3—7

Run Representation Based Minutiae Extraction in Fingerprint Image

Hui-Yeoun Hwang, Jung-Hwan Shin, Sung-Il Chien[1]
School of Electronic and Electrical Engineering
Kyungpook National University

Joon-Jae Lee[2]
School of Internet Engineering
Dongseo University

Abstract

Fingerprint matching depends on the comparison of characteristics of the local ridges and their relationships. Widely used local ridge's characteristics, called minutiae in automatic fingerprint identification systems (AFIS), are ridge termination and bifurcation. Automatic minutiae extraction is an extremely critical process in AFIS. Though several approaches for the minutiae detection have been proposed in literatures, most of these methods are carried out on thinned images. As a result of a thinning process, some ridge smoothing and postprocessing are often needed to eliminate many spurious minutiae occurring due to the presence of undesired spikes and breaks. To overcome this problem we present a novel method of extracting fingerprint minutiae based on the horizontal and vertical run-length encoded from binary images without a thinning process. Once fingerprint images are represented by a cascade of runs, runs' adjacency is then checked. Minutiae can be defined as merging, splitting or ending runs, which are called characteristic runs. Finally, true minutiae are extracted by tracking adjacent runs from each characteristic run. Experimental results show that the proposed method for extracting minutiae is fairly reliable and fast, when compared to a thinning based method.

1 Introduction

A modern society is challenged by the need to identify individuals. Among all the biometrics, fingerprint matching is one of the most popular, mature, and advanced technologies. The uniqueness of an individual fingerprint is exclusively determined by the local ridge characteristics and their relationships. There are various types of local characteristics, called minutiae, in a fingerprint but widely used fingerprint features are restricted to two types of minutiae. The first is a ridge termination defined as the point where a ridge ends abruptly. The second is a bifurcation defined as the point where a ridge merges or splits into branch ridges. AFIS are usually based on these minutiae. Fingerprint matching process starts with image preprocessing and then minutiae are extracted, compared with pre-stored features in order to prove one's correspondence. The performance of AFIS heavily relies on how well these minutiae are extracted. In this paper, we focus our attention on the minutiae extraction algorithm. This paper focuses on a binary image based technique without a thinning process. The main problem in the minutiae extraction method using thinning processes comes from the fact that minutiae in the skeleton image do not always correspond to true minutiae in the fingerprint image. In fact, a lot of spurious minutiae are observed because of undesired spikes, breaks, and holes, while true minutiae are less than 100. Therefore, postprocessing is usually adopted to avoid spurious minutiae, which is based on both statistical and structural information after feature detection [2][3].

Here, we propose a novel method for the fast extraction of fingerprint minutiae that are based on the horizontal and vertical run-length encoding from binary images without a computationally expensive thinning process. Fingerprint images are represented by a cascade of runs after run-length encoding. Then runs' adjacency is checked and characteristic runs are detected. But all characteristic runs cannot be true minutiae. So, some geometric constraints are introduced for checking validity of characteristic runs.

2 Preprocessing

As shown in Figure 1, we first convert a grayscale image into a binary image. Although there are many ways in binarization, we carried out the binarization process using a two-dimensional Gabor filter. Since fingerprints possess locally parallel ridges and valleys with well-defined local spatial-frequency and orientation, noise can be reduced and true ridge and valley structures can be preserved by filtering the fingerprint using Gabor filters. For the Gabor filter, two parameters related to the orientation and the frequency should be determined. Here, we use the least mean square orientation estimation algorithm and the ridge frequency estimation algorithm [5][7] to set the parameters with respect to the orientation and the frequency, respectively.

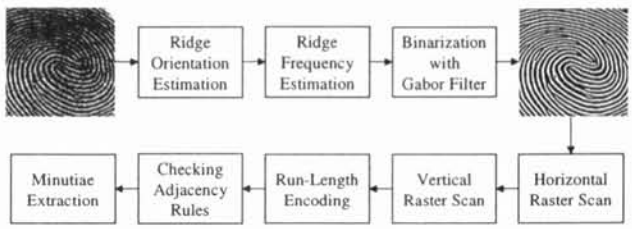


Fig. 1. Block diagram of proposed minutiae extraction algorithm using run-length encoding.

3 Proposed algorithm

We adopt run-length encoding for the binary fingerprint images [4][8]. A binary fingerprint image is completely specified by a linked list of its run. In a fingerprint image, the ridge termination is the starting or ending point of a ridge flow and the ridge bifurcation is the merging point or

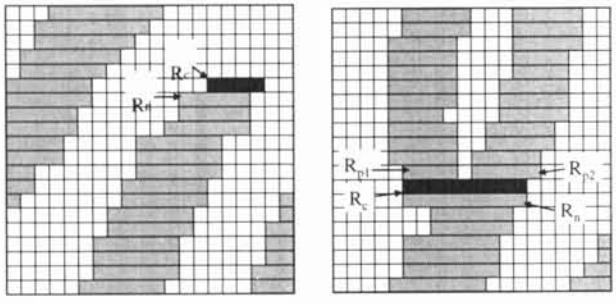
[1] Address: 1370, Sangyeok-dong, Buk-gu, Daegu, 702-701 Korea. E-mail: sichien@ee.knu.ac.kr
[2] Address: San 69-1, Jurye2-Dong, Sasang-Gu, Busan, 617-716, Korea. E-mail: jjlee@dongseo.ac.kr

the splitting point of two ridges. The underlying idea of the proposed method is detecting these points and tracking ridge flows connected to them not in a binary image but in a run-length code. The following steps are performed in our proposed algorithm.

3.1 Run-length encoding

A run-length encoding is an efficient coding scheme for binary or labeled images because it can not only reduce memory space but also speed up image processing time [8]. In the binary image, successive black pixels along the scan line are defined as a run.

Generally, a run-length encoding of a binary image is a list of contiguous horizontal runs of black-pixels. For each run, a location of the starting pixel of a run and either its length or the location of its ending pixel must be recorded. Figure 2 shows runs in a binary fingerprint image.

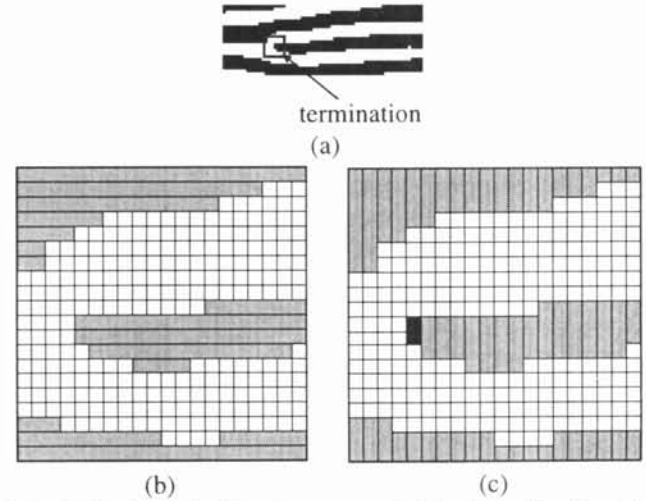


(a) (b)

Fig. 2. The minutiae in run representation. (a) Termination in horizontal runs. (b) Bifurcation in horizontal runs.

By observing carefully, we can find that the ridge is made up of connected runs. From this point of view, the endmost run, the R_c in Fig. 2(a), stands for the ridge termination, while the merging or splitting run, the R_c in Fig. 2(b), signifies the bifurcation. For reliable detection of these runs, we carry out line scans with two orthogonal directions from top to bottom and repeated from left to right. The major reason of using horizontal and vertical line scans is that a minutia where a ridge is running parallel to the direction of line scan cannot be often detected properly as shown in Fig. 3. Figure 3(a) shows a ridge of fingerprint image, Figs. 3(b) and (c) show ridges represented by horizontal and vertical scans, respectively. While scanning a binary fingerprint image in Fig. 3(b) from top to bottom, we find that the length of runs for a horizontal ridge is much longer than the width of ridge. However, in Fig. 3(c) scanned from left to right, the length of vertical runs is close to the width of ridge. So, the minutiae on the horizontal ridge are not represented in the same direction raster scan, while the minutiae are well expressed by using the vertical raster scan. Similarly, horizontal runs generate a vertical ridge in the same manner. For a slanted ridge, both horizontal and vertical runs have little difference in width. In this case, both can well represent the minutiae.

Horizontal and vertical run-length codes are generated along the two directions, which are scanned from top to bottom and from left to right, respectively. Each run in the binary image is encoded by the locations of its starting and ending pixels. In a horizontal line scan, a horizontal run is specified by three parameters, x, y, and z, where x is the number of a row, while y and z denote the position of a starting column and an ending column of the run, respectively with the obvious condition $z \geq y$. Similarly, in a vertical line scan, a vertical run is also specified by x, y, and z, where x is the number of a column, while y and z denote the position of a starting row and an ending row of the run, respectively.



(a)

(b) (c)

Fig. 3. Horizontal ridge is represented by two directional scans. (a) Termination in binary image. (b) Ridge is represented by horizontal run. (c) Ridge is represented by vertical run.

3.2 Checking adjacency rule

After the current run is found, it is checked against the run on the previous scan line (l_p) and the next scan line (l_n) for adjacency, where the adjacency is based on 8-connectedness. Before checking adjacency, we need to elaborate several definitions [4].

Definition 1. Two runs are said to be adjacent if they lie in adjacent scan lines and touch one another in an 8-neighbor sense.

Definition 2. A ridge is composed of a sequence of runs. Ridge = $\{P_1P_2P_3\ldots P_n$, where, P_{i+1} is adjacent to P_i, $1 \leq i \leq n-1\}$. If S is a ridge in binary fingerprint image and P_1, P_n are any pair of runs in S. Then we say that P_1 and P_n are connected in a ridge S and a sequence from P_1 to P_n is a *path*.

Definition 3. A run is *regular* if it has one upper-adjacent run (R_p) and one lower-adjacent run (R_n). However, a run is *singular* if it is not *regular*. A *singular* run is a candidate for minutiae.

Definition 4. A start run is a run with no upper-adjacent run. An end run is a run with no lower-adjacent run. A merging run is a run that has two upper-adjacent runs (R_{p1}, R_{p2}) and one lower-adjacent run (R_n). A splitting run is a run that has one upper-adjacent run (R_p) and two lower-adjacent runs (R_{n1}, R_{n2}).

After we check adjacency for a current run, we can find the following five cases:

Case 1: There are no adjacent runs both on the previous

and the next scan line.
Case 2: There are two adjacent runs on the previous and the next scan line.
Case 3: There is one adjacent run on either the previous or the next scan line.
Case 4: There are two adjacent runs on either the previous or the next scan line.
Case 5: There are more than two adjacent runs on either the previous or the next scan line.

The first case means the run is one pixel spot or an isolated line with more than one pixel width. The second case means the run is part of a ridge flow. The third case means the run is a ridge termination, either the starting or the ending points of ridge flow. The fourth case means two runs on the previous scan line are merging or one run is splitting into two runs on the next scan line. Finally, the fifth case has not been considered in the current experiment because a confluence point which is composed of more than two ridge flows is not a minutiae in AFIS. The runs in both the third and the fourth cases are called characteristic runs, whereas the runs in the second case are called regular runs. Characteristic runs of the third case correspond to candidates for termination minutiae in a fingerprint image and those of the fourth case stand for bifurcation in a fingerprint image.

3.3 Validity of singular runs

Although the characteristic runs are extracted from the run-length codes, we now examine whether the characteristic runs are valid or not. Because very short ridges, generated by the noise in an input image or small protrusions on the contours of the ridge, tend to result in false characteristic runs. These false characteristic runs can be eliminated without removing true characteristic runs by the proposed methods. The validity for eliminating false characteristic runs is examined on three parameters; the first one is a length of each run. If the length of run is longer than the threshold, it is removed. The second one is the gap length between R_{p1} and R_{p2} on upper-branches or R_{n1} and R_{n2} on lower-branches. If the gap is larger than threshold, it is considered as a curved ridge with small curvature. The third one is the path length calculated by descending or ascending along the adjacent runs from the detected current characteristic run R_c as far as a sequence threshold, in other words, the tracking path connected to a characteristic run. If another characteristic run is found within the threshold, R_c is determined to be a false characteristic run. The threshold is determined by twice the average ridge width. The structures of typical false characteristic runs are shown in Fig. 4 and the proposed method successfully removed the false runs as shown in Fig. 5.

3.4 Extracting minutiae

As a termination minutia we calculate a center pixel point of a valid starting or ending run which correspond to case 3. The location of a bifurcation minutiae is determined by a pair (X, Y), where X is the number of a characteristic run's scan line, x and Y is determined by a center of the gap between R_{p1} and R_{p2} or a center of the gap between R_{n1} and R_{n2}.

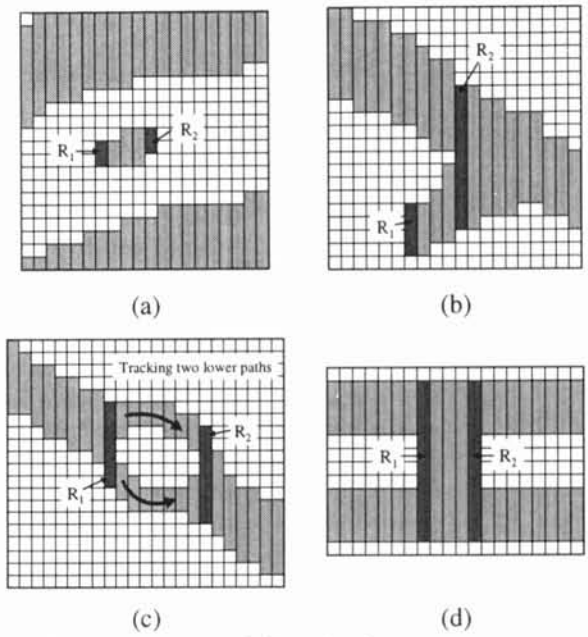


Fig. 4. Examples of false singular run structures.

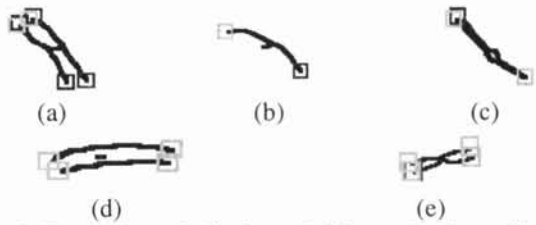


Fig. 5. Examples of eliminated false minutiae with the proposed method (□: detected ridge terminations).

4 Experimental results

The minutiae extraction algorithm described in this paper has been implemented and tested on a sample set, which is composed of 80 fingerprint images from 20 fingers with four fingerprint images for each finger. The fingerprint images are taken through an optical sensor based on the prism. Each gray-level fingerprint image is 256×256 pixels with the resolution of 500 dpi. In the following subsection, we will describe a quantitative method to compare the proposed method (A) with a thinning-based method (B).

4.1 Performance evaluation and Comparison

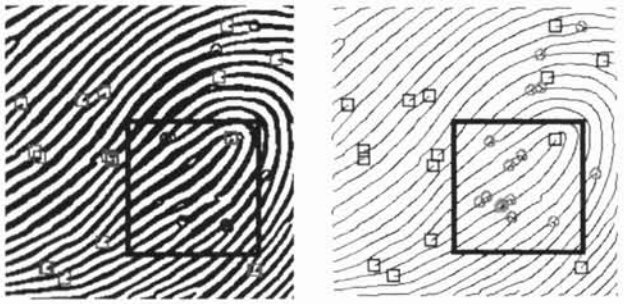
The performance of different methods has been reliably evaluated by comparing the detected minutiae with the set of minutiae obtained by a human expert from the same image. This requires a great deal of time so that we used the small size of sample images in comparison.

In manual minutiae detection, an expert marked the true minutiae ignoring the minutiae located in poor quality regions. In automatic minutiae extraction, two methods A and B have common process until obtaining binary fingerprint images. The method A represents the binary image by run-length codes, while the method B applies the additional thinning process offered in [6] to the binary image. To extract minutiae in the method B, a count of the neighbor pixels' transition at a point of interest in a 3×3 window is used [2]. The count at a point M is defined by

the following equation.

$$C_N(M) = \sum_{k=1}^{8} |R(k+1) - R(k)| \qquad (1)$$

where R(1), R(2), ..., R(8) are the neighboring pixels of M. In particular, the pixel M which has $C_N(M)$=2 corresponds to a termination, while the pixel which has $C_N(M)$=6 corresponds to a bifurcation. Extracted minutiae points using the methods A and B are shown in Figs. 6(a) and (b), respectively. In the same marked region in Figs. 6(a) and (b), the method B produces some false minutiae but the method A successfully removes the false minutiae.



(a) (b)

Fig. 6. Comparison of thinning-based minutiae extraction and proposed method (□: termination, ○: bifurcation).

The following definitions are needed for the purpose of comparing the experimental results.

- True minutia: A minutia point detected by an expert, $\mathbf{m}_t$.
- Paired minutia: If a minutia extracted by the algorithm, $\mathbf{m}_a$, coincide with $\mathbf{m}_t$, it is said to be paired minutiae.
- False minutia: A minutia $\mathbf{m}_a$ which does not coincide with $\mathbf{m}_t$ is said to be a false minutia.
- Dropped minutia: When a minutia $\mathbf{m}_t$ is not detected in $\mathbf{m}_a$, $\mathbf{m}_t$ is said to be a dropped minutia.
- Exchanged minutia: A minutia $\mathbf{m}_a$ which corresponds to $\mathbf{m}_t$ with their types exchanged.

The entire minutiae extraction procedures are executed on a Pentium III-500Mhz. Table 1 reports the performance of the minutiae extraction algorithms. The method A indicates the results of the proposed method and the method B indicates the results of thinning-based method. The table shows the decrease of 26.1% in false minutiae ratio (FMR) while only 3.88% in true minutiae ratio (TMR) when our method is adopted. That means the proposed method eliminates great deal of false minutiae while retaining most of true minutiae. But the method A still detects false termination in broken structures, while the method B shows much higher FMR. Because thinning is very sensitive to noise, the method B produces many false minutiae in hole and short ridge structures after a thinning process. These false minutiae will increase FAR and FRR in a fingerprint matching step. Dropped minutiae ratio (DMR) of the method A is larger than that of the method B. Since the proposed method sometimes missed two adjacent bifurcations in crossover structures and also dropped a bifurcation in delta points where two ridges are nearly orthogonal with each other. Exchanged minutiae ratio (EMR) is mainly due to the preprocessing step that converts a gray image to a binary image. In practice, if a termination is very close to another ridge, the preprocessing algorithm may connect this point to an adjacent ridge and make a false bifurcation. This situation also rises in the thinned image.

In terms of computational complexity, the proposed method is faster than the thinning-based method. The average computational time of the method A is about one third of that of the method B, because the method A does not adopt a computationally expensive thinning process.

Table 1. Comparison of performance and computational time between method A (proposed method) and B (thinning-based method).

Factors	Method A	Method B
FMR (%)	22.50	48.60
TMR (%)	75.32	79.20
DMR (%)	10.18	6.20
EMR (%)	14.50	14.60
Average Computational Time (ms)	35.9	105

5 Conclusions and future work

We have proposed a novel method for reliable and fast feature extraction based on run-length encoding from binary fingerprint images. We detected characteristic runs as features in run-length codes by simple adjacency rules. We checked the validity of characteristic runs directly from run-length codes. The experimental results show that our proposed method is effective in extracting minutiae and has less computational time. We are now in the process of testing the proposed algorithm on the NIST fingerprint database, which contains thousands of fingerprint images.

References

[1] *Biometric Technology Today*, vol. 9, no. 6, pp. 7-8, Elsevier Science, June 2001

[2] Q. Xiao and H. Raafat, "Fingerprint Image Postprocessing: A Combined Statistical and Structural Approach," *Pattern Recognition*, vol. 24, no. 10, pp. 985-992, 1991

[3] S.J. Kim, D.J. Lee, and J.H. Kim, "Algorithm for Detection and Elimination of False Minutiae in Fingerprint Images," *The Third International Conference, AVBPA 2001*, pp. 235-240, 2001

[4] S.D. Zenzo, L. Cinque, and S. Levialdi, "Run-Based Algorithms for Binary Image Analysis and Processing," *IEEE Trans. PAMI*, vol. 18, no. 1, pp. 83-88, January 1996

[5] L. Hong, Y. Wan, and A.K. Jain, "Fingerprint Image Enhancement: Algorithm and Performance Evaluation," *IEEE Trans. PAMI*, vol. 20, no. 8, pp. 777-789, August 1998

[6] T.Y. Zhang and C.Y. Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236-239, March 1984

[7] 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

[8] L.G. Shapiro and G.C. Stockman, *Computer Vision*, pp.62-63, Prentice Hall, 2001