Hierarchically linked extended features for fingerprint processing

Dissertation

zur Erlangung des Doktorgrades
der Mathematisch-Naturwissenschaftlichen Fakultäten
der Georg-August-Universität zu Göttingen

vorgelegt von

Krzysztof Mieloch
aus Wrocław, Polen

Göttingen 2008

D7

Referent:	Prof. Dr. Axel Munk
Koreferent:	Prof. Dr. Preda Mihăilescu
Tag der mündlichen Prüfung:	8. Mai 2008

Contents

Abstract

This thesis discusses a novel approach for fingerprint feature extraction. A new fingerprint structure has been proposed as a basis for the extraction of extended features. These new features provide us with a link between the global and local zoom-levels generally utilised separately, which can contribute to bridging the gap between the automatic procedure for fingerprint recognition and that of a human expert. Furthermore, a novel development process based on interactive testing has been proposed. The results of an application of the new features in matching and classification confirm the goodness of the features.

Chapter 1

Introduction

1.1. Biometric authentication

Nowadays, the identification of a person is a fundamental task for many applications such as access control, authorised users authentication, or credit card transactions. A person may be identified based on her possessions, e.g. a key, or her knowledge of a piece of information, e.g. a pin or password. However, a token such as a physical key or a smart card can be lost, stolen, duplicated or left at home, while a password can be forgotten, shared or compromised. Furthermore, since we live in a rapidly developing, electronic world we are obliged to remember a multitude of passwords and personal identification numbers for computer accounts, cash dispensers, e-mail accounts, mobile phones and so forth. Consequently, alternative authentication methods based on identifying physical characteristics of a person have been developed. Such characteristics are:

- physiological traits, e.g. fingerprints, face, hand geometry;
- behavioural characteristics, e.g. voice, signature, gait.

A method of identification of a person based on her distinctive physiological or behavioural characteristics is called *biometrics* [1].

A significant difference between a biometrics-based person identification and other, conventional methods is that the conventional methods do not involve any complex pattern recognition and hence they almost always perform accurately as intended by their system designers. A conventional method provides us with a clear answer if a person is who he claims to be, whereas a biometric system provides us only with a similarity score. Hence a typical biometrics-based system is not perfectly accurate and basically commits two types of errors: a false accept refers to identifying an impostor to be a genuine user and a false reject refers to rejecting a genuine user as an impostor [2]. Whereas the false reject leads to inconvenience for users, a false acceptance provides the access to a non-authorised user.

As biometric based identification methods are becoming more popular, big privacy concerns arise. A part of one's very personal information is stored in a database. In opposition to passwords which are usually stored encoded with a one-way function, due to intra-user variability in the acquired biometric traits, ensuring the security of the template while maintaining the recognition performance is a challenging task [3].

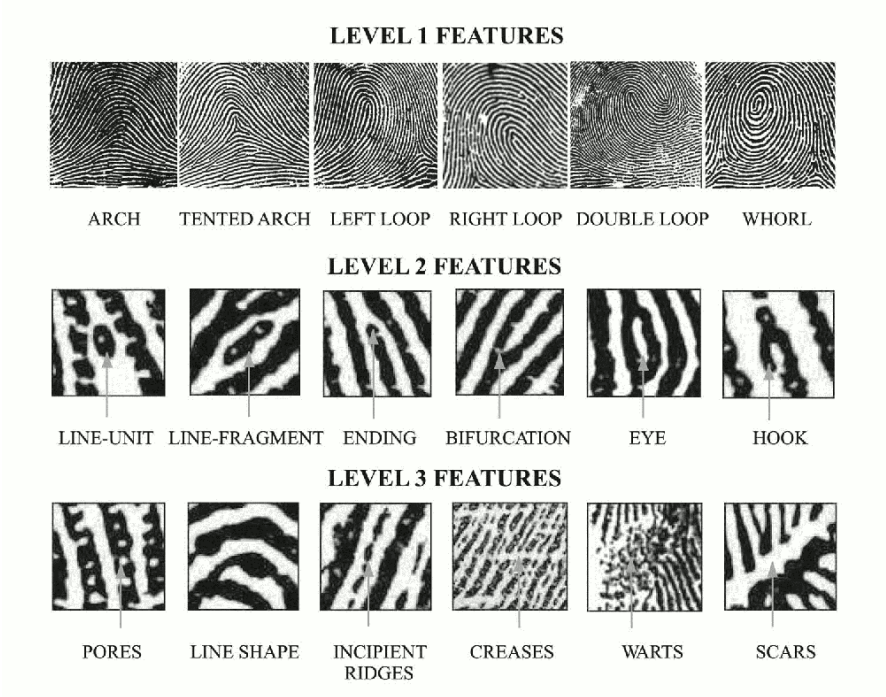


Figure 1.1. Fingerprint feature levels [6]

1.2. Fingerprints as biometrics

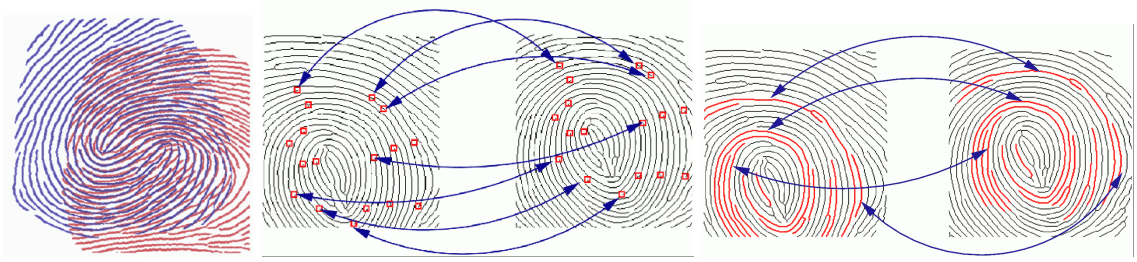
Among various biometric techniques, fingerprint recognition is the most popular one for automatic person identification [4]. The fingerprint of an individual is unique, so that even identical twins have different fingerprints [5]. Furthermore, the structure of fingerprints hardly changes over a lifetime.

A fingerprint is a pattern of ridges and valleys on the surface of a fingertip [4]. Two imprints are matched by comparison of characteristic features which are generally categorised into three levels [6] (Fig. 1.1):

- **level 1 features (patterns)** - macro details of a fingerprint such as ridge flow and pattern type which are used for fingerprint classification, global level;
- **level 2 features (points)** - minutiae, which have sufficient discriminating power to establish the individuality of fingerprints, local level;
- **level 3 features (shapes)** - all dimensional attributes of a ridge such as ridge path deviation, width, shape, sweat pores, edge contour, incipient ridges, breaks, creases, scars and other permanent details, intermediate level between global and local level.

The tasks in fingerprint recognition are generally categorised into four groups:

- **Fingerprint image preprocessing:** preparation of an acquired fingerprint image for further processing. This step contains image processing algorithms such as image enhancement or discrimination between the foreground containing the fingerprint and the background. For an example of a fingerprint image before and after preprocessing see Fig. 1.5(a) and 1.5(b).



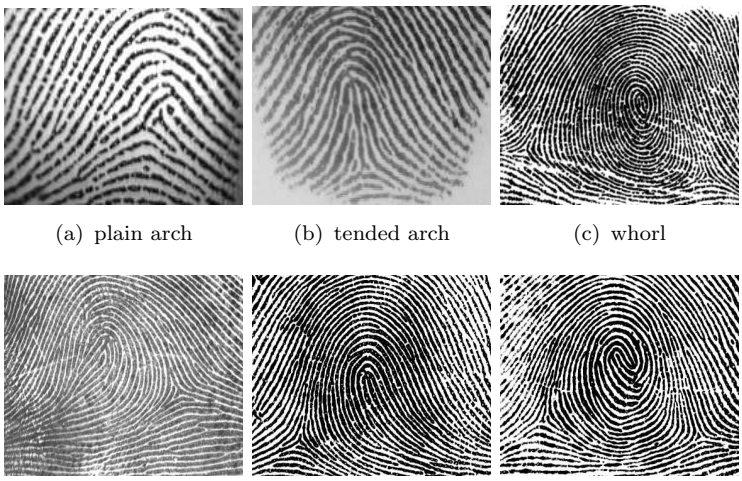
(a) correlation-based (b) minutiae-based (c) ridge feature-based

Figure 1.2. Three approaches to fingerprint matching [8]

- **Fingerprint feature extraction:** although there exist correlation-based methods for fingerprint matching which compare images directly, gray-scale image intensities are known to be unstable representations of fingerprints. Most of the fingerprint recognition algorithms are feature-based [7].
- **Matching:** comparison of two given fingerprints providing a degree of similarity. A large number of approaches can be classified into three families [7]:
  - correlation-based matching (Fig. 1.2(a)): two fingerprint images are superimposed and the correlation between corresponding pixels is computed for different alignments.
  - minutiae-based matching (Fig. 1.2(b)): most popular and widely used technique based on finding the alignment between two minutiae sets, resulting in the maximum number of minutiae pairings.
  - ridge feature-based matching (Fig. 1.2(c)): the approaches belonging to this family compare fingerprints in terms of features extracted from ridge patterns such as local orientation and frequency, ridge shape or texture information.
- **Classification:** in order to identify a person, her fingerprint has to be compared with each fingerprint in a database. In case of a large database the identification typically has an unacceptably long response time. A common strategy to speed up the query is to narrow the search by dividing the fingerprint database into a number of bins (based on some predefined classes). A given fingerprint to be identified is then compared only to the fingerprints in a single bin. The most important and widely used classification schemes are variants of Henry's classification scheme [7]. The six most common classes, naturally defined by the pattern flow, are presented in Fig. 1.3. Unfortunately fingerprints are not uniformly distributed among these six classes:

plain arch	tented arch	left loop	right loop	whorl	double loop
3.7%	2.9%	33.8%	31.7%	23.1%	4.8%

Hence, for applications which do not require compliance with an existing classification scheme, other indexing methods have been proposed. In indexing (i.e. continuous classification), fingerprints are not partitioned into disjoint classes, but associated with a point in a normed vector space so that similar fingerprints are mapped to close points. During the identification process the input fingerprint is matched with those in the database whose corresponding vectors are close to the searched one [9].



(a) plain arch (b) tended arch (c) whorl

(d) left loop (e) right loop (f) double loop

Figure 1.3. The widespread Henry's classification scheme

1.3. Fingerprint acquisition

The traditional way of capturing a fingerprint is inking, in which black ink is spread on the finger which is then pressed against a paper card; afterwards the card is scanned with a common paper-scanner [7] (a sample of an inked fingerprint is presented in Fig. 1.4(a)). Inking has a long tradition and is still used in some law enforcement applications [10]. However, this method is unsuitable for security systems in which online acquisition is required and hence, live-scan devices must be employed. There are a lot of scanner types available on the market; the most common ones are listed below:

- optical (Fig. 1.4(b)) – the image is captured by a CCD (charged coupled device) or a CMOS (complementary metal oxide semiconductor) camera. In general, the finger is placed on a glass plate and the camera takes a picture. A disadvantage of an optical system is its vulnerability, that is, the sensor cannot distinguish whether the presented finger is real or a fake. Another problem is related to latent fingerprints left by a finger previously placed on the sensor surface.
- capacitive (Fig. 1.4(c)) – the most popular one. A capacitive sensor with electrical current is employed for measuring the fingerprint. The main advantages of capacitive sensors are low-cost and built-in liveness detection. However, capacitive sensors have problems with fingerprints from wet or dry fingers.
- surface pressure sensor – the principle of pressure sensing is as follows: when a finger is placed on the sensor area only the ridges of the fingerprint touch the sensor's piezo array. In contrast, the valleys have no contact with the sensor cells.
- touchless (Fig. 1.4(d)) – in all sensor types mentioned above, a finger has to be pressed onto the sensor surface, which can introduce non-linear distortions (for details see Section 5.1). A solution to this problem is a touchless fingerprint sensor technique which is currently under development. With touchless sensors, the finger is not placed on a surface but only held over the sensor at about 5 cm. A disadvantage is that dust and dirt between the sensor and the fingerprint may contribute to bad quality of the images.

(a) inked fingerprint (b) optical-type sensor (c) capacitive-type sensor (d) touchless sensor

Figure 1.4. Sample fingerprint impressions acquired with help of different methods

1.4. Thesis motivation

As a consequence of the requirements of law enforcement and interest on the part of the developers of biometric systems, efficient fingerprint-based identification systems are becoming rapidly widespread, and are extensively researched by the pattern recognition community. It is a common misconception that automated fingerprint identification is a solved problem: despite significant research efforts over the past four decades, the state of the art in fingerprint matching technology is nowhere near the theoretically possible performance [11].

The general framework for fingerprint identification systems is well-established in the literature. A majority of current publications accepts this framework, and attempts are made to improve specific algorithms at various levels of processing. This approach has beneficial effects as the minor improvements contribute to higher accuracies. However, as the state of the art is still far from its theoretical potential, approaches radically different from those currently in fashion appear to be worth exploring [12].

Most fingerprint matching algorithms are based on matching small fingerprint details known as minutiae. However, minutiae are only a small subset of all the information contained in a fingerprint. Consequently, while comparing two imprints, experts do not focus exclusively on minutiae, taking among others the level 3 features as well as unusual formations of the fingerprint lines into account and viewing fingerprint image at both global and local scales at the same time. As a result, though the information contained in minutiae is sufficient for automatic recognition of fingerprint images unaffected by noise because a sufficient number of genuine minutiae can be extracted, it is not reliable enough for disturbed images. There, in order to improve the recognition, non-minutiae-based information present in fingerprints should be explored.

Requirements for identifying and defining additional fingerprint features beyond minutiae, even not limited to level 3 details, has already been addressed by the community [13]. However, new feature proposals are based on refinements of existing features (e.g. a finer level of classification), or on the introduction of new single features (e.g. 3-D level features such as the ridge height) [14]. The new set of features proposed in this work does not only include additional fingerprint features individually but it also contains the information about their relationships such as line adjacency information at minutiae points or links between neighbouring fingerprint lines.

As has already been mentioned, an expert can look at a fingerprint image at various scales at the same time, whereas common automatic methods utilise different feature levels separately in different fingerprint processing steps. Thus the concept put forward in this thesis is to extract a universal set of features which can be used in all the tasks of fingerprint processing. As a consequence, the time required for extraction, and the space demanded for storing those features might be reduced. An example of such a universal feature is presented in Fig. 1.5(e). The extracted red line provides us with both information about the class of the imprint (left loop) and part of the information for matching, since the set of all such lines allows us to discriminate between fingerprints deriving from different fingers, whereas the commonly used features for classification, such as orientation fields or singular points, do not provide enough discriminating information for matching. Moreover, the utilisation of local information can decrease the misclassification rate as explained in Chapter 4.

In spite of the fact that a fingerprint is a structure build up from both ridges and valleys, common algorithms extract information only from ridges. At first sight, it appears to be reasonable as the processing of both ridges and valleys would be redundant. On the other hand, the apparent redundancy provides us with information which can be used to correct defects caused by noise. If either one is disturbed by noise, the other one may be correct and may allow to correct the first one. In Fig. 1.5(a) the ridges (dark lines) are disconnected due to a dry fingertip skin. Nonetheless, since fingerprint lines change their direction rather smoothly, the valleys can be extracted properly and then be used for connecting the ridges. Such errors might also be removed with image processing tools. A directional smoothing filter, such as the Gabor filter [15], results in an enhancement of disconnected lines as presented in Fig. 1.5(b). Unfortunately, at the same time some level 3 features, e.g. scars, might also be smoothed out as shown in Fig. 1.5(c) and 1.5(d).

Assessing both ridges and valleys allows to validate the extracted information by the duality principle: a bifurcation of ridge corresponds to an ending of valley and vice versa. The green ending of the ridge in Fig. 1.5(f) corresponds to the blue bifurcation of the valley. The only exception to this principle arises in singular areas, as shown in Fig. 1.5(f): the red marked ending in the core area does not posses a corresponding minutia. There exists exactly one such minutia for each singular point. Consequently, minutiae without corresponding dual minutia outside the singular areas are indicators for errors in acquisition, preprocessing or feature extraction in that region. Thus the region can be marked and processed again more thoroughly.

In order to improve the matching, it is vital that the extracted features set remains stable over multiple acquisitions. However, even if a feature is missing in one impression, the extracted structure around the missing feature ideally should provide enough information to confirm the fact of the existence of that feature in the original fingerprint.

Another important challenge is the reduction of the space required for storing acquired fingerprint impressions. Let it suffice to say that the FBI databases already contain over 200 million fingerprint records [7] and as the number of records is growing rapidly, the requirements on storage are similarly increasing. Consequently it is crucial to find a low-memory representation for a fingerprint containing all necessary information.

To summarise, the guiding principle of this thesis might be expressed as:

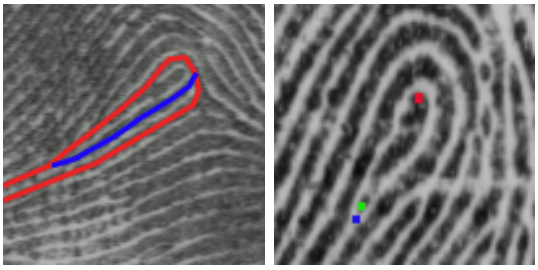
Extract a maximal amount of stable information robustly.

(a) A noisy fingerprint image with disturbed lines

(b) The disturbed lines after a preprocessing step

(c) A fingerprint with scars

(d) The scars after a preprocessing step



(e) Local feature for classification

(f) Duality principle

Figure 1.5.

1.5. Thesis contributions

The presented work contributes to various stages of the fingerprint recognition process. A brief summary of the specific contributions is listed below:

- In order to bridge the gap between automatic procedures and human experts, a new class of fingerprint features which combine the traditional global and local features, is in this thesis.
- A is novel low-memory representation of fingerprints is proposed as a basis for extracting of these new features.
- Additional information contained in a fingerprint, such as duality between ridges and valleys, is utilised to obtain a reliable feature set.
- Two improvements to the preprocessing stage are made, resulting in better segmentation, and enhancement of areas disturbed by noise.
- A novel approach to fingerprint classification based on the extracted features is proposed, as a result the misclassification error rate has been decreased by 0.2% to 3.5%.
- For the first confirmation of the quality of extracted features in application for the matching, the precision of the extracted minutiae is measured. The experiments show a decreased error rate when supplying the minutiae to a standard minutiae-based matcher.
- A broad and careful study of the extracted features for fingerprint matching is conducted with the objective to develop a matcher which fully utilises the advantages of all those features.
- A novel development process based on interactive testing is proposed. As a consequence, an interactive software environment for fingerprint research has been created.

1.6. Thesis outline

This thesis starts in Chapter 2 with the description of a new approach for fingerprint feature extraction. Following chapters discuss the application of the extracted features to different stages of fingerprint recognition. Chapter 3 addresses preprocessing, Chapter 4 presents classification, followed by matching in Chapter 5. The development process and interactive software environment for fingerprint research is presented in Chapter 6. Finally, Chapter 7 contains a summary of the major contributions and results of the thesis, and puts forward suggestions for future research in this direction.

1.7. Acknowledgements

This work would not be possible without the help of many people. I wish to thank my thesis advisor Axel Munk for proposing the problem and supervision. My thanks go also to my coadvisor Preda Mihăilescu for fruitful discussions and drawing my attention to new topics in fingerprint recognition field. I would like to thank the Institute for Mathematical Stochastics at the University of Göttingen and the DFG's Graduiertenkolleg "Identifikation in mathematischen Modellen: Synergie stochastischer und numerischer Methoden" for the financial and scientific support. I would like to thank Thomas Hotz for proof-reading this thesis from which it benefited greatly. Last but not least I want to thank my colleagues at the institute for discussions and the general good working atmosphere.

Chapter 2

Entracer - an extended feature extractor

2.1. Fingerprint features

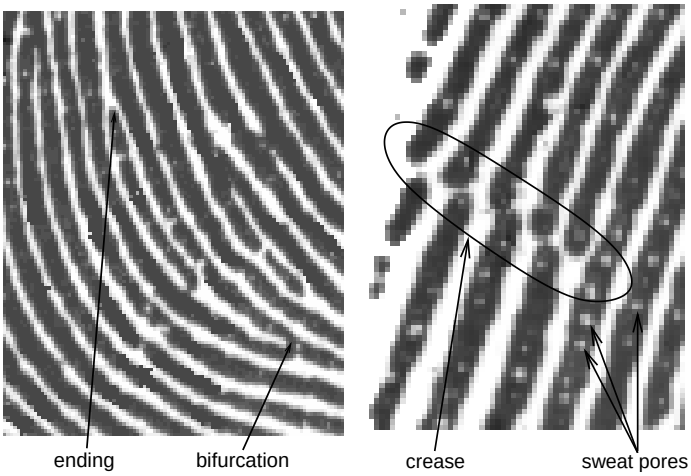
The ways a forensic expert and a machine treat a fingerprint image (Fig. 2.1(a)) differ in many aspects. A machine utilises the different levels separately for different applications: for classification the global level (Fig. 2.1(b)) is used, whereas for matching exclusively local features (Fig. 2.1(c)) are used, and the global information is not used directly. An expert, however, approaches a fingerprint impression at different scales at the same time.

Furthermore, the most common matching algorithms use only level 2 features and no level 3 features (Fig. 2.1(d)), although they – like the level 2 features, are also claimed to be permanent, immutable, and unique [1]. If properly utilised, they can provide discriminatory information, and that is why forensic experts use them together with the level 2 features in a latent print examination.

One reason why level 3 features are hardly used by commercial software is that for an automatic extraction of most of those features high resolution (at least 1000 dpi) devices are required [1]. However, some of the features (such as a ridge shape, breaks, creases) can also be easily found in images acquired by 500 dpi scanners which are commonly used in automatic systems. In this section we will define such features and describe an extraction method for them. Since we will also extract additional features (such as ridge connection information or neighbourhood information) we propose to call them *hierarchically linked extended features*.

Our data extraction approach differs from other current approaches in more than one aspect. We tackle the problems of orientation flow extraction, ridge and minutiae recognition at the same time, taking advantage of their interdependence. The features are extracted by following the fingerprint lines. In doing so, we use a simple adaptive - or *entropy sensitive* - approach. Its two main characteristics give the name to our method: entropy sensitive tracer - **entracer**.

(a) Sample fingerprint image (b) Global features - level 1 features



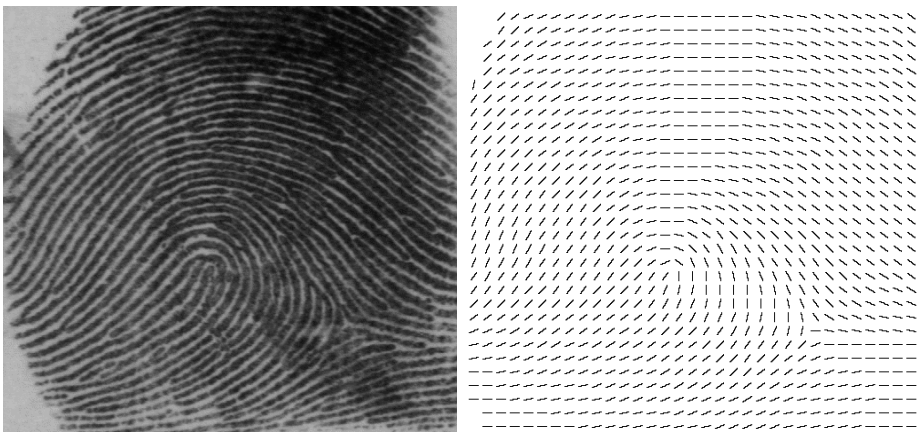
(c) Local features - level 2 features (d) Local level 3 features

Figure 2.1. Different zoom-levels at which a fingerprint image can be viewed

2.2. Fingerprint structure extraction

In order to identify fingerprint features, first a fingerprint structure containing the fingerprint's crucial information has to be extracted. The essential engine which we are going to use for that task is a fingerprint line tracer. Various approaches for minutia detection which are based on following lines have been investigated in the literature [16; 17; 18; 19]. However, there are principial differences between the existing methods and ours:

- We additionally extract the so-called *extended features*, whereas other approaches focus solely on minutiae detection.



(a) (b)

Figure 2.2. A fingerprint (a) and its orientation field (b).

- We perform the tracing in an already binarised image. In contrast to grey level images, in binarised images recognising the line's border while following it is much simpler and faster. Thus we can skip the costly centring on the line but on the other hand we have to binarise the input image. Still, it is advantageous since binarisation is much faster than centring. Furthermore, we do not fully dismiss the grey level image, but come back to it if necessary; for example in regions where the binarised image is strongly disturbed by noise.
- Both ridge and valley lines (or simply the white and black lines) are processed. The objective of the apparently redundant information is the correction of defects caused by noise. For example, in regions where black lines are broken the information from connected white lines is used to repair the black lines.
- We do not need to compute the orientation field[1] before tracing. Moreover, the orientation field can easily be computed from the extracted features.

2.2.1. Preprocessing. Before we can apply the tracer algorithm to an acquired fingerprint image, some preprocessing steps on the image have to be performed.

Since common matching algorithms utilise only level 1 and 2 features, the additional features are treated as noise and are removed during the enhancement stage. The application of a directional smoothing filter results in an enhancement of the input image. A widespread smoothing method is based on Gabor filters [15]. As already mentioned, fingerprints are flow-like patterns which consist of locally parallel ridges and valleys, and their local frequency and orientation are well defined. Gabor filters have both frequency- and orientation-selective properties and have a good joint resolution in both spatial and frequency domains. Hence, Gabor filters are sensibly used to remove the noise and preserve true ridge/valley structures [20].

[1]The orientation field is the set of local ridge orientations as illustrated in Fig. 2.2.

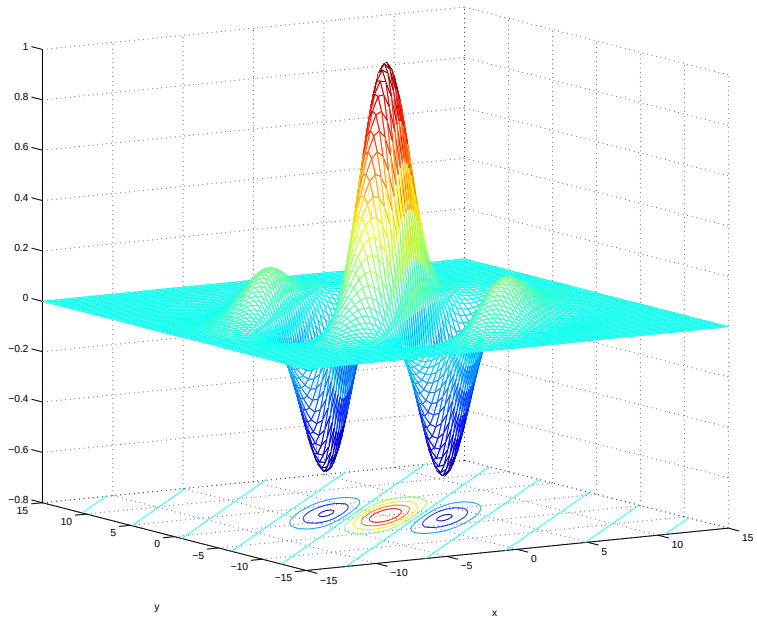


Figure 2.3. Graphical representation of the Gabor filter.

As shown in Fig. 2.3, the Gabor filter is defined by a cosinusoidal plane wave tapered by a Gaussian. The even symmetric two-dimensional Gabor filter has the following form [7]:

$$G(x, y; \theta, f) = \exp\left(-\frac{1}{2}\left(\frac{x_\theta^2}{\sigma_x^2} + \frac{y_\theta^2}{\sigma_y^2}\right)\right) \cdot \cos\left(2\pi f x_\theta\right)$$

$$x_\theta = x \sin\theta + y \cos\theta$$

$$y_\theta = y \cos\theta - x \sin\theta$$

where θ is the orientation of the Gabor filter, f is the frequency of the cosinusoidal plane and σ_x and σ_y are the standard deviations of the Gaussian envelope along the $x-$ and $y-$axes, respectively.

To apply Gabor filters to an image, each pixel (x, y) of an image is convolved, in the spatial domain, with the filter $G(x, y; \theta, f)$ such that θ is the ridge orientation at the pixel and f its ridge frequency. An example of the application of Gabor filters to a fingerprint is presented in Fig. 2.4. Whereas the disrupted ridges have been connected (Fig. 2.4(a) and 2.4(b)), the level 3 features (Fig. 2.4(c) and 2.4(d)) are smoothed out simultaneously. In order to preserve the information about the level 3 features, we use a very weakly smoothing Gabor filter and additional non-directional smoothing enhancement, such as holes filling and a median filter.

Segmentation[2], that is, the division between the foreground containing information and the background, is very important for our line following algorithm since only fingerprint lines and not the surrounding area should be traced. We apply the algorithm proposed by Bazen and Garez [21] for this task.

Finally, the image is binarised by means of the common adaptive mean filtering [7].

[2]For a detailed description of segmentation see Section 3.1.

(a) Input image

(b) The disconnected lines have been repaired

(c) Input image

(d) The scars have been smoothed-out

Figure 2.4. Problems which arise while smoothing of fingerprint lines

2.2.2. Tracing. A fingerprint line has three possible end types:

- bifurcation (point where three lines join together),
- border point (point where a line hits the background),
- ending.

Additionally, a whorl type fingerprint can contain special lines without ends, that is closed lines.

Black and white lines are traced independently. The correspondence between the black and the white area is added in the "offline" stages after all lines have been traced.

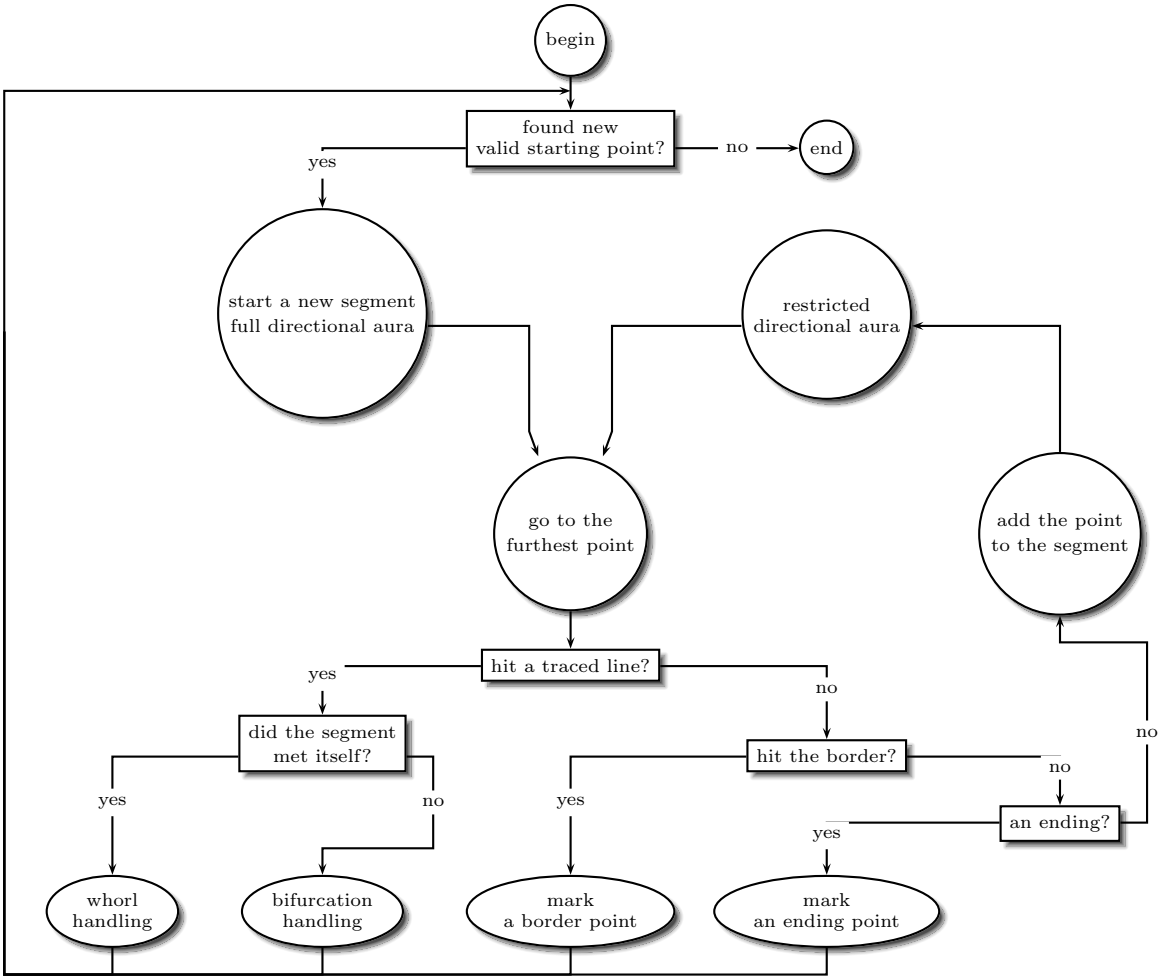
The tracing process briefly presented in Table 2.1 is described in detail in the following subsections.

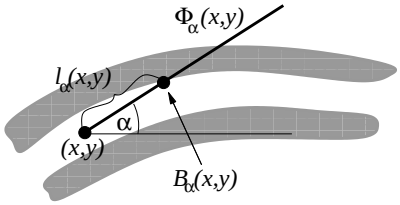
Aura. The main tracer tool is an **aura** which comprises **feelers** and **ridge border location**. The aura determines the optimal direction which the tracer should follow from a given point.

The feeler Φ_α is a directed line section with direction $\alpha \in \Omega$ starting at the position (x, y):

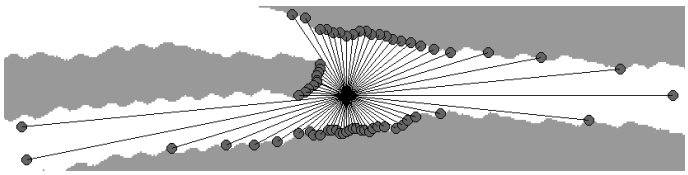
$$\Phi_\alpha(x, y) = \{(x, y) + \lfloor m \cdot (\cos\alpha, \sin\alpha) \rfloor : m \in \mathbb{N}\},$$

Table 2.1. Tracing process





(a) Sample values of functions for aura computation



(b) Set of angular border distances.

Figure 2.5.

For each direction α the maximal value of m is computed, such that the section $\Phi_\alpha(x, y)$ stays within the fingerprint line. The section length for that m is computed as:

$$l_\alpha(x, y) = |B_\alpha(x, y) - (x, y)|,$$

where $B_\alpha(x, y) : \mathbb{N}^2 \to \mathbb{N}^2$ returns the point where the section $\Phi_\alpha(x, y)$ crosses the line border and $|\cdot|$ denotes the Euclidian distance, see Fig. 2.5.

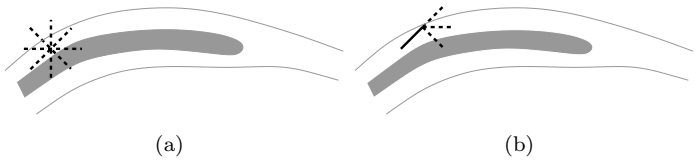


Figure 2.6. Aura in a starting point (a) and in a next point (b).

The set of directions Ω at the first point of the traced line is a discrete set of equidistant angles:

$$\Omega = \left\{ \frac{i}{n} 2\pi : i = 0, ..., n-1 \right\}$$

As ridges and valleys do not take abrupt turns but change direction rather smoothly, we can restrict Ω to some subset $\Omega_0 \subset \Omega$ for the following points on the line so that only directions in a cone around the previous direction are taken into account. An example is shown in Fig. 2.6. An extracted fingerprint line between two minutiae or border points will be called a segment.

Valid starting point. A point in a fingerprint image at which a single tracing process is started has to fulfil the following conditions:

- it is located in the foreground,
- it is within a fingerprint line,
- the line has not been traced yet.

In order to check the last two conditions, the full aura is computed for a given point. It has to have significantly larger values for two opposite directions compared with their orthogonal directions, and none of the feelers should meet an already traced line at a distance below a given threshold.

The candidates for the starting points are taken from a grid, that is from the set

$$\left\{ (i \cdot x_\Delta, j \cdot y_\Delta) : i = 1 \ldots \left\lfloor \frac{x_{size}}{x_\Delta} \right\rfloor,\ j = 1 \ldots \left\lfloor \frac{y_{size}}{y_\Delta} \right\rfloor \right\}$$

where x_{size} and y_{size} are the vertical and horizontal size of the input fingerprint image while the values of x_Δ and y_Δ depend on the resolution of the fingerprint sensor was used for acquisition. On the one hand, overly small values result in unnecessarily large computation times for processing the whole fingerprint. On the other hand, if the values are too large some lines may not be traced. For a 500 dpi sensor, we choose $x_\Delta = y_\Delta = 4$.

The tests performed have shown that the order in which the points on the grid are examined does not influence the result of the tracing.

Branches of a bifurcation. In order to properly identify the segments connected to a bifurcation, adjacent segments are numbered according to a certain scheme: the first branch is the one on the opposite side of the smallest angle and all branches are ordered anticlockwise (Fig. 2.7(a)).

To find the correct order, the distances a, b, c (Fig. 2.7(b)) between the points on the connected segments at a given distance from the bifurcation point are computed. The smallest distance (in this case c) is between the second and the third branch.

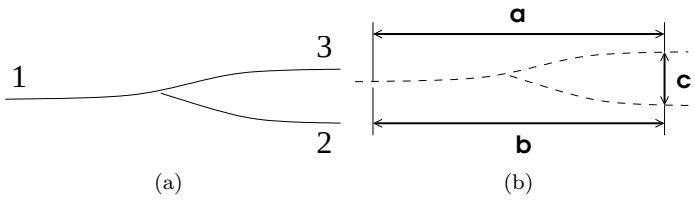


Figure 2.7. The order of bifurcation branches

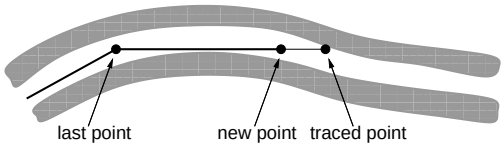


Figure 2.8. Simple centring

2.2.2.1. *Centring.* With the existing grey level tracing methods, in order to follow a ridge correctly the tracer has to stay at the top of the ridge, which requires the application of an expensive centring method. This is not necessary in case of tracing in a binarised image as the algorithm for line border recognition uses only transitions between the black and the white area, but no grey level intensity differences. However, if one prefers that the traced points are near the line centre, a simple and efficient centring can be used. After proceeding to the next traced point, the traced point is moved back toward the last point by 20% of the distance between the current and the last point. An example is presented in Fig. 2.8.

2.2.3. Fingerprint structure. The information obtained from the tracing process is:

- minutiae and border points;
- closed segments (whorls) which are usually not captured by a typical minutiae extraction algorithm;
- ordinary segments with two ends from which one is called *front end* and the other *back end*.

Information stored for a minutia or a border point:

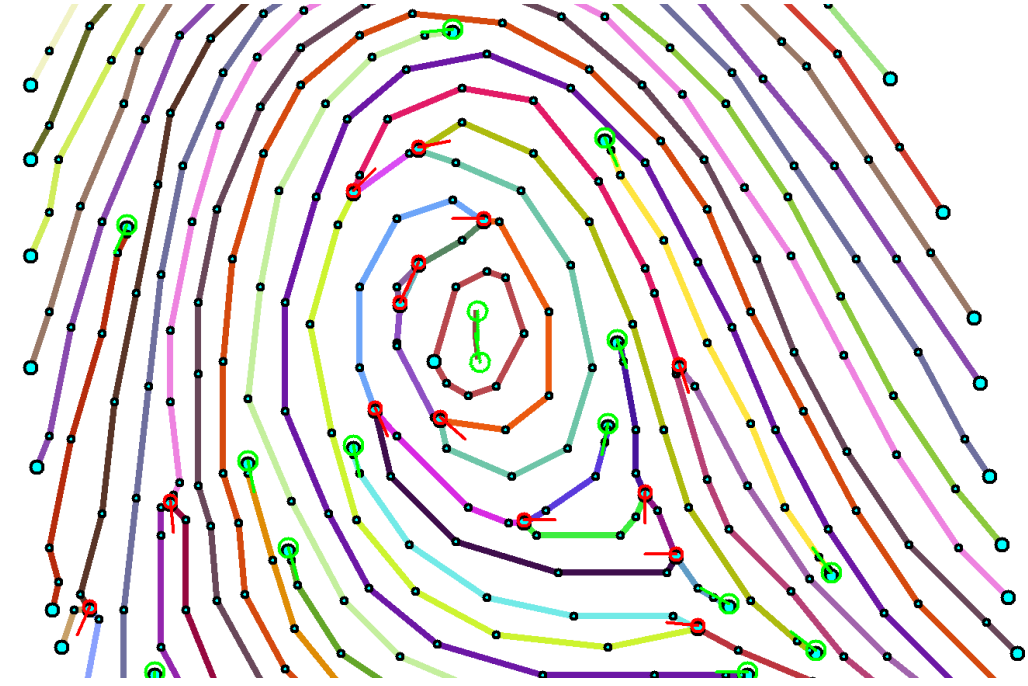
- position
- adjacent segments (3 for a bifurcation and 1 for an ending or a border point) with information whether the minutia is connected at the front or back end of the segment.

The following data are stored for segments:

- type (simple or closed)
- in case of a simple segment: minutiae or border points at both ends of the segment
- line course as a sequence of points.

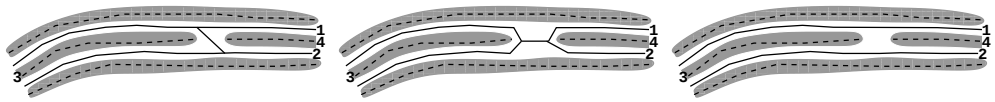
A sample traced fingerprint is presented in Fig. 2.9.

(a) Fingerprint structure with underlying binarised image. Each segment is marked with a different colour. Endings are marked in green and bifurcations in red. Note the unvalidated minutiae at the core points where only endings and no bifurcation are marked and and at the delta point where a bifurcation but no ending is marked.



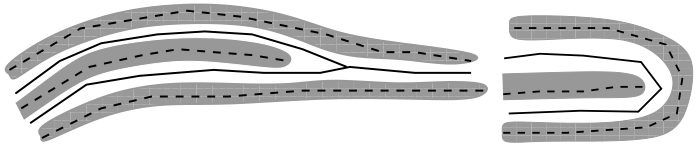
(b) Fingerprint structure for the black lines. Each segment is marked with a different colour. Endings are marked in green and bifurcations in red. The black points on the segments are the traced points.

Figure 2.9. A sample traced fingerprint.



(a) (b) (c)

Figure 2.10. Deleting of short segments



(a) A bifurcation in white with the corresponding ending in black

(b) The exception for a singular point

Figure 2.11. Duality principle

2.3. Postprocessing

Due to existing noise in fingerprint impressions as well as the specific way in which the entracer works, the obtained structure has to be postprocessed.

2.3.1. Short segments. Spurious bifurcations extracted by the entracer in noisy areas are removed in this step. If two bifurcations are connected over a short segment and the segment is connected to both minutiae at the branch with the same number (Fig. 2.10(a) and 2.10(b)), the short segment and the bifurcations are removed and the remaining segments are connected (Fig. 2.10(c)). The two segments on the disrupted black line are connected in one of the later postprocessing stages.

2.3.2. Validation. The white and black areas are naturally dual to each other (see Fig. 2.11(a)). Generally, bifurcations in one area correspond bijectively to endings in the other area. Natural exceptions can be encountered only at singular points (Fig. 2.11(b)); further exceptions are consequences of noise (Fig. 2.12(a) and 2.12(b)) or additional features (Fig. 2.12(c)) and thus the existence of dual minutiae is an indicator for a reliable minutia point. Consequently, this – obviously not novel ([7], p.86) – observation of the duality of ridges and valleys becomes a fruitful resource for validating minutiae.

During the validation step, for every bifurcation its corresponding ending is searched for in the area between the second and third branch of the bifurcation, as shown in Fig. 2.13. Once the dual ending has been found, the bifurcation position is reset to the ending position. This eliminates the problem of false positions of bifurcations determined during the tracing step. For each validated minutia, the pointer to the corresponding minutia is stored in the structure, which introduces the first connection between dual areas.

2.3.3. "Bifurcations ladder". It may happen in some fingerprints that due to an unusual grey-level distribution the background may be falsely recognised as foreground, which results in a foreground stripe around the fingerprint (Fig. 2.14(a)). During the tracing process false bifurcations are found in such regions (Fig. 2.14(b)). As the bifurcations are connected between the first and the second or third branch, they are not removed during

(a) Due to noise, two unvalidated endings have been extracted

(b) Correction of the situation presented in (a)

(c) Crease which is marked as an additional feature for matching

Figure 2.12. Unnatural exceptions from the duality principle

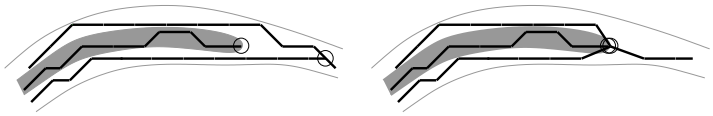
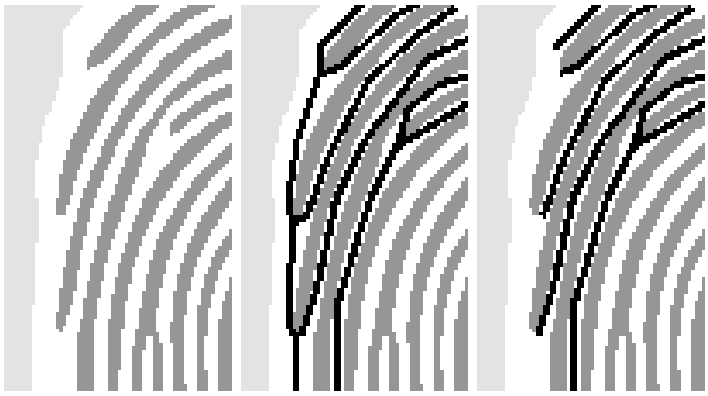


Figure 2.13. Process of minutia validation



(a) Wrongly extracted foreground

(b) Spurious bifurcations

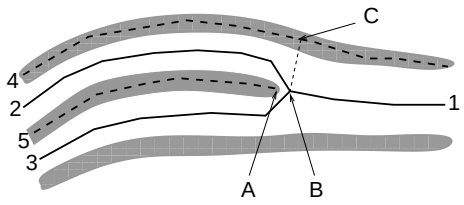
(c) Result of the postprocessing

Figure 2.14. An example of "bifurcation ladder" processing. For better illustration only lines traced in white are shown.

the short segment removal stage but in an extra postprocessing step in which such a sequence of at least 3 shortly connected bifurcations (which we call **bifurcations ladder**) is identified and removed. The bifurcations are resolved and the corresponding dual false endings are changed into border points. The result is shown in Fig. 2.14(c).

2.4. Neighbour information

In order to provide for the required robustness of the features, additional information contained in the fingerprint structure must be evaluated. The most important information for that purpose is the neighbourhood information described in this section.



At point C line 4:

side:	right
position:	25
before:	2
after:	1
minutia:	B

At point A line 5:

side:	left
position:	0
before:	none
after:	3
minutia:	none

At point A line 5:

side:	right
position:	0
before:	none
after:	2
minutia:	none

Figure 2.15. Sample change points for segment neighbours, front ends of segments 4 and 5 are on their right.

2.4.1. Segment neighbours. For further postprocessing and applications to other fingerprints tasks (classification, matching), neighbouring dual lines, which are the lines running parallel in the dual colour area, are identified for each traced line or, in case of an outermost segment, the background is marked as the neighbour.

The neighbour information is stored as a list of change points - the points on a line where the neighbours change. The information stored in one list element is:

- side of change - right or left looking from the front towards the back end of the segment;
- position of change - the length of the segment between the front end and the change point;
- neighbouring segment towards the front end (before);
- neighbouring segment towards the back end (after);
- minutia causing the change of the neighbour.

Some examples for the stored information are shown in Fig. 2.15.

For sake of simplicity, in the following description we assume that minutiae are adjacent at the corresponding segment's front ends.

Clearly, in a non-disturbed area the neighbour of a segment changes only if there is a minutia on the neighbouring segment. Hence, it suffices to find the neighbours for all minutiae and border points and in order to create the neighbour lists. Consequently, the computation time is reduced in comparison to identifying neighbours by following segments and checking at fixed lengths.

The points are processed in the following order:

- validated bifurcations
- border points
- unvalidated minutiae

For a validated bifurcation the information about *inside* neighbours already exists due to the duality principle (compare Fig. 2.16(a)). Consequently, the right neighbour of the second branch and the left neighbour of the third branch is the segment connected to the dual ending. Hence the neighbours of the segment connected to the ending are defined as well. In order to find the *outside* neighbours, we follow the directions orthogonal to

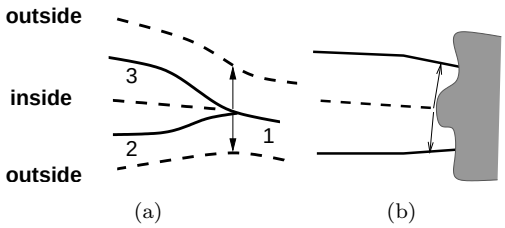


Figure 2.16. Finding neighbours for a bifurcation (a) and a border point (b)

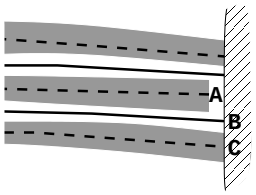


Figure 2.17. An unvalidated ending next to the background

the bifurcation direction until a segment in the dual area is met. For border points and unvalidated minutiae the strategy is similar to the one for "outside" neighbours at a bifurcation. We look for the neighbours in the directions orthogonal to a border point or an unvalidated minutia direction (Fig. 2.16(b)).

2.4.2. Neighbours at border points and unvalidated endings. In addition to the information about segment neighbours, the neighbour point (if one exists) is stored for each unvalidated ending or border point. For the situation presented in Fig. 2.17 at the border point `B` the following data is stored: the left neighbour is the border point `C` and the right neighbour is the unvalidated ending `A`.

2.4.3. Postprocessing using neighbours information.

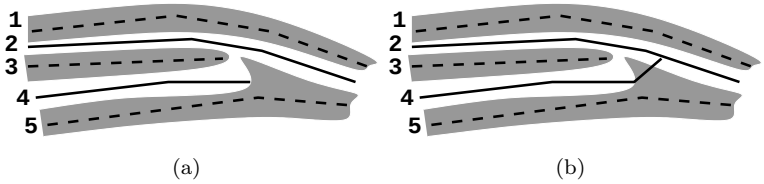
Bridging. For connecting disconnected lines dual information is used. Generally, if a ridge is disconnected, their dual neighbours are not broken. See Fig. 2.10(c). We use this information to repair broken ridges.

In order to distinguish between a line break caused only by noise and that contained in a fingerprint (creases, scars), the number of neighbouring, disconnected lines is counted. If it is less than or equal 3, we treat it as noise and consequently connect the lines. Otherwise the disconnection is marked as a scar for further processing.

Correcting unvalidated endings. As already mentioned, an unvalidated minutia, apart from one for each singular point, is an indicator for an extraction error. The neighbour information can be used to repair such errors.

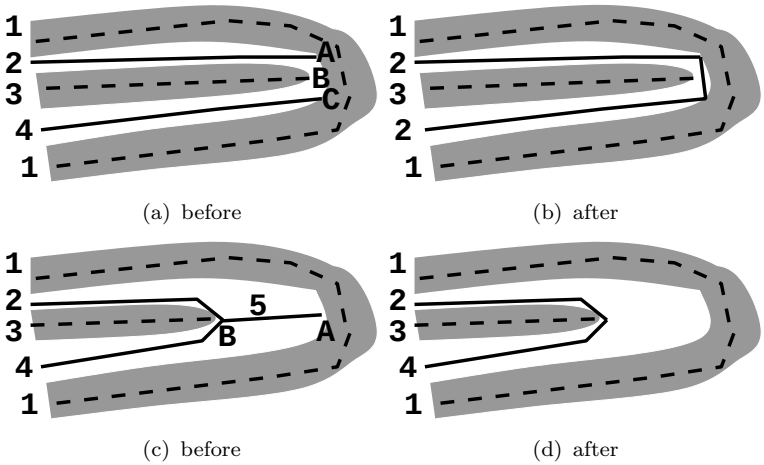
If an unvalidated ending has a border point as its minutia neighbour (compare Fig. 2.17), its type is changed from ending to border point.

Another unvalidated ending occurs when the tracer gets stuck instead of finding a bifurcation as presented in Fig. 2.18(a). In order to find and correct these errors, the neighbourhood information can be used and the segment adjacent to one of the endings is connected to its neighbour segment in such a way that the resulting bifurcation is the



(a) (b)

Figure 2.18. Correcting an unvalidated ending



(a) before (b) after

(c) before (d) after

Figure 2.19. Correcting an unvalidated endings located in a core area; before and after postprocessing

dual one for the second unvalidated ending (Fig. 2.18(b)). Because of the virtual minutiae introduced in Section 2.6.2, we can choose arbitrary which ridge pair is connected (4 to 2 or 3 to 5 in Fig. 2.18(a)).

Yet other rules apply for an unvalidated ending in a core area. First, the core has to be identified, which is an easy task using the neighbour information. The possible situations are presented on the left side of Fig. 2.19. Points `A` and `C` have the same segment neighbour on opposite sides in the first case, and in the other case `A` has the same neighbours on both of its sides and the length of segment 5 is below the threshold. The corrections are presented on the right side of Fig. 2.19.

2.5. Features

The fingerprint structure presented above, containing minutiae, segments and connectivity information, is an important basis for extracting the information contained in a fingerprint which is used for recognition. The generally extracted features are presented below while the features for the particular applications are described in the following chapters.

2.5.1. Standard features. In addition to the new features described below, the standard features such as orientation field or minutiae are also contained in the extracted structure. For example, the orientation field can be straightly computed from the angle between a linear piece of a traced line and the horizontal line. The singular points can be extracted as well. The approach for deltas is described in Section 4.5.2. For detection of a core point, an approach based on the duality as described in Section 2.3.2 could be

developed. However, due to noise the duality condition saves only as a necessary condition since there can exist an unvalidated minutia other than at a singular point which has been introduced by noise. Hence additional information – still contained in the extended feature set – curvature and windings of the line and their neighbours have to be used.

2.5.2. Minutia and border point direction. As the fingerprint structure changes frequently during the postprocessing, additional information such as the direction of a minutia, curvature or winding number of a line is computed directly from the extracted data. The direction of an ending or a border point is computed from the line course of the connected segment. For a bifurcation the direction is taken as the average of the second and third branch direction.

2.5.3. Line distance. To compute the distance between two points p_1 and p_2 lying on the same segment, their Euclidian distance can be used. However, in order to capture the shape of the line, we use the notion of **line distance** which is defined as the length of the segment's interval between p_1 and p_2.

2.5.4. Segment line shape. In order to describe the flow and the shape of a segment, the following descriptors can be introduced:

- length of the segment - the line distance between the end points which will be denoted with $l(s)$ for a given segment s,
- cumulative direction change - the sum of direction changes along the segment line,
- winding number, and
- curvature.

In order to compute the curvature two approaches have been investigated. The first is the computation directly from the natural piecewise linear representation of a segment [22]. Although this method is not computationally intensive and provides good results in clear areas of a fingerprint, it performs poorly in the disturbed areas of high curvature such as regions around singular points. One solution could be the approximation of the segment line by a smooth curve, which leads us directly to the second approach: B-splines posses good smoothing and approximation capabilities for a segment [23]. Each segment is represented as a plane curve in parametric form:

$$\mathbf{r}_s(t) : [t_0, t_m] \to \mathbb{R}^2$$

given by a linear combination of B-spline basis functions $b_{i,k}(t)$ using $m+1$ control points $\mathbf{p}_i = (x_i, y_i)$

$$\mathbf{r}_s(t) = \sum_{i=0}^{m} \mathbf{p}_i b_{i,k}(t) = \sum_{i=0}^{m} (x_i b_{i,k}(t), y_i b_{i,k}(t))$$

where k is the order of the B-spline curve [24]. The application of B-splines is also important for the reconstructing of the fingerprint image from the stored fingerprint structure. Additionally, the B-spline representation can be used for the comparison of fingerprint ridges during matching. This two applications have not yet been thoroughly investigated but are left for future research.

2.5.5. Scars and creases. As has already been mentioned, in contrast to simple line breaks, scars or creases are not being connected. They are marked as level 3 features to be used in the matching process.

Structures which at first sight look like a scar or crease may turn out to be only noise, and consequently scars and creases should be validated by means of multiple acquisition. Furthermore, some scars can be temporary and disappear over time (e.g. a wound that heals). To distinguish between a permanent and a temporal scar some correction over time might have to be applied. If a fingerprint with scar was stored as a template and the scar has not been detected in a series of future acquisitions, the scar should be deleted from the template.

2.6. Fingerprint structure in detail

2.6.1. Fingerprint graphs. We can regard the obtained fingerprint structure (extracted minutiae, border points, and segments) as a graph in which minutiae and border points are vertices and segments are edges between them. Since it may happen that two edges join the same bifurcations (e.g. for an eye) it is in general a *multigraph* (a graph in which multiple edges and loops are allowed) [25]. In order to include duality information we add special edges that connect corresponding validated minutiae. The graph is defined as follows:

$$G_M = (V_M, E_M, E_D),$$

where V_M is the set comprising minutiae and border points. The sets[3] of edges $E_M, E_D \subset V_M \times V_M$ fulfil the following conditions:

$\{v_1, v_2\} \in E_M \iff v_1$ and v_2 are connected via a single segment.

$\{v_1, v_2\} \in E_D \iff$ one of v_1, v_2 is a bifurcation and the other one is an ending, and they are dual to each other.

Hence the vertex degree $d(v) \in \{1, 2, 3, 4\}$:

- border points have degree 1,
- endings have 1 (unvalidated) or 2 (validated),
- bifurcations have 3 (unvalidated) or 4 (validated).

We can also create a graph in which segments are vertices, as Isenor and Zaky proposed [27]. Let us define

$$G_S = (V_S, E_S, E_N),$$

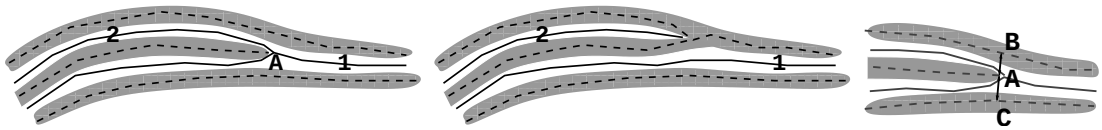
where V_S is the set of segments, and the edge sets E_S, E_N fulfil the following conditions:

$\{v_1, v_2\} \in E_S \iff v_1$ and v_2 have a common bifurcation, or v_1 and v_2 are connected over a dual minutia pair, that is, there exists a path $P = (e_1, e_2, e_3)$ in the graph G_M in which $e_1 = v_1$, $e_3 = v_2$ and $e_2 \in E_D$

$\{v_1, v_2\} \in E_N \iff$ the segments v_1 and v_2 are neighbours.

The graph G_S also contains information about closed segments which are represented by vertices adjacent only to edges from the E_N set.

[3] Actually E_M is a multiset, that is a collection of objects in which order is ignored and elements may occur more than once [26].



(a) (b) (c)

Figure 2.20. An example of minutia type change and a virtual minutia

2.6.2. Virtual minutiae. Because of the instability of minutia type, a challenge in fingerprint matching is the switching of the minutia type between an ending and a bifurcation. Hence, in the most automatic fingerprint matching methods the type of a minutia is generally not taken into consideration [1].

An ending can change to a bifurcation[4] if the line adjacent to the ending gets connected to a neighbouring line. There are two stages at which it can happen:

(1) during the enhancement of the image (e.g. by directional smoothing),

(2) already during the acquisition due to skin elasticity (Fig. 5.2).

While the disruptive influence of the first one can be decreased by improving the enhancement method, the second problem cannot be avoided when contact-based sensors are used for acquisition as is commonly done.

Minutia A from Fig. 2.20(a) changes its type from an ending to a bifurcation which leads to the situation presented in Fig. 2.20(b). It bears more consequences: when minutia A is an ending, then points 1 and 2 lie in the same connected component but after the minutia type has changed, the points might even no longer belong to the same dual connected component.

To deal with this problem we propose *virtual bifurcations*, that is we introduce additional points on the neighbour segments and add links between them and the ending (Fig. 2.20(c)), which results in an extension of graph G_M:

$$G_V = (V_M \cup V_V, E_M, E_D, E_V),$$

where V_V is the set of the additional points on the segments and $\langle m, v\rangle \in E_V \subset V_M \times V_V$ is a link between a virtual minutia and an ending or an end of the segment on which the virtual minutia lies.

2.6.3. Connected components. Three commonly applied connectivity systems are:

- **simple** – in graph G_M. Two vertices are connected if there exists a path between them containing only edges from the set E_M.
- **dual** – in graph G_M. Two vertices are connected if there exists a path which may contain edges from both sets E_M and E_D.
- **extended** – a connectivity system can be extended with virtual minutiae. If there exists an edge in E_V connecting two different connected components, the two components are combined into one.

The descriptors of a segment listed above can naturally be extended (with respect to a defined connectivity system) to measures between two points lying on different segments. E.g. for computing the distance between two points not lying on the same segment the

[4]Note that the corresponding dual bifurcation changes to an ending.

following procedure applies: if two points are in the same connected component, the line distance between them is the length of the shortest path. Otherwise, the distance is set to infinity.

Chapter 3

Preprocessing using extended features

As already mentioned, the extended features will be applied in all fingerprint processing stages. This chapter presents the application of those features for two tasks in preprocessing and enhancement: image segmentation and reconstruction of low quality areas.

3.1. Segmentation

A fingerprint image acquired by a sensor is composed of two parts: the fingerprint area (ridges with valleys) and the background. As features are contained only in the foreground, background and foreground have to be separated, which is the aim of *segmentation*; an example is shown in Fig. 3.1.

It is desirable that areas disturbed by noise as well as those devoid of clear fingerprint structure within the imprint should also be included in the background [28] since they contain no information and most feature extraction algorithms extract many spurious features in those regions.

If the image background were always uniform and lighter than the fingerprint area, a simple approach based on local intensity could be effective for discriminating between foreground and background; in practice, the presence of noise requires more robust segmentation techniques [7].

Several approaches to fingerprint segmentation have been presented in literature. Mehtre *et al.* [29] divide the fingerprint in blocks of 16×16 pixels. In each block the local ridge orientation is estimated for each pixel. The existence of a dominant orientation denotes an oriented pattern (the foreground), whereas a flat histogram of orientations is a characteristic of the background. Mehtre and Chatterjee [28] extended that method to include an additional block feature: a grey-scale variance. Ratha, Chen and Jain [30] classify each block according to the greyscale variance in the direction orthogonal to the block's global orientation. Jain, Ratha and Lakshmanan [31] use the output of a set of Gabor filters as the input to a clustering algorithm that constructs spatially compact clusters. Maio and Maltoni [16] discriminate between foreground and background by using the average magnitude of the gradient in each image block. Bazen and Gerez [21] propose a segmentation

(a) (b)

Figure 3.1. An acquired fingerprint image (a) and its marked background (b)

algorithm which uses three pixel features: gradient coherence, mean and variance in the pixel neighbourhood. An optimal linear classifier is trained for a pixel-wise classification, while morphology is applied as postprocessing to obtain compact clusters and to regularise the external silhouette of the foreground.

The methods described above use the fact that the foreground contains a clear structure of ridges and valleys. However, fingerprint images can also contain some ridge-like structures in the background such as a second finger, the phalanx impression, or remainders of old skin impressions from a previous acquisition, which results in false segmentations as presented in Fig. 3.2. In order to circumvent that problem, the extended feature set is used as described in the next section.

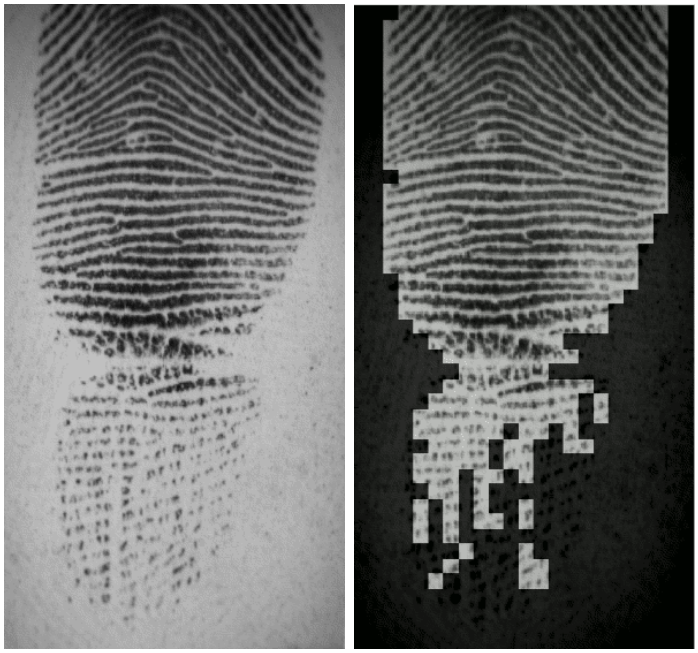
3.2. Segmentation using extended features

In order to correct common drawbacks of segmentation algorithms, we will evaluate additional information contained in the fingerprint structure extracted by the entracer. An input fingerprint image is segmented by a common segmentation algorithm and then the areas classified wrongly as foreground are identified and marked as background.

3.2.1. Additional features for segmentation. The following features are used as indicators for an area belonging to the background.

Average length of dual connected components. For each dual connected component d its average length

$$\bar{l}(d) = \frac{\sum_{s \in S(d)} l(s)}{|S(d)|}$$



(a) (b)

Figure 3.2. A fingerprint for which a segmentation algorithm has assigned a noisy area to the foreground

is being computed. $S(d)$ denotes the set of all segments in a component d, $|S(d)|$ is its cardinality, and $l(s)$ the length of segment s.

Experiments have shown that the average length of connected components is small in misclassified areas, which is a consequence of the increased number of spurious minutiae in such areas.

Number of unvalidated minutiae. As already mentioned, every singular point has a corresponding unvalidated minutia. Each remaining minutia has its corresponding minutia in the dual area. As there exist at most 4 singular points (2 cores and 2 deltas) in a fingerprint, a large number of unvalidated minutiae in a region is an indicator for the background.

Number of segments with inconsistent neighbours. In the disturbed area `A` in Fig. 3.3 the segment neighbours are not consistent, e.g. for segment 1 no dual neighbour has been found since all possible neighbours are too far away, whereas in the region `B` the line flow is well preserved and each segment has its dual neighbour.

3.2.2. Block averaging. Image points are assigned to the background in a block-wise manner. For descriptive features, their mean value in each block must be computed and quantity features must be summed up in each block.

The block average lengths for connected components are computed as follows. Let $L \in \mathbb{N} \times \mathbb{N}$ be a rectangular matrix with the same size as the fingerprint image. An element at position (i, j) in the matrix L is the average length of the component to which the segment passing through the point (i, j) belongs . If there is no such segment, the value is undefined. From the matrix L, the matrix B - the block average length matrix

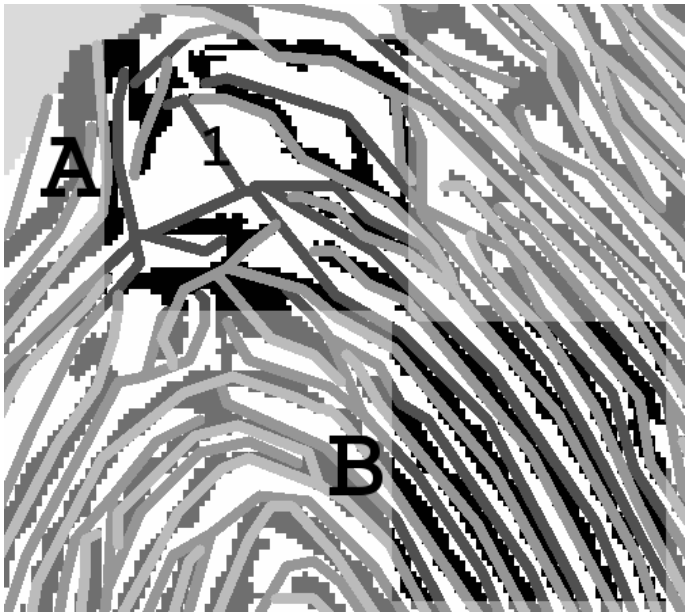


Figure 3.3. Low (A) and high (B) quality regions with extracted segments

- is computed. The matrix L is divided into blocks of a given size and the average of the defined values is computed for each block. An exemplary matrix B for connected components from Fig. 3.2(a) is plotted as a mesh in Fig. 3.4.

3.2.3. Identifying background. With the help of the features described above, a classification algorithm can be constructed for discriminating between background and foreground. Since the samples of features in both areas are available, we follow the supervised learning approach. Many different classification algorithms exist that can be applied to this problem. One can for instance think of $K-$nearest neighbour classifier, neural networks, among others to find decision boundaries. However, it is very important to use a classification algorithm that has as low a computational complexity as possible [21]. Hence we have decided to use a linear classifier:

$$y(c, m, s) = a_c c + a_m m + a_s s + b,$$

where

- c is the mean of the average dual connected component lengths in the block,
- m is the number of unvalidated minutiae in the block,
- s is the number of segments with inconsistent neighbours in the block.

And the decision for a block is:

$$\text{the block} \begin{cases} \text{remains foreground} & \text{if } y(c, m, s) \geq 0 \\ \text{changes to background} & \text{if } y(c, m, s) < 0 \end{cases}$$

The weights a_c, a_m and a_s are determined during a supervised learning phase with the help of images in which the background has been marked manually.

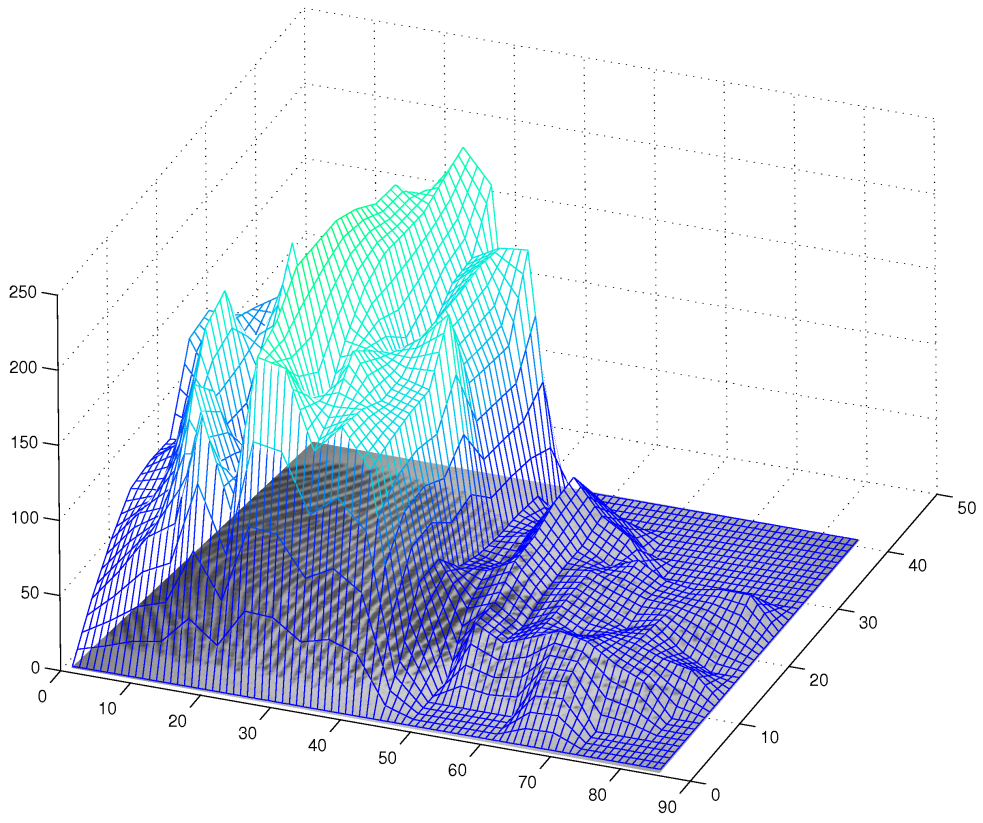


Figure 3.4. Average lengths of the dual connected components for the fingerprint from Fig. 3.2(a)

A sample result is presented in Fig. 3.5. The small area of the *second finger* in Fig. 3.5(b) has been falsely attached to the foreground since the area has ridge-valley structure which cannot be distinguished from a genuine foreground.

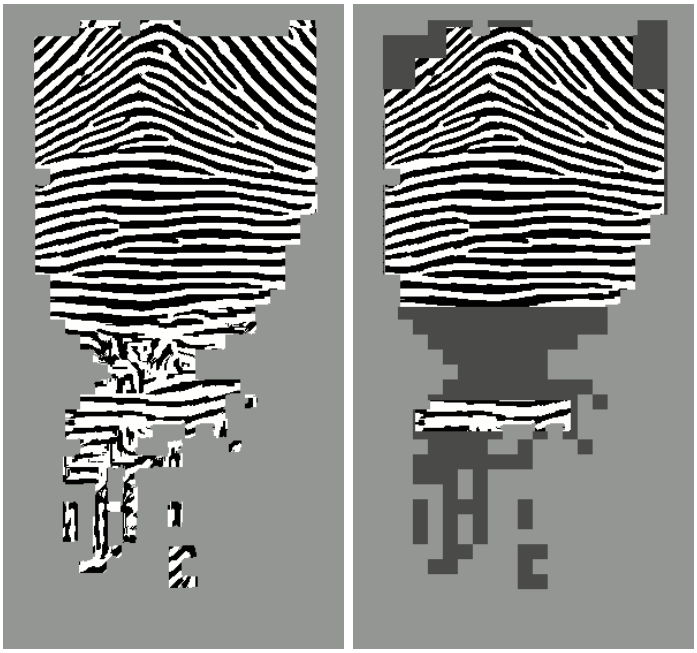
3.3. Low quality regions

Background, regarded as the area in which no information is contained, can be divided into:

- the *surrounding area* around the fingerprint,
- a *low quality region* within the foreground.

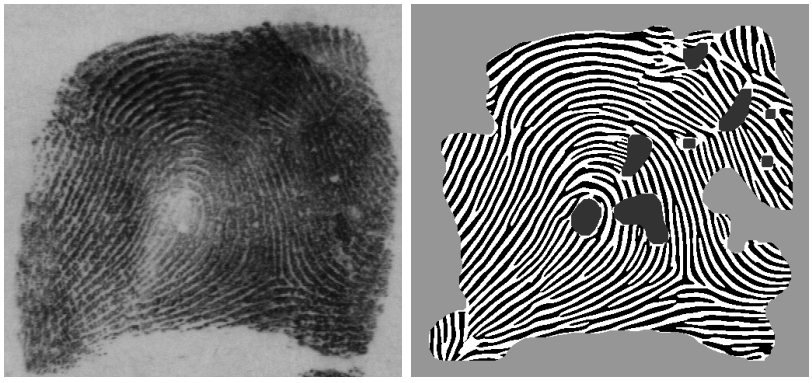
Since low quality regions are surrounded by non-disturbed foreground, a low quality region can be enhanced or even reconstructed with the help of the information contained in neighbourhood.

In order to distinguish between a surrounding area and a low quality region, the background must be divided into connected components. If there exists a point lying next



(a) (b)

Figure 3.5. The image from Fig. 3.2 binarised and segmented with commonly used method [21] and its improved segmentation



(a) (b)

Figure 3.6. A fingerprint image before and after binarisation and segmentation. The surrounding area is marked light grey and low quality regions - dark grey

to the image border in some component, the component is declared to be the surrounding area. Otherwise it is a low quality region. An example is shown in Fig. 3.6.

3.3.1. Connecting segments. In Section 2.4.2 border point neighbours have been introduced. Like a real border point each *noisy point* (the end of a segment at a low quality area border) knows its neighbours if they exist. With the help of that information, and additionally using the neighbouring non-disrupted segments, the trace lines can be connected: from the outermost line towards the centre of the noisy area. The result for a low quality region which has an equal number of segment ends at both sides (Fig. 3.7(a)) is

(a) Low quality area with the same number of segment ends at both sides

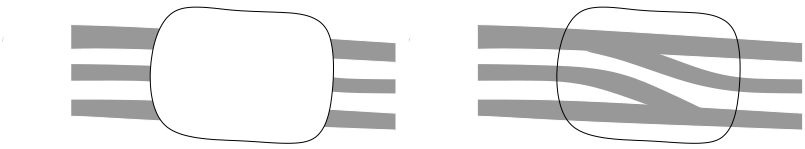
(b) After connection of disrupted segments

(c) Low quality area with different number of segment ends at both sides

(d) After simple connecting. One line could not be connected

(e) After identifying the minutia

Figure 3.7. Connecting segments



(a) (b)

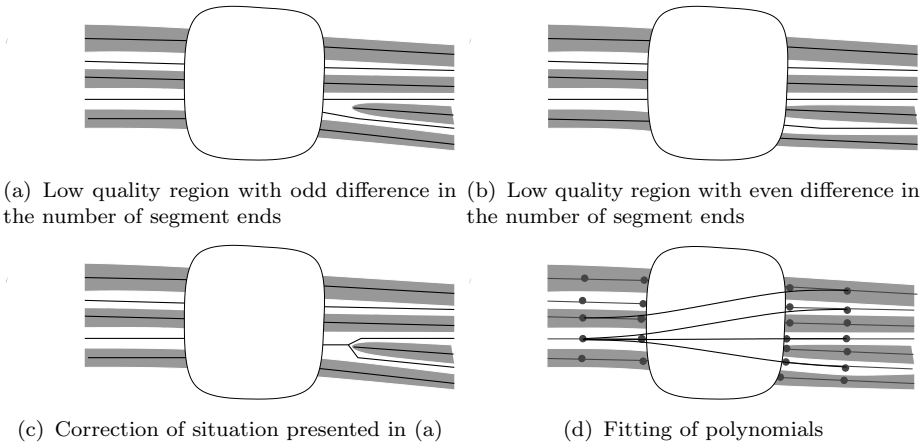
Figure 3.8. Hidden minutiae in a low quality region

presented in Fig. 3.7(b). An example for a region with different numbers of segment ends at both sides is shown in Fig. 3.7(c) and the result of the line connecting is presented in Fig. 3.7(d). Different numbers of line ends at both sides of a low quality area are an indicator for the existence of one or more minutiae in that region. A method for finding the approximate position and direction of the minutiae is described in the following section.

Obviously, if a noisy region does not contain any minutia, the number of segment ends at one of its sides is equal to the number at the other side. However, this statement cannot be reversed. Even if the numbers at both sides are equal, there might exist minutiae in the low quality region as presented in Fig. 3.8. Such minutiae cannot be reconstructed by our algorithm and will be missing.

3.3.2. Minutiae in low quality regions. In this subsection we will consider a situation in which the numbers of segment ends at both sides of a low quality area differ. For what follows we will assume, without loss of generality, that the left side has less end points than the right side.

The following two situations are possible:



(a) Low quality region with odd difference in the number of segment ends

(b) Low quality region with even difference in the number of segment ends

(c) Correction of situation presented in (a)

(d) Fitting of polynomials

Figure 3.9.

(1) There exist two neighbouring ends having same colour (Fig. 3.9(a)), i.e. both are black or both are white.

(2) Each neighbour of an end, if one exists, lies in the dual area (Fig. 3.9(b)), so that the difference in the numbers of ends at both sides is even.

In the first case, any two neighbouring ends lying in the same dual area are connected to make a bifurcation and consequently the number of ends at that side decreases by one. The result of connecting the ending for the situation presented in Fig. 3.9(a) is shown in Fig. 3.9(c). After processing of all such neighbouring endings the second condition is fulfilled. In that case the strategy is as follows.

In order to find likely connections between the disrupted segments, we use the fact that the flow of fingerprint lines is very smooth. We thus try to fit a polynomial curve to points at the ends of the disrupted segments. The connections with the best fit are chosen. Sample fits are presented in Fig. 3.9(d).

We use curves of degree 2 that can be written in the parametric form as

$$l(t) = \left(a_2t^2 + a_1t + a_0, b_2t^2 + b_2t + b_0\right)$$

For the approximation four points will be chosen, two for each segment. On each segment, one point is the end and the second is chosen at a given distance from the end. The residual sum of squares defines the cost of the connection. The lines should be connected in a way such that the overall cost is minimised and all lines are connected. If a segment end has been connected to two other segment ends, a bifurcation is placed in the middle between the left and the right side of the low quality area since the exact position cannot be determined. The corrected connections for the situation presented in Fig. 3.7(c) are shown in Fig. 3.7(e).

Chapter 4

Classification

4.1. Introduction

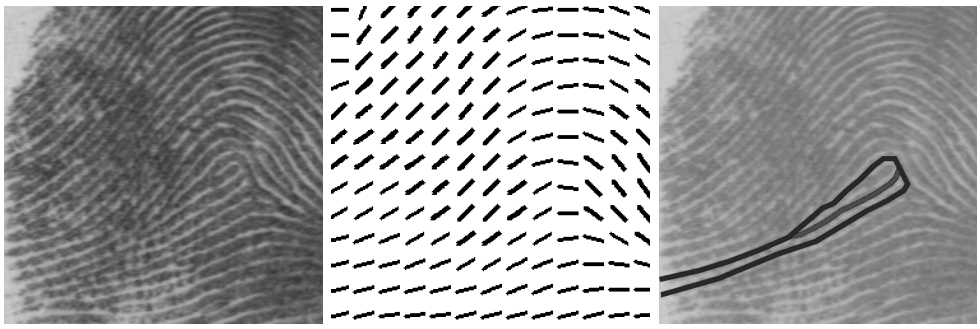
This chapter discusses applications of the extended feature set for another fingerprint recognition stage – classification.

Fingerprints have specific flow dynamics which manifest themselves in relatively distinct flow patterns – these help to define *classes* of fingerprints. The first classification of fingerprints was made by Purkinje in 1823. The most important and widespread one is Sir Edward Henry's classification, designed in 1902 as a refinement of the earlier classification system by Galton. Most of the currently used classification schemes are variants of Henry's classification scheme [7]. The six most common classes, shown schematically in Fig. 4.2, are the following: *plain arch, tented arch, left loop, right loop, whorl,* and *double loop.*

Whereas fingerprint matching is predominantly performed with the help of local features, fingerprint classification is generally based on global features, that is, a skin ridge flow direction and singular points. The orientation field can be used as a feature vector for classification [32; 33; 34]. Different procedures for a reduction of the dimension of the orientation field, e.g. Karhunen-Loève transformation [35] have been proposed. The statistical classifiers may then be applied to the reduced vector [36; 15].

Dass and Jain [37] have proposed the utilisation of the ridge shape. However, they actually use only level 1 features since they extract the shapes not directly from the ridge image but from the computed orientation field, which can lead to misclassification of fingerprints containing very few indicators for their class. For the left loop presented in Fig. 4.1(a)) the computed orientation field (Fig. 4.1(b)) does not contain any indicator for a left loop as the loop has been smoothed out. Consequently, the fingerprint is misclassified as an arch, whereas a flow line indicating a left loop can be found within the flow lines which are directly extracted from the ridge segments (Fig. 4.1(c)).

One of the main objectives of this thesis is to abandon the classical approach which distinguishes between the global and the local features. Some investigations aimed at finding alternatives to level 1 features have already been presented in literature. Chang and Fan [38] have proposed a combination of global ridge shape analysis – leading to an auxiliary classification of 10 ridge patterns – and the recognition of syntactic grammar languages with a non-deterministic finite automaton. Cho *et al.* [39] have proposed a method

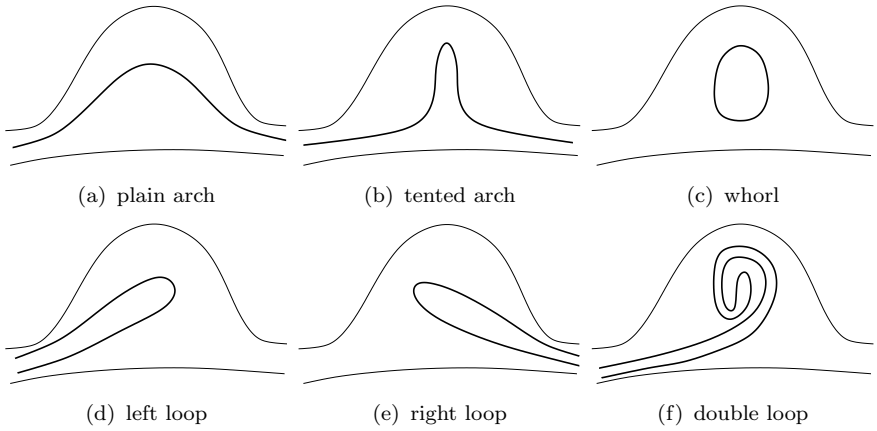


(a) A left loop image with a small class indicator

(b) Orientation field, note that the loop was smoothed-out

(c) Class indicator from the extended feature set

Figure 4.1. Small loop challenge



(a) plain arch (b) tented arch (c) whorl

(d) left loop (e) right loop (f) double loop

Figure 4.2. Characteristic lines for fingerprint classification

that classifies fingerprints according to the curvature and orientation of the fingerprint area near the core.

Principally, our method is based on the ridge flow extracted directly from the ridge segments. In order to deal with noisy or partial images, we will explore additional features, such as the ridge connectivity or a flow mode of ridges.

4.2. Classification

Each of Henry's classes has a certain winding of connected lines which are class-specific. Typically, they can be found around the centre of a fingerprint (see Fig. 4.2). This suggests that with the help of the connected entraced lines, a characteristic line for that class may be found. In order to determine a class it is sufficient to find one or, for robustness, a few such lines.

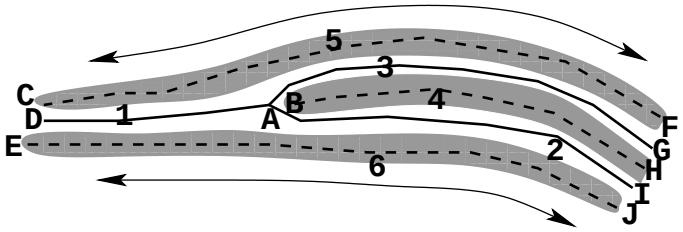


Figure 4.3. Extended line identification process

4.3. Extended lines

The obtained fingerprint structure (extracted minutiae, border points, and segments) can be considered as two graphs, using the definitions introduced in Section 2.6.1:

$$G_{\mathcal{M}} := (V_{\mathcal{M}} := V_M, E_{\mathcal{M}} := E_M \cup E_D)$$

and

$$G_S = (V_S, E_S),$$

where:

- $V_{\mathcal{M}}$ is the set containing all minutiae and border points,
- $\{v_1, v_2\} \in E_{\mathcal{M}} \iff v_1$ and v_2 are connected via a segment or correspond dual to each other. For the situation presented in Fig. 4.3, in addition to the "natural" edges $\{\mathtt{A},\mathtt{D}\}, \{\mathtt{A},\mathtt{G}\}, \{\mathtt{A},\mathtt{I}\}, \{\mathtt{B},\mathtt{H}\}, \{\mathtt{C},\mathtt{F}\}$, and $\{\mathtt{E},\mathtt{J}\}$ also the edge $\{\mathtt{A},\mathtt{B}\}$ is contained in the edge set.
- V_S is the set of all segments, and
- $\{v_1, v_2\} \in E_S \iff$ the segments v_1 and v_2 are neighbours. For the segments in Fig. 4.3, $E_S = \{\{1,5\},\{1,6\},\{2,4\},\{2,6\},\{3,4\},\{3,5\}\}$.

An **extended line** λ is a path in the graph $G_{\mathcal{M}}$:

$$\lambda = \langle v_1, e_1, v_2, \ldots, e_{m-1}, v_m \rangle$$

which connects two border points v_1, v_m with the additional condition that, in order to respect the natural flow, bifurcations are traversed only by following their larger angles. Hence, the only extended lines in Fig. 4.3 are: $\langle \mathtt{C}, 5, \mathtt{F} \rangle$, $\langle \mathtt{D}, 1, \mathtt{A}, 3, \mathtt{G} \rangle$, $\langle \mathtt{D}, 1, \mathtt{A}, \mathrm{e}_{\mathtt{A}-\mathtt{B}}, \mathtt{B}, 4, \mathtt{H} \rangle$, $\langle \mathtt{D}, 1, \mathtt{A}, 2, \mathtt{I} \rangle$, $\langle \mathtt{E}, 6, \mathtt{J} \rangle$, where $\mathrm{e}_{\mathtt{A}-\mathtt{B}} \in E_D$ denotes the dual link between $\mathtt{A}$ and $\mathtt{B}$. Moreover, we can equivalently use another representation of an extended line – its pointwise representation:

$$\lambda_p = \langle p_1, p_2, \ldots, p_n \rangle$$

where $p_1 \ldots p_n$ are the segment points contained in the edges $e_1 \ldots e_{m-1}$.

We denote the set of all extended lines contained in a fingerprint with Λ.

4.3.1. Extended line properties. An extended line can be characterised by two descriptors: *maximal change of direction* and *flow mode* defined, as follows:

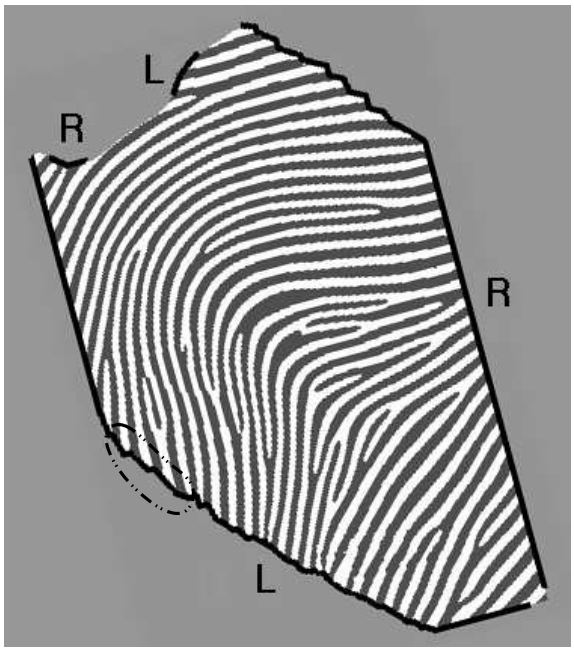


Figure 4.4. Determination of the border point side location

Maximal change of direction. The cumulative direction change of a segment defined in Section 2.5.4 can naturally be extended to extended lines.

Definition 4.1 (Cumulative change of direction)**.** *For an extended line* $\lambda_p = \langle p_1, p_2, \ldots, p_n \rangle$ *the* **cumulative direction change** $\phi_\lambda(i)$ *up to segment point* p_i *is the sum of direction changes at segment points* $p_2, \ldots, p_{i-1}$*:*

$$\phi_\lambda(i) := \begin{cases} \phi_\lambda(i-1) + \alpha_i & \text{for } 2 < i \leq m \\ 0 & \text{for } i \in \{1, 2\} \end{cases}$$

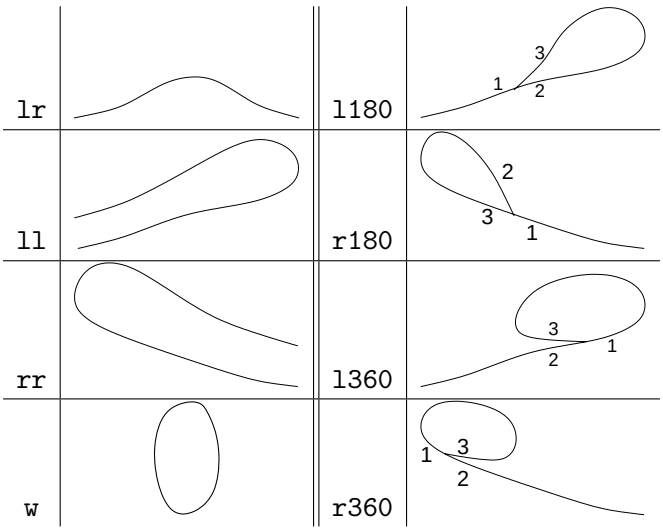
where $\alpha_i \in (-\pi, \pi]$ *is the change of direction at point* p_{i-1}*.*

Definition 4.2 (Maximal change of direction)**.** *For an extended line* λ *the* **maximal direction change** $\Phi(\lambda)$ *is defined as follows:*

$$\Phi(\lambda) := \max_i(\phi_\lambda(i)) - \min_i(\phi_\lambda(i))$$

4.3.1.1. *Flow mode.* Flow mode describes the shape of a line on a global scale. The information about the side location (on the left or the right side of a fingerprint) of border points required for the computation of flow modes will be introduced first. For fingerprint images unaffected by a rotation, i.e. being aligned vertically, the value of the second component of a border point direction vector is a good indicator for its side location. If it is smaller than 0 (the direction vector points to the left), then the border point is located on the right side, otherwise on the left. However, that method can fail if the finger had been rotated slightly before it was acquired. The border points in the marked area in Fig. 4.4 can then be falsely classified as left border points. However, since every border point knows its neighbour (if it has one) as described in Section 2.4.2, we can connect the neighbouring border points together in order to build larger components. For each component its mean direction is utilised for a side assignment.

Table 4.1. Flow modes



With the help of the image sides characteristic lines can be divided into the following classes, illustrated in Table 4.1:

- connecting two sides of the image (`lr`);
- having both ends at the same image side (`ll`, `rr`);
- closed line without ends (`w`);
- having one end at an image side and the other end within the line (`l180`, `r180`, `l360`, `r360`).

The 180° and 360° loops are discriminated by using the branches order of the bifurcation (introduced in Section 2.2.2) where the loop is connected. If the loop is between the second and the third bifurcation branch, it is a 180° loop. Otherwise, if the loop is connected at the first branch, it is a 360° loop.

We introduce the set of all possible flow modes

$$F = \{\texttt{lr}, \texttt{ll}, \texttt{rr}, \texttt{w}, \texttt{l180}, \texttt{r180}, \texttt{l360}, \texttt{r360}\},$$

and let $\Gamma : \Lambda \to F$ be the function which returns the flow mode for an extended line.

4.4. First decisions

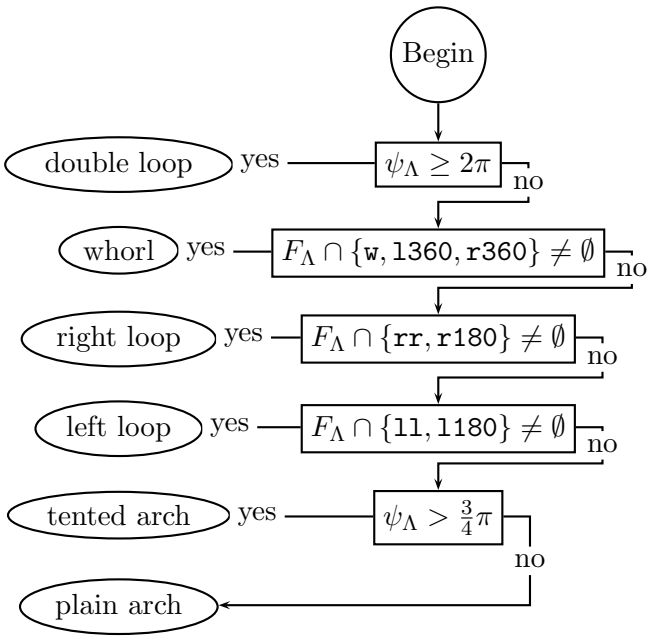
If a good quality fingerprint image contains an extended line with flow mode `w` it is clear that the fingerprint belongs to the whorl class. Similarly, if it has only `lr` extended lines and none of them has a maximal direction change larger than $\frac{\pi}{2}$, it is clearly a plain arch. Hence the classification of good quality images can be achieved with the help of the tree classifier shown in Table 4.2, where $F_\Lambda \subset F$ is the set of all flow mode among extended lines in the set Λ,

$$F_\Lambda := \{\Gamma(\lambda) : \lambda \in \Lambda\}$$

and ψ_Λ is the largest maximal direction change in Λ,

$$\psi_\Lambda = \max_{\lambda \in \Lambda} \Phi(\lambda)$$

Table 4.2. First decision tree



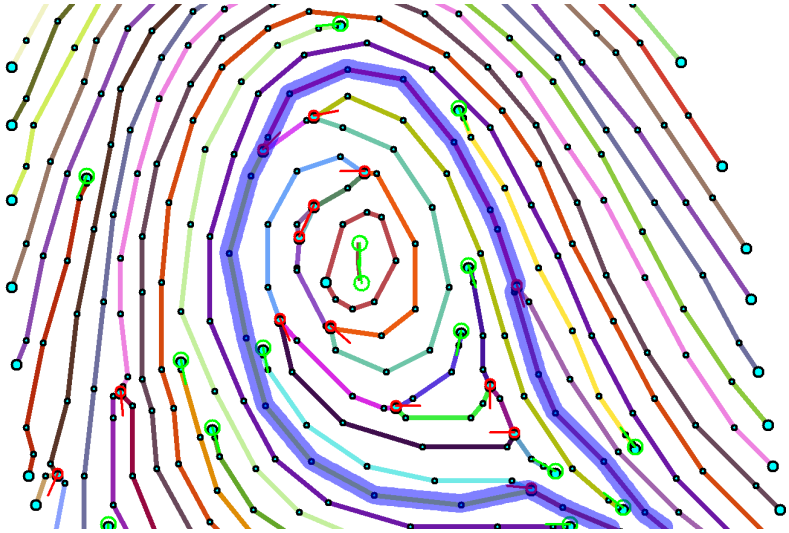
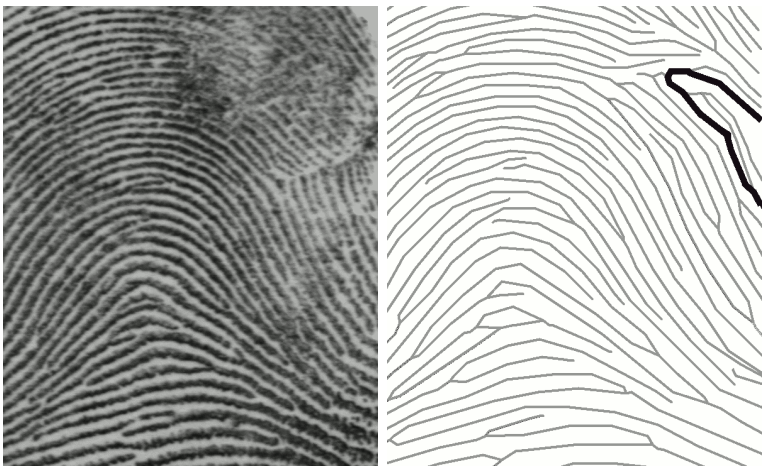


Figure 4.5. A sample characteristic line in a right loop fingerprint

An extended line which has fulfilled one of the five conditions resulting in the assignment of the fingerprint to a class other than plain arch is called a **characteristic line**. A sample characteristic line is presented in Fig. 4.5.

The classification described above works well for good quality fingerprint images. However, in case of a noisy image a characteristic line may not be found or a spurious characteristic line may be extracted (Fig. 4.6), which results in a false classification. Thus, additional features are required for images of lower quality.



(a) Original impression (b) Extracted lines; the bold line leading to the misclassification

Figure 4.6. False `rr` line introduced by noise to a plain arch class image; it results in a false classification as a right loop when the tree classifier from Table 4.2 is used

4.5. Additional features

4.5.1. Connected component graph. In Fig. 4.2 we can observe another important property of characteristic lines. Characteristic lines lie between the fingerprint top, which contains only `lr`-extended lines with maximal orientation change about $\frac{\pi}{2}$, and the fingerprint bottom which also contains only `lr`-extended lines but with the maximal direction change close to 0. This observation helps to eliminate spurious characteristic lines. For example, in case of the spurious characteristic line in Fig. 4.6(b), taking advantage of the fact that the lines below the characteristic line have a maximal direction change around $\frac{\pi}{2}$, the line can be discarded for further processing and a correct classification is obtained.

In order to capture and evaluate these properties, we introduce the graph of dual connected components G_C built up from dual connected components[1] of the graph $G_{\mathcal{M}}$:

$$G_C = (V_C, E_C),$$

where:

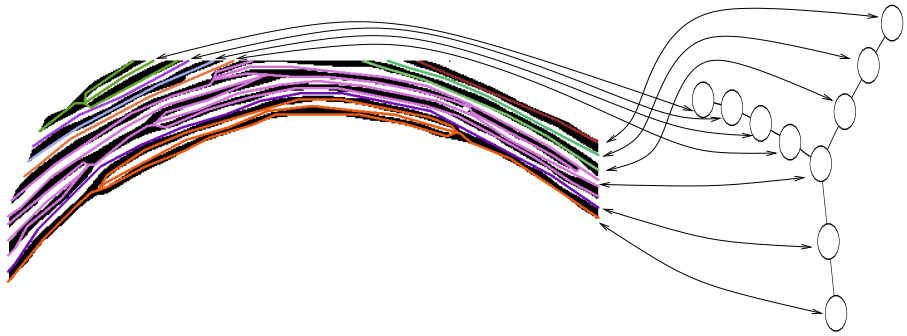
- V_C is the set of all dual connected components,
- $\{v_1, v_2\} \in E_C \iff$ one of the segments in v_1 is a neighbour of a segment in v_2.

In Fig. 4.7 an example for building connected components and the corresponding connected components graph is shown.

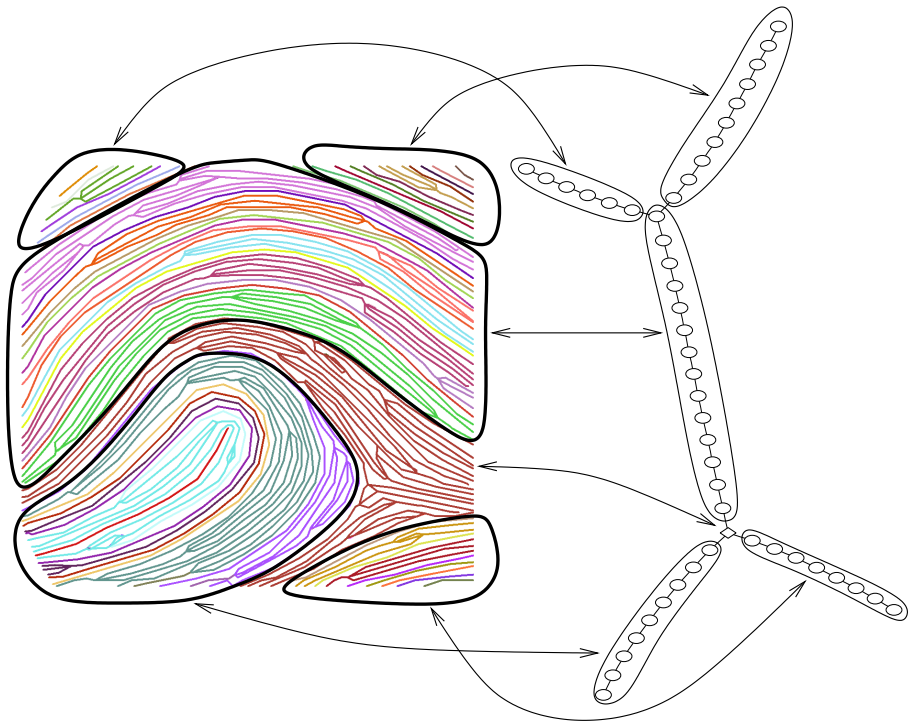
We will adapt the functions for *maximal change of direction* (Φ) and *flow mode* (Γ) for use with connected components. Consequently, $\widetilde{\Phi} : V_C \to \mathbb{R}$ returns the largest maximal direction change of all extended lines in a component, whereas $\widetilde{\Gamma} : V_C \to \mathcal{P}(F)$ provides all flow modes of all extended lines contained in a component.

The graph G_C is simplified with the following contraction rules:

[1] For the definition of dual connectivity see Section 2.6.3.



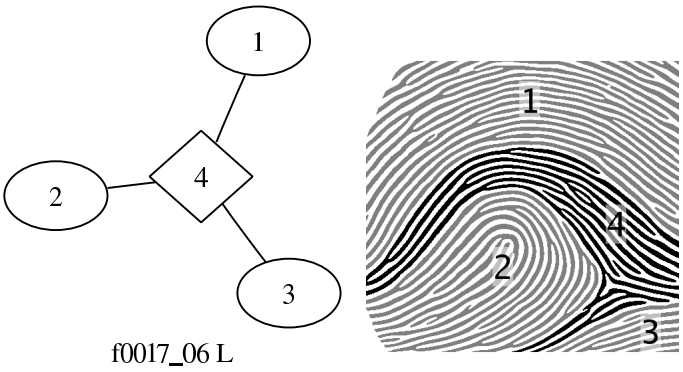
(a) For a part of a fingerprint.



(b) For a whole fingerprint

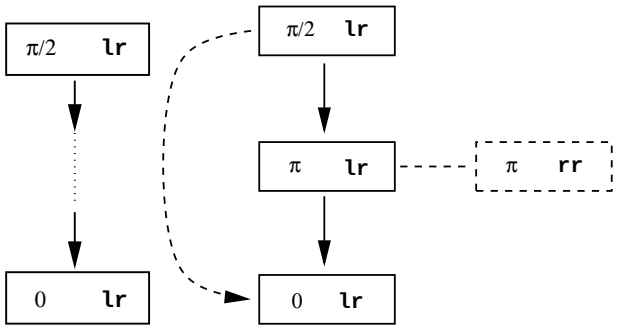
Figure 4.7. Dual connected components and a dual connected component graph with marked correspondences. Each dual connected component is marked with a different colour.

- Vertices which correspond to components not containing any extended line (e.g. containing only one line with two unvalidated endings) are removed from the graph.
- A vertex with degree 1 is merged with its neighbour if they are compatible, that is, they have the same flow modes and the difference in maximal direction change is below a given threshold.
- A vertex with degree 2 is deleted and its neighbours are connected if all 3 vertices are compatible.



(a) Reduced connected component graph (b) Corresponding regions

Figure 4.8.



(a) For an arch (b) For a right loop

Figure 4.9. Sample graph prototypes, dashed parts are optional

The result of this postprocessing step applied to the connected component graph from Fig. 4.7 together with the corresponding regions is presented in Fig. 4.8.

The postprocessed graph contains four vertices:

(1) component with `lr` flow mode and maximal direction change $\frac{\pi}{2}$;

(2) component with `ll` flow mode;

(3) component with `lr` flow mode and maximal direction change near 0;

(4) component with `lr` and `ll` flow modes, adjacent to all other components. It is also the component containing the delta point.

Consequently, prototypes for each class are constructed. The sample prototypes for an arch and a right loop are shown in Fig. 4.9. The dashed parts in Fig. 4.9(b) indicate optional parts of the graph as there can exist either a connection between the fingerprint top and bottom or the central part can be merged with the component containing delta but either one must be present.

An extracted characteristic line can be validated by comparing the obtained reduced connected component graph with the above proposed prototypes.

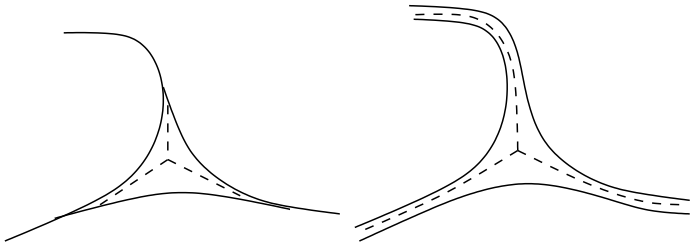


Figure 4.10. Different delta architectures

4.5.2. Delta points. For a fingerprint the number of core points always equals the number of delta points: for tented arches, right or left loops it equals 1, for whorls and double loops there are 2 while plain arches do not posses any.

Although delta points are not reliable features since they often lie outside of the visible fingerprint area, our experiments have shown that the classification error rate decreases when delta point information is used in the following way: if two deltas have been found in a fingerprint it is declared to be a whorl or a double loop; if no characteristic line but one delta can be found, it is most probably a tented arch.

We will utilise only delta point information and no core point information because of the following reasons:

(1) Generally, if no characteristic line can be found in a noisy fingerprint which is not a plain arch, a core point will also not be found.

(2) Delta points can easily and reliably be computed with the help of extended lines and a computation of orientation field is not required.

Deltas exist in various types. The easiest one to detect is presented in Fig. 4.10(a). Two attributes of such a delta are:

- there exists a triangle build up of three bifurcations,
- the second and the third branch of the uppermost bifurcation are adjacent to the segments having their other ends at different image sides.

The most difficult delta architecture is presented in Fig. 4.10(b). Since there is no bifurcation triangle, the above listed conditions cannot be applied directly. The best solution is to utilise neighbourhood information. Another indicator for a delta point is the existence of a star-like unvalidated bifurcation with all three angles around 60°.

4.6. Final decisions

The final decision tree for classification is presented in Table 4.3 using the first classifier presented in Table 4.2.

4.7. Results and conclusion

We have tested our method on the NIST Special Database 4 [41]. This database contains 4000 fingerprint images acquired from 2000 fingers. The fingerprints are equally distributed over five classes assigned by human expert: plain arch, tented arch, left loop, right loop,

Table 4.3. Final decision tree

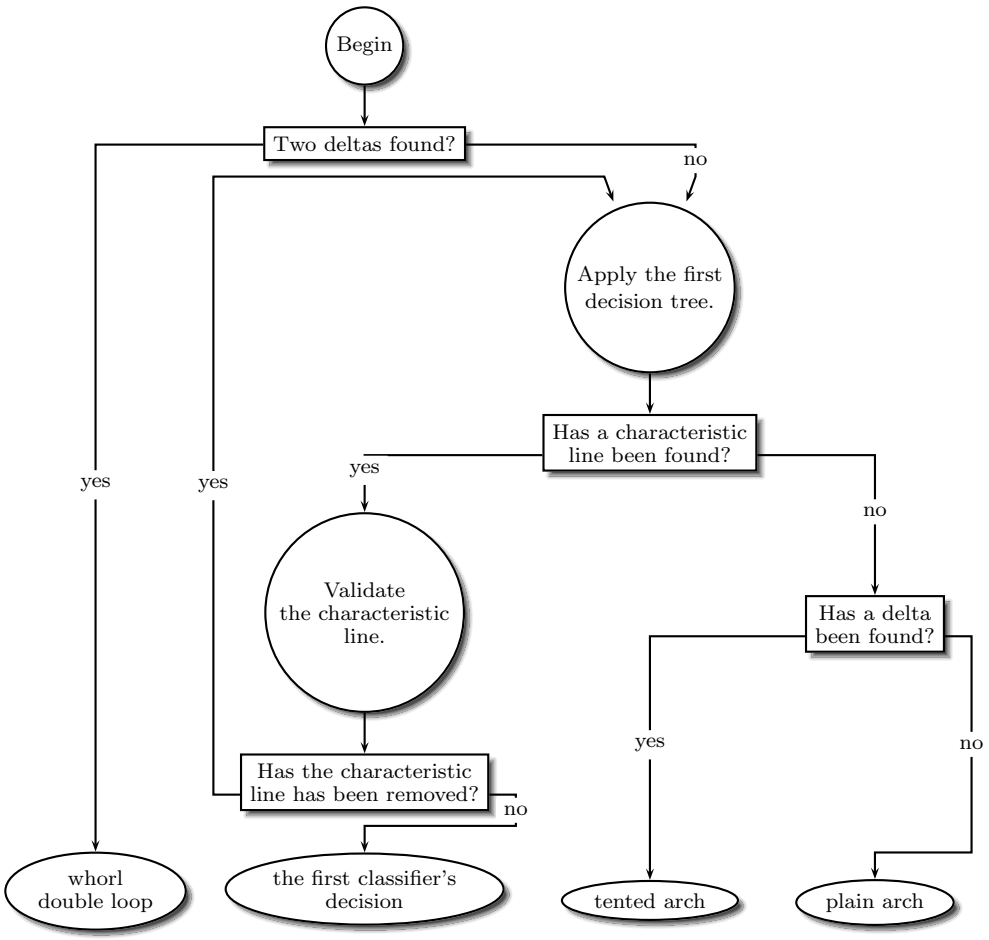


Table 4.4. Comparison of error rates

	4 class problem	5 class problem
Dass and Jain [37]	6.6%	—
Li, Yau, and Wang [40]	5%	6.5%
our method	4.5%	6.3%

Table 4.5. Confusion matrix of the results on the NIST 4 Database

true class	predicted class PA	TA	LL	RL	W
PA	**751**	35	15	12	0
TA	39	**747**	16	9	7
LL	8	6	**763**	11	10
RL	12	10	6	**733**	12
W	8	7	14	16	**753**

and whorl. The whorl class contains fingerprints from both the whorl and the double loop class.

The performance of a fingerprint classification system is usually measured in terms of *error rate* which is the ratio between the number of misclassified fingerprints and the total number of samples in the test set [7]:

$$\text{error rate} = \frac{\text{number of misclassified fingerprints}}{\text{total number of fingerprints}} \times 100\%$$

Since the discrimination between a plane and a tented arch is a difficult task, we have also computed, as many researches do, the error rate for the four class problem, in which plane and tended arches are identified as a single class. For the four class problem we obtained 3.5% error rate, and 6.6% for the five class problem. A comparison with error rates from the literature is presented in Table 4.4.

A more detailed analysis of the behaviour of a classifier can be obtained by examining the *confusion matrix* which has one row for each true class and one column for each predicted class; each cell at row r and column c reports how many fingerprints belonging to class r are assigned to class c [7]. Table 4.5 shows the confusion matrix of the results on the NIST 4 database for the presented approach. As even experts have problems assigning a given fingerprint to a class, some fingerprints in the database had been assigned to two classes. In such a case, a fingerprint is concerned as classified correctly if it has been assigned to one of the two classes. For building the confusion matrix we report an image as falling into the class we assigned in such a case.

In the remaining part of the section we will discuss misclassifications which occurred and the reasons.

Fingerprint classification is a difficult task due to small inter-class but large intra-class variability (Fig. 4.11). Furthermore, in a non-negligible series of images even a trained expert may have difficulty in determining a unique class in which to classify a given fingerprint so that multiple class decisions are a possible solution.

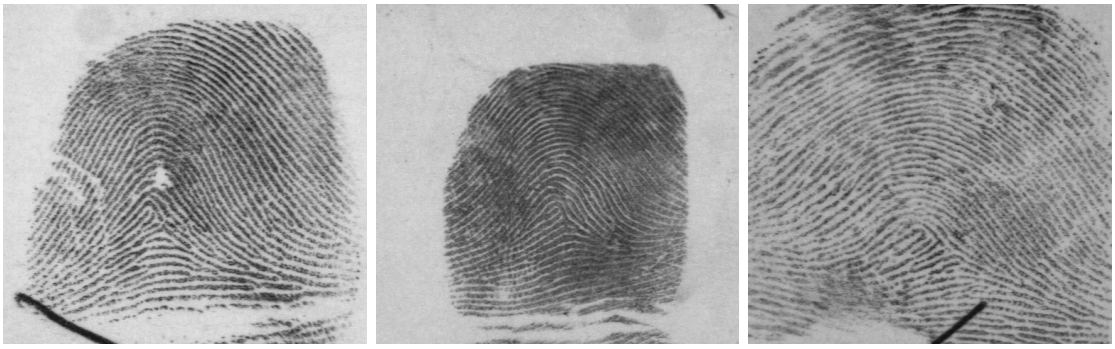
In spite of the multiple class decisions, one can find in the NIST 4 database sets of images in which a visual confrontation may render it difficult to follow the expert's decision (Fig. 4.12), for example where a delta was identified in an image which appears to show a plain arch (Fig. 4.13(b)) or where an apparent left loop appears to have been classified as a tented arch only because it is so acute (Fig. 4.13(c)). The distinctions are blurred and, unless one trains an algorithm on a given database, there is no apparent *a priori* criterion which might enable the algorithm to duplicate the expert's decisions. In order to construct a database independent method, we have decided to reject such a training.

We have been confronted with considerable problems when trying to distinguish between the two arch classes (Fig. 4.13(a) and 4.13(b)) as well as in case of the distinction between whorls and some loops, mostly due to the presence of small pocket whorls. Moreover, the quality of images is quite an important factor. Some low quality images have their discriminating features removed in the enhancement stage (Fig. 4.14(b) and 4.14(c)), whereas some are difficult to classify even for a human (Fig. 4.14(a)).

Since the tests have proven the extended features' usefulness for classification, it might be worth applying them for indexing. Last but not least, the refinement of the existing classes, e.g. by dividing a class into subclasses by means of the number of characteristic lines or their curvature appears to be worth a deeper examination.

(a) plain arch (b) left loop (c) left loop (d) left loop (e) tended arch

Figure 4.11. An example of small inter-class and large intra-class variability.



(a) `s1936_07` TA (b) `f0008_10` TA LL (c) `f0045_06` LL

Figure 4.12. Contentious assignments to the classes proposed by an expert. We would suggest to at least assign all image as TA LL. `s1936_07` was horizontally mirrored for better comparison.

(a) `f0791_03` PA (b) `f1070_07` TA (c) `f0401_08` TA

Figure 4.13. Fingerprints and the expert's decisions

(a) f1463_02 TA (b) f0310_05 W (c) f0310_05 W

Figure 4.14. Low quality images

Chapter 5

Matching

5.1. Introduction

In general, matching two fingerprint images is a difficult task, mainly due to a great dissimilarity of impressions of the same finger. The main factors for those large variations are summarised below [7].

- Translation: the finger may be placed at different locations on the sensor. Although a well-designed sensor housing can prevent large finger displacements, even a 2 mm finger translation on the sensor results in a translation of about 40 pixels in the acquired image while scanning at a resolution of 500 dpi. For an example compare Fig. 5.1(a) and 5.1(d).
- Rotation: the finger may be placed at different angles relative to the sensor. Like in the translation case, even using a well-designed sensor cannot completely avoid rotations and as a result a rotations of about $\pm 20^\circ$ can be observed. An example of relatively rotated fingerprint acquisitions is shown in Fig. 5.1(a) and 5.1(c).
- Partial overlap: the above mentioned linear distortions can result in a small overlap between the acquired images. The common part of the impressions presented in Fig. 5.1(d) and 5.1(e) covers only about 30% of the foreground in each fingerprint.

As mentioned in Section 1.3, the commonly used fingerprint sensors are touch-based, that is a finger has to be pressed onto the sensor surface. As a result additional factors resulting in a large intra-finger variability come into play:

- Noise: finger pressure, dryness of the skin, sweat, dirt, grease and humidity in the air may contribute to non-uniform contact with the sensor surface. As a consequence, the acquired images are noisy.
- Skin elasticity: may contribute to distortions at all levels. On the global level it may induce non-linear deformations of the ridge flow (Fig. 5.2). On the local level it may lead to change of minutiae type from an ending to a bifurcation (or the other way round), as presented in Fig. 5.2 within the region marked in red.

(a) 10_1 (b) 10_2 (c) 10_3

(d) 10_4 (e) 26_1 (f) 18_6

Figure 5.1. Sample fingerprint impressions from the FVC2002 database db2 [42]

In this chapter we will first investigate the quality of minutiae extracted by the entracer using a standard minutiae-based matching algorithm. The minutiae-based matching algorithm is then extended to include some of the extended features. Finally, we will present a sketch of a matcher which fully utilises all the extracted features.

(a) (b)

Figure 5.2. Two impressions of the same finger affected by non-linear deformations

5.2. Measuring the performance of a fingerprint-based identification system

In order to compare two fingerprint-based identification systems each system can be run on a given test database and error rates can be computed and compared. This section provides the details of this process.

When testing the similarity of two input fingerprints, an automatic identification system provides us with a score $t \in [0, 1]$. The input fingerprints are considered as genuine, if t is greater than a given threshold t_0.

In order to measure the performance of an identification system, we introduce the following notation:

- F: set of fingerprints in the test database
- $F_g := \{\{f_1, f_2\} \in F \times F : f_1 \text{ and } f_2 \text{ come from the same finger}\}$: set of genuine pairs
- $F_i := \{\{f_1, f_2\} \in F \times F : f_1 \text{ and } f_2 \text{ come from different fingers}\}$: set of impostor pairs
- $I : F \times F \to [0, 1]$: a matcher

For a given threshold t_0 we can compute two types of errors:

- false reject rate: for a given threshold t_0 the percentage of genuine pairs which have falsely been recognised as non-matching

$$FRR(t_0) = \frac{|\{\{f_1, f_2\} \in F_g : I(f_1, f_2) < t_0\}|}{|F_g|}$$

- false acceptance rate: for a given threshold t_0 the percentage of impostor pairs which have falsely been recognised as matching

$$FAR(t_0) = \frac{|\{\{f_1, f_2\} \in F_i : I(f_1, f_2) > t_0\}|}{|F_i|}$$

To allow a simple and quick comparison of two systems, we can compute their equal error rates – the rates at which both false reject rate and false acceptance rate are equal ($FRR(t) = FAR(t)$). The lower the equal error rate, the higher the accuracy of the biometric system.

As test databases we have chosen two databases from the FVC databases: db2 from FVC2002 [42] and db3 from FVC2000 [43]. All FVC databases contain impressions provided by volunteers. Since the acquisition conditions and the volunteer crew were different for each database, the difficulty level between the databases varies. The easier database db2 from FVC2002 has been acquired with the help of a good quality sensor which offers a relatively large acquisition area, whereas database db3 from FVC2000 is a more difficult one as the volunteers' age is spread between 5 and 73, one-sixth of the volunteers being under 7 and one-third over 55. Furthermore, complete overlap between consecutive images taken from a given volunteer was prohibited.

Each database comprises 100 fingers with 8 impressions for each finger (i.e., it contains 800 fingerprint images). We have computed equal error rates by matching the fingerprint impressions according to the FVC protocol [43]: each image in a database is matched against the remaining images of the same finger to compute the false rejection rate and the first image of each finger is matched against the first image of the remaining fingers in the database to compute the false acceptance rate. The described protocol results in 2800 genuine and 4950 impostor pairs for each database and algorithm.

5.3. Minutiae-based matching

The most widespread methods for fingerprint matching are minutiae-based. The extracted minutiae are stored as points on the plane. In order to compute a similarity score of two fingerprint images, the optimal alignment between the template and the input minutiae sets has to be found. To align them the minutiae sets are rotated and translated in order to compensate for linear distortions. To compensate for non-linear distortions, rectangles around minutiae are defined and consequently two minutiae are considered as matching if one falls within the surrounding rectangle of the other. However, such a method cannot compensate for all non-linear distortions and additionally will increase the false match rate.

In order to check the quality of the extracted minutiae, we have taken a commercial extractor developed by Neurotechnologija [44] (VeriFinger SDK, version 5.0). We have extracted minutiae with both the entracer and with VeriFinger's extractor. Then, we

Table 5.1. Equal error rates for minutiae matching

	Neurotechnologija	Entracer
FVC2002 db2	1.056%	0.953%
FVC2000 db3	7.692%	7.253%

have used the matcher from the VeriFinger SDK to match the extracted minutiae sets[1]. The results are presented in Table 5.1.

These tests show that the error rates are reduced even when no additional extracted features are used. In the next step, we extended the matching algorithm so that it exploits the advantages of some of the extended features.

5.4. Consistency-based matching

First, the extracted minutiae are matched with the help of a minutiae-based matching method. Next, the following consistency test is applied.

Let $G_M = (U, E_M, E_D)$ and $H_M = (V, F_M, F_D)$ be the extracted fingerprint graphs (see Section 2.6.1) for the fingerprints to be matched.

Let M be the set of all matched minutiae pairs

$$M := \{(u_i, v_i)\}_{i=1}^{n} \subset U \times V$$

M is a relation between the elements of the set U and the set V. Let $U_m \subseteq U$ and $V_m \subseteq V$ be its domain and its range, respectively, that is:

$$[\forall_{u \in U_m} \exists_{v \in V_m} (u, v) \in M] \wedge [\forall_{v \in V_m} \exists_{u \in U_m} (u, v) \in M]$$

For the consistency test we take two minutiae $u_1, u_2 \in U_m$ that lie in the same extended dual connected component of the graph G_M. For the corresponding matched minutiae in the second fingerprint

$$v_1, v_2 \in V_m : (u_1, v_1) \in M \wedge (u_2, v_2) \in M$$

we check the following conditions in a top-down manner, that is, if any of earlier conditions is not fulfilled, the following conditions will not be checked:

- v_1 and v_2 lie in the same extended dual connected component of the graph H_M,
- the difference of line distances is below a given threshold:
$$|d_l^{ed}(u_1, u_2) - d_l^{ed}(v_1, v_2)| < t_d,$$
- the difference in the cumulative direction change on the shortest path between the vertices is below a certain threshold
$$|\phi^{ed}(u_1, u_2) - \phi^{ed}(v_1, v_2)| < t_\phi.$$

Consequently, the scores for pairs (u_1, v_1) and (u_2, v_2) are updated according to the result of the consistency test.

This method does not exhaust the full power of the extended features as it only decreases scores for falsely matched minutiae, whereas the main reason for using extended features is to decrease the false rejection rate in case the level 2 features (usually minutiae) do not provide enough information to declare a pair as genuine [6]. However, even with

[1] For the tests we have not used any additional extracted information but only minutia position and direction.

Table 5.2. Equal error rates for consistency based matching

	Neurotechnologija	Entracer
FVC2002 db2	1.056%	0.784%
FVC2000 db3	7.692%	6.727%

the simple extension described above we were able to decrease the equal error rates, as shown in Table 5.2.

5.5. Extended features-based matching

In this section a new approach to fingerprint matching is described, taking into account the information about minutiae, ridges and particularly the connectivity information contained in the extracted fingerprint structure. Consequently, global and local features are connected so that the automatic matching resembles more closely the way human experts proceed.

Although the consistency-based method (described in the last section) already uses extended features, it still utilises a minutiae-based matching algorithm. The method described in this section adopts a thoroughly different approach to matching. Instead of globally aligning minutiae sets, each pair of minutiae is compared with the help of the information contained in the connected component to which the minutiae belong. This results in a minutiae similarity score. After all minutiae in a fingerprint have been compared, a global similarity score for the two fingerprints is determined.

In the following subsections the concepts of segment similarity, minutiae score and finally fingerprint similarity score are introduced. $G_M = (V_M, E_M, E_D)$ denotes the minutiae-vertex graph introduced in Section 2.6.1.

5.5.1. Segment similarity. In order to define segment similarity the following segment features are compared:

- Length: their relative difference is computed and if it is below a given threshold, the lengths are considered to be similar:
$$\frac{|l_1 - l_2|}{\max(l_1, l_2)} < c;$$
where c is an empirically chosen constant.
- Maximal curvature or curvature course: provides a good measure for fingerprints which have only linearly been disturbed. In case of non-linear distortions the local curvature can change diametrically as for example in the top left region of the fingerprint presented in Fig. 5.2. Hence the following features are more suitable.
- Maximal change of direction: see Section 4.3.1. Their difference is computed relatively as in the case of the length.
- Sign of the global direction change: provides the information whether a line is left- or right-twisted.

Since the set of features and the comparison techniques used for similarity tests have been specified only partially so far, for further investigation the following binary similarity function is defined:

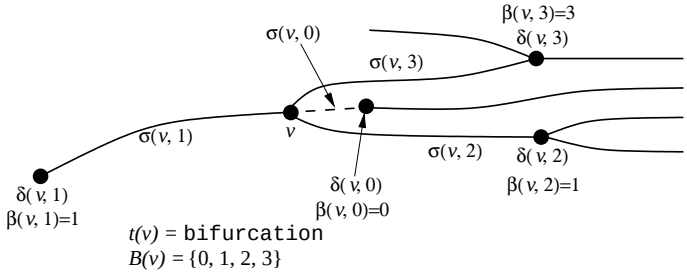


Figure 5.3. Sample values of functions for minutiae similarity score computation

Definition 5.1 (Similar segments - $s(s_1, s_2)$)**.** *The function* $s : E_M \times E_M \to \{0, 1\}$ *returns* 1 *if the segments* s_1*,* s_2 *are similar, i.e. pass the similarity tests, and* 0 *otherwise.*

5.5.2. Minutiae similarity score. In this subsection we proceed to the comparison of two minutiae. This approach explores the connectivity information. The score is computed recursively that is the minutiae and their adjacent segments are compared; if they can be considered as similar, the minutiae at the other ends of the segments are compared. As a result, both a similarity score and a connected component which contains minutiae and segments included into the comparison are obtained. In order to write down this comparison concept, some additional functions have to be defined.

Definition 5.2 (Type of a vertex – $t(v)$)**.** $t : V_M \to \{\texttt{ending}, \texttt{border point}, \texttt{bifurcation}\}$.

Definition 5.3 (Valid branches)**.** *The function* $B : V_M \to \mathcal{P}(\{0, 1, 2, 3\})$ *returns the set of all the valid branches of a vertex. The return values are presented below. The branch with number* 0 *is a dual connection.*

v	$B(v)$
non-validated ending, border point	$\{1\}$
validated ending	$\{0, 1\}$
non-validated bifurcation	$\{1, 2, 3\}$
validated bifurcation	$\{0, 1, 2, 3\}$

Definition 5.4 (Adjacent minutia – $\delta(v, b)$)**.** *The function* $\delta : V_M \times \{0, 1, 2, 3\} \to V_M$ *returns the minutia at the opposite end of the segment adjacent to vertex* v *at branch* b*. The valid values of* b *are contained in the set* $B(v)$*. In case minutia* v *is validated,* $\delta(v, 0)$ *is the corresponding dual minutia.*

Definition 5.5 (Adjacent segment – $\sigma(v, b)$)**.** *The function* $\sigma : V_M \times \{0, 1, 2, 3\} \to E_M \cup E_D$ *returns the segment adjacent to vertex* v *at branch* b*. The valid values of* b *are contained in the set* $B(v)$*. Like for* δ*, if minutia* v *is validated,* $\sigma(v, 0)$ *is an edge from the set* E_D*.*

Definition 5.6 (Branch at the adjacent minutia – $\beta(v, b)$)**.** *The function* $\beta : V_M \times \{0, 1, 2, 3\} \to \{0, 1, 2, 3\}$ *returns the index of the branch at which segment* $\sigma(v, b)$ *is connected to minutia* $\delta(v, b)$*.*

Exemplary values for the functions introduced above are presented in Fig. 5.3. In the next part of this section we will proceed with the comparison of two minutiae.

Let $G = (U, E_M, E_D)$ and $H = (V, F_M, F_D)$ be the extracted fingerprint graphs (see Section 2.6.1) for the fingerprints to be matched. For two minutiae $u \in U$ and $v \in V$ the

Table 5.3. Computation rules for $m^s_{in}(u, b_u, v, b_v, s)$. The introduced vertex v' is a new bifurcation created while connecting the neighbouring segment ending to the line being followed. In order to find whether there exists an appropriate ending, the virtual minutiae are utilised.

	G	H	return value	example on figure
1	$t(u) =$ `ending`	$t(v) =$ `ending`	$s + 1$	5.4(a)
2	$t(u) =$ `border point`	$t(v) =$ `border point`	$\begin{cases} s+1 & \text{if same image side,} \\ 0 & \text{otherwise} \end{cases}$	5.4(b)
3	$t(u) =$ `border point`	$t(v) \in \{$`ending`, `bifucarion`$\}$	$s + 0.75$	5.4(c)
4	$t(u) =$ `bifurcation`	$t(v) =$ `bifurcation`, $b_v = b_u$	$s + 1 + \sum_{b \neq b_v} m_{out}(u, b, v, b, 0)$	5.4(d)
5	$t(u) =$ `bifurcation`, $b_u \in \{1, 2\}$	$t(v) =$ `ending`	$s + \frac{3}{4} + m^s_{in}(u, b_u, v', b'_v)$	5.4(e)
6	$t(u) =$ `bifurcation`, $b_u \in \{1, 2\}$	$t(v) =$ `bifurcation` $b_v \in \{1, 2\} \wedge b_v \neq b_u$	$s + \frac{1}{2} + m^s_{in}(u, b_u, v', b'_v)$	5.4(f)
7	$t(u) =$ `bifurcation`, $b_u = 0$	$t(v) =$ `bifurcation`, $b_v \in \{1, 2\}$	$s + \frac{1}{4} + \max_{b \in \{1,2\}} (m_{out}(u, b, v, 0, 0))$	5.4(g)
8	$t(u) =$ `bifurcation`, $b_u = 0$	$t(v) =$ `ending`	$s + \frac{1}{4} + \max_{b \in \{1,2\}} (m_{out}(u, b, v', 0, 0))$	5.4(h)

similarity score between them is denoted with $m(u, v)$. The function $m : U \times V \to \mathbb{R}$ is defined for two vertices of the same type ($t(u) = t(v)$) as follows:

$$m(u, v) = \sum_{b \in B(u) \cap B(v)} m_{out}(u, b, v, b, 0),$$

where the recursive function $m_{out} : U \times \{0, 1, 2, 3\} \times V \times \{0, 1, 2, 3\} \times \mathbb{R} \to \mathbb{R}$ is defined below.

Definition 5.7 (Outgoing match – $m_{out}(u, b_u, v, b_v, s)$)**.** *The function $m_{out} : U \times \{0, 1, 2, 3\} \times V \times \{0, 1, 2, 3\} \times \mathbb{R} \to \mathbb{R}$ compares the segments adjacent to the input minutiae. The function takes the following arguments:*

- *u, v - minutiae from the sets U and V respectively;*
- *b_u, b_v - the indices of branches at which the segments to be compared are connected;*
- *s - the similarity score computed from the minutiae compared so far.*

and is defined by the following equation:

$$m_{out}(v, b_v, u, b_u, s) =$$

$$= \begin{cases} m^s_{in}(\delta(v, b_v), \beta(v, b_v), \delta(u, b_u), \beta(u, b_u), s) & \text{if } s(\sigma(v, b_v), \sigma(u, b_u)) = 1; \\ \dot{m}^d_{in}(\delta(v, b_v), \beta(v, b_v), \sigma(u, b_u), s) & \text{otherwise, if } l(\sigma(v, b_v) < l(\sigma(u, b_u); \\ \ddot{m}^d_{in}(\sigma(v, b_v), \delta(u, b_u), \beta(u, b_u), s) & \text{otherwise, if } l(\sigma(v, b_v) > l(\sigma(u, b_u). \end{cases}$$

The functions m^s_{in}, $\dot{m}^d_{in}$ and $\ddot{m}^d_{in}$ are defined below.

Definition 5.8 (Incoming match for similar segments – $m^s_{in}(u, b_u, v, b_v, s)$)**.** *The function $m^s_{in} : U \times \{0, 1, 2, 3\} \times V \times \{0, 1, 2, 3\} \times \mathbb{R} \to \mathbb{R}$ returns the score for two minutiae reached by segments which have been regarded as similar. The function has the following arguments:*

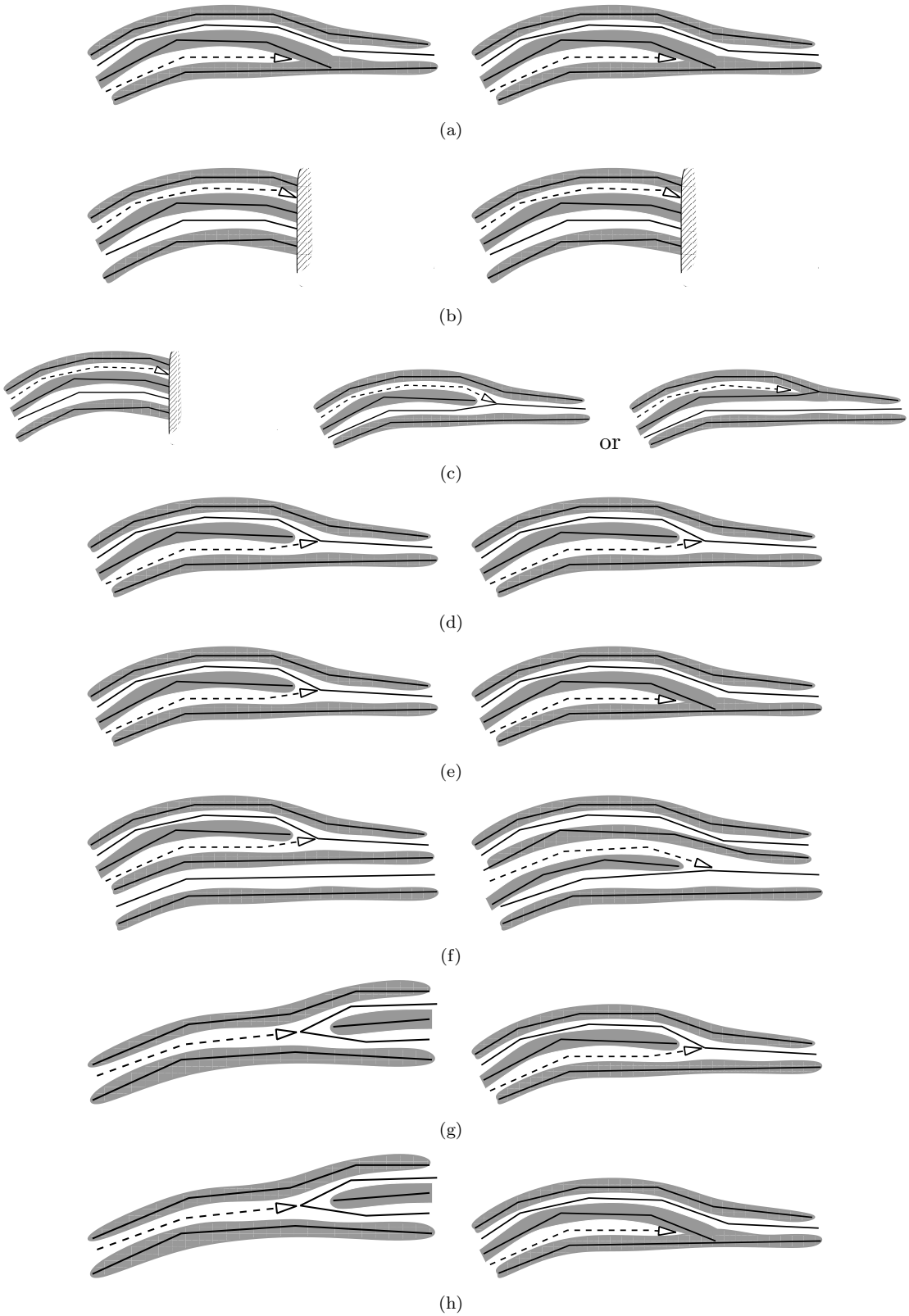


Figure 5.4. The situations which may arise in the case of similar segments. The dashed lines are processed in the direction of the arrow which points at the vertex u or v, respectively.

Table 5.4. Computation rules for $\dot{m}^d_{in}(u, b_u, e, s)$

		additional condition	return value	example on figure
1	$t(u) =$ `border point`		s	5.5(a)
2	$t(u) =$ `bifurcation`	an appropriate ending on the neighbour of e exists	$s + \frac{1}{2} + m^s_{in}(u, b_u, v', b'_v)$	5.5(b)
3	$t(u) =$ `bifurcation`	no appropriate ending	s	5.5(c)
4	$t(u) =$ `ending`	an appropriate ending on the neighbour of e exists	$s + m^s_{in}(u', b'_u, v', b'_v)$	5.5(d)
4	$t(u) =$ `ending`	no appropriate ending	s	5.5(e)

- *u, v - vertices in fingerprint graphs G and H, respectively;*
- *b_u, b_v - the branches at which the vertices u and v, respectively, have been reached;*
- *s - the similarity score computed from the minutiae compared so far.*

The computation rules are presented in Table 5.3.

The inclusion of the "score computed so far" to the parameter list allows us to introduce a stopping condition for aborting the computation of the similarity score when a clear indicator for non-matching arises. An example of such a condition is the second rule in Table 5.3.

Definition 5.9 (Incoming match for different segments – $\dot{m}^d_{in}(u, b_u, e, s)$)**.** *The function $\dot{m}^d_{in} : U \times \{0, 1, 2, 3\} \times (F_M \cup F_D) \times \mathbb{R} \to \mathbb{R}$ returns the similarity score in the case of non-similar segments. The function takes the following arguments:*

- *u - a vertex in the fingerprint graph G;*
- *b_u - the branch at which vertex u has been reached;*
- *e - the segment in graph H compared with the segment adjacent to vertex u at branch b_u;*
- *s - the similarity score computed from the minutiae compared so far.*

The computation rules are presented in Table 5.4.

Since function $\ddot{m}^d_{in} : (E_M \cup E_D) \times V \times \{0, 1, 2, 3\} \times \mathbb{R} \to \mathbb{R}$ is symmetric to function $\dot{m}^d_{in} : U \times \{0, 1, 2, 3\} \times (F_M \cup F_D) \times \mathbb{R} \to \mathbb{R}$, its definition is not presented.

While the similarity score for minutiae u and v is being computed, minutiae belonging to the connected components which contain u and v, respectively, are matched onto each other. As a result we get a match set defined as follows.

Definition 5.10 (Match set for a vertex pair (u, v) – $M(u, v) \subset U \times V$)**.** *The set $M(u, v)$ contains all minutia pairs matched during the approach described above when starting at the vertices $u \in U$ and $v \in V$:*

$$M(u, v) = \{(u_i, v_i) \in U \times V\}$$

Obviously $(u, v) \in M(u, v)$ and $M(u, v)$ defines a relation on $U \times V$. Let us denote its domain and its range with $U_{M(u,v)}$ and $V_{M(u,v)}$[2], respectively. A crucial property of match sets is their consistency, which means that every minutia has been matched with at most one minutia:

[2]In the following, we omit the dependence on (u, v) when it does not lead to confusion.

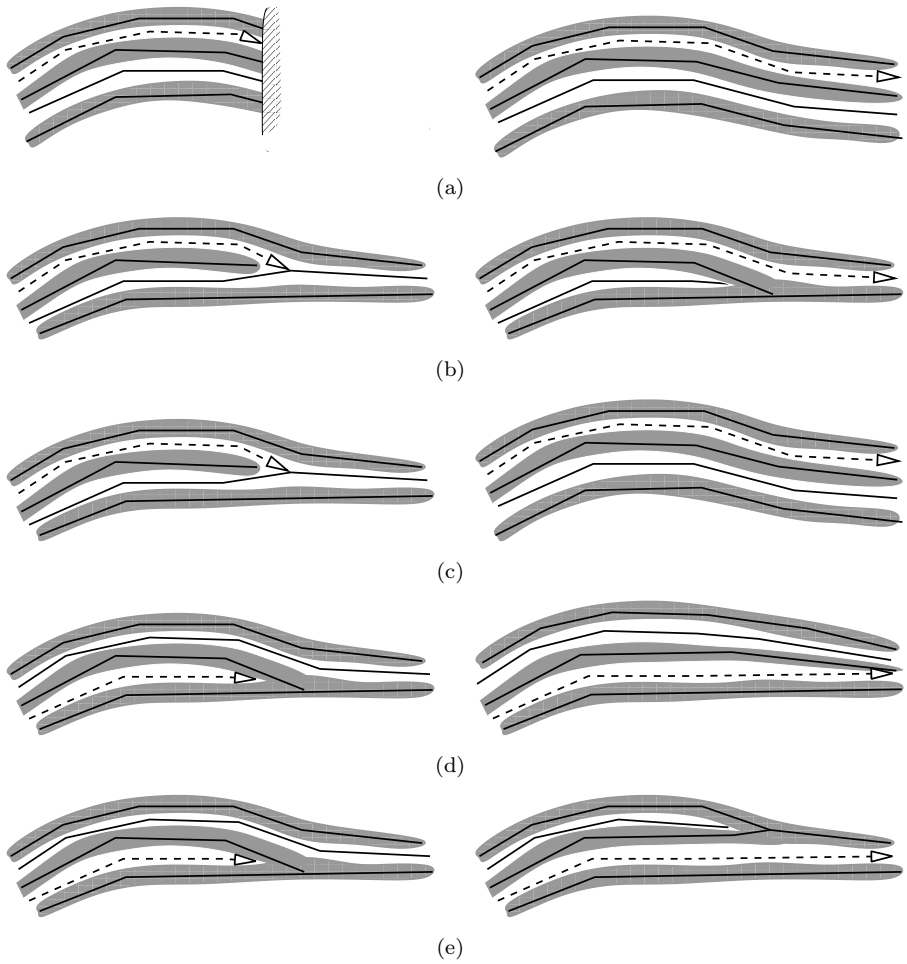


Figure 5.5. The situations which may arise in case of non-similar segments. The dashed lines are processed in the direction of the arrow which points at the vertex u or v, respectively.

Definition 5.11 (Match set consistency)**.** *A match set $M \subset U \times V$ is consistent if M defines a bijective function $M : U_M \to V_M$, that is, the following two conditions are fulfilled:*

$$(u_1, v_1) \in M \wedge (u_2, v_1) \in M \Rightarrow u_1 = u_2$$

$$(u_1, v_1) \in M \wedge (u_1, v_2) \in M \Rightarrow v_1 = v_2$$

5.5.3. Fingerprint similarity score. For two minutiae $u \in U$ and $v \in V$, the comparison algorithm described in the previous subsection provides us with a match score $m(u, v)$ and a consistent match set $M(u, v)$. Samples of computed score values for two impostor fingerprints are presented in Table 5.5 and for two genuine ones in Table 5.6. From these partial results a global similarity score has to be computed. Three approaches are proposed in the following subsections.

Bipartite graph. Let

$$\mathfrak{m} := \{(u, v) \in U \times V : m(u, v) > 0\}$$

Table 5.5. A sample subset of minutiae scores for an impostor fingerprint pair

minutiae in the first fingerprint	minutiae in the second fingerprint									
	1	2	3	4	5	6	7	8	9	10
1	0	0.09	0	0	0	0	0	0	0.31	0
2	0	0	0	0	0	0	0.49	0	0	0
3	0.13	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0.21	0	0	0	0	0.19
5	0	0	0	0	0	0	0	0	0	0
6	0.83	0	0	0	0	0	0	0	0	0
7	0.16	0	0	0	0	0	0	0	0	0

Table 5.6. A sample subset of minutiae scores for a genuine fingerprint pair. The scores for the corresponding minutiae are marked bold

minutiae in the first fingerprint	minutiae in the second fingerprint					
	1	2	3	4	5	6
1	0	0	**0.66**	0	0	0
2	0	0.15	0	**0**	0	0
3	0	0	0.28	0	0	0
4	**0.50**	0	0	0	0.12	0
5	0	0.12	0	0	**0.66**	0
6	0	0	0	0.21	0	0
7	0	0	0	0	0	**0.83**
8	0	**0.73**	0	0	0	0.23

be the set of all non-zero matching minutiae pairs. The goal is to find a consistent subset $\widehat{\mathfrak{m}}$ of $\mathfrak{m}$ which maximises the global score

$$\sum_{(u,v)\in\widehat{\mathfrak{m}}} m(u,v).$$

In order to find such a subset, let us consider a bipartite graph

$$G_B := ((U,V), E_B)$$

in which the edge set E_B contains an edge (u,v) with weight w iff there exist a non-zero match between minutiae $u \in U$ and $v \in V$ with the score w. Such a graph for the match matrix from Table 5.6 is presented in Fig. 5.6. The problem of finding the maximal consistent matching is then equivalent to the problem of finding the maximal matching in the bipartite graph G_B. This is the well-known weighted marriage problem [45] which can be solved in polynomial time.

Unfortunately, this method computes a high similarity score also for impostor pairs as it does not take the dependencies between the matched pairs into account. Methods taking this problem into account are presented in the next subsections.

Displacement vectors. In order to decrease the similarity score for impostor pairs in the last method, some connections between minutiae pairs have to be introduced.

Let us consider a displacement vector for a matched minutia pair $(u,v) \in U \times V$, that is a vector originating at the position of minutia u and pointing to the position of minutia v. Examples of displacement vectors for minutia pairs with a non-zero score of the fingerprints

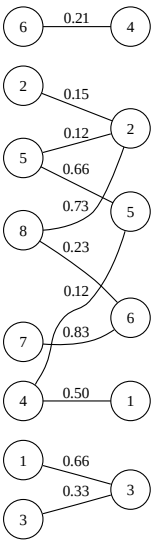
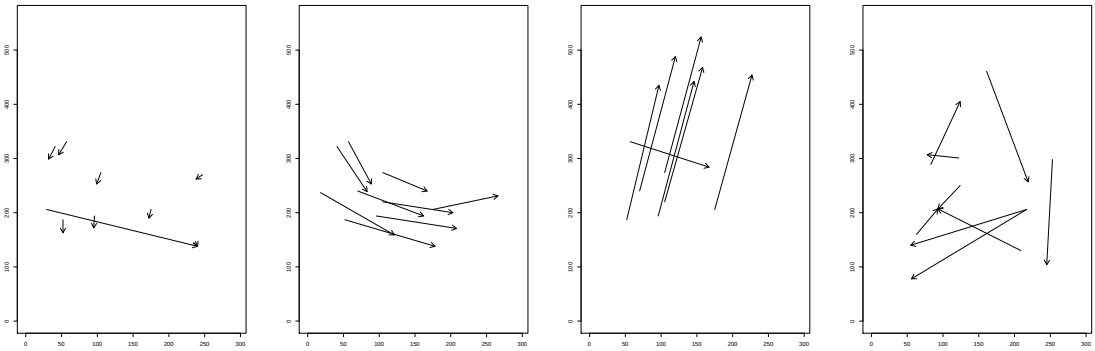


Figure 5.6. Bipartite graph for the scores presented in Table 5.6



(a) 10_1 10_2 (b) 10_1 10_3 (c) 10_1 10_4 (d) 18_6 26_1

Figure 5.7. Displacement vectors for fingerprints from Fig. 5.1. Three genuine pairs and one impostor pair.

given in Fig. 5.1 are presented in Fig. 5.7. We can observe that most displacement vectors for a genuine fingerprint pair correlate with some linear transformation, whereas such a linear transformation cannot be found for an impostor pair. Consequently, one can find a linear transformation $T_{\varphi,\mathbf{o},\mathbf{t}}$ containing a rotation by an angle φ around a point $\mathbf{o}$ and translation by $\mathbf{t}$ approximating the displacement of a majority of the minutiae pairs in a match. The transformation $T_{\varphi,\mathbf{o},\mathbf{t}}$ for a point $\mathbf{p}$ in the plane is defined by

$$T_{\varphi,\mathbf{o},\mathbf{t}}(\mathbf{p}) := (\mathbf{p}-\mathbf{o}) \cdot \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} + \mathbf{o} + \mathbf{t}$$

Thus, in order to compute a global score the weighted marriage problem described above can be applied but with the additional constraint that the displacement of matched pairs should conform to a linear transformation. Obviously, due to the non-linear deformations and the inexact determination of a minutia position, the conformance to a linear

transformation should not be strict. The degree of accepted non-linear deformations is correlated with the degree of the conformance relaxation. Obviously, the higher the relaxation, the larger is the false match rate. The following subsection introduces a method for dealing with this problem.

Clique approach. In the matching approaches described above two minutiae pairings (u_1, v_1) and (u_2, v_2) are accepted if the following condition is fulfilled:

$$u_1 = u_2 \iff v_1 = v_2$$

However, even if the above condition is fulfilled it may happen that there exist two pairs in the corresponding match sets : $(u_1', v_1') \in M(u_1, v_1)$ and $(u_2', v_2') \in M(u_2, v_2)$, so that:

$$\left(u_1' = u_2' \wedge v_1' \neq v_2'\right) \vee \left(u_1' \neq u_2' \wedge v_1' = v_2'\right).$$

This leads us to the following definition.

Definition 5.12 (Compatible matching sets)**.** *Two consistent match sets M_1, M_2 are compatible if $M_1 \cup M_2$ is a consistent match set for a match between $U_{M_1} \cup U_{M_2}$ and $V_{M_1} \cup V_{M_2}$, i.e. it complies with the following requirements:*

$$(u_1, v_1) \in M_1 \wedge (u_2, v_1) \in M_2 \Rightarrow u_1 = u_2,$$
$$(u_1, v_1) \in M_1 \wedge (u_1, v_2) \in M_2 \Rightarrow v_1 = v_2.$$

Let

$$\mathfrak{M} := \{M(u, v) : u \in U \wedge v \in V \wedge m(u, v) > 0\}$$

be the set of all minutiae match sets with a non-zero match score. The goal is to find a subset $\widehat{\mathfrak{M}}$ of $\mathfrak{M}$ so that all match sets in $\widehat{\mathfrak{M}}$ are compatible to each other and the global score

$$\sum_{u,v:M(u,v)\in\widehat{\mathfrak{M}}} m(u, v)$$

is maximal.

In order to find such a subset $\widehat{\mathfrak{M}}$, let us consider the graph

$$G_{\mathfrak{M}} := (V_{\mathfrak{M}}, E_{\mathfrak{M}})$$

in which the vertex set $V_{\mathfrak{M}} \subset U \times V$ contains all the matched minutiae pairs with a positive score and there exists an edge $((u_1, v_1), (u_2, v_2)) \in E$ iff match set $M(u_1, v_1)$ is compatible with match set $M(u_2, v_2)$. The problem of finding a subset $\widehat{\mathfrak{M}}$ which maximises the score is then equivalent to the weighted clique (complete subgraph) problem. Unfortunately the problem of finding a maximal clique is NP-hard [46]. However, as the graph $G_{\mathfrak{M}}$ is usually small and sparse the computation time required to find a desired clique is not unreasonable. Empirical tests have shown that a long computation time is needed only for few fingerprints.

5.5.4. Conclusion. In this section a novel approach to fingerprint matching has been described. The method has however not been completely developed. There are a few steps in which the approach calls for improvements. It may be worthwhile considering a continuous function which returns a segment similarity score instead of a binary function. Furthermore, it would also be reasonable to additionally introduce segment neighbourhood information so that the segments not adjacent to any bifurcation or ending might be included in the matching process.

Chapter 6

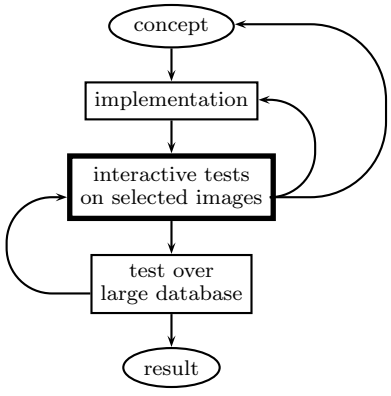
Development process and software environment

6.1. Design and development

The design and development process which has been used for constructing the methods described above is briefly presented in Table 6.1 and explained in greater detail in what follows. After the implementation of a new approach, the research is carried on to the main step - the interactive testing on chosen fingerprint images. These visually controlled tests give a valuable feedback for perfecting the methods and their implementation. After completing the interactive step, the approach is non-interactively tested over a large fingerprint image database. The outliers, that is the impressions for which the method fails or the achieved error rates are high, can then be identified and processed in the interactive stage in order to find improvements.

Using an interactive approach simplifies more or less technical procedures such choosing between methods or parameter tuning. Methods can be applied on a small sector of

Table 6.1. Stages in the development process



the image, which provides us with a more detailed view. Furthermore, in order to find an appropriate method from a large set of candidates, as for example in image processing steps, different methods can easily be applied at the same time in order to judge their influences. After the right method has been found, its parameters can easily be tuned. Consequently, the interactive step supplies us with an answer to the question: *How do the parameters or methods influence the outcome?* However, for a fruitful research it is even more important to know the answer to the question: *Why do the parameters or methods influence the outcome in that manner?* The support gained for getting the answer to the latter question is the most valuable, though in a sense indirect, contribution of the interactive step. The interactive investigation provides us with a feeling for the complicated structure of a fingerprint: one can precisely observe where problems arise and one's approach is improved gradually through assessing one's results.

The interactive development can be well illustrated in the example described below. During the development of the entracer, a single line was traced at the beginning. The starting point for the tracer was manually chosen by pointing with the mouse. The line following process was done step by step so that the influence of the different parameters such as feeler length or number of angles could be monitored. The behaviour of the tracer in difficult areas such as ones disturbed by noise or close to singular points, with high ridge curvature, could be tracked. In the next step, the tracer was applied to the whole image, and again the results could be utilised for further improvement.

6.2. Software environment for research and development

The approach described above for conducting the research has contributed to the creation of an environment for fingerprint investigation. It is a software platform which includes implementations of common methods from image analysis and fingerprint processing, the new methods presented in this thesis as well as interactive tools. The environment is composed of three parts:

- faul: the main part with a graphical interface,
- kfinger: a package for statistical evaluation,
- fate: a library with a functional programming approach to selected problems.

For the development of the whole platform only free software has been used. Although the environment has been developed under the Linux operating system, the libraries and compilers used are platform independent and thus the environment can easily be migrated to other operating systems.

The following sections provide for a brief description of the platform.

6.2.1. faul. This part, written in C++ and compiled under the GNU GCC compiler [47], is built up of three modules:

- fingerprint processing methods: the part containing both common and new methods for fingerprint processing,
- graphical user interface: the front-end implemented with the wxWidgets toolkit [48], allowing interaction,
- command line interface: allows scripted execution of the methods which is useful e.g. for tests over large databases.

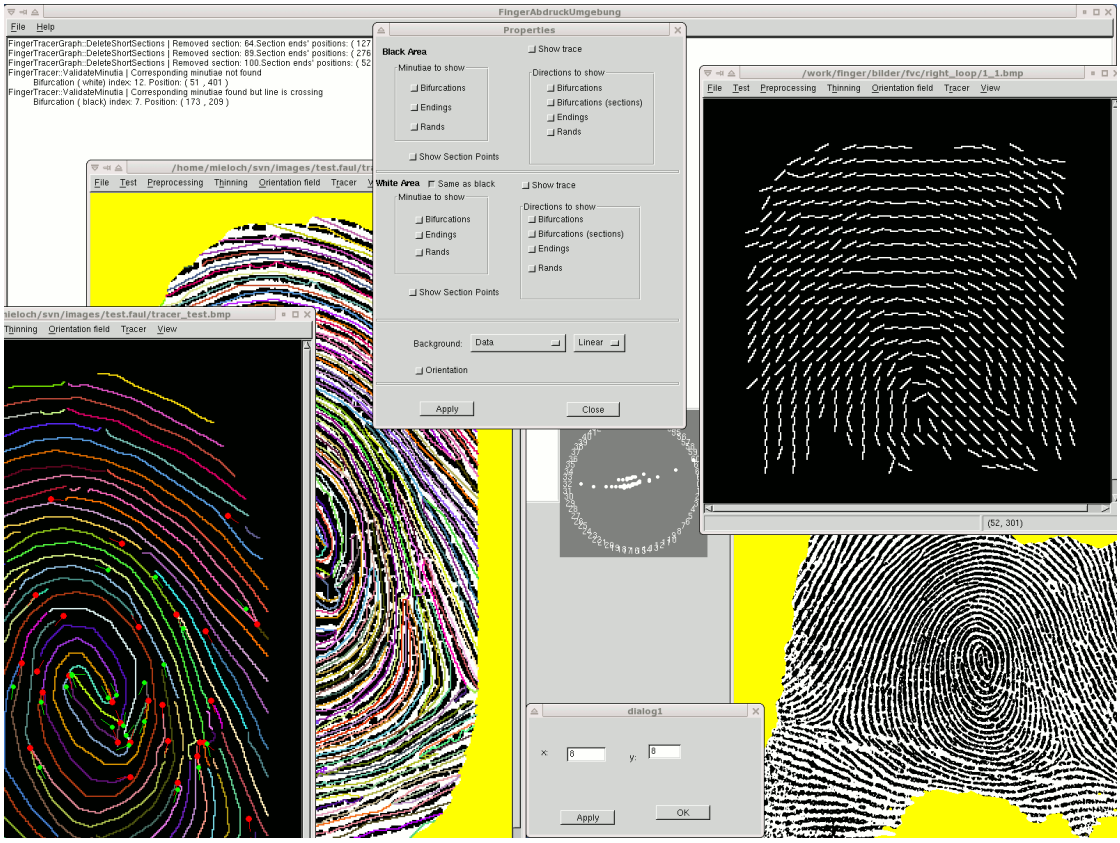


Figure 6.1. Graphical user interface of faul

Fingerprint processing methods. The software is built modular and implemented in an object-oriented style. The core classes are listed below:

- `Finger` - class holding the fingerprint image, containing I/O-procedures for loading and saving, functions for image processing as well as implementations of common methods;
- `Minutia` - class holding a single minutia;
- `FingerMinutiae` - container holding all the minutiae in a fingerprint;
- `FingerTracerBase` - contains the entracer methods;
- `TracerMinutia` - class derived from the `Minutia` class. The base class has been extended with members and methods required for the tracing process such as links to adjacent segments;
- `FingerTracerMinutiae` - class derived from the `FingerMinutiae` class. Like for the last class, the base class has been extended with members and methods required for the tracing process;
- `Segment` - class holding a single segment, containing methods for computing descriptive characteristics of a segment such as its orientation change or curvature course;
- `FingerTracerGraph` - class holding all the segments, and the connectivity information between segments and minutiae which define the fingerprint graph.

Graphical user interface. A screenshot of the software in use is presented in Fig. 6.1; the main features of the graphical front-end are listed below:

- interactive
  - parameter adjustment,
  - method selection,
  - pointwise or regional application,
  - keeping track of how an algorithm proceeds;
- log-window which displays the textual output of the methods;
- graphical display results when applying a method;
- selection of features to be displayed.

6.2.2. kfinger and fate. Although the main part of the platform has been implemented in the C++ language, two parts have been separately implemented in other programming languages more suitable for particular tasks.

Accordingly, the R programming language and software environment [49] has been used for the statistical evaluation.

Furthermore, some of the methods has been developed with the help of a functional programming language. Since functional programming provides one with a higher level of abstraction than imperative programming, the former is more suitable for handling complicated abstract mathematical structures. For the present platform, the functional approach has been used for graph processing and recursive matching. OCaml (objective caml) [50] has been chosen for the implementation as it unifies functional, imperative, and object-oriented programming under an ML-like type system [51]. The OCaml's tool set includes an interactive top-level interpreter and an optimising native code compiler which provides a fast running executable for integration with the faul platform.

Conclusion

Traditionally, minutiae have been used as primary features for fingerprint comparison and global features only for classification. The primary goal of this thesis has been to investigate the use of alternative features that can be extracted from fingerprint images, which includes a proposal for introducing a hierarchically linked extended feature set and a new entropy-sensitive extraction approach.

The universal class of features defined in this thesis can be applied in various stages of fingerprint processing since the features combine the global and local scales which are ordinarily used separately. Consequently, this approach makes the automatic fingerprint recognition procedure more similar to the way human experts work. The tests performed for classification and matching have shown a performance increase as the result. Similarly, new approaches for other stages in fingerprint processing have been proposed. The addressed stages are presented in Fig. 7.1.

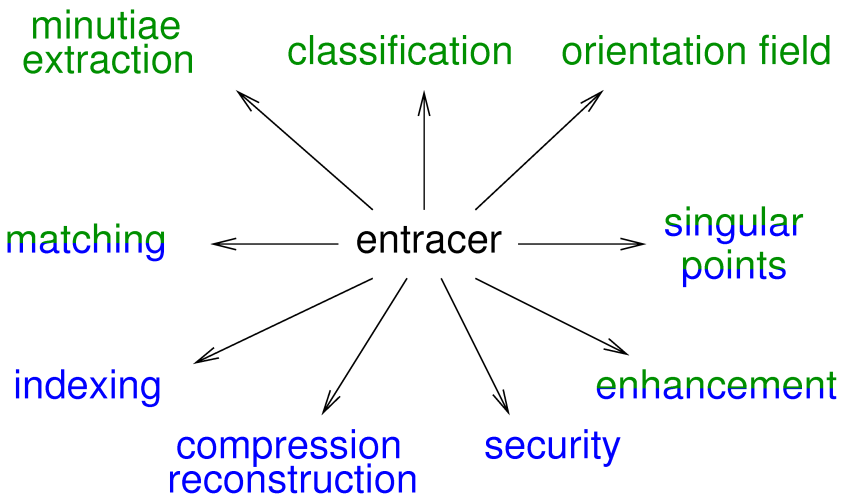


Figure 7.1. Potential applications of the entracer. The green marked ones have been already successfully performed. For the double coloured applications a realisation sketch has been presented in this thesis. The blue ones have not been investigated yet.

The entracer introduced in Chapter 2 is a novel approach to fingerprint feature extraction. The extraction proceeds in two stages. First, information contained in a fingerprint is extracted and stored in the fingerprint structure as a basis for the feature extraction. The features required for a particular task are then computed from the structure.

Features such as lengths of connected components in the fingerprint graph or number of unvalidated minutiae have been employed for the segmentation improvement described in Chapter 3. Furthermore, connectivity information has been used for reconstructing fingerprint lines in low quality areas.

In Chapter 4 the extended lines extracted from the fingerprint structure have successfully been applied to fingerprint classification and a possible application to fingerprint indexing has been proposed.

Minutiae - the conventional features used for matching - are also contained in this structure. Their quality has been tested in Chapter 5. Additionally, a sketch of a matching approach which fully utilises the extended features has been presented.

The thesis provides a good basis for applications of the new features for other tasks in fingerprint processing, as well. The extracted features are robust, which is crucial for securing a template. Finally, since the features are extracted from the underlying fingerprint structure exclusively, storing of the structure instead of the whole fingerprint requires less memory, and can save resources. Furthermore, it would be desirable to investigate the problem of reconstruction of a fingerprint image from its fingerprint structure.

It would be also worth considering an application of the methods described in this thesis for recognition of other structures build up of parallel running ridges, such as tree-rings.

Index

Bibliography

[1] S. Pankanti, S. Prabhakar, and A. K. Jain, "On the individuality of fingerprints," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 8, pp. 1010–1025, Aug. 2002.

[2] A. Jain and S. Pankanti, "Biometric systems: Anatomy of performance," *IEICE transactions on information and systems*, vol. 84, no. 7, pp. 788–799, 2001.

[3] A. K. Jain, K. Nandakumar, and A. Nagar, "Biometric template security," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, 2008, article ID 579416, 17 pages.

[4] J. Zhou and J. Gu, "Modeling orientation fields of fingerprints with rational complex functions," *Pattern Recognition*, vol. 37, pp. 389–391, 2004.

[5] A. K. Jain, S. Prabhakar, and S. Pankanti, "On the similarity of identical twins fingerprints," *Pattern Recognition*, vol. 35, no. 11, pp. 2653–2663, 2002.

[6] A. K. Jain, Y. Chen, and D. Meltem, "Pores and ridges: high-resolution fingerprint matching using level 3 features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 15–27, 2007.

[7] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*. New York: Springer, 2003.

[8] A. K. Jain and S. Pankanti, *Automated Fingerprint Identification and Imaging Systems*, ser. Crc Series in Forensic and Police Science. CRC Press, 2001, ch. 8, pp. 275–326.

[9] A. Lumini, D. Maio, and D. Maltoni, "Continuous vs. exclusive classification for fingerprint retrieval," *Pattern Recognition Letters*, vol. 18, no. 10, pp. 1027–1034, 1997.

[10] X. Xiongwu and L. O'Gorman, "Innovations in fingerprint capture devices," *Pattern Recognition*, vol. 36, no. 2, pp. 361–369, 2003.

[11] N. Yager and A. Amin, "Fingerprint verification based on minutiae features: a review," *Pattern Analysis Application*, vol. 7, pp. 94–113, 2004.

[12] N. Yager, "Hierarchical fingerprint verification," Ph.D. dissertation, School of Computer Science and Engineering, University of New South Wales, 2006.

[13] *CDEFFS: the ANSI/NIST Committee to Define an Extended Fingerprint Feature Set.* [Online]. Available: http://fingerprint.nist.gov/standard/cdeffs

[14] A. Hicklin, *CDEFFS – Extended Feature Sets*, Latent Testing Workshop, National Institute of Standards and Technology, 2006. [Online]. Available: http://www.itl.nist.gov/iad/894.03/latent/presentations.htm

[15] L. Hong, Y. Wan, and A. K. Jain, "Fingerprint image enhancement: Algorithm and performance evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 777–789, Aug. 1998.

[16] D. Maio and D. Maltion, "Direct gray-scale minutiae detection in fingerprints," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 1, pp. 27–40, 1997.

[17] D. Maio and D. Maltoni, "Minutiae extraction and filtering from gray-scale images," in *Intelligent Biometric Techniques in Fingerprint & Face Recognition*, L. Jain, U. Halici, I. Hayashi, and S. Lee, Eds. CRC Press, 1999, pp. 153–192.

[18] X. Jiang, W.-Y. Yau, and W. Ser, "Detecting the fingerprint minutiae by adaptive tracing the gray-level ridge," *Pattern Recognition*, vol. 34, pp. 999–1013, 2001.

[19] H. Liu and Chang, "Direct minutiae extraction from gray-level fingerprint image by relationship examination," in *Proc. International Conference on Image Processing*, vol. 2, 2000, pp. 427–430.

[20] L. Hong, A. K. Jain, S. Pankanti, and R. Bolle, "Fingerprint enhancement," in *Proc. of IEEE Workshop on Applications of Computer Vision*, 1996, pp. 202–207.

[21] A. M. Bazen and S. H. Gerez, "Segmentation of fingerprint images," in *Proc. ProRISC 2001 Workshop on Circuits, Systems and Signal Processing*, 2001, pp. 276–280.

[22] F. Leymarie and D. Levine, "Curvature morphology," Computer Vision and Robotics Laboratory, McGill University, Tech. Rep., 1988.

[23] M. M. S. Chong, T. H. Ngee, L. Jun, and R. K. L. Gay, "Geometric framework for fingerprint image classification," *Pattern Recognition*, vol. 30, no. 9, pp. 1475–1488, 1997.

[24] C. de Boor, *A practical guide to splines.* New York: Springer, 2001.

[25] B. Bollobas, *Graph Theory.* New York: Springer, 1979.

[26] W. D. Blizard, "Multiset theory," *Notre Dame Journal of Formal Logic*, vol. 30, no. 1, pp. 36–66, 1988.

[27] D. K. Isenor and S. G. Zaky, "Fingerprint identification using graph matching," *Pattern Recognition*, vol. 19, no. 2, pp. 113–122, 1986.

[28] B. M. Mehtre and B. Chatterjee, "Segmentation of fingerprint images - a composite method," *Pattern Recognition*, vol. 22, no. 4, pp. 381–385, 1989.

[29] B. M. Mehtre, N. N. Murthy, S. Kapoor, and B. Chatterjee, "Segmentation of fingerprint images using the directional image," *Pattern Recognition*, vol. 20, no. 4, pp. 429–435, 1987.

[30] 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.

[31] A. K. Jain, N. K. Ratha, and S. Lakshmanan, "Object detection using gabor filters," *Pattern Recognition*, vol. 30, no. 2, pp. 295–309, 1997.

[32] M. Ballan, F. A. Sakarya, and B. L. Evans, "A fingerprint classification technique using directional images," in *Proc. Asilomar Conf. on Signals Systems and Computers*, 1997, pp. 101–104.

[33] K. Mardia, A. Baczkowski, X. Feng, and T. Hainsworth, "Statistical methods for automatic interpretation of digitally scanned fingerprints," *Pattern Recognition Letters*, vol. 18, no. 11-13, pp. 1197–1203, 1997.

[34] M. Kawagoe and A. Tojo, "Fingerprint pattern classification," *Pattern Recognition*, vol. 17, no. 3, pp. 295–303, 1984.

[35] R. Cappelli, D. Maio, and D. Maltoni, "Multi-space kl for pattern representation and classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 9, pp. 977–996, 2001.

[36] A. Senior, "A hidden markov model fingerprint classifier," in *Proc. Asilomar Conf. on Signals Systems and Computers (31st)*, 1997, pp. 306–310.

[37] S. C. Dass and A. K. Jain, "Fingerprint classification using orientation field flow curves," in *Proc. of the Fourth Indian Conference on Computer Vision Graphics & Image Processing*, 2004, pp. 650–655.

[38] J. Chang and K. Fan, "Fingerprint ridge allocation in direct grey-scale domain," *Pattern Recognition*, vol. 34, pp. 1907–1925, 2001.

[39] B. H. Cho, J. S. Kim, J. H. Bae, I. G. Bae, and K. Y. Yoo, "Core-base fingerprint image classification," in *Proc. Int. Conf. on Pattern Recognition*, 2000, pp. 863–866.

[40] J. Li, W.-Y. Yau, and H. Wang, "Combining singular points and orientation image information for fingerprint classification," *Pattern Recognition*, vol. 41, no. 1, pp. 353–366, 2008.

[41] C. I. Watson and C. L. Wilson, *NIST Special Database 4: Fingerprint Database*, National Institute of Standards and Technology, Mar. 1992.

[42] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, and A. K. Jain, "FVC2002: Second fingerprint verification competition," in *Proc. Int. Conf. on Pattern Recognition (16th)*, vol. 3, 2002, pp. 811–814.

[43] ——, "FVC2000: Fingerprint verification competition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 402–412, 2002.

[44] *Neurotechnologija Verifinger SDK*. [Online]. Available: www.neurotechnologija.com

[45] D. B. West, *Introduction to Graph Theory*. London: Prentice-Hall, 1996.

[46] C. M. Papadimitriou, *Computational complexity*. Reading, Massachusetts: Addison-Wesley, 1994.

[47] GNU Project, *GNU Compiler Collection*, 2008. [Online]. Available: http://gcc.gnu.org

[48] wxWidgets Developers and Contributors, *wxWidgets: A portable C++ and Python toolkit*, 2007. [Online]. Available: http://www.wxwidgets.org

[49] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2008. [Online]. Available: http://www.R-project.org

[50] X. Leroy, D. Doligez, J. Garrigue, D. Rémy, and J. Vouillon, *The Objective Caml system*, Institut National de Recherche en Informatique et Automatique, Paris, France, 2007. [Online]. Available: http://caml.inria.fr

[51] A. Webber, *Modern programming languages: a practical introduction.* Wilsonville: Franklin, Beedle & Associates, 2003.

Curriculum Vitae

Krzysztof Mieloch (M.Sc.)

born 20$^{\text{th}}$ September 1980 in Wrocław, Poland
married, Polish

September 1985 – June 1998
Schooling
Matura at *Liceum Ogólnokształcące, Strzelin, Poland*

September 1998 – October 2003
Study of Computer Science (Master)
Institute of Computer Science, University of Wrocław, Poland
master thesis: *Jenga: analysis and implementation*
supervised by *Prof. Dr. Krzysztof Loryś*

September 1998 – June 2003
Study of Mathematics (Bachelor)
Mathematical Institute, University of Wrocław, Poland

October 2003 – November 2004
Study of Mathematics (Master)
Faculty of Mathematics, University of Göttingen
master thesis: *Mathematical models and statistical analysis of fingerprints*
supervised by *Prof. Dr. Axel Munk*

October 2003 – November 2004
Student Assistant
Institute for Mathematical Stochastics, University of Göttingen

since November 2004
Ph.D. Studies in Mathematics
Faculty of Mathematics, University of Göttingen
supervised by *Prof. Dr. Axel Munk*
member of the *DFG Graduate Program 1023*
"Identification in Mathematical Models"