Segmentation of Fingerprint Image Based on Automatic-parameter Normalization

WU Jian-fei

(*Shanghai Institute of Technical Physics, China Academic of Science, Shanghai* 200083)

Abstract　A combined method to segment fingerprint images, based on automatic-parameter normalization, is presented in this paper. Taking directional field and gray variance information in the fingerprint image into account, the method holds an efficient and robust feature. Compared with fixed-parameter normalization in former segmentation methods, the automatic-parameter normalization presented in this paper can normalize the fingerprint image to a maximum extent while not deteriorating any local image feature.

Keywords　Fingerprint segmentation, Histogram, Threshold, Valid area window, Gray variance

中图法分类号:TP301.6　　**文献标识码**:A　　**文章编号**:1006-8961(2008)04-0672-06

基于自动参数标准化的指纹分割方法

吴健飞

(中国科学院上海技术物理研究所,上海　200083)

摘　要　提出了一种合成的指纹分割方法:基于自动参数标准化的指纹分割方法。这个方法应用指纹图像中方向图和灰度变化信息,具有高效性和强壮性的特点。与以往指纹分割方法中固定参数标准化相比,基于自动参数标准化的指纹分割方法可以把指纹图像最大程度的标准化而不会恶化指纹图像细节。

关键词　指纹图像分割　直方图　阈值　有效窗口　灰度差异

1　Introduction

As a unique feature of every person, fingerprint is widely used in automatic personal identification. There are many steps included in order to extract a fingerprint minutia[1]. The segmentation of fingerprint images is a crucial step in the whole process. Several approaches to segment fingerprint image have been presented in recent years[2~6]. The simplest way to segment a fingerprint image is to perform a threshold operation with a suitable threshold, as presented in [2]. But it is far more than satisfiing. Bazen and Gerez presented a combined method, which used the directional fields and gray variance information[5,6]. Xinjian Chen, Jie Tian presented a Linear Classifier algorithm[4]. All methods presented in [5]、[6] and [4] are more promising than that of [2].

A combined method of segmenting fingerprint image, which is based on automatic-parameter normalization, is presented in this paper. Though the method is still using directional fields and gray variance information, the mathematical model in the method is more promising than that of [5]、[6]。

2　Segmentation of fingerprint image

2.1　Outline of the algorithm

The method includes following steps:

收稿日期:2006-08-28;**改回日期**:2006-12-08

第一作者简介:吴健飞(1979～　),男。助理研究员,2006 年于中国科学院上海技术物理所获硕士学位。主要从事数字信号处理和图像处理工作。E-mail:alphare@126.com

(1) A smart ‘valid area window’ is first used to roughly locate foreground of a fingerprint, which decrease the image areas needed to be computed afterwards.

(2) Normalization parameters are automatically calculated from the ‘valid area window’ and performed in the ‘valid area window’.

(3) Divide the ‘valid area window’ into blocks, with width and height of 16 pixels.

(4) Divide the ‘Valid area window’ into blocks, and calculate each block’s orientation degree.

(5) Calculate each block’s gray variance degree.

(6) Calculate each block’s score which means of how possible the block belongs to foreground.

(7) Identify the foreground blocks.

2.2 Calculate the ‘valid area window’

In order to reduce computation, it is necessary to estimate the location of the fingerprint, which ensures subsequent steps only operating in a ‘valid area window’. The process is as follows:

(1) Calculate the origin image’s gray threshold

For a high-quality fingerprint image, there are always at least two distinct peaks in the gray histogram of the image, the peak of the highest gray representing background and the peak of the lowest gray representing foreground, as figure 1(a) shows. While for a low-quality fingerprint image, normally there is only one peak, representing background, or no peak at all, as figure 1(b) shows.

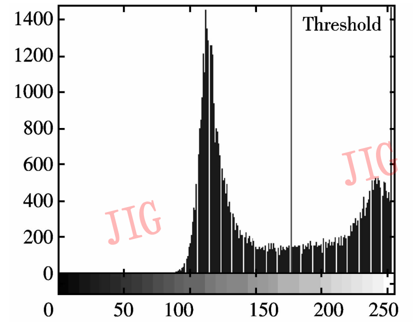
If a fingerprint image has two peaks in its gray histogram, we calculate the gray values of both peaks G_1 and G_2. In between G_1 and G_2, we find the valley, whose gray is G_3. Then the gray threshold of the whole image would be

$$Thr = G_3 \tag{1}$$

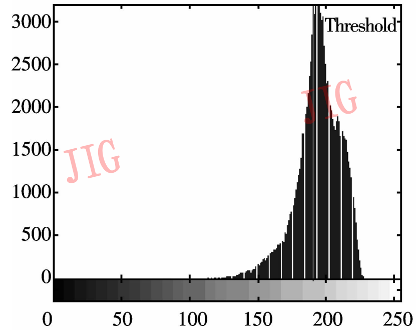
Else if the fingerprint image has more than two peaks in it’s gray histogram, we calculate the gray values of both the first peak G_1 and the last peak G_2, which represent foreground and background respectively. Then the gray threshold of the whole image would be

$$Thr = \frac{G_1 + G_2}{2} \tag{2}$$

Else if the fingerprint image has no more than one peak in its gray histogram, then the gray threshold of the whole image would be:



(a) A high-quality fingerprint image gray histogram



(b) A low-quality fingerprint image gray histogram

Fig. 1 Gray histogram of a high-quality fingerprint image and low-quality fingerprint image

$$Thr = \frac{1}{W \times H} \sum_{i=0}^{W-1} \sum_{j=0}^{H-1} Gray(i,j) \tag{3}$$

Where Width is the image’s width, Height is the image’s height and $Gray(i,j)$ is *pixel* (i,j)’s gray.

(2) Find the ‘valid area window’

In the whole fingerprint image, search a window, which exactly include the foreground.

The process is as follows:

- Find all pixels whose grays less than Thr.
- From top to bottom of the image, find the first line which has at least 5 pixels whose grays less than Thr, indicating the top of the ‘valid area window’. And in the same way find the window’s left, right and bottom, as figure 2 shows.

2.3 Normalization

Normalization is done in the ‘valid area window’ to remove the effects of sensor noise and finger pressure difference:

$$I_1(i,j) = \begin{cases} M_0 + \sqrt{VAR_0 (I(i,j) - M)^2 / VAR} & I(i,j) > M \\ M_0 - \sqrt{VAR_0 (I(i,j) - M)^2 / VAR} & \text{otherwise} \end{cases} \tag{4}$$

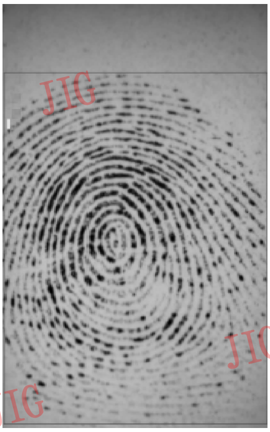


Fig. 2 A ‘valid area window’

I_1: normalized image.

I: original image.

M_0: desired mean.

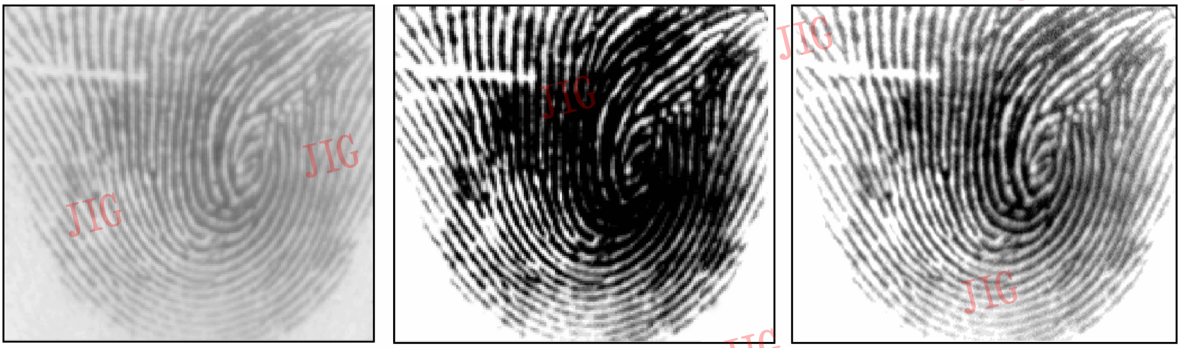
VAR_0: desired variance.

M: mean in origin ‘valid area window’.

VAR: variance in origin ‘valid area window’.

It was also used in several previous approaches[1,3,7,8]. But all of them use fixed parameters: M_0 and VAR_0, which in some cases will deteriorate some local image features, as figure 3 (b) shows.

While in the proposed algorithm, M_0 and VAR_0 are directly calculated in the ‘valid area window’.



(a) Original image (b) Image normalized using fixed parameters (c) Image normalized using automatic parameters

Fig. 3 Normalization using fixed parameters and automatic parameters

Auto-parameter Normalization:

(1) Calculate gray valiance and gray mean in the ‘valid area window’: *ValidGrayVar*, *ValidGrayMean*. Then in the equation (4), $M = ValidGrayMean$, $VAR = ValidGrayVar$.

(2) Calculate high-gray end: G_1, low-gray end: G_2, and valley: G_3 in the ‘valid area window’ respectively. high-gray end means: in the gray histogram of the fingerprint image, where the biggest gray locate, as figure4 shows. And in similar way we get low-gray end.

If the ‘valid area window’ has a valley in it's gray histogram, then $G_3 = valley$.

Else if the ‘valid area window’ has no valley in it's gray histogram at all, then G_3 is assigned:

$$G_3 = \frac{G_1 + G_2}{2} \tag{5}$$

(3) Adjust all pixel's gray in the ‘valid area window’.

Suppose the adjust value is *ads*. Then the adjusted ‘valid area window’ would be:

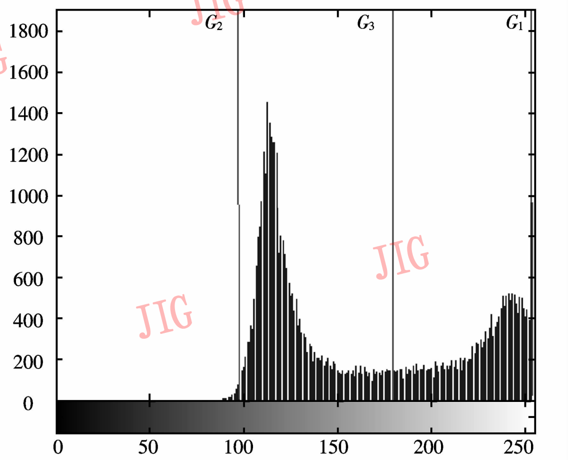
$$Gray'(i,j) = Gray(i,j) + adj \tag{6}$$

There exists an *adj*, when the adjusted image $Gray'(i,j)$ is normalized with equation (4), the low-gray end tends to 0 and the high-gray end tends to 255, that is:

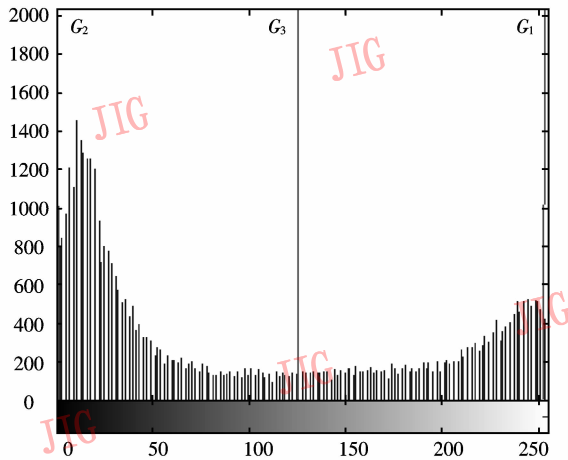
$$\begin{cases} \left(M_0 - \sqrt{\dfrac{VAR_0 (G_2 - M)^2}{VAR}} \right) \to 0 \\ \left(M_0 + \sqrt{\dfrac{VAR_0 (G_1 - M)^2}{VAR}} \right) \to 255 \end{cases} \tag{7}$$

By a recursive method, the exact value of *adj* can be easily got. Then from equation (7), M_0 and VAR_0 can be also got.

The histogram of the normalized ‘valid area window’ will perfectly span the whole gray domain, as figure 4(b) shows. Any local image feature will not be deteriorated, as figure 3(c) shows. This method normalized the ‘valid area window’ to a maximum extent while not deteriorating any local image feature.



(a) Histogram of the original ‘valid area window’



(b) Histogram of normalized ‘valid area window’

Fig. 4　Histogram of ’valid area window’ before and after normalization

2.4　Calculate each block’s orientation degree

(1) Divide the ‘valid area window’ into blocks, which have both width and height of 16 pixels. *block X* and *block Y* are horizontal and vertical block numbers.

(2) Calculate each pixel’s orientation in every block.

$$AnglePixel_{(p,q)}(i,j) = \tan^{-1}\frac{\partial_y(i,j)}{\partial_x(i,j)} \tag{8}$$

Where $AnglePixel_{(p,q)}(i,j)$ is the $pixel(i,j)$’s orientation angle in $block(p,q)$. $\partial_x(i,j)$ and $\partial_y(i,j)$ are the gradients at $pixel(i,j)$.

(3) Calculate each block’s orientation[9].

$$\begin{cases} V_x(i,j) = \sum\limits_{u=i-8}^{i+8}\sum\limits_{v=j-8}^{j+8} 2\partial_x(u,v)\partial_y(u,v) \\ V_y(i,j) = \sum\limits_{u=i-8}^{i+8}\sum\limits_{v=j-8}^{j+8} [\partial_x^2(u,v) - \partial_y^2(u,v)] \\ OriBlock(p,q) = \dfrac{1}{2}\tan^{-1}\left[\dfrac{V_y(i,j)}{V_x(i,j)}\right] \end{cases} \tag{9}$$

Where $OriBlock(p,q)$ is the least square estimate of the local ridge orientation angle at the block (p,q) centered at pixel(i,j).

(4) Due to the presence of noise, $OriBlock(p,q)$ may not always be a correct orientation angle for *block* (p,q). A low-pass filter can be used to modify the incorrect orientation angle in some blocks[9]. In order to perform the low-pass filtering, the orientation image needs to be converted into a continuous vector field, which is defined as follows:

$$\begin{cases} \Phi_x(p,q) = \cos[2OriBlock(p,q)] \\ \Phi_y(p,q) = \sin[2OriBlock(p,q)] \end{cases} \tag{10}$$

Where Φ_x and Φ_y are the x and y components of the vector field respectively. With the resulting vector field, the low-pass filtering can then be performed as follows:

$$\begin{cases} \Phi'_x(p,q) = \sum\limits_{u=-w_\Phi/2}^{w_\Phi}\sum\limits_{v=-w_\Phi/2}^{w_\Phi/2} W(u,v)\Phi_x(p-uw, q-vw) \\ \Phi'_y(p,q) = \sum\limits_{u=-w_\Phi/2}^{w_\Phi}\sum\limits_{v=-w_\Phi/2}^{w_\Phi/2} W(u,v)\Phi_y(p-uw, q-vw) \end{cases} \tag{11}$$

Where W is a 2-dimensional low-pass filter with unit integral and $w_\Phi \times w_\Phi$ specifies the size of the filter.

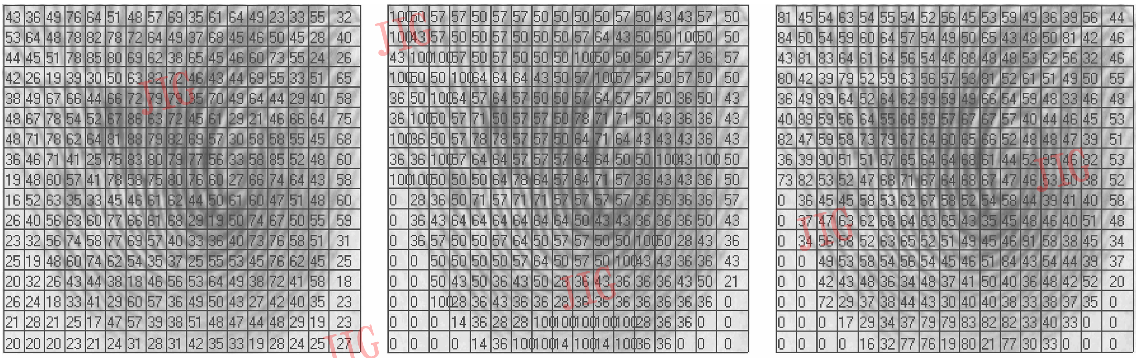
So, the smoothed orientation angle of the block(p,q) is:

$$AngleBlock(p,q) = \frac{1}{2}\tan\left[\frac{\Phi'_y(p,q)}{\Phi'_x(p,q)}\right] \tag{12}$$

(5) Calculate each block’s orientation degree.

$$\begin{cases} ORI\ degree(i,j) = 100N/(16\times 16) \\ N = \sum\limits_{i,j\in Block(p,q)} \operatorname{sgn}(AnglePixel(i,j), AngleBlock(p,q)) \\ \operatorname{sgn}(a,b) = \begin{cases} 1 & \operatorname{abs}(a-b) < 25° \\ 0 & \text{otherwise} \end{cases} \end{cases} \tag{13}$$

$ORIdegree(i,j)$: block(i,j)’s orientation degree, as figuer 5(a) shows.



(a) Block's orientation degree (b) Block's gray variance degree (c) Block's score

Fig. 5 Block's orientation degree, gray variance degree and score

2.5 Calculate each block's gray variance degree

(1) Calculate each block's gray variance: *GrayVar*(i,j).

(2) Calculate each block's gray variance degree:

$$GRAYdegree(i,j) = 100\,GrayVar(i,j)/VAR_0 \tag{14}$$

VAR_0: gray variance of the 'valid area window' after normalization.

$GRAYdegree(i,j)$: block (i,j)'s gray variance degree, as figure 5(b) shows.

$$\begin{cases} Score(i,j) = \begin{cases} 0 & \text{if}(ORIdegree(i,j) < ThOri \parallel GRAYdegree(i,j) < ThGray) \\ [A \times ORIdegree(i,j) + B \times GRAYdegree(i,j)]/(A+B) & \text{otherwise} \end{cases} \\ ThOri = [\max(ORIdegree(i,j)) - \min(ORIdegree(i,j))]/5 + \min(ORIdegree(i,j)) \\ ThGray = [\max(GRAYdegree(i,j)) - \min(GRAYdegree(i,j))]/5 + \min(GRAYdegree(i,j)) \end{cases} \tag{15}$$

2.6 Calculate each block's score

$Score(i,j)$: $block(i,j)$'s score which means how possible the block belongs to foreground.

A and B: weight factor of $ORIdegree(i,j)$ and $GRAYdegree(i,j)$. In our algorithm, $A = 1$, $B = 2$.

$ThOri$: block's orientation degree threshold.

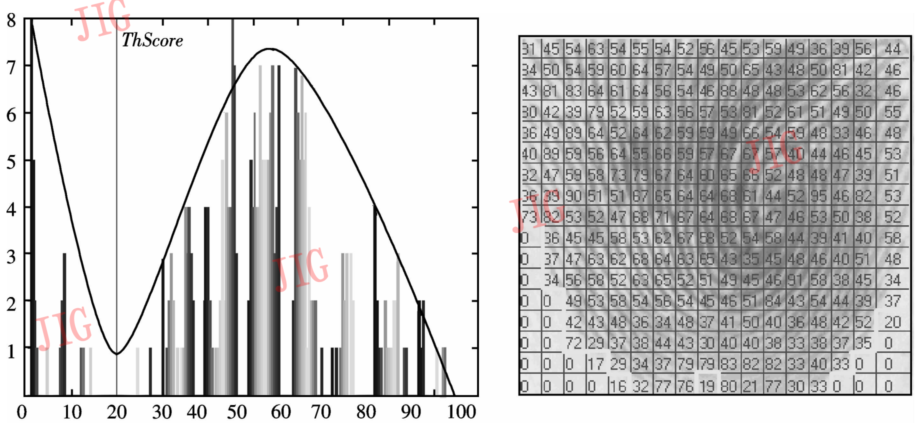
$ThGray$: block's gray valiance degree threshold.

2.7 Identify the foreground blocks

In the histogram of $Score(i,j)$, find the fist valley, as figure 6(a) shows, then

$$\begin{cases} ThScore = valley \\ block(i,j) \in \begin{cases} Forground & Score(i,j) < ThScore \\ Background & \text{otherwise} \end{cases} \end{cases} \tag{16}$$

Where $ThScore$ is block's score threshold. So the segmented foreground is as figure 6(b) shows:



(a) Histogram of $Score(i,j)$ (b) Foreground and background

Fig. 6 Block's score threshold and foreground and background

3 Experiment results

10 sample fingerprints in DB1_B, DB2_B, DB3_B and DB4_B of both FVC2002 and FVC2000, and 10 sample fingerprints in DB1_A, DB2_A, DB3_A and DB4_A of FVC2004 which are selected randomly are tested with the proposed algorithm, and the accuracy is assessed manually. as Table 1 shows.

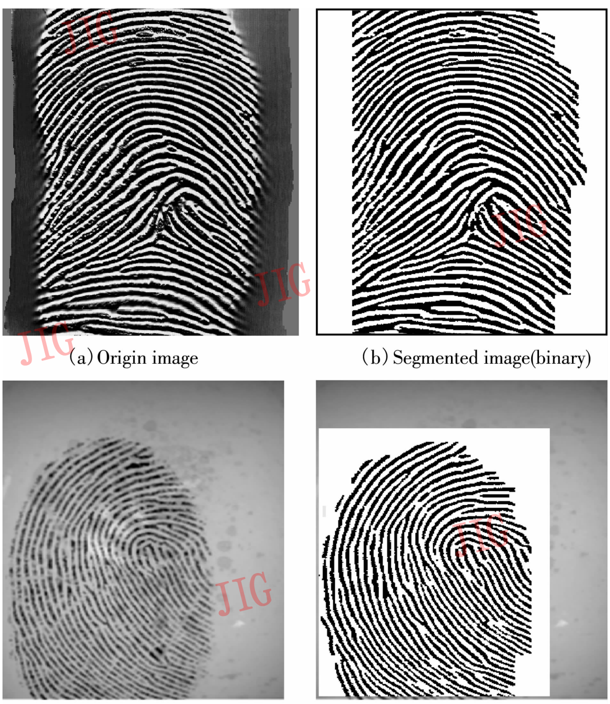
Some examples of the effect of the segementation also shown in figure 7.

Tab. 1　Accuracy of Segmentation (%)

	DB1_B or DB1_A	DB2_B or DB2_A	DB3_B or DB3_A	DB4_B or DB4_A
FVC2000	98.3	98.9	96.4	97.6
FVC2002	99.2	99.4	97.2	97.4
FVC2004	99.7	98.4	95.8	97.4

Tab. 2　Reduction of area in fingerprint image needed to be calculated using ‘valid area window’ (%)

	DB1_B or DB1_A	DB2_B or DB2_A	DB3_B or DB3_A	DB4_B or DB4_A
FVC2000	13.6	14.7	16.6	9.76
FVC2002	28.9	31.1	12.2	29.9
FVC2004	65.1	13.2	13.7	27.6



(a) Origin image　(b) Segmented image(binary)

(c) Origin image　(d) Segmented image(binary)

Fig. 7　Examples of the effect of the method

4 Conclusion

This paper presented a combined segmentation algorithm based on auto-parameter normalization method. This method utilizes the information of orientation degree and gray variance of each block. Compared with the fixed-parameter normalization in previous approaches, the auto-parameter normalization step in the proposed method normalizes the fingerprint image to a maximum extent while deteriorating any local image feature.

From the experiment results, only averagely 2.2% of the blocks in FVC2000, 1.7% of the blocks in FVC2002 and 2.2% of the blocks in FVC2004 are misclassified, which testifies the effectiveness of the proposed method.

参考文献(References)

1 Hong Lin. Automatic Personal Identification Using Fingerprints[D]. Michigan, USA: Michigan State University, June 25, 1998.

2 Ahmed Wathik Naji, Abd Rahman Ramli. A segmentation algorithm based-on histogram equalizer for fingerprint classification system[A]. In: Proceedings of Second International Conference on Electrical and Computer Engineering ICECE 2002[C], Dhaka, Bangladesh, 2002: 26～28.

3 Bazen A M. Gerez S H. Segmentation of fingerprint images[A]. In: ProRISC 2001 Workshop on Circuits, Systems and Signal Processing [C], Veldhoven, The Netherlands, 2001: 276～280.

4 Chen Xin-jian, Tian Jie. Segmentation of fingerprint images using linear classifier[J]. EURASIP Journal on Applied Signal Processing, 2004, **4**(4): 480～494.

5 Mehtre B M, Murthy N N, Kapoor S. Segmentation of fingerprint images using directional image[J]. Pattern Recognition, 1987, **20**(4): 429～435.

6 Mehtre B M, Chatterjee B. Segmentation of fingerprint image-Acomposite method[J]. Pattern Recognition, 1989, **22**(4): 381～385.

7 Chen Xin-jian, Tian Jie, Zhang Yang-yang. A robust orientation estimation algorithm for low quality fingerprints[A]. In: proceedings of International Workshop on Biometric Recognition Systems 2005 (IWBRS2005)[C], Beijing, China, 2005: 22～23.

8 Hong Lin, Wan Yi-fei, Anil Jain. Fingerprint image enhancement: algorithm and performance evaluation[J]. IEE Transactions on Pattern Analysis and Machine Interlligence, 1998, **20**(8): 777～789.