**QUEEN'S
UNIVERSITY
BELFAST**

FPGA-based minutia matching for biometric fingerprint image database retrieval

Jiang, R. M., & Crookes, D. (2008). FPGA-based minutia matching for biometric fingerprint image database retrieval. *Journal of Real-Time Image Processing*, 3(2)(3), 177-182. <https://doi.org/10.1007/s11554-008-0079-8>

Published in:
Journal of Real-Time Image Processing

Queen's University Belfast - Research Portal:
[Link to publication record in Queen's University Belfast Research Portal](#)

General rights

Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.

FPGA-based Minutia Matching for Biometric Fingerprint Image Database Retrieval

Richard M. Jiang, Danny Crookes

Institute of Electronics, Communications and Information Technology
School of Electrical Engineering, Electronics & Computer Science,
Queen's University Belfast, Belfast, UK
Email: d.crookes@qub.ac.uk

Abstract

In this paper, a parallel-matching processor architecture with early-jump-out control is proposed to carry out high-speed biometric fingerprint database retrieval. The processor performs the fingerprint retrieval by using minutia point matching. An early-jump-out(EJO) method is applied to the proposed architecture to speed up the large database retrieval. The processor is implemented on a Xilinx Virtex-E, and occupies 6825 slices and runs at up to 65 MHz. The software/hardware co-simulation benchmark with a database of 10,000 fingerprints verifies that the matching speed can achieve the rate of up to 1.22 million fingerprints per second. EJO results in about a 22% gain in computing efficiency.

Index Terms Biometric Image Retrieval - Fingerprint Matching - Reconfigurable Computing

I. Introduction

Biometric image retrieval [1-5] has become an important research topic due to its wide application in every day life. Fingerprint [1], iris [2] and facial images [3] are frequently-applied biometric identification modes in various security verification systems. Shoeprint images are an emerging additional mode [4]. Subsequently, biometrical image retrieval becomes a key challenge for both research and engineering.

Unlike conventional text-oriented database retrieval that has been addressed extensively, content-based biometric image retrieval [6] usually need to deal with non-textual data. The sensed information is of much higher dimensionality than textual information. Content-based retrieval techniques can be based on many visual cues such as colour, texture, and shape. Ref. [6] proposed the use of colour histograms for indexing into a large database of models. Ref. [7] integrates many cues such as colour, texture, and shape to efficiently retrieve content from the image database. In face image retrieval [4], colour, edge maps and other topological features from faces (e.g. eye-distance) have been extracted as indexing features. As the database size grows, these techniques are not always successful and efficient. Additionally, some image database retrieval requires real-time performance, while image retrieval using low-level content features are usually compute intensive.

Fingerprint matching has become one of the most popular biometric techniques used in automatic person identification [5], such as routine criminal identification for law enforcement agencies, access control for high security installations, credit card usage verification, and employee identification. However, the complexity of fingerprint matching increases with the size of the image database, which can vary from a few hundred records to several million records. Fingerprint verification has been adopted by countries such as US and UK for airport security. This means the fingerprint database can be expanded to cover potentially the whole population, of the order of 100 million fingerprints. High-speed database retrieval from such a large biometric database will be a challenging task that needs both an efficient search algorithm and high-speed hardware implementation.

Although current computers can cope with the computational complexity of automatic fingerprint recognition, there are still many applications for which General Purpose Processors (GPPs) or Digital Signal Processors (DSPs) cannot meet the required cost or performance, such as portable fingerprint verification equipments. With the recent rapid development of FPGA technology [8], FPGAs have become a promising way to tackle these compute-intensive tasks efficiently.

In Ref.[16-20], a number of parallel architectures were presented simply by replicating the compute-intensive matching procedure in parallel to increase the retrieval speed. In this paper, we presented an efficient implementation of fingerprint matching on FPGAs, which takes advantage of a well-known “Early-Jump-Out” mechanism [9] in the searching procedure. The rest of the paper is organized as follows. The high-level features extracted for use in the retrieval stage are described in Section 2. The proposed parallel matching architecture with an early-jump-out mechanism is proposed in Section 3. Section 4 gives the hardware implementation. The experimental results are discussed in Section 5. Section 6 concludes the paper.

II. High-Level Features for Fingerprint Matching

A fingerprint can be classified in general into one of the following classes: arch, tented arch, whorl, left loop, and right loop, as shown in Fig.1. In an automatic fingerprint matching system, the matching pyramid consists of two levels, namely, class-based and minutiae-based. In the class-based step, the query fingerprint is classified into one of the four classes. Different classes can have different search databases. This will reduce the overall search range of the database.

Fig.1 Different Classes of Fingerprint Patterns

Since a captured fingerprint usually appears randomly in an image, preprocessing is needed for adjusting the location of a fingerprint and setting the reference area, which includes the whole area for fingerprint feature extraction. The basic preprocessing steps are shown in Fig.2. In the first step, a reference point of a fingerprint is detected, and the location of the reference point is adjusted to the image centre. The reference point detection algorithm, based on the orientation field of fingerprint images, has been described in Ref.[10-12]. To align two fingerprint images, we must locate a reference point as well as the orientation of each image. The most commonly used reference point is the core point, which is defined as the point at which there is maximum direction change. In step two, the fingerprint is rotated based on the ridges structure around the reference point. The last step sets a reference area after the location of a fingerprint is adjusted.

Fig.2 Fingerprint Registration

In order to facilitate the fingerprint matching procedure, high-level structural features need to be extracted from the fingerprint image for the purpose of representation and matching. A closer analysis of the fingerprint has revealed [13-15] that the ridges or the valleys exhibit anomalies of various kinds, such as ridge bifurcations, ridge endings, short ridges, and ridge crossovers. Collectively, these features are called minutiae. For automatic feature extraction and matching, the set of fingerprint features can be simplified by exploiting two types of minutiae: ridge endings and ridge bifurcations. Fig.3 is the extracted minutia point features of the fingerprint in Fig.2.

Fig.3 Fingerprint Minutiae Extraction

Fig.4 Description of Minutia Features.

Fig.5 A Fingerprint Featured by a set of Minutiae.

A good quality, rolled fingerprint image can have about 70 to 80 minutiae points. In a latent, or partial fingerprint, the number of minutiae is much less. The structural features which are commonly extracted from the gray level input fingerprint image are the ridge bifurcations and ridge endings. Each of the two types of minutiae has three attributes, namely, the x-coordinate, the y-coordinate, and the local ridge direction β as shown in Fig.4. Therefore, a fingerprint can be simple presented by the collection of its unique minutia distribution, as shown in Fig.5. Given the minutiae representation of fingerprints, matching a fingerprint against a database reduces to the problem of point matching of these minutia features.

III. Proposed Fingerprint Matching with Early-Jump-Out

A typical minutiae-based fingerprint verification process works in two phases: fingerprint enrolment phase and fingerprint matching phase, as shown in Fig.6. In the fingerprint enrolment phase, a sensor captures the fingerprint image from which the minutiae are extracted, processed, and stored as a ‘master template’ in the fingerprint database. In the fingerprint verification phase, the above process repeats, resulting in the generation of a ‘live template.’ The two templates are matched to determine a similarity score of the two fingerprints.

Fig.6 Overview of Fingerprint Matching Process

The following notation is used in the sequential and parallel matching algorithms described below. Let the query fingerprint be represented as a set of q minutiae points: $Q = \{f_0, f_1, \dots, f_{q-1}\}$. Note that each of the q elements is a feature vector containing three components,

$$f_i = (fx_i, fy_i, f\beta_i)$$

The components of a feature vector are shown geometrically in Fig.4. Similarly, let the k -th reference (database) fingerprint be represented as a set of p minutiae points $D_k = \{d_0^k, d_1^k, \dots, d_{p-1}^k\}$, $d_j^k = (dx_j, dy_j, d\beta_j)$.

The matching algorithm is based on finding the number of paired minutiae between each database fingerprint and the query fingerprint. A tolerance range $t_i = (tx_i, ty_i, t\beta_i)$ should be generated for each query feature f_i as the matching criteria. The size of the tolerance range depends on the ridge widths and distance from the core point in the fingerprint. The query feature f_i can be considered as matching with the reference feature D_k only when they satisfy the following conditions:

$$\begin{aligned} |fx_i - dx_j^k| &< tx_i, \\ |fy_i - dy_j^k| &< ty_i, \\ |f\beta_i - d\beta_j^k| &< t\beta_i. \end{aligned}$$

This distance is usually called Manhattan or cityblock distance. Once the count of matching minutiae is obtained, a matching score is computed. The matching score is used for deciding the degree of match. Finally, a set of the top 10 scoring reference fingerprints is obtained as a result of matching. Note that the query fingerprint Q may or may not belong to the fingerprint database $D \supset \{D_k\}$.

In this paper, we propose a parallel matching procedure with an early-jump-out mechanism, which can be listed as follows:

Proposed Parallel-Matching Algorithm with Early Jump Out control

Input:

Query fingerprint Q defined by a set of q minutiae points:

$$Q = \{f_0, f_1, \dots, f_{q-1}\}, f_i = (fx_i, fy_i, f\beta_i)$$

The matching criterion is defined by a tolerance tensor:

$$T = \{t_0, t_1, \dots, t_{q-1}\}, t_i = (tx_i, ty_i, t\beta_i)$$

Enrolled fingerprint D_k in Database $D = \{D_0, D_1, \dots, D_{N-1}\}$, which are featured by p minutiae points:

$$D_k = \{d_0^k, d_1^k, \dots, d_{p-1}^k\}, d_j^k = (dx_j, dy_j, d\beta_j)$$

The size of fingerprint database: N

Output

A list of top 10 records from the database with highest matching scores: $mscore[1:10]$;

Begin

```

search()
{
    for (k = 0; k < n; k++) // done in series
    {
        missCount = 0;
        EJO_Control = 1;
        j = 0;
        while (j < p AND EJO_Control = 1) do // j'th feature of database item - (done in series)
        {
            // match feature  $d_j^k$  of  $D_k$  against ALL query features  $f_i$  in parallel:
            for (i = 0; i < q; i++)
            {
                matchFound = false;
                if (distance between  $f_i$  and  $d_j^k$  is within threshold)
                    matchFound = true;
            }
            if (!matchFound)
            {
                missCount++;
                if missCount >= missScore[10]
                    EJO_control = 0;
            }
            j++;
        }
        // No of matches is  $q$  - missCount
        if (missCount < missScore[10])
            insert (missCount, k) into missCount[], maintaining sorted order;
    }
}

```

In the above procedure, the early-jump-out mechanism can make an early decision to reject the reference fingerprint at an early stage, and the further computation of feature compares is saved when the unmatched features fall out of the stored top 10 most fit fingerprint candidates. This procedure can be depicted as in Fig.7.

Fig.7. Early-Jump-Out Process Flow for Fingerprint Matching

IV. Hardware Implementation

With recent advances in embedded system development, fingerprint matching can migrate to the mobile world, supporting applications like standalone indoor access systems, PDAs/notebooks, or even cell phone login systems. Such embedded devices are usually limited by its power and computation speed. The above parallel-matching early-jump-out method can fit these applications well by speeding up the matching while also reducing the computing power.

Hardware based fingerprint matching usually adopts a fixed number of minutia features so that the parallel processing architecture can be predefined. The number of features may vary. As in many other papers, the selection of features is usually the 64 minutia points most central to the reference point, as shown in fig.5. In this paper, we standardize the high-level features of a fingerprint as 64 minutiae around the reference point. If the fingerprint is incomplete and has less than 64 minutia, the remaining points are filled by ‘-1’ to avoid matching. Therefore, we have 64 processing elements (PEs) in the processor.

Fig.8 is the basic architecture that implements the proposed EJO algorithm. The standardized 64 features of the query fingerprint are deployed into 64 processing elements in the processor initialization. Then the next reference

fingerprint from the database is loaded. Each reference fingerprint has a standardized size of 64 features. The control unit broadcasts the 64 features of the reference fingerprint one by one to all 64 processing elements. Each PE performs a compare operation and can vote through an OR logic bus to confirm if the feature is matched or not. If the feature is matched, the score accumulator increases by one, and the EJO unit makes the decision whether or not to jump out of this reference fingerprint. Once the EJO unit emits a ‘jump’ command to the address unit, the address unit will skip over the remaining minutia feature points, and move directly to the next reference fingerprint.

The top- k searching is widely used for image retrieval because the matching is usually based on a similarity measure. The number k of top-matched fingerprints can be selected from 5 to 20. The number k has little influence on the EJO’s performance, because the fingerprint database size (usually $> 10,000$) is much larger than k . In the following experimental benchmark, k is set to 10.

Fig.8. Co-Processor Architecture

V. Results and Discussion

The above fingerprint matching co-processor was implemented in VHDL and synthesized for Xilinx Virtex-E FPGAs. Table 1 is the synthesized result. The processor has 64 processing elements, and each processing element has 85 slices. The whole processor occupies 6825 slices, and runs at up to 65 MHz.

A software/hardware co-simulation test bench is set up. Each fingerprint has 64 minutia features. Similar to Ref.[16], 10,000 fingerprint samples were generated by random scattering from 20 real fingerprint data for the coprocessor benchmark.

The matching processor was tested with the fingerprint database generated by random scattering from 10 fingerprint samples. The coprocessor without EJO control can process 1 million fingerprints per second. With EJO control, the processor can process 1.22 million fingerprints per second. On average, each matching with a reference fingerprint needs 64 feature-comparing operations without EJO control, which was reduced on average to 52 feature compare operations with EJO control, as shown in Table 2. In total, with the proposed Early-Jump-Out mechanism, the performance can obtain a 22% improvement. Although 22% improvement may be considered not very impressive, it should still be valuable, especially for large fingerprint database retrieval, such as millions of fingerprints.

Table 1. Implementation of the Co-Processor

Area per PE	85 slices
Total Area	6825 slices
Speed	65 MHz

Table 2. Performance Improvement with EJO

	FPs/s	Compares/FP
Without EJO[16]	1M	64
With EJO	1.22M	52
Improvement	22%	18.7%

Table 3. Comparison with PC-based Software Performance (Search in 1 Million Fingerprints)

	Speed
Intel Celeron 2.8GHz	38,165 ms
FPGA Virtex-E, 65 MHz	820 ms
Improvement	~ 45 times

In fingerprint retrieval, we keep the top 10 most fit fingerprints as the retrieval results. This number can also be selected by users. After the user gets the top- k best matching results, the user can make a further judgement by themselves. This kind of top- k searching method is a basic strategy for image retrieval, because there is no guarantee that the top one is the exact candidate that the user really wants, due to noise and many other factors, such as deviations in reference point location.

The performance is also compared to a PC-based software solution. The PC has a 2.8GHz Pentium Celeron processor. The software was written in C++ and compiled by Microsoft Visual Studio. The software solution spent 38,165 ms to finish a query search of a 1 million fingerprint database, while the FPGA-based coprocessor with EJO control can finish the retrieval in 820 ms, as shown in Table.3. The retrieval speed was easily improved almost 45 times by the FPGA solution.

VI. Conclusion

In conclusion, a compute-efficient architecture is proposed for performing high-speed fingerprint matching from a large-scale fingerprint database. The processor architecture performs the fingerprint matching by using minutia point match. An early-jump-out control is added to the matching procedure, which can improve the processing speed while reducing the computing operations. The processor was implemented on a Xilinx Virtex-E. It occupies 6825 slices and runs at up to 65 MHz. The software/hardware co-simulation benchmark with 64 features per fingerprint and 10,000 fingerprints in the database verifies that the matching speed can achieve the rate of one million fingerprints per second without EJO control, and 1.22 million fingerprints with the proposed EJO control.

References

- [1] L. Hong, A. Jain, "Integrating faces and fingerprints for personal identification", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, Issue 12, Dec. 1998, pp.1295.
- [2] L. Ma, T. Tan, Y. Wang, D. Zhang, "Personal identification based on iris texture analysis", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.25, Issue 12, Dec. 2003, pp.1519.
- [3] C. Chen, S.-P. Chiang, "Detection of human faces in colour images", IEE Proceedings of Vision, Image and Signal Processing, Vol. 144, Issue 6, Dec. 1997, pp.384.
- [4] M. Gueham, A. Bouridane, D. Crookes, "Automatic Recognition of Partial Shoeprints Based on Phase-Only Correlation", 2007 IEEE International Conference on Image Processing, Vol. 4, Sept.16, 2007, pp.438-441.
- [5] B. Miller, "Vital Signs of Identity," IEEE Spectrum, vol. 31, no. 2, pp. 22-30, Feb. 1994, San Jose, Calif., 1994.
- [6] L. Kotoulas, I. Andreadis, "Colour histogram content-based image retrieval and hardware implementation", IEE Proceedings Circuits, Devices and Systems, Vol. 150, Issue 5, 6 Oct. 2003, pp. 387.
- [7] Yong Man Ro, Ho Kyung Kang, "Hierarchical rotational invariant similarity measurement for MPEG-7 homogeneous texture descriptor", Electronics Letters, Vol. 36, Issue 15, 20 July 2000 pp.1268.
- [8] D. Crookes, "Architectures for high performance image processing: The future", J. of Systems Architecture, 45 (10), Apr. 1999, pp.739.
- [9] M. Jiang, D. Crookes, S. Davidson, R. Turner, "A Low-Power Systolic Array Processor Architecture for FSBM Video Motion Estimation", Electronics Letters, 2006, Vol. 42, No. 20, pp.1146.
- [10] W. M. Koo and A. Kot, "Curvature-based singular points detection", in Audio-and Video-Based Biometric Person Authentication, Sweden, 2001, pp.229.
- [11] X. Luo, J. Tian, and Y. Wu, "A minutiae matching algorithm in fingerprint verification," in Proc. 15th Int. Conf. Pattern Recognition, vol. 4, 2000, pp.833.
- [12] D. Maio and D. Maltoni, "Direct gray-scale minutiae detection in fingerprints," IEEE Trans. Pattern Anal. Machine Intell., Vol. 19, Jan. 1997, pp.27.
- [13] C. V. K. Rao and K. Black, "Type Classification of Fingerprints", IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 2, no. 223-231, pp.31,1980.
- [14] R. Cappelli, A. Lumini, D. Maio, and D. Maltoni, "Fingerprint classification by directional image partitioning," IEEE Trans. Pattern Anal. Machine Intell., vol. 21, pp.402, May 1999.
- [15] K. Rerkrai and V. Areekul, "A new reference point for fingerprint recognition", in Proc. Int. Conf. Image Processing, Vol. 2, 2000, pp.499.
- [16] K. R. Nalini, S. Y. Chen, A. K. Jain, "A Real-Time Matching System for Large Fingerprint Databases", IEEE Trans. Pattern Analysis & Machine Intelligence, Vol.18, No.8, Aug. 1996, pp. 799.
- [17] M. Fons, F. Fons, E. Canto, "Design of FPGA-based Hardware Accelerators for On-line Fingerprint Matcher Systems", PhD Research in Microelectronics and Electronics 2006, 0-0 0, pp.333.
- [18] A. Lindoso, L. Entrena, J. Izquierdo, "FPGA-Based Acceleration of Fingerprint Minutiae Matching", 2007 3rd Southern Conference on Programmable Logic, 28-26 Feb., 2007, pp.81.
- [19] M. Fons, F. Fons, E. Canto, "Hardware-Software Co-design of a Fingerprint Matcher on Card", 2006 IEEE International Conference on Electro/information Technology, 7-10 May 2006, pp.113.
- [20] A. Lindoso, L. Entrena, J. Izquierdo, "Correlation-based fingerprint matching using FPGAs", 2005 IEEE International Conference on Field-Programmable Technology, 11-14 Dec. 2005, pp.87.

Richard M. Jiang graduated with a BEng degree in Electronics from Huazhong University of Science & Technology, China. He completed his PhD at Queen's University Belfast in 2008. He is currently investigating high performance image and vision processing.

Danny Crookes graduated with BSc in Mathematics and Computer Science in 1977, and PhD in Computer Science in 1980, both from Queen's University Belfast. He was appointed Professor of Computer Engineering at Queen's University Belfast in 1993. His research interests centre round software tools for high performance computing, with particular emphasis on image processing.

a) Arch b) Whorl c) Left Loop

Fig.1 Different Classes of Fingerprint Patterns

Fig.2 Fingerprint Registration

Fig.3 Fingerprint Minutiae Extraction

Fig.4 Description of Minutia Features.

Fig.5 A Fingerprint Featured by a set of Minutiae.

Fig.6 Overview of Fingerprint Matching Process

Fig.7. Early-Jump-Out Process Flow for Fingerprint Matching

a) Proposed Co-Processor Architecture

b) Proposed Early-Jump-Out Control Unit

Fig.8. Co-Processor Architecture

Table 1. Implementation of the Co-Processor

Area per PE	85 slices
Total Area	6825 slices
Speed	65 MHz

Table 2. Performance Improvement with EJO

	FPs/s	Compares/FP
Without EJO[16]	1M	64
With EJO	1.22M	52
Improvement	22%	18.7%

Table 3. Comparison with PC-based Software Performance (Search in 1 Million Fingerprints)

	Speed
Intel Celeron 2.8GHz	38,165 ms
FPGA Virtex-E, 65 MHz	820 ms
Improvement	~ 45 times