Fingerprint Recognition

Wenxin Fang, Zimou Zhang

***Abstract*—Based on minutiae points algorithm, this paper proposes a fingerprint recognition technique to match one fingerprint among various fingerprint images accurately. First, in order to raise quality of blurred and broken images, we implement the preprocess using enhancement and Gabor filter. Then two kinds of minutiae points are extracted from the print and validated. Finally, during the matching step, points are matched using Ransac under an affine transformation model. After testing upon database FVC2002 and FVC2004, we prove that our method is robust and accurate with 77% accuracy.**

I. Introduction

Fingerprints are one of the most common and trusted biometrics for personal identification. Because of their uniqueness and consistency over time, fingerprints have been used for identification for over a century. Fingerprint identification is popular because of the inherent ease in acquisition, the numerous sources available for collection, and their established use and collections by law enforcement and immigration.

In this paper, main objective is to present and implement an accurate and robust fingerprint recognition method. We adopt two most prominent kinds of minutiae: ridge ending and ridge bifurcation. To improve the robustness, we use Gabor filter several times to raise the image quality. For finding correspondences for points, we use geometric hashing instead of SIFT because of uniqueness of fingerprint.

II. Fingerprint Segmentation

An important step in fingerprint recognition is the segmentation of the region of interest (ROI). The objective of fingerprint segmentation is to extract the ROI which contains the desired fingerprint impression.

In our fingerprint recognition, we use modified gradient based method proposed in [1].Steps for this method are summarized as follows:

1) Divide the input image $I(i,j)$ into non-overlapping blocks with size $w \times w$. In our implementation, we use $w = 8$.
2) Use histogram equalization to enhance the contrast of each block.
3) Compute the gradient $\partial_x(i,j)$ and $\partial_y(i,j)$ at each pixel (i,j) which is the center of the block.
4) Compute the standard deviation for both $\partial_x(i,j)$ and $\partial_y(i,j)$.
5) Compute the gradient deviation using equation 1.

$$\text{grddev} = \text{std}_x + \text{std}_y \tag{1}$$

6) Select a threshold value empirically. If grddev is greater than threshold value, the block is considered as foreground otherwise it belongs to background.

III. Fingerprint Enhancement

A. Orientation Estimation

We use gradient based orientation estimation method in [2]. Given an image $I(i,j)$, the main steps of the algorithm are as follows:

1) Divide $I(i,j)$ into blocks of size $w \times w$. In our implementation, we choose$w = 8$.
2) Compute the gradient $\partial_x(i,j)$ and $\partial_y(i,j)$ at each pixel. In our implementation, we use Sobel operator to compute the gradient.
3) Estimate the local orientation of each block centered at pixel (i,j) using the following equations:

$$V_x(i,j) = \sum_{u=i-w/2}^{i+w/2} \sum_{v=j-w/2}^{j+w/2} 2\partial_x(u,v)\partial_y(u,v) \tag{2}$$

$$V_y(i,j) = \sum_{u=i-w/2}^{i+w/2} \sum_{v=j-w/2}^{j+w/2} (\partial_x^2(u,v) - \partial_y^2(u,v)) \tag{3}$$

$$\theta(i,j) = \frac{1}{2}\tan^{-1}\left(\frac{V_y(i,j)}{V_x(i,j)}\right) \tag{4}$$

where$\theta(i,j)$ is the estimation of the local ridge orientation at the block centered at pixel (i,j).
4) Due to the presence of noise, corrupted ridge, minutiae, etc. some $\theta(i,j)$ may not always be a correct estimate. Because local ridge orientation varies slowly, we can add a smooth filter to smooth the ridge orientations. In our implementation, we first use Gaussian filter to smooth

$V_x(i,j)$ and $V_y(i,j)$ as follow:

$$V'_x(i,j) = \sum_{i-w_g/2}^{i+w_g/2} \sum_{j-w_g/2}^{j+w_g/2} K(u,v)V_x(u,v) \quad (5)$$

where $K(u,v)$ is Gaussian kernel, w_g is the block size of the filter. We choose sigma=2, $w_g = 9$. After smoothing, we can calculate $\theta(i,j)$ using equation 4 except that we use $V'_x(i,j)$ and $V'_y(i,j)$ instead.

B. Ridge Frequency Estimation

Local fingerprint image patches are spatially and spectrally similar to s sinusoidal signal, where the dominant peaks in the magnitude spectrums of the two signals are co-located. The location of the dominant peak in the magnitude spectrum of a local image area carries information about the local orientation and frequency of the fingerprint pattern. Because the sample rate in the image is 1, so when the spectrum maximum is in the edge of the image, the ridge frequency is 0.5. So we can estimate the local ridge frequency using equation

$$f = \frac{D}{N}$$

where D is the distance from the spectrum maximum to the center of the image. We can easily to obtain wavelength by reciprocating f.

C. Gabor Filter

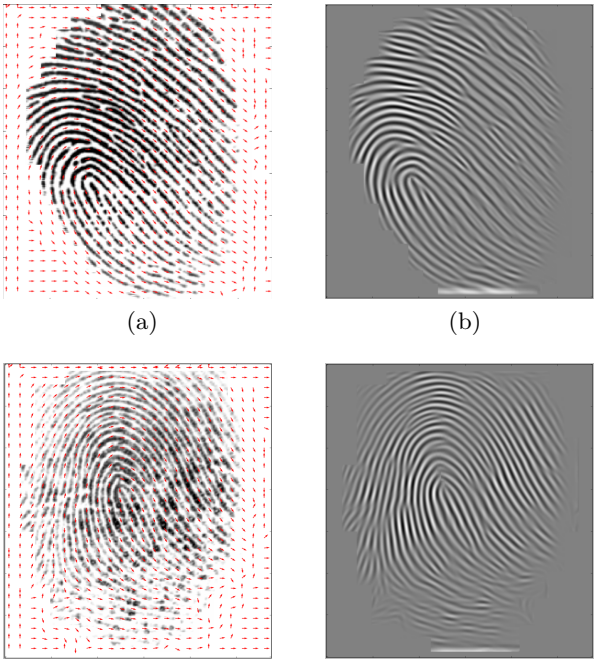
Gabor filter is a linear filter used for edge detection. In the spatial domain, a 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave. Gabor filter have both frequency-selective and orientation-selective properties and have optimal joint resolution in both spatial and frequency domains. Therefore, it is appropriate to use Gabor filter as bandpass filter to remove the noise and preserve true ridge structures.

$$g(x,y) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)\cos\left(2\pi\frac{x'}{\lambda} + \psi\right) \quad (6)$$

where

$$x' = x\cos\theta + y\sin\theta$$
$$y' = -x\sin\theta + y\cos\theta$$

θ is the ridge orientation and λ is the ridge wavelength, both of which can be obtained using methods we mentioned above. In our implementation , we set $\gamma = 1$, $\psi = 0$ and $\sigma = 5$.



(a) (b) (c) (d)

Fig. 1: Some experimental results. (a) Orientation image of a fingerprint image. (b) Enhanced image. (c) Orientation image of another fingerprint image. (d) Enhanced image.

P_9 $(i-1, j-1)$	P_2 $(i-1, j)$	P_3 $(i-1, j+1)$
P_8 $(i, j-1)$	P_1 (i, j)	P_4 $(i, j+1)$
P_7 $(i+1, j-1)$	P_6 $(i+1, j)$	P_5 $(i+1, j+1)$

Fig. 2: Neighbors designations in a 3×3 window

D. Experimental Results

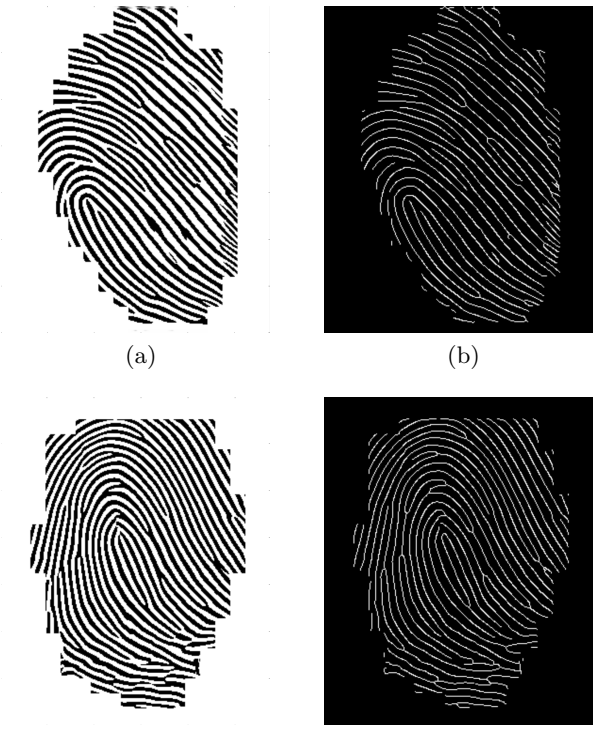
Some experimental results are shown in Fig. 1. We can see that using Gabor filter, we can remove the noise and preserve true ridge structures.

IV. Binarisation and Thinning

After the fingerprint image is enhanced, it is then converted to binary form, and submitted to the thinning algorithm which reduces the ridge thickness to one pixel wide[3]. In our implementation, we simply set a threshold 0, all values large than 0 will be changed to 1, and all values less than 0 will be changed to 0.

After binarisation, we use thinning algorithm proposed in [4]. The main steps of the algorithm is as follows:

1) Define neighbor points as in Fig. 2.



(a) (b) (c) (d)

Fig. 3: (a)(c) Binarisation images. (b)(d) Thinning images.

2) First subiteration. The point P_1 is deleted from the image if it satisfies the following conditions:
   a) $2 \leq B(P_1) \leq 6$, where $B(P_1)$ is the number of nonzero neighbors of P_1, that is, $B(P_1) = P_2 + P_3 + \cdots + P_9$.
   b) $A(P_1) = 1$, where $A(P_1)$ is the number of 01 patterns in the ordered set $P_2, P_3, \cdots, P_9$.
   c) $P_2 \times P_4 \times P_6 = 0$
   d) $P_4 \times P_6 \times P_8 = 0$
3) Second subiteration, only conditions c) and d) are changed as follows:
   c') $P_2 \times P_4 \times P_8 = 0$
   d') $P_2 \times P_6 \times P_8 = 0$
4) Repeat step 2 and step 3, until no point in the image can be deleted.

After binarisation and thinning, we can obtain images as shown in Fig. 3.

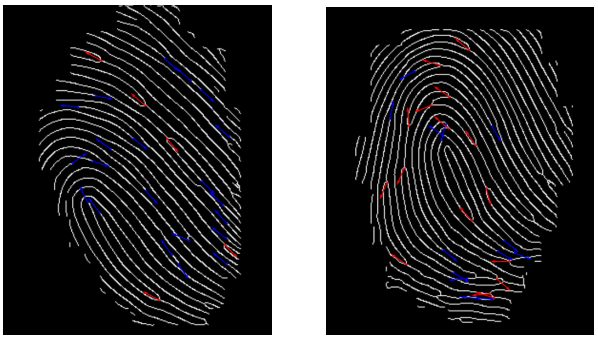
V. Minutiae Extraction

A. Minutiae Extraction

After a fingerprint image has been enhanced and thinned, the next step is to extract the minutiae from the image. The most commonly employed method of minutiae extraction is the Crossing Number (CN) concept. This method extracts the ridge endings and bifurcations from the skeleton image by examining the local neighborhood of each ridge pixel using a 3×3 window. The CN for a ridge pixel P is given by

$$\text{CN} = 0.5 \sum_{i=2}^{9} |P_i - P_{I+1}|, \quad P_{10} = P_2 \tag{7}$$

where P_i is the neighbor pixel as in Fig. 2. After the CN for a ridge pixel has been computed, the pixel can then be classified according to the property of its CN value. If CN value equals 1 then it is an ending point, if CN value equals 3, then it is a bifurcation point.



(a) (b)

Fig. 4: Minutiae extraction

B. Minutiae Validation

False minutiae may be introduced into the image due to factors such as noisy images, and image artifacts created by the thinning process. Hence, after the minutiae has been extracted, it is necessary to employ a validation process in order to validate the minutiae. We use the method proposed in [5] to validate the minutiae.

C. Experimental Results

After the minutiae has been extracted and validated, we can obtain the position and orientation of ending points and bifurcation points as shown in Fig. 4.

VI. Matching

A. Problem Formulation

Let $T(i,j)$ and $I(i,j)$ be the template and input fingerprint we are trying to match. $I(i,j)$ and $T(i,j)$ may not have the same number of feature points: let m and n be the number of point for $T(i,j)$ and $I(i,j)$.

$$T = \{m_1, m_2, \cdots, m_m\}, \text{ with } m_i = \{x_i, y_i, \theta_i\} \ i = 1...m$$
$$I = \{m'_1, m'_2, \cdots, m'_n\}, \text{ with } m'_j = \{x'_i, y'_j, \theta'_j\} \ j = 1...n$$

Two minutiae points m_i and m'_j are considered to match if their position and orientation are close. This can be written according to equation 8 and 9, using the spatial distance sd, and direction difference dd [6].

$$sd(m_i, m'_j) = \sqrt{(x_i - x'_j)^2 + (y_i - y'_j)^2} < R_0 \quad (8)$$

$$dd(m_i, m'_j) = \min(|\theta'_j - \theta_i|, 360 - |\theta'_j - \theta_i|) < \theta_0 \quad (9)$$

where R_0 and θ_0 are two tolerance parameters we can adjust to improve matching accuracy.

Let md be the function that says if two minutiae points are close or not i.e. $md(m_i, m'_j) = 1$ if $sd(m_i, m'_j) < R_0$ and $dd(m_i, m'_j) < \theta_0$, 0 otherwise.

B. Finding Correspondences

As a preprocessing step of Ransac, we should find for each point in the input image which points in the template are matching point possibly. We don't use conventional solution SIFT because a minutiae point looks like another point; then distance between them would be not accurate. We present a method called geometric hashing [7] which compute for each minutiae point a feature vector. And the smaller the feature distance, the more likely will be the points to match. First, we need to consider a minutiae point m_i in the template set and three closest points from template set: we number them $m_{i,1}$, $m_{i,2}$, $m_{i,3}$. To make feature vector independent of the rotation in the image, we rotate those points of an angle equal to the local orientation of m_i.

After we have a rotation invariant feature vector in the template and input set, we can compute Euclidean distance between every permutation of the points. Then we combine those numbers. To make it more robust, we allow one point to be wrong and require two points to match as in equation 10

$$d_2(m_i, m'_j) = \min_{k_1 \neq k_2, l_1 \neq l_2} (D_{k_1,l_1} + D_{k_2,l_2}) \quad (10)$$

Finally, for each m_i in the template, we choose the 5 points m'_j in the input set having smallest d_2 distance and assign them as being the correspondences.

C. Ransac

It's impossible to try every affine transformations due to combinatorial complexity. Thus, we use Ransac to try at random transformations and expect to find a good one.

Let k be number of iterations required, d be number of maximum matching points and *score* be threshold of matching degree. As shown before, R_0 and θ_0 be thresholds used to identify that a point fits well. The algorithm is based on three key processes as follows:

1) Take three random points both in template and input set.
2) Find the affine transformation mapping the template random points on the input points.
3) Apply the transformation to all template points. Count the number of points which match and give a score of match. If score is over *score*, then we decide these two images are from the same fingerprint.

D. Experimental Results

In the test, we use $R_0 = 20$, $\theta_0 = 10$,$k = 2000$, $score = 0.3$. We choose FVC2002_DB1 as database which give us 16 prints and eight times in different conditions for each print. We take first image as input dataset and 80 images as template dataset; therefore. Among 80 comparisons, 60 of them are matched correctly and 20 of them are decided wrongly; 75% accuracy is obtained. Due to limited time for project, accuracy and running time still need to be improved. For further research, we hope we can improve our approach in following ways: use fingerprint classification to speed up the algorithm; allow no-linear transformation in matching process; compute the local ridge frequency in Gabor filters.

References

[1] M. U. Akram, S. Nasir, A. Tariq, I. Zafar, and W. S. Khan, "Improved fingerprint image segmentation using new modified gradient based technique," in *Electrical and Computer Engineering, 2008. CCECE 2008. Canadian Conference on*. IEEE, 2008, pp. 001 967–001 972.

[2] L. Hong, Y. Wan, and A. Jain, "Fingerprint image enhancement: algorithm and performance evaluation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 8, pp. 777–789, 1998.

[3] R. Thai, "Fingerprint image enhancement and minutiae extraction," *The University of Western Australia*, 2003.

[4] T. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, 1984.

[5] M. Tico and P. Kuosmanen, "An algorithm for fingerprint image postprocessing," in *Signals, Systems and Computers, 2000. Conference Record of the Thirty-Fourth Asilomar Conference on*, vol. 2. IEEE, 2000, pp. 1735–1739.

[6] D. Maltoni, D. Maio, A. Jain, and S. Prabhakar, *Handbook of fingerprint recognition*. Springer Science & Business Media, 2009.

[7] R. S. Germain, A. Califano, S. Colville *et al.*, "Fingerprint matching using transformation parameter clustering," *IEEE Computational Science and Engineering*, vol. 4, no. 4, pp. 42–49, 1997.