ hierarchy and associated angles as (u,v,β). Aforesaid aperture can be defined as:

$$\begin{cases} A_h(x,y,\alpha,w) = \{(u,v,\beta) = (x + \rceil w\cos(\alpha)\lfloor, y + \rceil w\sin(\alpha)\lfloor, \beta) \mid w \in Z_w\}, \\ A_h^-(x,y,\alpha,w) = \{(u,v,\beta) = (x + \rceil w\cos(\alpha)\lfloor, y + \rceil w\sin(\alpha)\lfloor, \beta) \mid w \in Z_w^-\}, \end{cases} \quad (3)$$

$$A_h(x,y,w) = \bigcup_{\alpha \in Z^*} A_h(x,y,\alpha,w), \quad (4)$$

where $(x,y) \in X_h \times Y_h$ is the centre of aperture; $(u,v) \in X_h \times Y_h$ is the point of aperture; w is the size of aperture; $Z_w = 1..w$; $Z_w^- = -w..-1 \cup 1..w$; α is the direction of aperture; $\rceil a \lfloor$ - the nearest integer part of a. The direction from the centre of aperture (x,y) to the point (u,v) is defined as:

$$\beta = \operatorname{arctg}\left(\frac{v-y}{u-x}\right) + \pi n \mid n \in 0..1 .$$

Fingerprint image preprocessing and enhancement must meet time restriction requirements and also algorithm must ensure acceptable quality of minutiae recognition, which can be verified with test base. List of minutiae is defined as:

$$L_m = \{M_i = \{x_i, y_i, \alpha_i, t_i\} \mid i \in 1..n\}, \quad (5)$$

where M_i – minutiae with i index number; (x_i, y_i), α_i, t_i – coordinates, direction and type of minutiae; $n = |L_m|$ – cardinal number. Minutiae are detected in informative area of an image.

Then trade-off decision of speed – quality violation is implementation of six steps of fingerprint image processing, like this:

– orientation estimation;

– smoothing;

– period estimation;

– segmentation;

– binarization;

– skeletonization and minutiae recognition.

2. QUICK PROCESSING

The main goal of most fingerprint recognition algorithms [8] is minutiae, which can be used to calculate additional parameters of the fingerprint image. Quick processing is executed for initial fingerprint image $F_0^{(0)} = \{f_0^{(0)}(x,y)\}$ (see Figure 1).

2.1 Integral Image

An integral image (also known as a summed-area table) is a tool that can be used whenever we have a function from pixels to real numbers $f(x,y)$ (for instance, pixel intensity), and we wish to compute the sum of this function over a rectangular region of the image.

Without an integral image, the sum can be computed in linear time per rectangle by calculating the value of the function for each pixel individually. However, if we need to compute the sum over multiple overlapping rectangular windows, we can use an integral image and achieve a constant number of operations per rectangle with only a linear amount of preprocessing.

To compute the integral image, we store at each location, $I(x,y)$ the sum of all $f(x,y)$ terms to the left and above the pixel (x,y). This is accomplished in linear time using the following equation for each pixel (taking into account the border cases),

$$I(x,y) = f(x,y) + I(x-1,y) + I(x,y-1) - I(x-1,y-1). \quad (6)$$

Figure 2 illustrates the computation of an integral image. Once we have the integral image, the sum of the function for any rectangle with upper left corner (x_1, y_1), and lower right corner (x_2, y_2) can be calculated in constant time using the following equation,

$$\sum_{x=x_1}^{x_2} \sum_{y=y_1}^{y_2} f(x,y) = I(x_2,y_2) - I(x_1-1,y_2) - I(x_2,y_1-1) + I(x_1-1,y_1-1). \quad (7)$$

4	1	2	2
0	4	1	3
3	1	0	4
2	1	3	2

4	5	7	9
4	9	12	17
7	13	16	25
9	16	22	33

Figure 2: The integral image. Left: A simple input of image values. Right: The computed integral image.

2.2 Orientation estimation

The orientation estimation is a fundamental step in the processing and enhancement. It consists of two successive procedures. The simplest and most natural approach for extracting local ridge orientation is based on computation of gradients in the fingerprint image. The gradient $\nabla_x(x,y)$ at point (x,y) of $f_0^{(0)}(x,y)$, is a two-dimensional vector $[\nabla_x(x,y), \nabla_y(x,y)]$ where ∇_x and ∇_y components are the partial derivatives of $f_0^{(0)}(x,y)$ in (x,y) with respect to the x and y directions, respectively.

Evaluation of the orientation matrix. The essence of the method is to divide fingerprint image into nonoverlapping blocks $S_h(x,y)$ corresponding to Equation (1) and estimate the orientation $0 \le \delta_h^{(0)}(x,y) < 180$ block-wise on hierarchy $h = 3(8 \times 8)$. An orientation image is then calculated, which is a matrix $\Lambda_h^{(0)}$ of direction vectors representing the ridge orientation at each

location in the image. The matrix $\Lambda_h^{(0)}$ can be estimated using following equation:

$$\Lambda_h^{(0)} = \left\{\delta_h^{(0)}(x,y)\right\} = \left\{\frac{\pi}{2} + \frac{1}{2}\operatorname{arctg}\left(\frac{2J_{12}(x,y)}{J_{22}(x,y) - J_{11}(x,y)}\right)\right\}, \quad (8)$$

where

$$J_{12}(x,y) = \sum_{(u,v)\in S_h(x,y)} \nabla_x \nabla_y,$$

$$J_{11}(x,y) = \sum_{(u,v)\in S_h(x,y)} \nabla_x \nabla_x,$$

$$J_{22}(x,y) = \sum_{(u,v)\in S_h(x,y)} \nabla_y \nabla_y.$$

Here ∇_x and ∇_y are the x- and y-gradient components in $(u,v) \in \overline{X}_{hk} \times \overline{Y}_{hk}$ corresponding to Equation (1) of the blocks $S_h(x,y)$ computed through 3×3 Sobel masks as:

$$\nabla_x = \mathbf{H}_x ** f_0^{(0)}(u,v),$$

$$\nabla_y = \mathbf{H}_y ** f_0^{(0)}(u,v),$$

where $\mathbf{H}_x$ and $\mathbf{H}_x$ is direction error reduced convolution kernel [6]

$$\mathbf{H}_x = \begin{bmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{bmatrix}, \mathbf{H}_y = \begin{bmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{bmatrix}.$$

The steps for calculating the orientation matrix $\Lambda_h^{(l)}$ are as follows:

1. Firstly, integral images $IG_{xy}(x,y), IG_{xx}(x,y)$ and $IG_{yy}(x,y)$ must be computed according to Equation (6) in order to compute $J_{12}(x,y), J_{11}(x,y), J_{22}(x,y)$ for different-sized rectangular apertures $A_h(x,y,w)$ in constant time.
   For instance:
   $$IG_{xy}(x,y) = G_{xy}(x,y) + IG_{xy}(x-1,y) + IG_{xy}(x,y-1) - IG_{xy}(x-1,y-1),$$
   where $G_{xy}(x,y) = \nabla_x(x,y)\nabla_y(x,y)$. The integral images $IG_{xx}(x,y)$ и $IG_{yy}(x,y)$ can be calculated similarly.

2. After computing $IG_{xy}(x,y), IG_{xx}(x,y)$ and $IG_{yy}(x,y)$ we can calculate $J_{12}(x,y)$, $J_{11}(x,y)$, $J_{22}(x,y)$ by using following equations:
   $$J_{12}(x,y) = IG_{xy}(x_2,y_2) - IG_{xy}(x_1-1,y_2) - IG_{xy}(x_2,y_1-1) + IG_{xy}(x_1-1,y_1-1),$$
   $$J_{11}(x,y) = IG_{xx}(x_2,y_2) - IG_{xx}(x_1-1,y_2) - IG_{xx}(x_2,y_1-1) + IG_{xx}(x_1-1,y_1-1),$$
   $$J_{22}(x,y) = IG_{yy}(x_2,y_2) - IG_{yy}(x_1-1,y_2) - IG_{yy}(x_2,y_1-1) + IG_{yy}(x_1-1,y_1-1),$$
   where the point (x_1, y_1) is the upper left corner of the aperture $A_h(x,y,w)$ and the point (x_2, y_2) is the lower right corner of the aperture $A_h(x,y,w)$.

Actually, elements in $\Lambda_h^{(l)}$ are computed by averaging in blocks $\{S_h(x,y)\}$ by means of structural point-wise tensor operator J as:

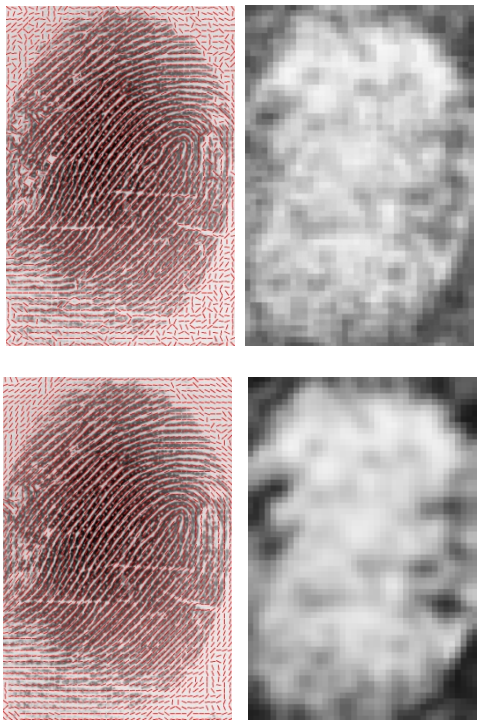
$$J = \begin{bmatrix} J_{11} + J_{22} \\ J_{22} - J_{11} \\ 2J_{12} \end{bmatrix}. \quad (9)$$

Analyze and correction the orientation matrix. The reliability of the estimate $\delta_h^{(0)}(x,y)$ can be derived by the coherence of the orientation vectors in the local apertures $A_h(x,y,w)$. For the gradient-based approach corresponding to Equation (8) we can calculate coherence matrix, which defined as:

$$\mathrm{M}_h^{(0)} = \left[\mu_h^{(0)}(x,y)\right] = \left[\frac{\sqrt{(J_{22}(x,y) - J_{11}(x,y))^2 + 4J_{12}^2(x,y)}}{J_{11}(x,y) + J_{22}(x,y)}\right]. \quad (10)$$

An example of local coherence map computed with Equation (10) is shown in Figure 3. The coherence of ideal lines is equal to one, but the coherence of isotropic structures is equal to zero [6].

Due to the integral images $IG_{xy}(x,y), IG_{xx}(x,y)$ and $IG_{yy}(x,y)$ we can compute orientation matrices $\Lambda_h^{(0)}$, $\Lambda_h^{(1)}$, $\Lambda_h^{(2)}$, $\Lambda_h^{(3)}$ and coherence matrices $\mathrm{M}_h^{(0)}$, $\mathrm{M}_h^{(1)}$, $\mathrm{M}_h^{(2)}$, $\mathrm{M}_h^{(3)}$ with different apertures $A_h(x,y,w)$ like these $A_h(x,y,8)$, $A_h(x,y,24)$, $A_h(x,y,40)$, $A_h(x,y,56)$ respectively ($h = 3$). Figure 3 illustrates the results of these calculations.



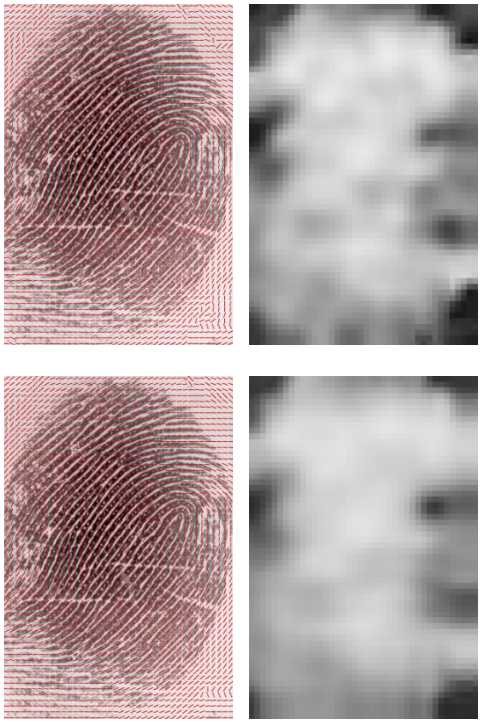


Figure 3: Top-down: matrices $\Lambda_h^{(0)}$, $\Lambda_h^{(1)}$, $\Lambda_h^{(2)}$, $\Lambda_h^{(3)}$ and their coherence matrices $\mathrm{M}_h^{(0)}$, $\mathrm{M}_h^{(1)}$, $\mathrm{M}_h^{(2)}$, $\mathrm{M}_h^{(3)}$ respectively.

After calculation of the orientation matrices $\Lambda_h^{(0)}$, $\Lambda_h^{(1)}$, $\Lambda_h^{(2)}$, $\Lambda_h^{(3)}$ and their coherence matrices $\mathrm{M}_h^{(0)}$, $\mathrm{M}_h^{(1)}$, $\mathrm{M}_h^{(2)}$, $\mathrm{M}_h^{(3)}$ we can analyze them. Let's compute a derivatives matrix $\mathrm{N}_h^{(0)}$ and an orientation matrix $O_h^{(0)}$, which contain the orientation with maximum coherence from different orientation matrices. Matrices $\mathrm{N}_h^{(0)}$ and $O_h^{(0)}$ can be defined as:

$$\mathrm{N}_h^{(0)} = \left[\nu_h^{(0)}(x,y)\right] = \left[\frac{1}{n}\sum_{l=1}^{n-1}(\mu_h^{(l)}(x,y) - \mu_h^{(l-1)}(x,y))\right],$$

$$O_h^{(0)} = \left[o_h^{(0)}(x,y)\right] = \left[\lambda_h^{(c)}(x,y)\right],$$

where $c = \arg\max_l(\mu_h^{(l)}(x,y))$; n is a quantity of the layers, in this implementation $n=4$.

Thus, $\mathrm{N}_h^{(0)}$ helps us assess how the coherence changes. It allows finding blocks with singularities like delta, loop and whorl.

Then, the orientation matrix $\Lambda_h^{(4)}$ containing reliable and accurate orientations can be computed by analyzing matrices $N_h^{(0)}$ and $O_h^{(0)}$. Matrix $\Lambda_h^{(4)}$ is defined as:

$$\Lambda_h^{(4)} = \left[\nu_h^{(4)}(x,y)\right] = \begin{cases}\Lambda_h^{(3)}, & \text{if } N_h^{(0)} > T,\\ O_h^{(0)}, & \text{else,}\end{cases}$$

where T is a threshold, which can be calculated experimentally (in this implementation $T = -0.1$).

Elements of $\Lambda_h^{(4)}$ can be corrected by using following equation:

$$\Lambda_h^{(5)} = \left\{\delta_h^{(5)}(x,y)\right\} = \left\{\frac{1}{2}\mathrm{arctg}\left(\frac{im_h^{(0)}(x,y)}{re_h^{(0)}(x,y)}\right)\right\}, \qquad (11)$$

where

$$re_h^{(0)}(x,y) = \sum_{k=-1}^{1}\sum_{m=-1}^{1}\mu_h^{(0)}(x+k,y+m)\cos(2\delta_h^{(4)}(x+k,y+m)),$$

$$im_h^{(0)}(x,y) = \sum_{k=-1}^{1}\sum_{m=-1}^{1}\mu_h^{(0)}(x+k,y+m)\sin(2\delta_h^{(4)}(x+k,y+m)).$$

Thereby, the orientation matrix $\Lambda_h^{(4)}$ is more accurate and more reliable than the matrix $\Lambda_h^{(4)}$. Figure 4 illustrates the matrix $\Lambda_h^{(4)}$ and its correction, the matrix $\Lambda_h^{(5)}$.

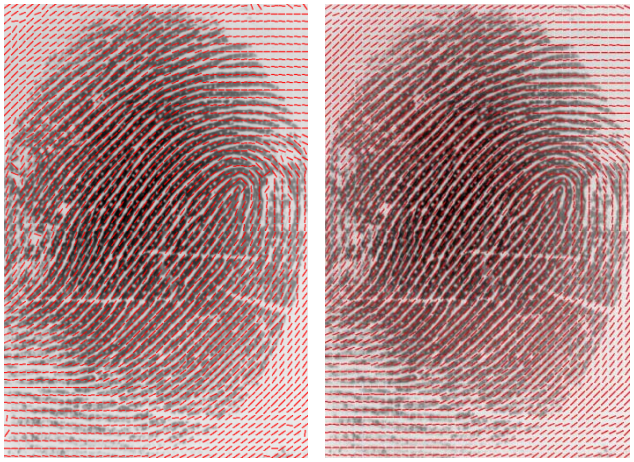


Figure 4: Fingerprint's orientation. Left: the orientation matrix $\Lambda_h^{(4)}$. Right: the orientation matrix $\Lambda_h^{(5)}$.

2.3 Smoothing

Smoothing eliminates ruptures and smudges. Before smoothing irregularity matrix $IR_h^{(0)}$ is computed by using following equation:

$$IR_h^{(0)} = \left[ir_h^{(0)}(x,y)\right] = \left[1 - \frac{\left\|\sum_{m=-1}^{1}\sum_{n=-1}^{1}d(x+n,y+m)\right\|}{\sum_{m=-1}^{1}\sum_{n=-1}^{1}\left\|d(x+n,y+m)\right\|}\right],$$

where $d(x,y) = \left[\cos 2\delta_h^{(5)}(x,y), \sin 2\delta_h^{(5)}(x,y)\right]$.

After that apply smoothing filter for each block $S_h(x,y)$ of the image $F_0^{(0)}$. Values from the base of the block $S_h(x,y)$ are smoothed by using following equation:

$$F_0^{(1)} = \begin{cases} f_0^{(1)}(x,y) = \mathbf{H}_1 * \Xi_0^{(\alpha)}(x,y), & if \quad ir_h^{(0)}(x,y) < t,\\ f_0^{(1)}(x,y) = f_0^{(0)}(x,y), & \text{else,}\end{cases} \qquad (12)$$

where $\mathbf{H}_1$ is a kernel of the one dimensional convolution; t is a threshold (in this implementation $t = 0.5$); a set $\Xi_0^{(\alpha)}(x,y) = \left\{\xi_0^{(\alpha)}(u,v)\right\}$ consists of the elements, which are chosen from $F_0^{(0)}$ with aperture according to Equation (3), as $\{\xi_0^{(\alpha)}(u,v)\} = \{f_0^{(0)}(u,v) \mid (u,v) \in A_0^{-}(x,y,\alpha,w) \cup (x,y)\}$;

$\alpha = \delta_h^{(5)}(x,y) \in \Lambda_h^{(5)}$ – direction of the aperture, that is identical for all elements of the block $S_h(x,y)$; w – size of the aperture.

Renumber ordered counts of the set $\Xi_0^{(\alpha)}(x,y) = \left\{\xi_0^{(\alpha)}(u,v)\right\}$, which is generated by slotted rectangular aperture according to Equation (3), as $k \mapsto (u_k, v_k)$, where $k \in 0..N$; $N = 2w+1$. Then the convolution kernel $\mathbf{H}_1$ is calculated as:

$$\mathbf{H}_1 = \exp\left(-\frac{(w-k)^2}{2\sigma^2}\right),$$

where σ is the standard deviation, that defines Gaussian steepness [8] (2-4 in this implementation); $k \equiv w$ is center of the aperture.

Smoothing filter is inherently first factor of an even-symmetric Gabor Filter [8]. Smoothing result is shown in Figure 5.

Figire 5: Smoothed image.

2.4 Period estimation

In addition to the orientation matrix, another important parameter that is used in the fingerprint recognition is the local ridge period. This step of fingerprint image processing is also consists of two successive procedures and performed in the same hierarchy $h = 3$.

Evaluation of the period matrix. Method is based on autocorrelation function.

Definition 1. The local ridge period is a value $t = w/n$ inversely to the number of ridges per unit length along a hypothetical segment centered at (x,y) and orthogonal to the local ridge orientation [8].

Specify a slit $C(x,y) = A_0^-(x,y,\alpha,w) \cup (x,y)$, which is generated by slotted rectangular aperture according to Equation (3), and renumber counts $(u,v) \in C(x,y)$ as $k \mapsto (u_k, v_k)$, where $k \in 0..N$; $N = 2w+1$. In the slit $C(x,y)$ with center $k \equiv w$ k-ordered values of the image $f_0^{(1)}(k)$ is collected ($w = 16$ in this implementation). Orientation of the slit $C(x,y)$, that is represented angle α, can be defined as:

$$\alpha = \frac{\pi}{2} + \delta_h^{(5)}(x,y).$$

The autocorrelation function can be calculated by using following equations [5]:

$$r(i) = \frac{1}{N}\sum_{k=0}^{N-i-1} \hat{f}_0^{(1)}(k)\hat{f}_0^{(1)}(k+i), \qquad (13)$$

$$\hat{f}_0^{(1)}(k) = f_0^{(1)}(k) - \bar{f},$$

$$\bar{f} = \frac{1}{N}\sum_{k=0}^{N-1} f_0^{(1)}(k).$$

Set difference as: $\Delta r(i) = r(i+1) - r(i)$; N is quantity counts for calculating of autocorrelation function. Then elements of the period matrix $\mathrm{T}_h^{(0)} = \left\{t_h^{(0)}(x,y)\right\}$ can be estimated as:

$$t_h^{(0)}(x,y) = \underset{j}{\operatorname{argmin}}\left\{\left(\Delta r(0),...,\Delta r(j)\right) \mid \Delta r(j-1) > 0 \wedge \Delta r(j) \le 0\right\}. \quad (14)$$

Actually, for each block $S_h(x,y)$, where $h = 3$, we find the center of the block and get the slit perpendicular to flow. Thereafter the slit is analyzed by the autocorrelation function according to Equation (13) and local period is calculated from Equation (14). Figure 6 illustrates results of the autocorrelation function for the white slit.

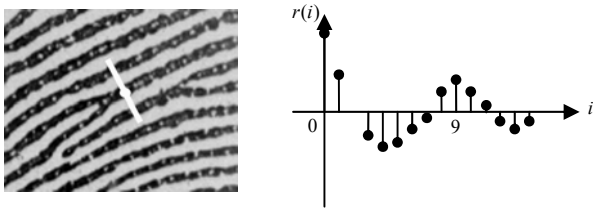


Figure 6: Calculating a local period.

Analyze and correction of the period matrix. On the principle that ratio of the constant component autocorrelation function and its value $r(t_h^{(0)}(x,y))$ equal 1 for sine or cosine, we can calculate reliability matrix $\mathrm{Z}_h^{(0)} = \left\{\zeta_h^{(0)}(x,y) = r(t_h^{(0)}(x,y))/r(0)\right\}$, $h = 3$.

It is well known that the local period can be in the range $4 \le t_h^{(0)}(x,y) \le 17$ for fingerprint image with resolution 500 dpi [8]. It allows deleting some mistakes of the local period calculations, by setting $t_h^{(0)}(x,y) = 0$.

The corrected period matrix can be calculated by using following equation:

$$t_{h,j}^{(1)}(x,y) = \frac{\sum_R t_{h,j-1}^{(1)}(u,v)\zeta_h^{(0)}(u,v)}{\sum_R \zeta_h^{(0)}(u,v)}, \qquad (15)$$

where condition of summing is $R = t_{h,j-1}^{(1)}(u,v) > 0$ and $n = \sum_R 1$ is quantity of nonzero elements (in this implementation $n > 4$).

Thereby, calculation errors are deleted, ridge periods are averaged and are predicted. The result of the calculated period matrix and its reliability matrix is shown in Figure 7. Period with value equal zero is black.

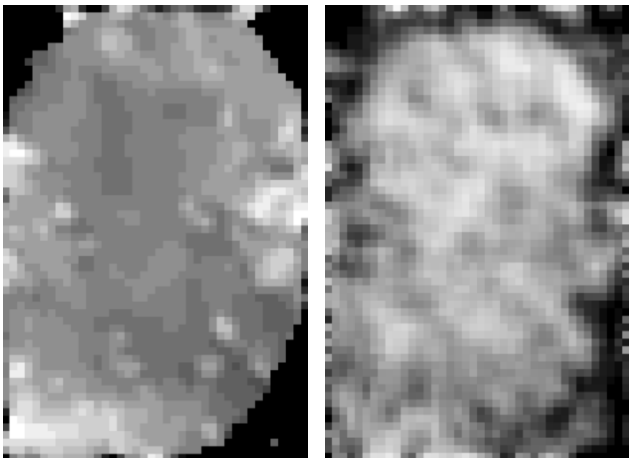


Figure 7: Left: the period matrix. Right: The reliability matrix.

2.5 Segmentation

Segmentation is the process of separating the foreground regions in the image from the background regions. The foreground regions correspond to the clear fingerprint area containing the ridges and valleys, which is the area of interest. The background corresponds to the regions outside the borders of the fingerprint area, which do not contain any valid fingerprint information. When minutiae extraction algorithms are applied to the background regions of an image, it results noisy and false minutiae. Thus, segmentation is employed to discard these background regions, which facilitates the reliable extraction of minutiae.

Segmentation is accomplished in the same hierarchy $h=3$ and consists in calculating a matrix $\mathrm{C}_h^{(0)} = \left\{c_h^{(0)}(x,y)\right\}$ as:

$$c_h^{(0)}(x,y) = \begin{cases} 1, & \text{if } k_1\zeta_h^{(1)}(x,y) + k_2\mu_h^{(1)}(x,y) > \kappa_0, \\ 0, & \text{else,} \end{cases} \quad (16)$$

where $\zeta_h^{(1)}(x,y)$ – period reliability value, smoothed with 3×3 mask; $\mu_h^{(1)}(x,y)$ – coherence according to Equation (10), smoothed with 3×3 mask; k_0, k_1, k_2 – learning coefficients.

Essentially, segmentation relies on two characters: coherence and period reliability. These characters are complex and their combination allows enhancing segmentation accuracy.

After segmentation islands of "diverse" regions can appear. They can be processed with mathematical morphology operations [1, 6], but under time restriction it is not advisable.

2.6 Binarization

Most minutiae extraction algorithms operate on binary images where there are only two levels of interest: the black pixels that represent ridges, and the white pixels that represent valleys. Binarization is the process that converts a grey level image into a binary image. This improves the contrast between the ridges and valleys in a fingerprint image, and consequently facilitates the extraction of minutiae.

Binarization is based on two earlier calculated matrices $\mathrm{T}_h^{(1)}$ and $\mathrm{C}_h^{(0)}$. The binarization process involves examining the grey-level value of each pixel of the block $S_h(x,y)$ of the image $F_0^{(1)}$ with mark $c_h^{(0)}(x,y) \in \{1\}$, and, if the value is greater than the local threshold, then the pixel value is set to a binary value one; otherwise, it is set to zero.

$$f_0^{(2)}(x,y) = \begin{cases} 1, \text{if } f_0^{(1)}(x,y) < km, \\ 0, \text{else.} \end{cases}$$

Threshold km is calculated from an aperture $A_h(x,y,w)$, where m is a mean value of $A_h(x,y,w)$; k - some coefficient (in implementation $k=0.98$); w is size of the aperture $A_h(x,y,w)$, that can be get from the matrix $\mathrm{T}_h^{(1)}$.

The mean value is calculated with the integral image, which allows accelerating binarization, according to Equation (6) and Equation (7). Figure 8 illustrates the result of binarization.

Figure 8: Binary image.

2.7 Skeletonization and minutiae recognition

The final fingerprint image quick processing step performed prior minutiae recognition is skeletonization (thinning). Skeletonization is the process of getting a skeleton by thinning of binary image (see Figure 8). In order to get accurate skeleton we can use mathematical morphology [1] to fill holes, remove small break and other artifacts. Work in some definitions.

Definition 2. Skeleton is a simple chain $\langle u,v \rangle$ with nodes u and v in eight-neighborhood and each node $p_1 \in \langle u,v \rangle$ has two adjacent to it nodes p_2 and p_3, though nodes p_2 and p_3 are non-adjacent.

Definition 3. Ending is a node p_1, which has only one adjacent node p_2.

Definition 4. Bifurcation is a node p_1, which has only three adjacent nodes p_2, p_3 and p_4, though any two nodes of a set $\{p_2, p_3, p_4\}$ are pair-wise non-adjacent.

Skeletonization is based on painting the points of $f_0^{(2)}(x,y) \in \{0\}$ according to the rules $P(\xi(x,y))$ defining in special table form. The rules $P(\xi(x,y))$ are based on a neighborhood identifier as:

$$\xi(x,y) = \sum_{i \in I} f(i) \cdot 2^i ,$$

where $f(i)$ is 1 for line (skeleton), otherwise 0; $i \in I = 0..7$ is a number of a aperture sector according to Equation (4). Value $\xi(x,y) \in 0..255$ determines table cell. According to [3], iterative application the rules $P(\xi(x,y))$ allows us to compute the skeleton. Endings and bifurcations are got from the skeleton's vertices [4], which situated in fingerprint image informative area, and placed to a list according to Equation (5). Skeleton and minutiae with their directions are shown in Figure 9. Endings are drawn with black color and bifurcations are drawn with grey color.

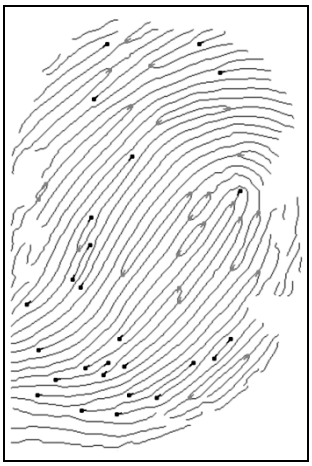


Figure 9: Skeleton and minutiae.

3. CONCLUSION

The primary focus of the work is on the quick processing of fingerprint images. In the article, it is proposed the group of techniques, which provide acceptable quality recognition with time restriction. Techniques include: estimation and correction the orientation matrix and its coherence, smoothing along ridges, estimation and correction period matrix and its reliability, segmentation of fingerprint image, binarization, skeletonization and minutiae recognition. Calculation of the orientation matrix is based on tensor analysis. Calculation of the period matrix is based on autocorrelation function. Total calculation time is about 100 milliseconds (Intel Pentium 4 CPU 3.0 GHz).

The algorithm also identifies the unrecoverable corrupted regions in the fingerprint image and removes them from further processing. This is a very important property because such unrecoverable regions appear in some of the corrupted fingerprint images and they are extremely harmful to minutiae extraction.

Also processing is based on convolutions, that allows integrating the processing into boards TMS and processors DSP [5] and using embedded convolution. The last makes possible realization of simple portable biometric systems, which work online.

Similarly to smoothing, one-dimensional Gabor filter can be used to filter image perpendicular to the orientation matrix. It recovers image and reduces time processing. Computed binary image can be smoothed again for more reliable and more accurate results.

4. REFERENCES

[1] R.C. Gonzales, R.E. Woods. Digital image processing, second edition. Prentice Hall, 2002.

[2] V.U. Gudkov. The methods of the first processing of dactiloscopic images, monography. Geotur, Miass, 2008.

[3] V.U. Gudkov, A.A. Kolyada, A.V. Chernyavskiy. New technology of dactiloscopic images skeleton formation. Methods, algorithms and software of flexible information technology for automated identification systems, Minsk, BSU, 1999, pp. 71-82.

[4] F.A Novikov. Discrete mathematics for programmers. Piter, 2008.

[5] A.B. Sergienko. Digital signal processing. Piter, 2002.

[6] B. Jahne. Digital image processing. Technosphere, 2007.

[7] R.M. Bolle, J.Y. Connel, S. Pankanti, N.K. Ratha. Guide to biometrics. Springer Verlag, 2004.

[8] D. Maio, D. Maltoni, A. K. Jain, and S. Prabhakar. Handbook of Fingerprint Recognition. Springer Verlag, 2003.

About the author

Vladimir Gudkov is a doctor at Chelyabisk State University, Department of Applied Mathematics. His contact e-mail is mailto:diana@sonda.ru.

Maxim Bokov is a Ph.D. applicant at South Ural State University, Department of Applied Mathematics. His contact email is mailto:guardianm@mail.ru.