MVA2005 IAPR Conference on Machine VIsion Applications, May 16-18, 2005 Tsukuba Science City, Japan
13-29

Efficient Minutiae-Based Fingerprint Matching

G. Eckert, S. Müller, T. Wiebesiek
Institut für Theoretische Nachrichtentechnik
und Informationsverarbeitung
University of Hannover, Germany
{eckert,mueller,wiebesiek}@tnt.uni-hannover.de

Abstract

We propose a new and efficient method for minutiae-based fingerprint matching, that is invariant to translation, rotation and distortion effects of fingerprint patterns. The algorithm is separated from a prior feature extraction and uses a compact description of minutiae features in fingerprints. The matching process consists of three major steps: the first step is finding pairs of possibly corresponding minutiae in both fingerprint patterns, the second step is combining these pairs to valid tuples of four minutiae each, containing two minutiae from each pattern. The third step is the matching itself. It is realised by a monotonous tree search that finds consistent combinations of tuples with a maximum number of different minutiae pairs. The approach has low and scalable memory requirements and it is computationally inexpensive.

1. Introduction

Automatic and secure person identification is an essential means to meet today's increasing need of security. Use of biometric features has many advantages over conventional personal identification and verification methods. Biometric features are time-invariant throughout the whole life, easy to record, not transmittable, unique and always available for the owner. The probability of finding two persons with the same fingerprint is 1 in 1 million. Fingerprint verification has been generally associated with police work, but of late, fingerprint verification is applicable in several public uses, for example in access control systems. Manual fingerprint verification, used for police work, has some disadvantages like high costs and time consumption. The goal is to replace the manual verification process by an automatic system.
Different fingerprint matching procedures and approaches have been developed. Methods can be differentiated into image based and feature based methods, or combinations of both.

In [3] a method for feature based fingerprint matching is given. It is based on triangular matching in order to cope with distortion in different fingerprint images. In [5] two methods to reduce the computational complexity of finding an affine transformation, that transforms the test fingerprint pattern into the reference pattern are given. Most publications combine extracted features and directional image data e.g. [6]. An approach with exclusive use of minutiae features is presented in [1]. Here triplets of minutiae are used to generate an index used for decision. In [2] a combination of intensity based and feature based matching algorithms is presented. The translation between two fingerprint patterns is calculated by corresponding minutiae points. A rotation is calculated by matching the pattern ridges. An affine transformation is applied to the test pattern and the overlapping minutiae points are counted.

The structure of an automatic feature based fingerprint verification system consists of three main parts: fingerprint aquisition and image preprocessing, feature extraction and classification of test fingerprints.

The method presented in this paper deals only with the last step in a fingerprint verification system, the classification of the fingerprints using minutiae features. The fingerprint is represented by extracted minutiae features, so fingerprint images are not necessary for the matching process. Minutiae-based matching of fingerprints is a special case of "point-set matching", with minutiae of the fingerprints representing the points.

Our goal was to develop an approach that is well suited for application in embedded systems with limitations in both, computational power and available memory resources. The matching approach uses a list of tuples of four minutiae, containing two minutiae of each pattern. Every tuple is rated with a cost function and the tuple list is sorted to achieve an efficient search. The matching is realised by a monotonous tree search, that finds valid combinations of tuples with the maximum number of different corresponding minutiae pairs.

2. Fingerprint features

The features used for matching are extracted from fingerprint images. They can be divided into global features, minutiae features and interminutiae features. Global features are core and delta points. Their position is based on global ridge directions in a fingerprint. Core points are located where ridges of the fingerprints form a loop, delta points are found, where ridges enclose an imaginary triangle.
Most important fingerprint features are minutiae points and their properties. Minutiae points are ridge endings or ridge bifurcations. Minutiae properties are the position of a minutia point, the direction of a minutia (tangent direction at the appropriate ridge) and the type of a minutia, either ridge ending or ridge bifurcation. Interminutiae features are the Euclidean distances between the minutiae and the number of ridges between two minutiae points. The ridges between every two minutiae are directly counted in the fingerprint images.

3. Matching approach

Minutiae-based fingerprint matching is based on corresponding minutiae in two fingerprint templates. The presented algorithm consists of three consecutive steps. During the first step it determines possible pairs of corresponding minutiae in a reference fingerprint template (with N_{ref} minutiae) and in a test fingerprint (with N_{test} minutiae). During the second step these pairs of minutiae are combined to tuples of two pairs of minutiae each. From such a tuple it is possible to determine a unique 2d transformation (translation, rotation, scaling) that maps the minutiae of the test fingerprint onto the minutiae of the reference fingerprint. The transformation parameters are stored as tuple attribute. The fingerprint images are assumed to be roughly aligned, i.e. they do not heavyly differ in size, location and orientation. Therefore, a cost function is introduced that assigns individual costs to each tuple considering deviations from maximum transformation parameters. All tuples found are ordered by their costs. During the last step an as large as possible set of consistent tuples is determined. The more consistent minutiae pairs are found, the better the two fingerprint templates match each other. Following subsections describe the matching approach in detail.

3.1 Possible corresponding minutiae pairs

The first matching step is grouping the minutiae to pairs. Every minutia of the test fingerprint is compared with every minutia of the reference fingerprint. A valid pair of minutiae has to fulfil the following conditions:

- Minutiae types must be identical
- $translation \leq translation_{max}$
- Differences in minutiae direction $\leq rot_{max} + \Delta dir_{max}$

$translation$ is the Euclidian distance between both minutiae. $translation_{max}$ is the maximum translation between two fingerprint images, rot_{max} is the maximum rotation between two fingerprint images and Δdir_{max} is the maximum difference in two minutiae directions (with respect to the rotation). Cores and deltas are treated as special types of minutiae that do not have a direction. Pairs of two minutiae that fulfil the conditions above are possible correspondencies between minutiae of test and reference fingerprint. Minutiae of both fingerprints can be used for several pairs. During the following steps, pairs are either verified or rejected.

3.2 Combining minutiae pairs to tuples

Now the minutiae pairs are combined to tuples of two minutiae pairs each. Even though a minutia may appear in more than one minutiae pair, it is not allowed to occur more than once in a minutiae tuple. Each minutiae tuple has four different minutiae, that allow to determine a unique 2d transformation that maps the minutiae of the test fingerprint onto the minutiae of the reference fingerprint. The transformation is decomposed into translation, rotation and scaling. The $translation$ moves the midpoint of the considered minutiae from the test fingerprint onto the midpoint of the respective minutiae from the reference fingerprint. A rotation about an angle rot and a scaling by a factor $scaling$ maps the test fingerprint minutiae onto the reference fingerprint minutiae. The direction properties of the test minutiae are rotated likewise. A tuple is valid if it fulfils the following conditions:

- $translation \leq translation_{max}$
- $rot \leq rot_{max}$
- $|1 - scaling| \leq scaling_{max}$
- Difference in numbers of ridges in each template $\leq \Delta ridge_{max}$
- Difference in directions of rotated minutiae pairs $\leq \Delta dir_{max}$

$scaling_{max}$ is a maximum scaling threshold, $\Delta ridge_{max}$ is the maximum difference in numbers of ridges between two corresponding minutiae pairs. The transformation parameters are attributes assigned to each tuple.

3.3 Cost function

The valid minutiae tuples, described in the previous section, are now rated by a cost function C. It is based on the assumption, that the fingerprint images are roughly aligned,

i.e. they do not heavyly differ in size, location and orientation, and it is derived from the tuple conditions of section 3.2. It is a weighted sum of the relative deviations of the calculated transformation parameters from their respective maximum parameters:

$$C = \alpha \cdot \frac{translation}{translation_{max}} + \beta \cdot \frac{rot}{rot_{max}} + \\ + \gamma \cdot \frac{(1 - scaling)}{scaling_{max}} + \delta \cdot \frac{\Delta dir}{\Delta dir_{max}} \quad (1)$$

The weights (α, $\beta \ll \gamma$, δ) control the influence of the individual parameters. The maximum parameters are constraints derived for example from the fingerprint acquisition system. The tuples are ordered by their costs and are stored in an ordered list.

3.4 Tree search

During the last step of the matching process, an as large as possible set of consistent tuples is determined. Consistency means, that minutiae correspondencies within the set do not contradict, and that their differences of respective transformation parameters do not exceed given bounds. The larger the number of different minutiae pairs in this set is, the larger is the support for the corresponding 2d transformation.

The search for the largest set of consistent tuples is realised by means of search tree. Nodes of the tree are the tuples from the sorted list of valid minutiae tuples. To illustrate the search, the tuples are assigned uppercase letters: for example A_1 represents the first tuple of the tuple list. The tuples' indices represent their individual costs.

The construction of the tree starts with an empty root node and the tuple with the least costs is inserted as new node at depth 1. In figure 1 an example for a tree is given. Every node contains one tuple with its costs and accumulated costs, i.e. the sum of costs on the path from root to the node. Search starts with the tupel with least costs (here tuple A in node number 1). The tree is now extended at the leaf with the least accumulated costs. For every extension step, two positions are possible:

- a new node at depth of currently developed leaf, and
- a new node at depth+1 of currently developed leaf.

For every tree extension, four combinations of new nodes are possible:

- two nodes of the same tuple,
- two nodes of different tuples,
- only one new node at same depth of developed node,
- no new node at all.

Every new node has to be checked for

- minutiae consistency to all predecessor nodes, and
- tuple consistency with respect to transformation parameters of all predecessor nodes.

The tuple of a new node has to be minutiae consistent, i.e. its new minutiae correspondencies must not contradict minutiae correspondencies already contained in the current branch. A tuple with a correspondence between minutia a of the test fingerprint and minutia b of the reference fingerprint is minutiae inconsistent e.g. to a tuple with a correspondence of the same minutia a and a different minutia c of the reference fingerprint. The mutual minutiae consistency of tuples is illustrated in the consistency table in figure 1 (left).

Additionally, the tuple has to be transformation consistent with the tuples of all predecessor node, i.e. the differences between respective transformation parameters within one branch must not exceed given thresholds.

The tree can only be extended with tuples, that have higher costs than the tuple of the currently developed leaf. If one of the above checks fails, the next tuple from the list of tuples is taken into account. Thus, the consistent tuple with least costs is inserted as new node.

In the example, node 1 can be extended with tuples C to E. Tuples A and B are not consistent, so tuple C is attached to the tree at position of depth+1 as new node 2. At the same depth of node 1, tuple B is inserted into the tree as node 3, because at this position no consistency to node 1 is necessary. The leaf with least costs is now node 3. Tuple B and C are consistent, so two new nodes with tuple C are introduced as node 4 and 5. Now, node 5 is the leaf with least accumulated costs, tuple C and D are inconsistent, so the new node 6 contains tuple E. Additionally, node 7 with tuple D is attached to the tree. The construction of the tree continues in the described way. The result is the completely developed tree as depicted in figure 1.

The branch containing the largest set of different minutiae pairs represents the 2d transformation that is supported by the most minutiae pairs. The ratio $r_{matching}$ of matched minutiae pairs $N_{matching}$ in this branch and the minimum number of minutiae in test and reference pattern is

$$r_{matching} = \frac{N_{matching}}{\min(N_{ref}, N_{test})} \quad (2)$$

The actual fingerprint verification is performed by simply applying a threshold to $r_{matching}$. Test fingerprints with a matching ratio larger than the threshold are accepted, others are rejected.

The memory requirement of the algorithm may be controlled by limiting the length of the list of tuples and by limiting the maximum number of nodes in the search tree.

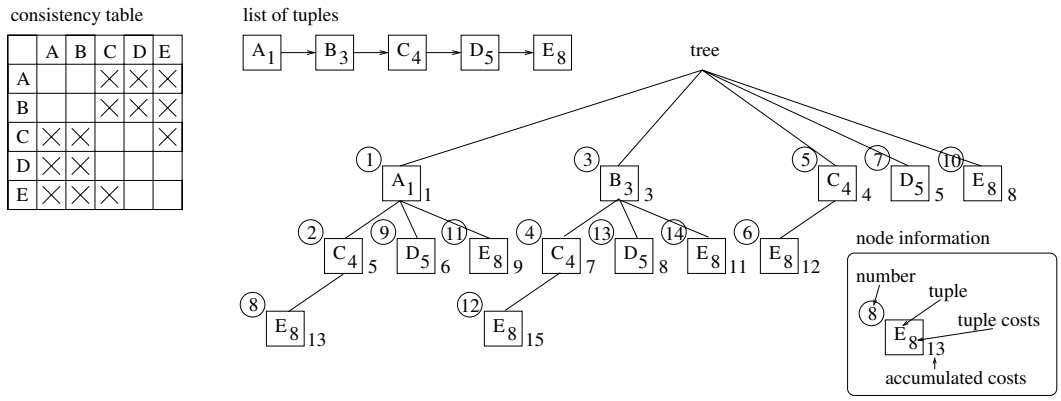

Figure 1. Consistent tuple combinations, list of tuples, search tree

4. Experimental results

The presented approach has been tested with fingerprints from the "Bologna FVC 2000 Database 2" [4]. This database contains images of 110 different fingers, 8 different samples of each finger. The images have been captured by a low-cost capacitive sensor with a resolution of 500dpi. The FVC deals with the complete processing chain including feature extraction. Our approach only deals with the minutiae matching (classification), not with the feature extraction. In order to show the applicability of our matching approach, we used fingerprint features, that had been extracted automatically by a third party.

The presented approach achieves an equal error rate of 4.4% on the complete FVC 2000 Database 2. Its major advantages are its efficiency and its low memory requirements. The average matching time is 47 ms (without special code optimisation) on an AMD Athlon 1,2 GHz, that is approximately equivalent to a matching time of 0,14 s on a Pentium III 450 Mhz (as used in the FVC 2000 and compared with the best result of 0,17 s). Our result was achieved with memory requirements of only 32 kB of working memory (dynamic data including fingerprint features). The availability of more memory would not have resulted in a better matching. An additional property of the approach is the scalability of memory consumption that yields to only a graceful degradation of the results without losing its principal functionality when limitating the available working memory. Therefore, our presented approach is well suited for application in embedded systems with limitations in both, computational power and available memory resources.

5 Conclusions

We presented an efficient approach for minutiae-based fingerprint matching using three major steps: minutiae pairing, tuple formation and tree search based matching. All available fingerprint features are evaluated during these processing steps. A cost function rating minutiae tuples allows an efficient search for the best matching result. Due to its low computational complexity, its low memory requirements, and its good scalability in terms of memory requirements, the approach is well suited for application in embedded systems with limited computational power and memory resources. The approach can easily be extended to use additional point features or attributes.

References

[1] R.S. Germain, A. Califano, S. Colville, "Fingerprint matching using transformation parameter clustering", IEEE Computational Science and Engineering, 4(4), Oct-Dec, 1997, pp. 42-49., 1997.

[2] A. K. Jain, L. Hong, and R. Bolle, "On-line Fingerprint Verification", IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(4):302-314, 1997.

[3] Zsolt Mikls Kovcs-Vajna, "A Fingerprint Verification System Based on Triangular Matching and Dynamic Time Warping", IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(11):1266-1276, 2000.

[4] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, and A. K. Jain, "FVC2000: Fingerprint Verification Competition", 15th IAPR International Conference on Pattern Recognition, Barcelona, Spain, Sep. 3-7, 2000.

[5] D. M. Mount, N. S. Netanyahu and Jacqueline L. Moigne, "Efficient Algorithms for Robust Feature Matching", Pattern Recognition, 32(1):17-38, June, 1998.

[6] N. K. Ratha, K. Karu, S. Chen, and A. K. Jain, "A Real-time Matching System for Large Fingerprint Databases", IEEE Transactions on Pattern Analysis and Machine Intelligence, 18(8):799-813, 1996.