PREPROCESSING FOR SKELETON-BASED FINGERPRINT MINUTIAE EXTRACTION

Feng Zhao and Xiaoou Tang

Department of Information Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
{fzhao0, xtang}@ie.cuhk.edu.hk

ABSTRACT

In this paper, we propose to use the fingerprint valley instead of ridge for the binarization-thinning process to extract fingerprint minutiae. We first use several preprocessing steps on the binary image in order to eliminate the spurious lakes and dots, and to reduce the spurious islands, bridges, and spurs in the skeleton image. Finally, by removing all the bug pixels introduced at the thinning stage, our algorithm can detect a maximum number of minutiae from the fingerprint skeleton using the Rutovitz Crossing Number.

1. INTRODUCTION

Fingerprints are graphical flow-like ridges and valleys present on human fingers [1]. They are widely used for personal identification [2]. Various approaches for automatic minutiae extraction have been proposed in the literature. Most of the techniques [3,4,5,6] extract the minutiae from the skeleton of the fingerprint image. The skeleton is computed by thinning the binary image, which is obtained by adaptive thresholding of the gray scale fingerprint image.

There are two types of minutiae, ridge endings and ridge bifurcations. Ridges are generally used for minutiae extraction, since most previous researches assume that the ridges and valleys in the fingerprint have a similar width and are equally spaced. In fact, this may not always be true for various fingerprints collected by different scanners. For example, the fingerprint images we collected using an optical scanner show that the average ridge width (typically 6 pixels) is thicker than the average valley width (typically 3 to 4 pixels), as illustrated in Figure 1. Since a thinner binary image is easier for skeleton computation, we propose to use the valley instead of ridge for minutiae extraction. We use valley endings and valley bifurcations as fingerprint minutiae.

After the valley skeleton is extracted from the binary image, ideally, the width of the skeleton should be strictly one pixel. However, this is not always true, especially at the intersection points, thus producing spurious minutiae points. In this paper, we use a new algorithm to remove such pixels to improve minutiae extraction.

Figure 1. Fingerprint images acquired using the StarTek FM100 sensor (White: valleys; Black: ridges).

2. PREPROCESSING

A critical step in an automatic fingerprint identification system (*AFIS*) is reliably extracting minutiae from the input fingerprint images. It generally consists of the following main steps:

1. Use an adaptive thresholding algorithm to compute the binary image from the input gray scale fingerprint image,
2. Use a thinning algorithm to compute the finger print skeleton from the binary image,
3. Use Rutovitz crossing number to extract minutiae from the skeleton of fingerprint image.
4. Post-processing the minutiae set according to some heuristic rules and the duality property.

In this work, we propose several preprocessing techniques before thinning of the binary image:

1. Use a morphological operator to separate some linked parallel valleys, thus to eliminate spurious bridges and spurs in the skeleton,
2. Fill in the small holes with an area (number of pixels) below a threshold T_{a1}, thus to eliminate the spurious lakes in the skeleton,
3. Remove the dots (isolated pixels) and the islands (short lines) with an area below a threshold T_{a2}, thus to eliminate the spurious lakes, dots, and some islands in the skeleton.

The thresholds should be selected appropriately. If T_{a1} and T_{a2} are too small, the above spurious minutiae in the skeleton will not be eliminated completely. If they are too large, the skeleton will be distorted. In our experiments, we empirically set T_{a1}=11 and T_{a2}=9. Figure 2 shows the effects of the preprocessing steps.

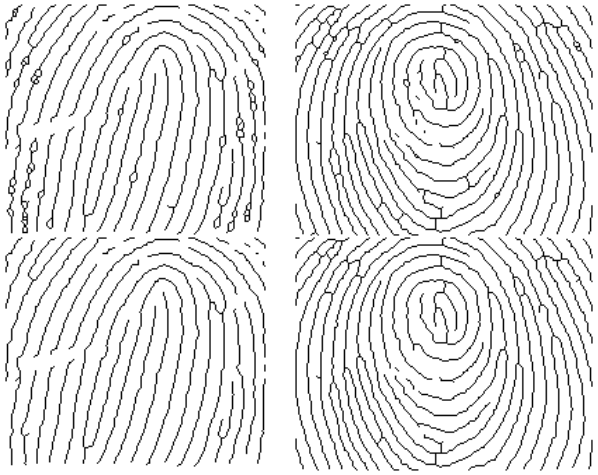

Figure 2. Examples showing the effect of the preprocessing steps. (Upper row: original skeleton images; lower row: skeleton images after preprocessing.)

3. MINUTIAE EXTRACTION

The concept of Crossing Number (*CN*) is widely used for extracting the minutiae [3,4,5]. Rutovitz's definition [7] of crossing number for a pixel *P* is:

P_4	P_3	P_2
P_5	P	P_1
P_6	P_7	P_8

$$CN = \frac{1}{2}\sum_{i=1}^{8}\left|P_i - P_{i+1}\right|$$

where P_i is the binary pixel value in the neighborhood of P with P_i = (0 or1) and P_1=P_9.

The skeleton image of fingerprint is scanned and all the minutiae are detected using the following properties of *CN*:

CN	**Property**
0	Isolated point
1	Ending point
2	Connective point
3	Bifurcation point
4	Crossing point

Ideally, the width of the skeleton should be strictly one pixel. However, this is not always true. Figure 3 shows some examples, where the skeleton has a two-pixel width at some bug pixel locations.

We define a bug pixel as the one with more than two 4-connected neighbors (marked by bold-italic ***1*** and ***0***). These bug pixels exist in the fork region where bifurcations should be detected, but they have *CN* =2 instead of *CN* >2. The existence of bug pixels may (i) destroy the integrity of spurious bridges and spurs, (ii) exchange the type of minutiae points, and (iii) miss detecting of true bifurcations, as illustrated in Figure 4. Therefore, before minutiae extraction, we develop a validation algorithm to eliminate the bug pixels while preserve the skeleton connectivity at the fork regions. By scanning the skeleton of fingerprint image row by row from top-left to bottom-right, we delete the first bug pixel encountered and then check the next bug pixel again for the number of 4-connected neighbors. If the number of 4-connected neighbors after the deletion of previous bug pixel is still larger than two, it will also be deleted; otherwise, it will be preserved and treated as a normal pixel. Some examples are shown in Figure 3. After this validation process, all the pixels in the skeleton satisfy the *CN* properties. Thus we can extract all the minutiae including true minutiae and false minutiae. The false minutiae can be eliminated at the post-processing stage.

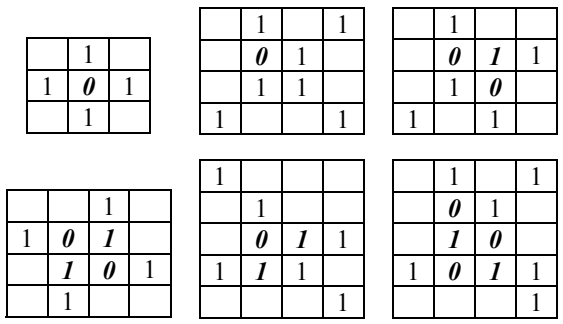

Figure 3. Examples of bug pixels and their validation. (Bold-italic 0: deleted bug pixels, bold-italic 1: preserved bug pixels that are changed to normal pixels.)

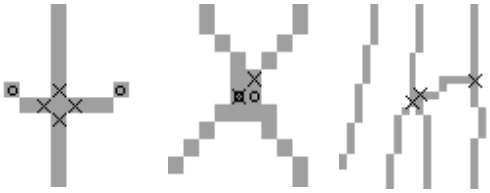

Figure 4. Without validating the bug pixels, we may have: (a) four bifurcations ("x") are missed; (b) two bifurcations are misdetected as two endings ("o"); (c) two bifurcations are missed including one true bifurcation.

4. EXPERIMENTS

To evaluate the performance of our algorithms, we randomly select 35 fingerprint images of medium quality from our fingerprint database. In the experiments, the scanned fingerprint images (256 x 256, 256 gray level, 500 dpi) are cropped into 170 x 180 in size in order to remove the very noisy border areas.

The valley skeleton and ridge skeleton are first obtained from the valley image and its dual ridge image respectively. The valley skeleton agrees rather well with the original valley image, while the ridge skeleton introduces a large number of spurious lakes and bridges. Consequently, the ridge skeleton will produce more spurious minutiae. Figure 5 shows a typical example.

The accuracy rates of applying the minutiae extraction algorithm on ridge skeleton and valley skeleton before and after preprocessing are reported in Table 1 and Table 2, respectively. In the tables, the total rate is calculated using the following formula:

$$Total \quad rate = \frac{E_t + B_t}{E_e + B_e},$$

where E_t and B_t are the number of true endings and true bifurcations, E_e and B_e are the number of extracted endings and bifurcations.

From the results, we can see that after preprocessing the accuracy rate of bifurcation is improved significantly, especially for the ridge skeleton. It demonstrates that the preprocessing algorithm does eliminate a large number of spurious lakes, bridges, spurs, which introduce false bifurcations. However, the accuracy rate of endings is only increased slightly since the preprocessing algorithm only eliminates some spurious islands that introduce false endings. In fact, the spurious dots also introduce false endings and are eliminated efficiently in the preprocessing stage. However, there are only a small number of dots in the skeleton image. The improvement of the accuracy rate of ridge bifurcation is greater than that of valley bifurcation. This shows that the ridge skeleton introduces more spurious minutiae. In addition, the computation speed for valley thinning is much faster than ridge thinning.

Table 3 shows some typical results of validating the bug pixels. From the results, we can see that the bug pixels exist in the fork region where bifurcations should be extracted. Some fingerprint skeletons may have more bug pixels and some may have none.

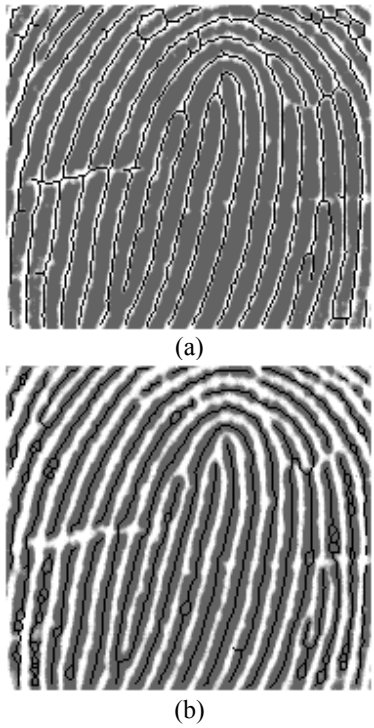

Figure 5. (a) Valley skeleton, (b) ridge skeleton (The skeleton is overlaid on the original gray scale fingerprint image).

Table 1. Accuracy rates for ridge minutiae.

	Before preprocessing	After preprocessing
Ending	10.84 %	10.92 %
Bifurcation	20.24 %	51.32 %
Total rate	13.54 %	17.08 %

Table 2. Accuracy rates for valley minutiae.

	Before preprocessing	After preprocessing
Ending	12.39 %	12.87 %
Bifurcation	16.35 %	26.58 %
Total rate	13.27 %	16.57 %

Table 3. Number of minutiae before and after validating bug pixels.

After preprocessing		After validating bug pixels	
Endings	Bifurcations	Endings	Bifurcations
67	47	67	47
87	27	87	27
55	123	55	125
62	110	62	118
77	55	75	57
106	20	93	27

5. CONCLUSION

In this paper, we develop several simple and efficient preprocessing techniques for minutiae extraction from the valley instead of ridge of fingerprint. Our minutiae extraction algorithm can detect all the minutiae, including both true and false minutiae, using the simple Crossing Number (*CN*) on the skeleton images after validating all the bug pixels introduced at the thinning stage. This allows the true minutiae preserved and false minutiae removed in later post-processing stages.

ACKNOWLEDGMENT

We thank Dr. Qiumei Yang for many constructive comments. The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region. (Project no. CUHK4378/99E).

REFERENCES

[1] A. K. Jain, R. Bolle, and S. Pankanti, *Biometrics: Personal Identification in Networked Society,* Kluwer Academic Publishers, 1999.

[2] H. C. Lee and R. E. Gaensslen, *Advances in Fingerprint Technology,* Boca Raton: CRC Press, 1994.

[3] N. K. Ratha, S. Chen, and A. K. Jain, "Adaptive flow orientation-based feature extraction in fingerprint images," *Pattern Recognition,* vol. 28, no. 11, pp. 1657-1672, 1995.

[4] A. K. Jain, L. Hong and R. Bolle, "On-Line Fingerprint Verification", *IEEE Trans. Pattern Anal. Machine Intell.,* Vol. 19, No. 4, pp. 302-314, 1997.

[5] B. M. Mehtre, "Fingerprint Image Analysis for Automatic Identification," *Machine Vision and Applications,* vol. 6, pp. 124-139, 1993.

[6] A. Farina, Z. M. Kovács-Vajna, and A. Leone, "Fingerprint minutiae extraction from skeletonized binary images," *Pattern Recognition,* vol. 32, no. 5, pp.877-889, 1999.

[7] D. Rutovitz, "Pattern Recognition," *J. Roy. Statist. Soc.,* vol. 129, pp. 504-530, 1966.