Presentation Attack Detection in Finger and Hand Vein Biometrics using Video Sequences

Master's Thesis

To obtain the academic degree
Master of Science in Engineering

Submitted by
Johannes Schuiki

Supervisor
Univ.-Prof. Dr. Andreas Uhl

Faculty of Natural Sciences

Department of Computer Sciences

Paris Lodron University Salzburg

Salzburg, December 2021

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Signature	Date, Place

Notice on Prior Publication

Partial results and paragraphs of this work were published in advance by the author of this thesis. The algorithms in sections 3.3 & 3.4 and parts of the results in section 6.3.2 were published in Proceedings of the 19th International Conference of the Biometrics Special Interest Group (BIOSIG 2020) [68]. Parts of the database introduced in section 5.1.1, a less detailed version of chapter 4 and parts of the results in section 6.2.2 were published in the Proceedings of the 9th International Workshop on Biometrics and Forensics (IWBF'21) [67]. Also the results from the threat analysis comparison with other publicly available finger vein datasets (section 6.2.2) and section 4.1 were published in the Proceedings of the 2021 IEEE International Joint Conference on Biometrics (IJCB 2021) [69].

Abstract

The times where access authorization checks for high security areas by using parts of the human body was solely possible on the TV screen are long over. Biometrics, as the generic term is called, has emerged to be a widely researched topic. Yet verifying a person's claim of identity via a behavioural or physical characteristic remains still a complex task.

One particular physical characteristic that is in use for this manner is the structure of blood vessels in the hand region. However, the last decade has brought forward several ideas on how to deceive an automatic verification system by using fake finger or hand veins as an input. The act of presenting such a forged finger or hand to a biometric sensor is known as presentation attack. To counteract such attacks, various approaches have been published during the last few years. The majority of the presented solutions to circumvent this problem operate on still images, though. Very scarce literature exists on achieving attack detection when dealing with video data.

This thesis focuses on looking for distinctive characteristics in consecutive frames of a given video sequence in order to perform presentation attack detection. Experiments include two video data sets that were recorded in the Multimedia Signal Processing and Security Lab from the University of Salzburg. The data sets contain blood vessel structures in the finger as well as on the dorsum of the hand, respectively, and include several attacks and illumination variants.

To do so, it is first evaluated whether the attacks would actually be able to deceive a real system. Afterwards, experiments are conducted that test already existing attack detection methods as well as methods that have been developed in the course of creating this thesis on their performance by using the two above mentioned data sets.

Contents

Abbreviations

This page lists the most recurring abbreviations that are used in this thesis and, if possible, reference its introduction.

APCER Attack Presentation Classification Error Rate 6.3
ASAVE Anatomy Structure Analysis-Based Vein Extraction 4.4.7
BPCER Bona Fide Presentation Classification Error Rate 6.3
CNN Convolutional Neural Network 4.4.12
DTFPM Deformation-Tolerant Feature-Point Matching 4.4.10
EER Equal Error Rate 4.1
FMR False Match Rate 4.1
FNMR False Non Match Rate 4.1
FPS Frames Per Second
GF Gabor Filters 4.4.5
IAPMR Impostor Attack Presentation Match Rate 4.1
IUWT Isotropic Undecimated Wavelet Transform 4.4.11
LBP Local Binary Pattern 4.4.11
MC Maximum Curvature 4.4.1
PC Principal Curvature 4.4.2
RLT Repeated Line Tracking 4.4.4
ROI Region of Interest
SIFT Scale Invariant Feature Transform 4.4.8
SURF Speeded Up Robust Feature 4.4.9
WLD Wide Line Detector 4.4.3

1 Introduction

Tasks such as logging into a computer system or withdrawing money from a teller machine require the user to authenticate themselves in order legitimate their access. Traditionally, this is being done by using either token based authentication (cards, badges, keys), knowledge based authentication (passwords, personal identification number or PIN) or a mixed form of the former two authentication methods (e.g. card together with a PIN). These methods however have several shortcomings that could lead to security breaches: Too weak passwords are easy to guess, too long passwords hold the risk of being forgotten, keys can be lost or stolen, badges can be duplicated and sometimes PIN codes are written directly on the card. Also these classic authentication methods could potentially be shared with other persons. Considering these shortcomings of established authentication methods, it is often difficult for a computer system to verify whether the operating user is authorized to do so or not [66].

An alternative to the well established authentication methods is given by a field of research named *biometrics* which uses one or more properties that are inherent to the users body or to their behaviour for authentication. It is crucial to note that there exists another, similarly named field of research that refers to statistical analysis of biological data [33]. Throughout this thesis however, the term biometrics will be used to denote the process of authentication to a system by using biometric features of the human body, also known as a *biometric trait*.

James Wayman defines the meaning of biometrics or biometric authentication as: "The automatic identification or identity verification of an individual based on physiological and behavioral characteristics [89]." Some key elements of this definition demand further elaboration:

The word *automatic* suggests that there is an automated process reaching from reading a biometric trait to making a decision. This process is com-

monly denoted as a *biometric system*. Essentially such a biometric system consists of four parts. (i) Data Acquisition: A biometric reader captures the biometric sample data that is presented to it, at the same time digitizing the data so that digital signal processing techniques can be applied in the next steps. (ii) After application of appropriate preprocessing to the digitized data, features are extracted in order to a) reduce the amount of data to be stored and b) allow a certain variance during the acquisition phase. (iii) A database where the extracted features of biometric samples are stored as so called *biometric templates* and finally (iv) a decision making or *comparison* module that is able to compare newly acquired biometric samples with already registered templates.

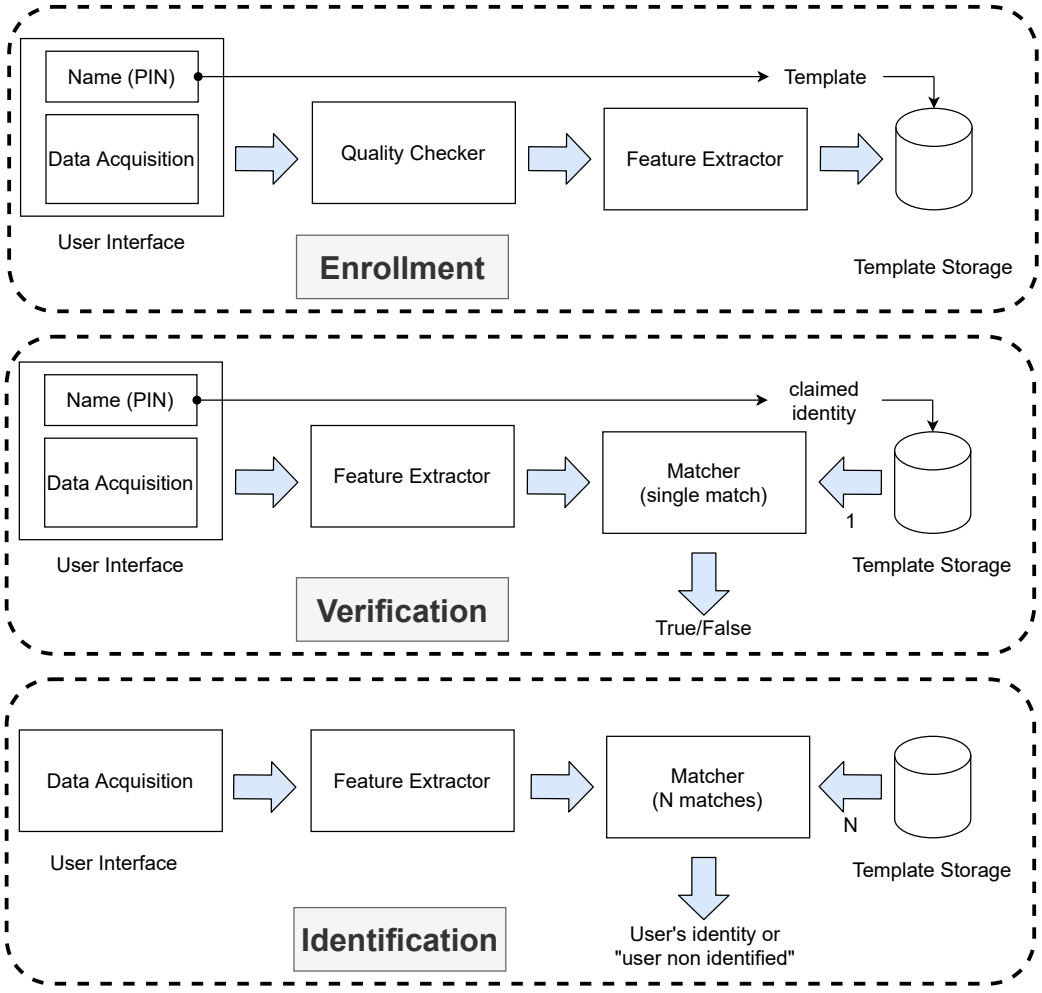

Figure 1.1: Modes of a biometric system, recreated from [32].

A biometric system is able to operate in two modes, namely *identification*

and *verification*. Both of which presuppose an initial step named *enrollment*. Figure 1.1 shows a block diagram of all modes of operation which was recreated from Jain et al. [32]. *Enrollment:* In order to perform any comparison at all, individuals need to be known or enrolled to the system. Therefore, as an initial step, preferably high-quality biometric samples have to be acquired and stored as a template. *Verification:* In verification mode, a user claims to be a certain enrolled individual and the biometric systems tries to answer the question whether the person is who they claim to be. This can be considered a one-to-one comparison since the system only needs to make the comparison between the presented biometric sample and the enrolled template corresponding to the claimed identity. *Identification:* In identification mode, the biometric system conducts a one-to-many search with the goal to identify the individual that presented its biometric data to the system. Therefore the system tries to answer the question to whom the biometric data belongs.
Historically in the field of biometrics, there has been inconsistent usage of the terms *recognition* and *authentication*. Biometric recognition is a generic term that can be used in the context of verification as well as identification task [90]. Recognition is also the successor of the word authentication which should not longer be used since it is labelled deprecated according to the standard for information technology on biometrics vocabulary [3] (i.e. ISO / IEC 2382-37:2017). The content of this thesis focuses solely on the verification task rather than identification, hence biometric recognition can be treated as a synonym to biometric verification throughout this thesis.

According to [30, 32, 31], any *physiological or behavioural characteristic* can be a biometric trait as long as it possesses the following properties:

- *Universality:* any relevant person should have the characteristic.
- *Distinctiveness:* any two persons should be sufficiently different in terms of the characteristic (i.e. have a high inter-class variability).
- *Permanence:* the characteristic should be sufficiently invariant over a period of time (i.e. have a low inter-class variability).
- *Collectability:* the characteristic can be measured quantitatively.

Furthermore, the following properties need to be considered before choosing a biometric trait for a practical application [32]:

- *Performance:* refers to the achievable recognition accuracy and speed, the resources required to achieve the desired recognition accuracy and

speed, as well as the operational end environmental factors that affect the accuracy and speed.

- *Acceptability:* indicates the extent to which people are willing to accept the use of a particular biometric trait in their daily lives.
- *Circumvention:* reflects how easily the system can be fooled using fraudulent methods.

Jain et al. [31] evaluated these properties for a selection of biometric traits based on the authors opinion using a three state scale consisting of low, medium and high. This selection includes face biometrics, fingerprint, hand geometry, iris texture, keystroke dynamics, signature and voice characteristics. Table 1.1 depicts their perception of these biometric traits in terms of the properties introduced above. This judgement suggests that there is no golden solution that fits every practical application. It is important to note however that this evaluation was published in 2004 and both processing power as well as methods for circumvention have developed since then.

Biometric trait	Universality	Distinctiveness	Permanence	Collectability	Performance	Acceptability	Circumvention
Face	H	H	M	H	L	H	H
Fingerprint	M	H	H	M	H	M	M
hand Geometry	M	M	M	H	M	M	M
Iris	H	H	H	M	H	L	L
Keystroke Dynamics	L	L	L	M	L	M	M
Signature	L	L	L	H	L	H	H
Voice Characteristics	M	L	L	M	L	H	H

Table 1.1: Comparison of several biometric identifier in terms of the properties mentioned above. The evaluation is based on the perception of the authors in [33] and is reported using a three state scale where L, M and H corresponds to low, medium and high.

1.1 Hand-Based Vascular Pattern as Biometric Trait

Another biometric identifier that has not been named so far is the vascular pattern in the hand region. Here, the network structure of blood vessels inside the human hand is used as biometric trait. According to [82], hand-based vascular pattern is a generic term that refers to on the following four locations: Finger vein, palm vein, hand vein and wrist vein. Similar to the hand, vascular pattern could also be extracted from the human eye either from the backmost layer of tissue, i.e. the retina, or from the visible area around the iris, also known as sclera [81]. The acquisition processes for eye based vascular pattern however are somewhat uncomfortable to the user and are solely addressed for the sake of completeness since both hands and eyes are the most popular regions used in the context of vascular biometrics [80].

Generally researchers assume that every subject (which can be either every finger or every hand) has a high distinctiveness to any other subject on earth and can therefore be labelled unique. In 2016 Ye et al. [96] conducted a large scale experiment on finger vein recognition involving over 350.000 subjects collected from Chinese middle schools that supports that assumption.

Since vessels in the hand area are often not visible neither to the human visual system nor to consumer imaging devices such as standard smartphone cameras, research has been going on to reveal these blood vessels using other approaches: Although not for the purpose of biometric recognition but medical imaging of the hand vascular structure in general, magnet resonance angiography, that is, injection of a contrast material followed by magnet resonance imaging, can be used for capturing blood vessels in the hand [15]. Iula et al. [29] used ultrasound scans to extract 3D palm vein patterns with the purpose of performing biometric recognition. While in the referenced research the authors label the results as not satisfactory, in later works the actual usefulness of this imaging technique for biometrics was shown [17, 28].

In 2018, a publication [49] used photoacoustic tomography (PAT, also called photoacoustic imaging) to visualize the palmar patterns of the human hand for medical analysis. PAT makes use of the photoacoustic effect which describes the conversion of electromagnetic energy into sound energy. When confronted with pulsed laser light, any material that absorbs that radiation experiences a slight change in temperature which causes the material to expand and contract itself. This change in volume, and therefore also pressure, represents an acoustic source emitting ultrasound waves. While standard

ultrasound imaging scans echos, this type of imaging measures ultrasound that is generated by various tissues in the body itself. Later that year, a different research group also used PAT for biometric recognition using palm vessels [88]. In 2020, researchers from the same group used PAT for finger vein recognition [97].
Lin and Fan [45] used the fact that veins and surrounding tissues such as fat, skin and bones differ in temperature and that the surrounding tissues possess a temperature gradient. They used a commercial infrared camera that is designed for thermal imaging of the back of the hand in a wavelength of 3400-5000 nanometers which is in the range of the so called mid-wavelength infrared. It was shown that this technique holds potential to be used for biometric recognition.
Another cost efficient acquisition technique that is being used in both commercial as well as academic area in is to use illumination in the wavelength range from 750 to 950 nm (see table 3.2 in [37]), which is a sub-band of the so called near-infrared spectrum. This particular wavelength bandwidth is based on the observation that both the oxygen saturated as well as the deoxygenated hemoglobin in the blood has a higher molar attenuation coefficient than other surrounding tissues like bones and flesh in this particular frequency range. In other words: Blood vessels absorb more light in the near infrared spectrum than its neighbouring tissues, resulting in dark structures in images if the illumination source is strong enough to penetrate the human skin. Additionally, an appropriate imaging device that is sensitive to near-infrared light is needed. Consumer cameras are usually equipped with an infrared blocking filter which could be removed in order to use such a device for imaging. There also exist infrared passing filters which block everything else except for infrared illumination that could be applied for quality enhancement [37]. This acquisition technique is also the one that will be used in the later part of this thesis.

Blood vessels in general are divided into arteries and veins depending on whether they deliver oxygen saturated blood (arteries) or they transport already consumed deoxygenated blood (veins) back to the heart. Although veins and arteries absorb more or fewer near infrared light depending on the exact wavelength that is being used as illumination source, the predominant term to refer to this type of biometric trait happens to be *vein recognition* [82].

Typically one differentiates between two perspectives, depending on which side of the finger or hand the imaging sensor is placed. Capturing the backside of the hand, or the dorsum, is denoted as *dorsal*. When image acquisition

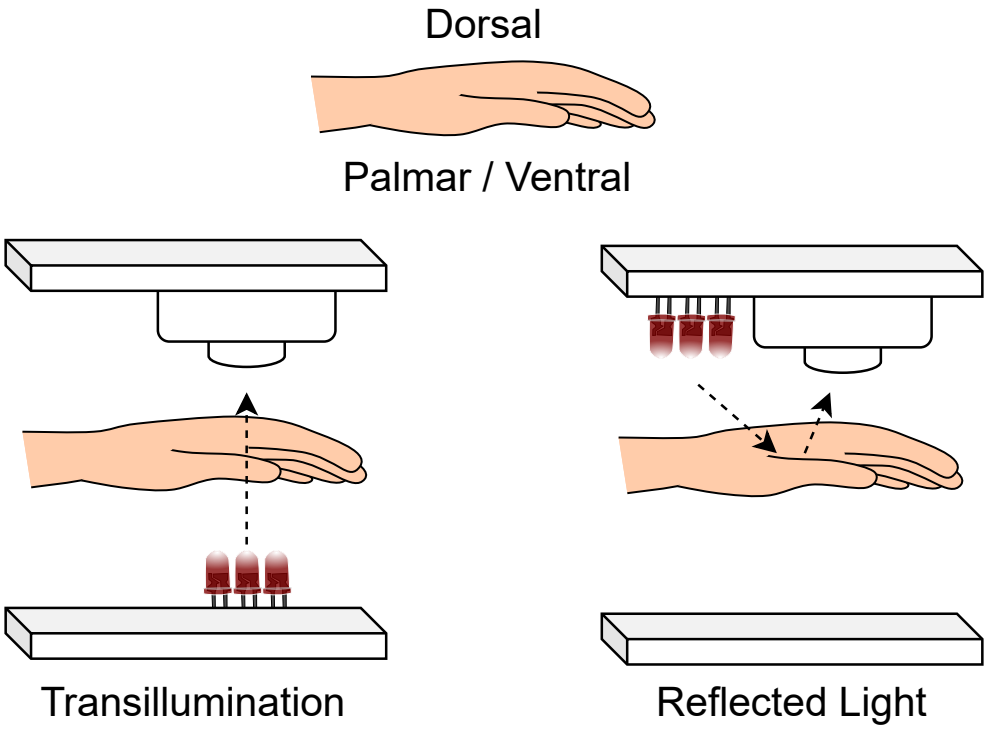

Figure 1.2: Perspectives (top row) and positioning of light source (bottom row illustrations) for vein acquisition. Illustration adapted from [68].

takes place on the inside of the hand, it is called *ventral* or *palmar*. In [36], it was tested whether comparison of dorsal and palmar finger vein samples from the same finger would yield a high similarity score, though without success. One reason for the failure was that the acquisitions also included wrinkles on the skin around the knuckles which are inherent to a certain side of the finger and influence the feature extraction.

Furthermore, the light source can also be positioned in various ways. The case where the light source is placed on the opposite side of the imaging sensor, therefore going all the way through the hand, is denoted as *transillumination*. This form of illumination also includes the case where the the light source is placed on the side in an 90 degree angle to the imaging sensor. Another illumination variant is denoted as *reflected light*. Here the light source is placed on the same side where also the imaging device is placed and as the name suggests the light which is not absorbed by the users hand but reflected is then captured again to by the imaging sensor [37]. Figure 1.2 illustrates where the light sources, the hand, and the imaging sensor is placed when

using the former mentioned illumination variants.

1.2 The Presentation Attack Problem

Security considerations for biometric systems are of high importance not only because biomtric data is inseparably linked to an individual and thus this data needs to be protected from being leaked, but also a faultless operation without fraudulent interference is expected. However, as illustrated in figure 1.3, several potential attack vectors exist. One particular attack scenario that is relevant for the scope of the present thesis (dashed box) is known as *presentation attack*. Here, a replica of a biometric sample is presented to the biometric reader with the goal of either impersonate someone or not being recognized, thereby interfering with the intended use of the biometric system. These kind of replica are defined by the ISO standard for biometric presentation attacks [1] as the generic term *presentation attack instrument* and can take various forms including face masks, replayed videos on smartphones, fingerprint mutilations, contact lenses etc.

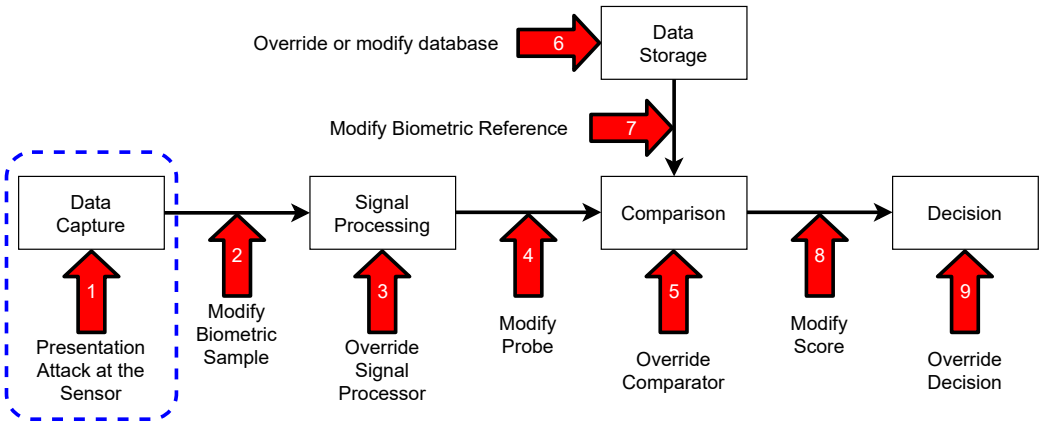

Figure 1.3: Possible points of attack in a biometric system as illustrated in the ISO/IEC 30107-1:2016 [1].

A potential security breach for a commercial dorsal hand vein system was reported in a document [22, 21] by the Future of Identity in the Information Society (FIDIS) consortium. They conducted experiments where backside of a human hand was captured and then the extracted vein pattern were printed on a piece of paper in order to first enroll their presentation attack instrument and afterwards making verification attempts. However this way to circumvent the system was only successful when the system internal live-

ness detection was disabled.

In 2013 [53], the first [1] reported successful attempts to fool a finger vein recognition algorithm was reported. They created a small scale presentation attack database from an existing finger vein database consisting of samples from seven individuals. For the generation of the presentation attacks, they printed the selected finger vein templates from the existing database on two types of paper and on overhead projector film using a laser printer.
Tome et al. [79] created presentation attacks by using a subset consisting of 50 subjects from an publicly available finger vein recognition dataset, that was initially released at the same time. Likewise, as Nguyen et al. in 2013 [53], a laser printer was used for presentation attack creation and additionally the contours of the veins were enhanced using a black whiteboard marker. One year later, in 2015, the first competition on counter measures to finger vein spoofing attacks was held [78] by the same authors as in [79]. With this publication, their presentation attack database was extended such that every biometric sample from their reference database has a corresponding spoofed counterpart. This was the first complete finger vein presentation attack database freely available for research purposes.

A selection of other approaches for the creation of hand and finger vein presentation attack instruments is described hereafter:

Instead of simply printing a vein sample, [60] suggested to print the sample on two overhead projector films and sandwich a white paper in between the aligned overhead films.

Another experiment was shown in [59]. Here dorsal hand veins were enrolled as intended. As presentation attack instrument, a smartphone and additional infrared illumination was used to first capture dorsal hand veins and afterwards the previously captured hand veins were shown on the smartphone display.

In 2016, researchers published [58] a working example of what they call 'wolf attack'. This type of attack has the goal to exploit the working of a recognition toolchain to construct an attack sample that will generate a high similarity score with any biometric template stored. They showed that their master sample worked in most of the finger vein verification attempts, therefore posing a threat to this particular recognition toolchain used.

[1] Several sources [16, 83, 79, 78] refer to a publication or its corresponding set of presentation slides from Prof. T. Matsumoto that fooled a commercial finger vein recognition system already in 2007, however no primary source could be found online neither in English nor in German at the time this thesis was created.

Although not aimed to be an attack instrument, [55] published preliminary results for a finger phantom that uses 3D print material, soap and titanium dioxide. Even though the vein structure is yet to be developed, this approach holds the potential for being used as a presentation attack instrument once custom 3D finger vein phantoms can be generated.
German hackers [40] used an extracted hand vein pattern that was encased by beeswax to spoof a commercial hand vein recognition device in a live demo at a hacking conference. The hand vein pattern was acquired from a distance of a few meters, which finally works towards giving an answer to a question what most publications do not consider, namely: "How does one acquire someones hand vein pattern without them noticing?". Besides that, they suggested that hand dryers in the restroom could be manipulated to secretly capture hand vein images.

Detailed descriptions of the presentation attack databases used in this thesis are given in chapter 5.

1.3 This Thesis

The main goal of the present master thesis is to investigate the usage of extracted vital information, such as heart rate, from video sequences of vascular patterns in hand regions, in order to perform presentation attack detection. For doing so, one dorsal hand vein and one finger vein video data set, both of which were acquired in the Multimedia Signal Processing and Security Lab at the University of Salzburg, are being used for this evaluation. This task can further be divided into three sub tasks and the corresponding research questions (RQ) be defined as:

(i) Finding applicable methodologies for presentation attack detection in finger and hand vein biometrics using video sequences.

RQ1: What is the state of research for vein presentation attack detection using video sequences?

(ii) Evaluation of threat emitted by a given finger vein and hand vein database by using a common threat evaluation protocol and a variety of different recognition methodologies.

RQ2: Are the attack databases used in this thesis capable of deceiving state of the art recognition methodologies?

(iii) Evaluation of liveness detection methodologies on the finger vein and hand vein dataset.

RQ3: How effective are the methodologies from RQ1 when performing presentation attack detection on the data sets introduced in RQ2?

The remainder of this thesis is structured as follows: Chapter 2 gives an overview of related hand vein presentation attack approaches, excluding methodologies that try to reconstruct vital signs from consecutive video frames. RQ1 will be covered in chapter 3. The experimental set up and recognition algorithms for RQ2 are described in chapter 4. Chapter 5 explains the data sets and 6 contains the experimental results of both threat analysis (RQ2) and liveness detection (RQ3). Finally a summary of the present thesis is given in chapter 7.

2 Still Image Presentation Attack Detection

Presentation attack detection in vascular hand biometrics is a widely researched topic where various publications exist. However, the scope of this thesis is restricted to using video sequences as input data to achieve presentation attack detection. Other works analyze properties like texture, quality or spatial frequency from still images. Therefore this chapter aims to give an overview of related publications that operate on a single-image basis. The methods are divided into approaches that were designed for finger or dorsal hand veins, respectively. However, judging by quantity of publications that descend from both categories, finger veins seems to be the more popular field of research. The methods are ordered chronologically.

Finger Vein Probably the first shown to be working solution for finger vein presentation attack detection was proposed by Nguyen et al. [53] in 2013. In this pioneer work, spatial information as well spatial frequency is extracted from every vein image by using Fourier transform, Haar wavelet and Daubechies wavelet transform to calculate three different scores. Using a support vector machine (SVM), the three scores are combined to arrive at a final decision.

In 2015, the *1st competition on counter measures to finger vein spoofing attacks* [78] was held where three teams participated along with the organizing team that delivered a baseline method. The baseline uses a method coined Fourier spectral bandwidth energy, in which first the average vertical energy from the Fourier spectrum is extracted and afterwards the bandwidth is calculated using a cutoff frequency of $-3dB$. The methods from the contesting teams are summarized as follows: (i) Binarized statistical image features (BSIF), where each pixel is represented as a binary code obtained by using filters that are learned using statistical properties of the images. (ii) One team uses monogenetic scale space based global descriptors, that capture

local energy and local orientation at a coarse level. (iii) The third team participated with two approaches, one fusing local phase information generated by Local Phase Quantization (LPQ) with a Weber Local Descriptor, and the other one using a local binary pattern approach. All approaches employ an SVM which led to decent attack detection results on the attack database.

Also in 2015, Triunagari et al. [77] presented a procedure that is based on Dynamic Mode Decomposition (DMD) which originally descends from the area of computational fluid dynamics. They proposed a windowed version (W-DMD) extracts texture information from an input image that was shown to be superior to other texture descriptors.

Later that year, Raghavendra and Busch [63] published a study where they used steerable pyramids for texture analysis together with SVM classifier on a private finger vein attack database. The effectiveness was demonstrated by comparing their approach to other presentation attack detection schemes.

In [39], Kocher et al. tested a variety of Local Binary Pattern (LBP) texture descriptors on its applicability to discriminate between real and fake finger vein samples. In addition to the baseline LBP scheme, seven LBP extensions are evaluated in their research. Their conclusion was that more sophisticated LBP variants do not necessarily imply an increase in detection accuracy and baseline LBP is perfectly suited for the task of detecting finger vein presentation attacks.

The first convolutional neural network (CNN) approach for detecting fraudulent finger vein samples was introduced in 2017 by [64]. Here, a pre-trained network which is often referred to as AlexNet [41] is used and extended by seven additional layers that alternate between fully connected and dropout layers. Their results clearly outperform handcrafted methods, although only tested on private data sets. A very similar approach was published in 2018 [65] by the same authors as a chapter in a book. Another CNN based approach in 2017 was published by Nguyen et al. [52]. Multiple experiments were carried out using a CNN that was adjusted using transfer learning. They applied principal component analysis dimension reduction after the CNN embedding and finally used an SVM for classification. Later in 2017, Qiu et al. [61] designed a shallow network expecially for the finger vein presentation attack detection task, named FPNet. They employed data augmentation to artificailly increase the amount of training data since they claim that existing finger vein attack datasets are of insufficient scale.

Also in 2017, Bhogal et al. [10] analyzed the applicability of non-reference image quality measures for the presentation attack detection task. In total, they included 6 image quality metrics in their study that were tested separately as well as in combination. Classification is done using k-nearest-neighbour classification. Results show that the use of quality metrics is dataset dependent, however accuracy values above 99% can be achieved. In 2018 this work was extended in [76] by using natural scene statistics (NSS) and adding support vector machine classification. Experiments were conducted for a variety of biometric traits, including finger veins and experimental results showed imnprovements to their prior work. Two years later parameters of asymmetrical generalized Gaussian distributions, which count to NSS, were used in Debiasi et al. [18] for a presentation attack detection study for finger and hand vein data.

Qiu et al. [60] used a method named Total Variation Decomposition to disassemble a given finger vein sample into a structure and a noise component. They then applied blockwise LBP feature extraction to feed a cascaded SVM model and could achieve perfect results on multiple datasets.

In 2019, Singh et al.[71] proposed a single image decomposition method that divides a given input image into a normal-map and a diffuse-map that contains 3D shape and material properties, respectively. This is achieved using the so called Sfs-Net [70]. Features are extracted from the resulting components using texture LBP, BSIF and LPQ. For classification, SVMs are employed whose results are analyzed independently as well as using score level fusion.

Later in 2019 Maser et al. [47] evaluated the use of Photo Response Non-Uniformity, a method commonly used for sensor identification by estimation of an image sensors 'fingerprint', in order to detect presentation attacks.

Another CNN-based approach was proposed in 2020 by Yang et al. [95]. They created a lightweight unified CNN model named FVRAS-Net to accomplish presentation attack detection as well as extract features for the recognition task at the same time.

Two recent publications that descend from the same research unit contributed additional hand-crafted approaches to the finger vein attack detection research. Lee et al. [44] analyzed the usage of three LBP variations together with three classification schemes and Ashari et al. [5] suggested to use histograms of oriented gradients.

Dorsal Hand Vein For the domain of dorsal hand vein biometrics, Wang et al. [87] proposed a presentation attack detection scheme that is based on spatial frequency. They divided the 2D frequency spectrum into three regions and calculated the spectral energy for each region. Using an SVM, a classification accuracy of up to 99% was reported.

In 2014, Wang et al. [85] proposed the following: First a training split consisting only of real hand vein data is used to generate a projection space using principal component analysis (PCA). Test data samples, now also including attack vein images, are then projected into this space to extract 1D noise information. Next, an autoregressive (AR) model is established in order to estimate the power spectrum of the extracted information which is then used for classification. In another publication from the same authors in 2016 [86], a similar procedure was reported.

In 2017, Jiang et al. [34] suggested to use a contrast ratio on self created attack samples. However it was not shown whether their attack samples would be able to deceive a hand vein recognition system.

In the same year, Bihlare et al. [9] developed an approach that first filters dorsal hand vein images with Laplacian of Gaussian filters and afterwards extracts histograms of oriented gradients from pixel blocks at three different block sizes. Classification is achieved for every scale using support vector machines. A combined decision is made using a majority voting.

3 Presentation Attack Detection using Video Sequences

Altogether this thesis includes four methods for presentation attack detection, where each operates by looking for vital signs in adjacent video frames, which are described in this chapter. All of the methods are evaluated on their performance on the video data sets described in section 5.1. The experiments and results can be found in chapter 6. The current chapter is dedicated to introduce the aforementioned methods. While the methods in section 3.3 and 3.4 were developed by the author in the course of this thesis and published in [68], the methods described in section 3.1 and 3.2 descend from different publications. With the exception of the algorithm that uses Eulerian Video Magnification (described in section 3.1), all methods build upon a common basis that transforms a given input video sequence into a one-dimensional time series. To do so, every frame (either as a whole, or a given region of interest where the rest of the frame is simply ignored) is averaged in terms of pixel grayscale values.

This can be seen as a form of Plethysmography which refers to the act of measuring changes in volume in various areas of the body such as lung capacity or blood volume. Sometimes it is also possible to make measurements in volume by analysing optical signals that are acquired through imaging devices, hence named Photoplethysmography (PPG). Blood has, when illuminated with infrared light, a higher absorption coefficient than its surrounding tissue components. Therefore, when using strong enough infrared light which penetrates the skin (either in transmissive or reflected light illumination) parts of the illumination get absorbed by the blood and are not recaptured again by an imaging sensor. Since the amount of blood oscillates with every cardiac cycle, a series of consecutive images captured with a high enough rate results in subtle variations in the detected light intensity [54]. In the medical area, this non-invasive method has been widely used

for monitoring purposes. A simple way to potentially extract vital information out of such a generated time series is to look for recurring peaks or looking for a dominant peak in the frequency domain that would indicate heartbeat. However, wearable photoplethysmography sensors, such as pulse oxymeters, are appended tightly to the human skin, eliminating unwanted extra illumination from external light sources. Also monitoring devices do not need to resolve vein structures and therefore simple illumination sensors are sufficient. Thus, since for the biometric recognition task vein images are captured from a distance, it is potentially prone to errors. Ding [19] for example showed that when using the average minima and maxima from the time series as features for classification it can be potentially tricked by a blinking LED that imitates pulse. Hence, the methods in sections 3.2, 3.3 and 3.4 try to avoid this issue by applying hand crafted transformation operations into another feature space. The resulting feature vector for every video sequence is then classified as either bona fide or attack video sequence.

3.1 Eulerian Video Magnification

This approach from Raghavendra et al. [62] can be split in two main parts that are applied in succession. The first part is Eulerian video magnification (EVM). The underlying algorithm was initially published by Wu et al. [93] and the core idea of EVM is to artificially amplify tiny motions in video sequences. The wording *Eulerian* is derived from fluid mechanics and is meant to differentiate this particular approach, which amplifies the variation of pixel values over time, from motion estimation approaches that track pixels over time (which, as reference to fluid mechanics, they call *Langragian* approaches). In short, EVM consists of three steps that are also shown in figure 3.1: (i) First, every frame is separately decomposed into spatial sub bands using a Laplacian pyramid. Frames of every pyramid level are still treated as a video such that the 3D signal can be processed with respect to the temporal axis. (ii) Second, the resulting sub band video signals are processed using temporal filtering by applying an ideal band-pass with a lower and upper cutoff frequency λ_l and λ_h. This operation is carried out on every pixel over the whole video sequence. Additionally, a spatial cutoff frequency λ_c needs to be defined beyond which the factor α has less or no effect. Next, the filtered signal is amplified using an amplification factor α, which is other than indicated in figure 3.1 always the same factor in the implementation for this thesis, and added to the unfiltered sub band signal. (iii) Finally, the signal is reconstructed by collapsing the pyramid.

The second part of the algorithm proposed by Raghavendra et al. [62] is

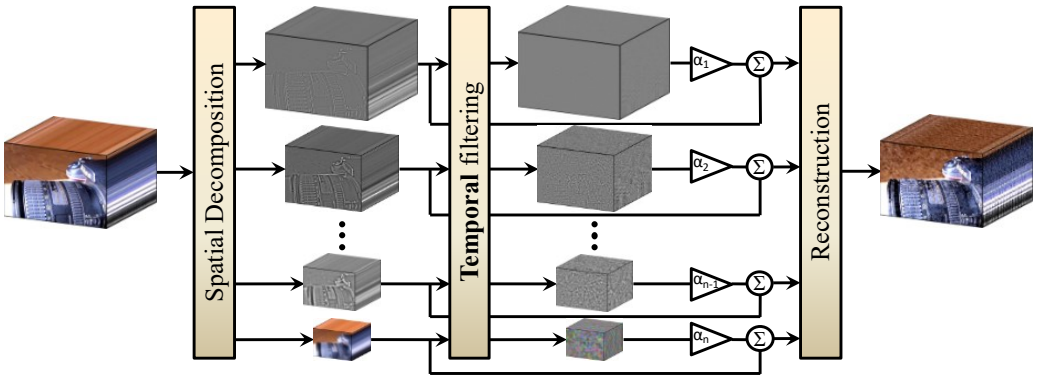

Figure 3.1: Block diagram of EVM algorithm; Taken from Wu et al. [93].

the computation of the so called optical flow. The term optical flow describes a pixel-wise displacement field between two images where either an element in the image moves, the camera position changes or a mix between many motions. It describes the velocity with which a certain pixel moved from its initial position in one image, to a different position in a second image [12]. For both, Motion Magnification [2] and Optical Flow [3], available Matlab implementations are used. An example of such a video magnification and corresponding optical flow estimation is given in figure 3.2.

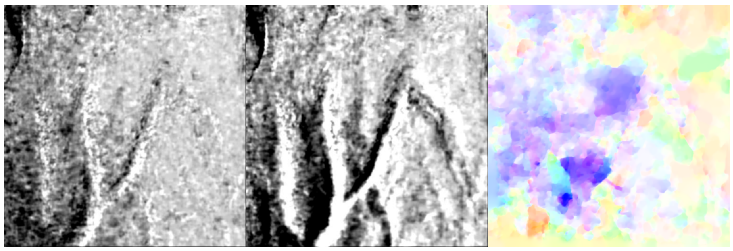

Figure 3.2: Left and center: Frame 1 and frame 7 from a motion magnified video sequence; Right: Optical Flow computed between the left images. The source video sequence used for the creation of this figure was taken from the database described in section 5.1.2.

The authors of [62] used video sequences of length 1.67 seconds recorded with a frame rate of 15 frames per second, resulting in 25 frames for each sequence. Let a single video sequence be denoted as V and the EVM processed video as $V_{EVM} = \{F_{E1}, F_{E2}, ..., F_{E25}\}$, where F_{En} are its processed frames. Then, using the optical flow operation OF, the flow in vertical M_y

[2] `https://people.csail.mit.edu/mrub/evm/#code`
[3] `https://people.csail.mit.edu/celiu/OpticalFlow/`

and horizontal M_x direction is calculated using the first and last frame as seen in equation 3.1.

$$[Mx,\ My] = OF(F_{E1},\ F_{E25}) \tag{3.1}$$

To arrive at a single number representing one particular video sequence, a motion magnitude $Motion_{Mag}$ is computed as seen in equation 3.2.

$$Motion_{Mag} = \sum_j \sum_k \left(\sqrt{[(M_x)^2\ +\ (M_y)^2]} \right) \tag{3.2}$$

Finally, a video sequence can be classified either bona fide or attack sample by simple thresholding (equation 3.3).

$$D_e = \begin{cases} bona\ fide, & \text{if } Motion_{Mag} \geq T \\ attack, & \text{otherwise} \end{cases} \tag{3.3}$$

Herzog and Uhl [24] adapted this approach to evaluate its functionality on the dataset described in section 5.1.2. They concluded, however, that applying additional motion to the attack instruments strongly reduces the ability to detect presentation attacks using this approach.

3.2 PPG-based by Bok et al.

This approach was proposed by Bok et al. [11] where they used it for separating real finger vein video sequences from forged ones. The starting point for this approach is the one-dimensional time series that was generated from calculation of the average pixel brightness in every frame as described in the beginning of this chapter. The basic idea of this approach is to transform the time series into Fourier space using the discrete Fourier transform and then use the magnitudes from individual frequency components as a feature vector. To ensure that the used frequency components are always the same, zero padding is applied to the time series prior to transformation into Fourier space, such that the following equation 3.4 for frequency resolution Δf is satisfied, i.e. a fixed spacing of 0.04 Hertz. The variable n_{ts} denotes the number of samples in the time series, fps denotes the frames per second with which a video sequence was captured and zp is the zero padding.

$$\Delta f = \frac{fps}{(s_{ts} + zp)} = 0.04Hz \tag{3.4}$$

As a next step, frequency components less than 1.0 and greater than or equal 3.0 Hertz are discarded, since normal heart rate of people appears to

be in the range of 60 and 180 beats per minute. What is left are 50 frequency components. By taking the magnitude of those frequency components, a 50-dimensional feature vector is constructed. For classification a support vector machine with linear basis function kernel is used. Figure 3.3 illustrates the feature vector extraction from the time series.

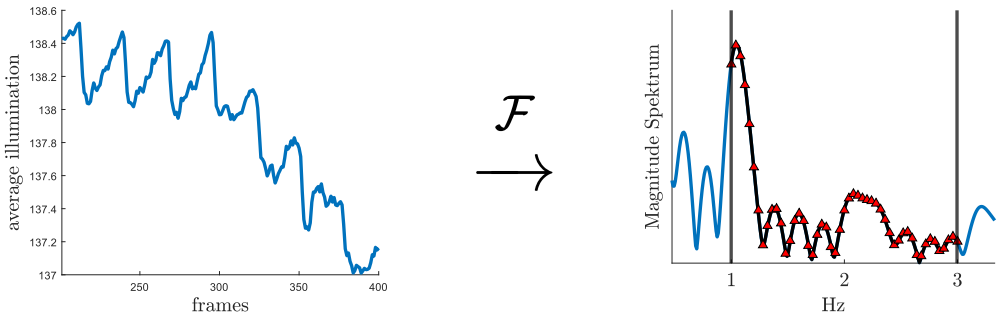

Figure 3.3: Left image: Time series containing averaged brightness values; Right image: Fourier domain, upside triangles have a spacing of 0.04 Hertz and are used for feature vector construction.

3.3 PPG-based using a windowed majority voting

As a starting point for this method, again the time series containing average pixel illuminations from a grayscale video sequence is used as described at the beginning of this chapter. As a next step, a rectangular window of size ws is applied to the time series and shifted over its length with a step size of ss data points (averaged frames) in the time series sequence. To achieve detrending and also to remove other artefacts that appear in the lower frequencies, a steep high-pass filter with cutoff frequency of $f_{cut} = 0.5\ Hz$ is applied to every window. Afterwards a zero padding of zp zeros is done by simply attaching zeros at the end of the signal. This is done to increase the resolution in frequency space after the next step. The windowed, filtered and zero padded signal is then transformed into Fourier space using the discrete Fourier transform.

For every window, the global $argmax(\mathcal{F})$, that is, the frequency where the magnitude spectrum has its highest peak, is temporary stored. From this sequence of maximum frequency per window, a histogram with bin size $bs = 0.05Hz$ is generated. Values below 0.5 Hz and above 2 Hz are ignored

and therefore do not contribute to the histogram. The values in the histogram are then normalized since some windows may have its peak outside the range and also video sequences do not necessarily need to be of same length. The final feature vector for this method is given by the normalized histogram and depicts a form of majority voting per window. The feature vectors, which are of dimension 31, are then classified using an SVM. The process of generating a feature vector with this method is shown in figure 3.4.

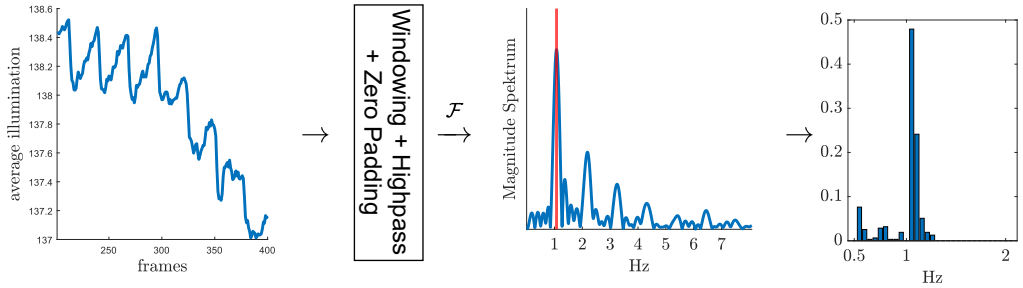

Figure 3.4: Feature vector creation using a windowed majority voting of the most dominant frequency and forming a histogram.

3.4 PPG-based with windowed analysis of harmonics

For this method, similar to the method described in section 3.3, the time series of averaged brightness values per frame is windowed, filtered using a high-pass filter, zero padded and finally transformed into frequency space using discrete Fourier transform. Instead of simply picking the frequency where the spectrum has its maximum magnitude, also the following observation is used.

Wei et al. [91] realized that blood pressure measurements do not only contain information about the heart rate, but also include harmonics, that is, integer multiples of a dominant fundamental frequency, whose magnitudes can be mathematically modelled through a decaying exponential function. This approach uses this observation for the construction of a feature vector that, besides to the single most dominant frequency in the spectrum like in 3.4, also takes into account the magnitudes of its assumed harmonics.

Therefore, let us define the global $argmax(\mathcal{F})$ as f_{HR} and the $max(\mathcal{F})$ as m_{HR}, where $\mathcal{F}$ is the magnitude spectrum (i.e. phase information of the frequency spectrum is ignored). If f_{HR} is the heart rate, then due to

the observations made in [91], one would expect local maxima at the harmonic frequencies (integer multiples of f_{HR}) with a certain magnitude as well. Therefore, for the first h harmonics (i.e. $n * f_{HR}$, $n \in \{2, ..., h\}$) a search window of $\pm\frac{f_{HR}}{5}$ is defined. Choosing h depends on the sampling frequency since it has a direct influence on f_{max}, which is the frequency at half the sampling frequency f_S. In addition to f_{HR} and m_{HR}, the local argmax and corresponding maxima in the search window are temporary stored as the a quotient with respect to the global argmax & maxima as f_i and m_i. The final feature vector x_v for every video sequence v is constructed by calculating the arithmetic means and medians (Md) for all stored values over every window as seen in equation 3.5.

$$x_v = \big(\overline{m_{HR}}\,,\,\overline{m_1}\,,\,...\,,\,\overline{m_4}\,,\,Md(m_{HR})\,,\,Md(m_1)\,,\,...\,,\,Md(m_4),\\ \overline{f_{HR}}\,,\,\overline{f_1}\,,\,...\,,\,\overline{f_4}\,,\,Md(f_{HR})\,,\,Md(f_1)\,,\,...\,,\,Md(f_4)\big) \tag{3.5}$$

Exceeding f_{max} with a search window counts as 0 for the entries in question. This case can only occur if $f_{HR} * 5 + \frac{f_{HR}}{5} \geq f_{max}$, which is usually out of range for a reasonable heart rate. Similar to the previous PPG-based approaches, the feature vectors are then classified using a support vector machine. Figure 3.5 depicts the process of extracting the information from f_{HR} and its harmonics.

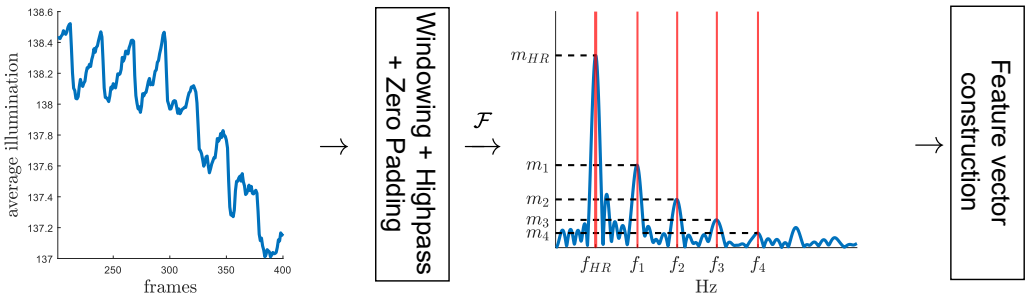

Figure 3.5: Construction of feature vector for the approach that analyses f_{HR} and its harmonics.

4 Recognition Algorithms and Framework for Threat Analysis

This chapter describes the general setup used for the threat evaluations in chapter 6 to answer *RQ2*. This setup includes the evaluation protocol that is being used to do so as well as the used recognition toolkit, its settings and the algorithms it provides. The used databases are described in chapter 5. Since these databases were not captured using commercial vein scanning devices but are collected using self built scanners, the databases simply consist of digital photographs. Therefore, in order to perform comparison experiments, one images is always treated as the one that would have been enrolled to a biometric system and another one is used for performing a verification attempt. In figure 4.1, the comparison pipeline is depicted where *Image A* is the quasi-enrolled template and *Image B* the one for performing a verification attempt. In the case where A and B are the same image, the similarity score would always be perfect. This case is avoided since in a real life scenario, two vein acquisitions made in different sessions will never yield identical digital images. When using two different images from the same subject (either the same finger or the same hand) one calls this case *genuine* attempt. The case where image A and image B descend from different subjects is usually denoted as *zero-effort impostor* or simplified *impostor* attempt. Finally, a third case is given by using a biometric sample from an actual human (also called *bona fide* sample) as image A and set a forged biometric sample (*presentation attack*) as image B with the goal of achieving a high similarity score such that the biometric system would be deceived. This case is denoted as *attack* attempt.

In order to evaluate the level of threat that presentation attack samples pose to various feature extraction and comparison schemes, a commonly known threat evaluation protocol is used that is described in section 4.1.

The threat evaluation experiments in this thesis employ twelve feature extraction schemes, together with appropriate pre- and postprocessing steps where the extracted features are finally compared using corresponding com-

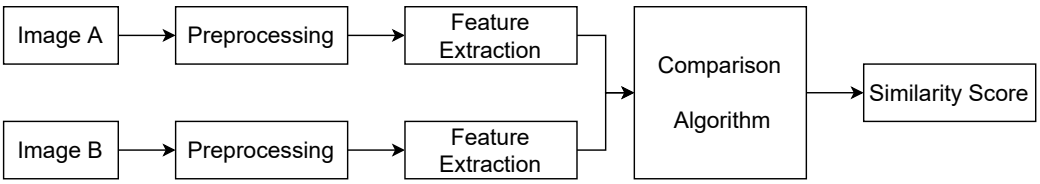

Figure 4.1: Comparison Pipeline

parison algorithms that yield a single similarity score. With the exception of one feature extraction and comparison algorithm (convolutional neural network described in 4.4.12), a publicly available vein recognition toolkit [38] is used that is described in section 4.2.

4.1 Threat Analysis Evaluation Protocol

To evaluate the level of threat exhibited by a certain database, an evaluation scheme which is known as "2 scenario protocol" [79, 59, 13], is adopted in this work. The two scenarios are briefly summarized hereafter (descriptioon taken from [69]):

- **Normal Mode**: The first scenario employs two types of users: Genuine (positives) and zero effort impostors (negatives). Therefore, both enrollment and verification is accomplished using bona fide finger vein samples. Through varying the decision threshold, the False Match Rate (FMR, i.e. the ratio of wrongly accepted impostor attempts to the number of total impostor attempts) and the False Non Match Rate (FNMR, i.e. the ratio of wrongly denied genuine attempts to the total number of genuine verification attempts) can be determined. The normal mode can be understood as a recognition experiment which has the goal to determine an operating point for the second scenario. The operating point is set at the threshold value where the FMR = FNMR (i.e. Equal Error Rate, EER).

- **Attack Mode**: The second scenario uses genuine (positives) and presentation attack (negatives) users. Similar to the first scenario, enrollment is accomplished using bona fide samples. Verification attempts are performed by comparing presentation attack samples against their corresponding genuine enrollment samples or templates. Given the threshold from the licit scenario, the proportion of wrongly accepted presentation attacks is then reported as the Impostor Attack Presen-

tation Match Rate (IAPMR), as defined by the ISO/IEC 30107-3:2017 [3].

4.2 PLUS OpenVein Finger- and Hand-Vein Toolkit

The PLUS OpenVein Toolkit [38] is a vein recognition framework that was created with the goal of providing a modular solution for finger an hand vein recognition to reduce the effort of implementing all necessary steps over and over again. Due to its modular design, new databases or algorithms can be added with relative small code and configuration adjustments. It is realised in Matlab and is available for download for research and non-commercial purposes from a publicly accessible gitlab repository [4]. The software provides a solution that includes reading input images, application of preprocessing, feature extraction, postprocessing and comparison of extracted features. The results can be output in terms of metrics over the whole database as well as table of similarity scores of the executed comparisons.

The toolkit can be used in "probe only" mode or in "gallery" mode. In the probe only mode, samples are taken from the same directory and therefore comparisons of samples with itself are omitted. This mode was used in the normal scenario from section 4.1. In the gallery mode, samples are taken from different directories, containing different samples using the same subject and sample ID (e.g. presentation attacks). Here an option setting also allows to compare samples with the same ID. This mode was used for the attack scenario as described in section 4.1. Figure 4.2 visualizes both modes.

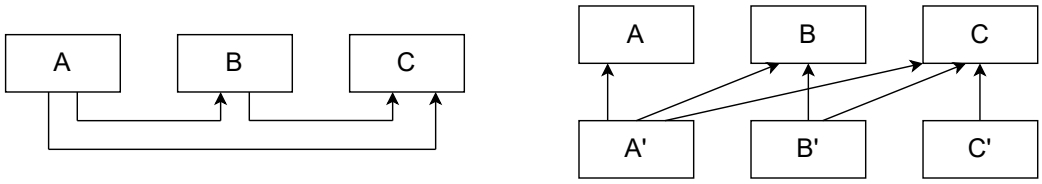

Figure 4.2: Comparisons done in probe only (l.) and gallery (r.) mode. Images A, B, C and A', B', C' are in one directory respectively.

For recognition experiments of an image database one can choose from multiple comparison protocols, that is, a definition of what samples should be compared. This allows that for large databases, exhaustively performing

[4]The Toolkit can be downloaded from `https://gitlab.cosy.sbg.ac.at/ckauba/openvein-toolkit`

every possible comparison can be avoided while the resulting error rates are still a good approximation. In this thesis, two comparison protocols are used that are described hereafter. The number of genuine comparisons is denoted as $n_{genuine}$, the number of impostor comparisons as $n_{impostor}$. $n_{subjects}$ is the number unique subjects. For the case of finger vein recognition, every individual finger is treated as unique subject. For the general formulas in this section, is is assumed that every subject has the same number of samples $n_{samples}$.

- FVC: The FVC protocol was named after a competition called fingerprint verification contest where this scheme was adopted from and with this protocol all samples are compared against all remaining samples descending from the same subject as genuine comparisons as given by the equation 4.1. Symmetric comparisons are omitted, meaning that after the comparison image A with image B is done, the comparison image B with image A will not be performed. This was chosen due to the fact that often comparison algorithms are symmetric and would not yield any new results. It is important to note however that the correlation based comparison algorithm described in section 4.4.1 for example is not symmetric.

$$n_{genuine} = \frac{n_{samples} * (n_{samples} - 1)}{2} * n_{subjects} \tag{4.1}$$

  For the Impostor comparisons, the FVC protocol defines that only the first sample of every subject is compared against the first of the remaining subjects. The number of impostor comparisons is given by equation 4.2 and only depends on the number of unique subjects regardless of the number of samples per subject.

$$n_{impostor} = \frac{n_{subjects} * (n_{subjects} - 1)}{2} \tag{4.2}$$

- Full: The full protocol defines that every possible comparison is performed while, similar to the FVC protocol, symmetric comparisons are omitted. Therefore the number of genuine comparisons is given by equation 4.1 and the number of impostor comparisons is given by equation 4.3.

$$n_{impostor} = \frac{(n_{subjects} * n_{samples}) * (n_{subjects} * n_{samples} - 1)}{2} \tag{4.3}$$

The algorithms described in sections 4.3 and 4.4 are, with the exception of the CNN related ones, implemented in this toolkit.

4.3 Preprocessing

As a first step in the comparison tool chain, preprocessing is applied to every vein sample in order to enhance the vein structures for easier feature extraction. Common image preprocessing functions such as re-scaling, conversion to double-precision floating point since Matlab reads images as 8-bit unsigned integer format, and 2D Gaussian smoothing filtering is provided by the Matlab image processing toolbox. More sophisticated algorithms that are implemented in the OpenVein toolkit and used in this thesis for preprocessing are explained hereafter.

4.3.1 Contrast Limited Adaptive Histogram Equalisation

Histogram equalisation, which is the basis of Contrast Limited Adaptive Histogram Equalisation (CLAHE) [100], has the goal to enhance the contrast of a digital image based on the assumption that adjusting the histogram of an image proves useful for many applications. An ideal uniform distribution in a histogram would result in a linear cumulative histogram. This is achieved by defining a transformation function T given in equation 4.4 that maps every pixel value to a new one as given in equation 4.5. The pixel intensity value k is in the range of 0 to 255 and p_n denotes the normalised occurrence of pixel intensities, i.e. the sum is the cumulative histogram. The histogram equalised image is denoted by $I'(u, v)$ and the input image as $I(u, v)$.

$$T(k) = \left\lfloor k_{max} * \sum_{n=0}^{k} p_n \right\rfloor \tag{4.4}$$

$$I'(u, v) = T(I(u, v)) \tag{4.5}$$

A modification known as adaptive histogram equalisation applies this technique using only local neighbourhoods around every pixel. The computation of both the histogram and the cumulative histogram for every pixels neighbourhood proves costly in terms if performance. Therefore it is suggested that an image should be divided into 8x8 non-overlapping contextual regions for which a histogram and a corresponding cumulative histogram is computed. A value for every pixel can then be received by applying bilinear interpolation of the mappings from the neighbouring contextual regions that includes the region where the pixel is as well as up to three neighbouring contextual regions. The weights for the interpolation are chosen according to the distance to the centers of the regions. The final step for CLAHE is

to define a threshold and apply a clipping on the histogram bins meant to limit the contrast. Since the overall number of pixels contributing to the histogram must not change, the clipped amount is equally divided and added to all bins.

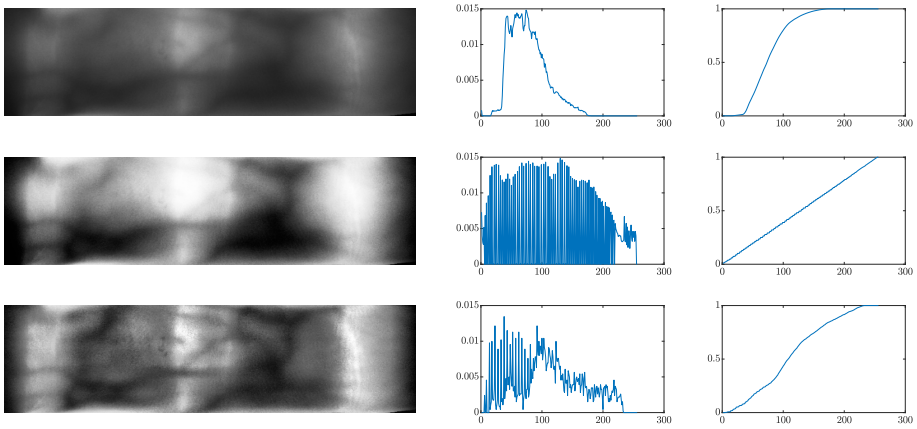

Figure 4.3: Top row: Normal image; Middle row: global histogram equalisation according to equations 4.4 and 4.5; Bottom row: *adapthist* function from Matlab that uses CLAHE with a clipping limit of 0.015.

Figure 4.3 shows examples of global histogram equalisation and contrast limited adaptive histogram equalisation together with intensity value histograms (second column) and cumulative histograms (third column). It is important to note that the CLAHE version (bottom row) does not show a perfect global linear cumulative histogram due to the fact that the adjustment is not made globally but using local multiple context regions and also interpolation as described earlier.

4.3.2 High-Frequency Emphasis Filtering

Zhao et al. [99] used 2D Butterworth high-pass filtering in the frequency domain due to the observation that vessel structures in vein images can be found using high frequency information since they appear relatively dark with respect to the surrounding tissues, thus having abrupt changes. Let the Fourier transformed input image be denoted as $F(u, v)$, the filtered image in Fourier domain as $G(u, v)$ and the Butterworth high-pass filter as $H(u, v)$, where u and v are pixel coordinates. The transfer function of the Butterworth high-pass filter of order n is defined as seen in equation 4.6.

$$H(u,v) = \frac{1}{1 + \left(\frac{D_0}{D(u,v)}\right)^{2n}} \tag{4.6}$$

$D(u,v)$ refers to the Euclidean distance of the pixel at coordinate (u,v) to the DC component. Usually 2D Fourier space is shifted such that this DC component can be found at the center. D_0 is a positive constant where the cutoff frequency can be adjusted. The transfer function $T(u,v)$ of the high-frequency emphasis filter is further defined as

$$T(u,v) = a + bH(u,v) \tag{4.7}$$

where b is an emphasis factor and the offset a has to goal to retain the gray level tonality from the lower frequency components. The convolution theorem tells us that for filtering in Fourier space, both the transformed filter and the target are simply multiplied. Therefore we derive the equation for the filtered image in Fourier domain as seen in equation 4.8.

$$\begin{aligned} G(u,v) &= T(u,v)F(u,v) \\ &= \left[a + \frac{b}{1 + \left(\frac{D_0}{D(u,v)}\right)^{2n}}\right] F(u,v) \end{aligned} \tag{4.8}$$

The high-frequency emphasis filtered image can now be obtained by application of inverse 2D Fourier transform to $G(u,v)$. Note that if the input image in Fourier domain was shifted at the beginning, one needs to reverse that shifting before applying inverse transformation.

4.3.3 Circular Gabor Filtering

Zhang and Yang [98] combined a contrast enhancement algorithm named gray-level grouping together with circular Gabor filters as a finger vein enhancement method. For the experiments in this thesis, the gray-level grouping was replaced with CLAHE (described in subsection 4.3.1), therefore it is not explained in detail. A Gabor filter is defined as the product of a Gaussian function complex sinusoidal signal of certain frequency and direction. For the case of a circular Gabor filter, the sinusoidal signal has no direction but starts at the center and expands simultaneously in every direction. The circular Gabor filter $G(u,v)$ can therefore be mathematically defined as given in equation 4.9.

$$G(u,v) = g(u,v)exp\left[i2\pi f_c\sqrt{u^2+v^2}\right] \tag{4.9}$$

Where $g(u,v)$ denotes an isotropic Gaussian envelope as given in equation 4.10.

$$g(u,v) = \frac{1}{2\pi\sigma^2}exp\left[-\frac{u^2+v^2}{2\sigma^2}\right] \tag{4.10}$$

The resulting filter function can be decomposed into a real and an imaginary part using the Euler formula. In [98], only the real part was used which is also called even-symmetric circular Gabor filter due to the cosine function. The formula for the even-symmetric circular Gabor filter ($G_c(u,v)$) is therefore given by formula 4.11.

$$G_c(u,v) = g(u,v)cos\left[2\pi f_c\sqrt{u^2+v^2}\right] \tag{4.11}$$

Figure 4.4 shows a visualization of the real circular sinusoidal function in the first column. The second column shows the isotropic Gaussian envelope with which the sinusoidal function is multiplied and the third column depicts the resulting even-symmetric circular Gabor filter.

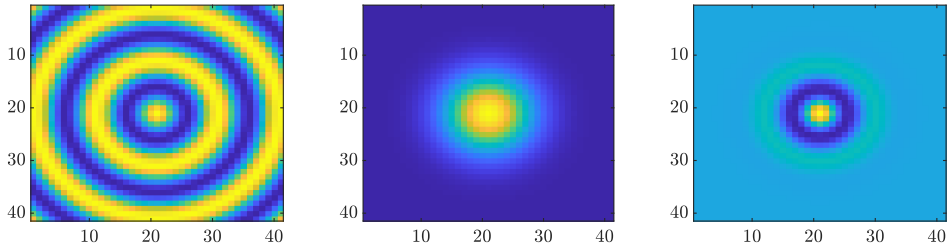

Figure 4.4: Left image: Real part of a circular sinusoidal function; Middle image: Isotropic Gaussian; Right image: Even-symmetric Gabor filter. Parameters: $\Delta F = 1.12$, $\sigma = 5$.

The filter has two parameters σ and f_c. Their relation is described as seen in equation 4.12. Here, ΔF denotes the bandwidth in octave. A good value for σ is suggested as 5 pixel and for ΔF as 1.12 octave.

$$\sigma f_c = \frac{1}{\pi}\sqrt{\frac{ln2}{2}}\frac{2^{\Delta F}+1}{2^{\Delta F}-1} \tag{4.12}$$

The filtered image $F(u,v)$ can be obtained by convolution of the filter $G_c(u,v)$ with the input image $I(u,v)$.

$$F(u,v) = G_c(u,v) * I(u,v) \tag{4.13}$$

4.4 Feature Extraction

The second step in the comparison toolchain (figure 4.1) is feature extraction. A feature is a generic term which can be any information about the contents of an image including the image as a whole or local neighbourhoods. The goal of feature extraction is to transform a digital image from image space into feature space. By doing so, often (but not always) the amount of information needed to represent an initial image is reduced (e.g. some feature extraction algorithms used in this thesis transform a grayscale image into a binary image, which reduces the bits per pixel from 8 bit down to 1 bit). Also every capturing of a vein image is a little bit different due to rotation of the hand or finger. Feature extraction can help to reduce this kind of inter class variability. This thesis employs twelve feature extraction schemes that can be categorized into three types of algorithms based on what type of feature they extract:

- Binarized Vessel Networks (subsections 4.4.1 - 4.4.7): Here, binarized versions of the digital images are extracted that are meant to separate vein structures from everything else. Usually features from this category are afterwards compared using a correlation measure as described in section 4.4.1.

- Keypoints (subsections 4.4.8 - 4.4.10): Keypoints are interesting points in an image, where the term interesting depends on the context. Two general purpose keypoint extraction methods and one that was especially tailored for the vein recognition task is used in this thesis. Every keypoint is stored by describing its local neighbourhood and its location.

- Texture (subsections 4.4.11 & 4.4.12): Image texture is a feature that describes the structure of an image. Shapiro and Stockman [75] define image texture as something that gives us information about the spatial arrangement of color or intensities in an image or selected region of an image. While two images can be identical in terms of their histograms, they can be very different when looking at their spatial arrangement of bright and dark pixels.

4.4.1 Maximum Curvature

Miura et al. [50] presented a finger vein feature extraction method that looks at curvature in the cross-sectional profiles in 4 orientations: vertical, horizontal, and both diagonal orientations. The resulting binary image, where white pixels are the vein networks and black pixels are the background, is obtained in three steps:

(i) The goal for every cross-sectional profile is to find valleys that indicate veins. Figure 4.5 shows an example of a vertical cross-sectional profile (white line) with three valleys A, B and C that indicate vein structures. The initial vein image $F(x, y)$ is therefore mapped to a cross-sectional profile $P_f(z)$ where z is a pixel position in a profile. To obtain the center pixels of the valleys, the curvature of this profile is calculated and the local maxima are calculated. The curvature $\kappa(z)$ is computed as seen in equation 4.14.

$$\kappa(z) = \frac{P_f(z)''}{\left[1 + P_f(z)'^2\right]^{\frac{3}{2}}} = \frac{\frac{d^2 P_f(z)}{dz^2}}{\left[1 + \left(\frac{dP_f(z)}{dz}\right)^2\right]^{\frac{3}{2}}} \tag{4.14}$$

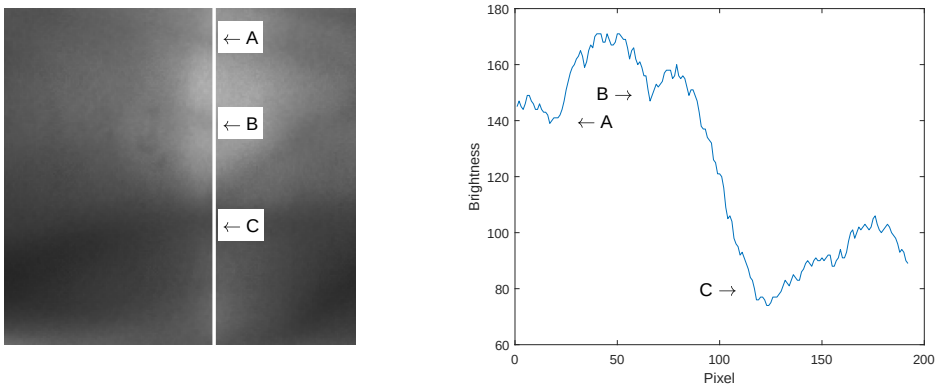

Figure 4.5: Vertical cross-sectional profile of vein image with valleys indicating vein structures.

Note that the actual implementation in the used toolkit applies filtering with the derivative of a Gaussian smoothing filter to generate $P_f(z)'$ and $P_f(z)''$. Doing so removes pixel artifacts and makes sure that only strong veins are found. The local maxima of every profile are denoted as z_i'. For every local maxima, a score is assigned that is calculated as seen in equation 4.15, where $W_r(i)$ is is the width of the region where the curvature around $\kappa(z_i')$ is positive and it represents the width of the found vein.

$$S_cr(z_i') = \kappa(z_i') * W_r(i) \tag{4.15}$$

These scores $S_cr(z_i')$ are then iteratively added onto a new plane $V(x, y)$ (i.e. an empty image) for every cross-sectional profile in the vertical, horizontal and both diagonal directions. Illustrations for this first step can be seen in figure 4.6.

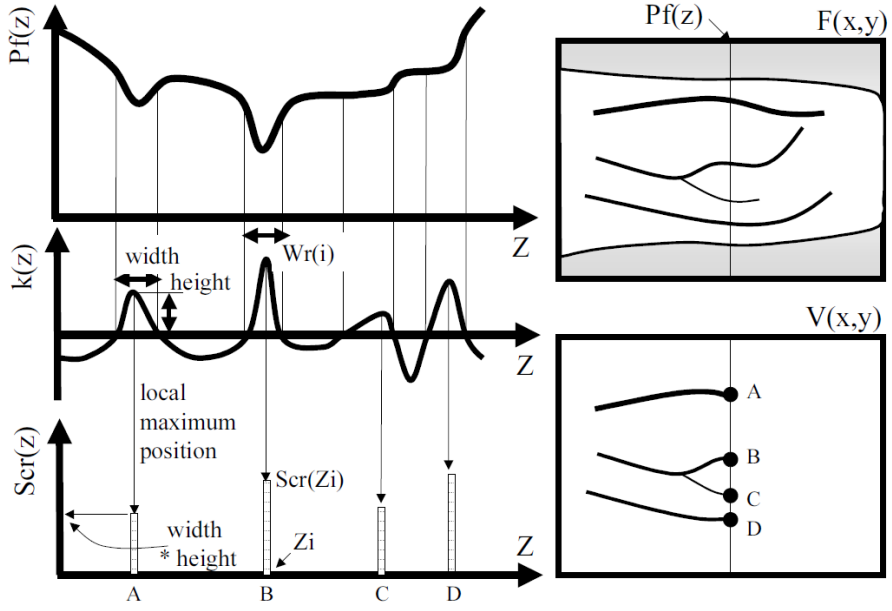

Figure 4.6: Illustrations of the score assignment for every valley center point, taken from [50].

(ii) In the second step, vein pixels are connected and noise should be eliminated. This is done by applying the following rule: For each of the four orientations, a temporary connection array C_{1-4} is created that is filled by applying a filtering operation as seen in equation 4.16 that looks at neighbouring pixels along one orientation. Note that this equation only gives an example for the horizontal orientation. For the other orientations, appropriate neighbouring coordinates need to be considered.

$$\begin{aligned} C_1(x, y) = min\{&max\left(V(x+1, y), V(x+2, y)\right), \\ &max\left(V(x-1, y), V(x-2, y)\right)\} \end{aligned} \tag{4.16}$$

All temporary connection images C_{1-4} are then combined into a single vein feature image $G(x, y)$ as using the maximum operator for each pixel:

$$G(x,y) = max\{C_1, C_2, C_3, C_4\} \tag{4.17}$$

(iii) To generate the final binarized vein feature image $G_{binary}(x,y)$, thresholding is applied. The threshold is chosen as the median of the pixel values in G.

$$G_{binary}(x,y) = \begin{cases} 1 & \text{if } G(x,y) \geq median(G) \\ 0 & \text{if } G(x,y) < median(G) \end{cases} \tag{4.18}$$

For comparison of two generated binary feature images, a customized version of the comparison algorithm described by Miura et al. in [51] is used. Essentially, a 2D correlation is computed between a registered image R and an input image I as seen in equation 4.19. The height and width of one image is denoted by h and w, assuming that both, registered and input image are of same size. To compensate smaller misalignments, a margin of c_w from the horizontal and c_h from the vertical border of the registered image is removed and shifted in x- and y-direction within the boundaries of the input image.

$$C(k,l) = \sum_{y=0}^{h-2c_h-1} \sum_{x=0}^{w-2c_w-1} I(k+x, l+y) \cdot R(c_w+x, c_h+y) \tag{4.19}$$

Let (k_0, l_0) be the coordinates where $C(k,l)$ has its maximum. The final similarity score S of registered image and input image is computed as seen in equation 4.20. Doing so ensures that the similarity score is within the interval $[0, 0.5]$.

$$S = \frac{C_{max}}{\sum_{y=k_0}^{k_0+h-2c_h-1} \sum_{x=l_0}^{l_0+w-2c_w-1} I(x,y) + \sum_{y=c_h}^{h-2c_h-1} \sum_{x=c_w}^{w-2c_w-1} R(x,y)} \tag{4.20}$$

4.4.2 Principal Curvature

A related approach to the algorithm described in section 4.4.1 was proposed by Choi et al. [14]. This approach uses the largest eigenvalue from the Hessian matrix of an image in each point to generate a binarized vein image. As a first step, the gradient field of an image, $\mathbf{G}(x,y)$, is computed using partial derivatives in x and y axis of image $L(x,y)$ the as seen in equation 4.21. The partial derivatives of $\mathbf{G}(x,y)$ are denoted as $g_x(x,y)$ and $g_y(x,y)$.

$$\mathbf{G}(x,y) = \nabla L(x,y) = \left(\frac{L(x,y)}{\partial x}, \frac{L(x,y)}{\partial y} \right) = \Big(g_x(x,y), g_y(x,y) \Big) \tag{4.21}$$

The gradient field is then normalised by dividing the gradient at every point by its magnitude $\|\mathbf{G}(x,y)\| = \sqrt{g_x(x,y)^2 + g_y(x,y)^2}$. Also, a thresholding is applied, which eliminates small noisy components in the normalisation process. The thresholded and normalised gradient is given in equation 4.22, where the threshold value γ is set as a percentage of the maximum gradient ($\gamma = \frac{percent}{100} * max(\mathbf{G}(x,y))$). If the gradient is lower than the threshold value, it is set to zero.

$$\mathbf{G}_T(x,y) = \begin{cases} \frac{\nabla L(x,y)}{\|\mathbf{G}(x,y)\|} & \text{if } \|\mathbf{G}(x,y)\| \geq \gamma \\ (0,0) & \text{if } \|\mathbf{G}(x,y)\| < \gamma \end{cases} \tag{4.22}$$

As a next step, a Gaussian smoothing filter with parameter sigma σ is applied to the two gradient images $g_{Tx}(x,y)$ and $g_{Ty}(x,y)$. Note that the index T should indicate that this step happens after thresholding and normalisation. After filtering, second order derivatives are calculated from the filtered first order derivatives $g_{Fx}(x,y)$ and $g_{Fy}(x,y)$, in order to create a modified Hessian matrix $\mathbf{H_m}$ as given by equation 4.23.

$$\mathbf{H_m}(x,y) = \begin{pmatrix} \frac{\partial g_{Fx}(x,y)}{\partial x} & \frac{\partial g_{Fx}(x,y)}{\partial y} \\ \frac{\partial g_{Fy}(x,y)}{\partial x} & \frac{\partial g_{Fy}(x,y)}{\partial y} \end{pmatrix} \tag{4.23}$$

Let the eigenvalues of $\mathbf{H_m}$ be denoted as λ_1, λ_2, where $|\lambda_1| \geq |\lambda_2|$. The principal curvature is now given by the bigger eigenvalue λ_1. To generate a binarized vein image, thresholding is applied. While [14] propose using Otsu's method [57] for thresholding , the Matlab implementation in the OpenVein toolkit uses median thresholding similar to the Maximum Curvature method in section 4.4.1.

For comparison of the generated feature images, the correlation based similarity measure described in section 4.4.1 is used.

4.4.3 Wide Line Detector

The Wide Line Detector method [26] for vein feature extraction considers a circular window around a given pixel $F(x_0, y_0)$ to decide whether it belongs to a vein structure. To do so, a set of pixels is defined as the neighbourhood region N_{x_0,y_0}, such that every pixel included in this set is within a certain radius r as defined in 4.24.

$$N_{(x_0,y_0)} = \{(x,y) | \sqrt{(x-x_0)^2 + (y-y_0)^2} \leq r\} \tag{4.24}$$

As a first step, for every pixel $F(x,y)$ included in the radius, the difference to the kernel origin, $F(x_0, y_0)$, is calculated. If the difference is smaller than

a certain threshold t, the helper variable $s(x, y, x_0, y_0, t)$ is set to 1 as seen in equation 4.25.

$$s(x, y, x_0, y_0, t) = \begin{cases} 0 & \text{if } F(x, y) - F(x_0, y_0) > t \\ 1 & \text{otherwise} \end{cases} \tag{4.25}$$

After the helper variable s has been calculated for every included position around the origin pixel, the sum of all $s(x, y, x_0, y_0, t)$ in the neighbourhood is built.

$$m(x_0, y_0) = \sum_{(x,y) \in N_{(x_0,y_0)}} s(x, y, x_0, y_0, t) \tag{4.26}$$

Finally, to create a binarized image $V(x, y)$, the computed sum is thresholded using a threshold value g.

$$V(x_0, y_0) = \begin{cases} 0 & \text{if } m(x_0, y_0) > g \\ 1 & \text{otherwise} \end{cases} \tag{4.27}$$

For comparison of the generated feature images, the correlation based similarity measure described in section 4.4.1 is used.

4.4.4 Repeated Line Tracking

Another approach for vein feature extraction, that was proposed by Miura et al. [51], moves along darker pixels in cross sectional profiles which indicate vein structures. The path is stored in a feature image, called locus space T_r, with the same size as the vein image $F(x, y)$. This procedure is repeated N times where the starting point (x_s, y_s) is determined using a uniform random distribution. Eligible starting positions are all pixel within a finger region R_f. After every repetition, the locus space is updated such that the pixel positions from the current path are incremented by one. This results in a feature image where bright spots indicate that multiple iterations of line tracking went this path. As a final step, the locus space is binarized. This is achieved by the following steps.

The randomly determined starting point (x_s, y_s) is the initial coordinate for the current tracking point (x_c, y_c), which is always the point at the current position. Next, a moving-direction for left-right D_{lr} and up-down D_{ud} is determined.

$$D_{lr} = \begin{cases} (1, 0) & \text{if } R_{nd}(2) < 1 \\ (-1, 0) & \text{otherwise} \end{cases} \tag{4.28}$$

$$D_{ud} = \begin{cases} (0,1) & \text{if } R_{nd}(2) < 1 \\ (0,-1) & \text{otherwise} \end{cases} \tag{4.29}$$

$R_{nd}(n)$ is a uniformly distributed random number in the real interval $(0, n)$. The vector values assigned to D_{lr} and D_{ud} in equations 4.28 and 4.29 are denoted as (D_x, D_y). After determining the moving-directions, the set of neighbouring pixels of (x_c, y_c), $N_r(x_c, y_c)$, is selected as shown in equation 4.30.

$$N_r(x_c, y_c) = \begin{cases} N_3(\mathbf{D}_{lr})(x_c, y_c) & \text{if } R_{nd}(100) < p_{lr} \\ N_3(\mathbf{D}_{ud})(x_c, y_c) & \text{if } p_{lr} \leq R_{nd}(100) < p_{lr} + p_{ud} \\ N_8(x_c, y_c) & \text{if } p_{lr} + p_{ud} + 1 \leq R_{nd}(100) \end{cases} \tag{4.30}$$

Here, p_{lr} and p_{ud} are probabilities in the real interval of (0,100). The authors of [51] suggest to define p_{lr} as 50 and p_{ud} as 25, assuming that the finger vein image is oriented horizontally and therefore vein structures mainly progress along the horizontal axis. $N_8(x, y)$ is the set of eight neighbouring pixels of the current pixel (x_c, y_c) and $N_3(\mathbf{D})(x, y)$ is the set of three neighbouring pixels of (x_c, y_c), whose direction is either by D_{lr} or D_{ud}. The construction of the set of neighbouring pixels for the N_3 cases is given in equation 4.31.

$$\begin{aligned} N_3(\mathbf{D}(x, y) = \{&(D_x + x, D_y + y), \\ &(D_x - D_y + x, D_y - D_x + y), \\ &(D_x + D_y + x, D_y + D_x + y)\} \end{aligned} \tag{4.31}$$

Figure 4.7 illustrates the pixels around the current pixel (x_c, y_c) that are chosen as neighbouring pixels for every possible case.

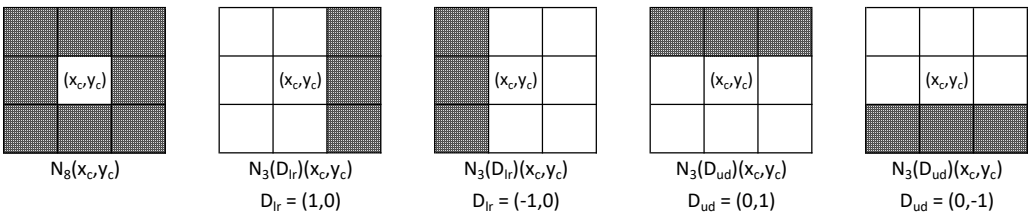

Figure 4.7: Set of neighbouring pixels around (x_c, y_c). Shadowed pixels belong to the set.

In every round, the path that was made along the veins is tracked in a temporary locus space T_c. A pixel to which the current tracking point (x_c, y_c) can move, needs to be within the region of the finger R_f, must not be a pixel that is already in the path of this round (i.e. in the set of the temporary locus space T_c) and must also be in the set of neighbouring pixels that was defined as $N_r(x_c, y_c)$. This constraint can be formulated using set theory as given in equation 4.32.

$$N_c = \overline{T_c} \cap R_f \cap N_r(x_c, y_c) \tag{4.32}$$

In the case that N_c is not empty, a pixel is determined where the current tracking point (x_c, y_c) should move based on a so called line-evaluation function V_l as given in equation 4.33. The line that this function evaluates is a cross sectional profile of width W, which lives at a distance of r away from the point (x_c, y_c). As can be seen in figure 4.8, the equation V_l is meant to find the deepest valley in the cross section.

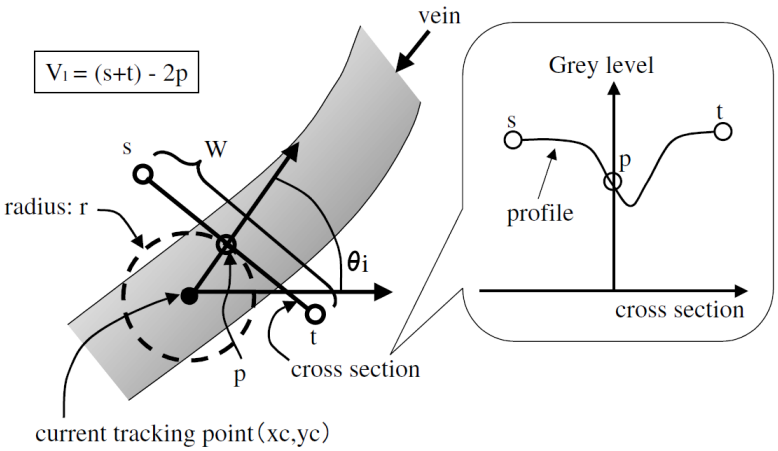

Figure 4.8: Illustration of RLT approach. Screenshot from [51].

$$\begin{aligned} V_l = \max_{(x_i,y_i)\in N_c} \Big\{ & F(x_c + r\,cos\theta_i - \frac{W}{2}\,sin\theta_i, y_c + r\,sin\theta_i + \frac{W}{2}\,cos\theta_i \\ & + F(x_c + r\,cos\theta_i + \frac{W}{2}\,sin\theta_i, y_c + r\,sin\theta_i - \frac{W}{2}\,cos\theta_i \\ & - 2F(x_c + r\,cos\theta_i, y_c + r\,sin\theta_i) \Big\} \end{aligned} \tag{4.33}$$

If V_l is positive, the current tracking point (x_c, y_c) is updated to the point where V_l is maximal (x_i, y_i). Now the process is repeated starting from

equation 4.30.
If V_l is negative or zero or there are no eligible options where the path could go, i.e. the set N_c is empty, the line tracking round is over, meaning that the global locus space $T_r(x, y)$ is updated by incrementing all path coordinates from the temporary locus space $(x, y) \in T_c$ by one. A new line tracking operation starts again by randomly selecting a new initial coordinate (x_s, y_s). This is repeated N times. Finally, a binarized vein image is created by applying a similar thresholding function as given in equation 4.18.

For comparison of the generated feature images, the correlation based similarity measure described in section 4.4.1 is used.

4.4.5 Gabor Filters

Gabor filters are texture descriptors that analyze a certain spatial frequency in a certain direction around the filter origin. The approach that is used in the OpenVein toolkit was proposed by Kumar and Zhou [42] and uses a filter bank consisting of an even symmetric Gabor filter in N orientations. An even symmetric Gabor filter consists of a sinusoidal part which only uses the cosine part, multiplied with an Gaussian part. The Gabor filter $h_{\theta_n}(x, y)$ is given in equation 4.34 where θ_n denotes the angle which the filter is rotated.

$$h_{\theta_n}(x, y) = exp\left\{-0.5\,\frac{x_{\theta_n}}{\sigma_x^2} + \frac{y_{\theta_n}}{\sigma_y^2}\right\} cos\left(\frac{2\,\pi}{\lambda} x_{\theta_n}\right) \tag{4.34}$$

Here, λ is the spatial wavelength of the filter and it defines, together with the bandwidth parameter bw, the variance σ due to the relationship given in equation 4.35.

$$\sigma = \frac{\lambda}{\pi}\sqrt{\frac{ln(2)}{2}}\frac{2^{bw}+1}{2^{bw}-1} \tag{4.35}$$

The variance in x-direction σ_x is assigned to the computed σ, while the variance in y-direction σ_y is further divided by a dilation factor γ. The rotated coordinates x_{θ_n} and y_{θ_n} are obtained by application of a rotation matrix R_n to $(x, y)^T$, i.e. $(x_{\theta_n}, y_{\theta_n})^T = R_n \cdot (x, y)^T$, where R_n is defined as follows.

$$R_n = \begin{pmatrix} cos\,\theta_n & -sin\,\theta_n \\ sin\,\theta_n & cos\,\theta_n \end{pmatrix} \tag{4.36}$$

The filter orientations θ_n are fixed into N steps as given in equation 4.37.

$$\theta_n = \frac{n\,\pi}{N}, \quad n \in \{0, 1, ..., N-1\} \tag{4.37}$$

For creation of the filters, a filter size fs has to be defined in pixel, which must be an uneven number. Figure 4.9 shows an example filter bank consisting of four orientations of a Gabor filter.

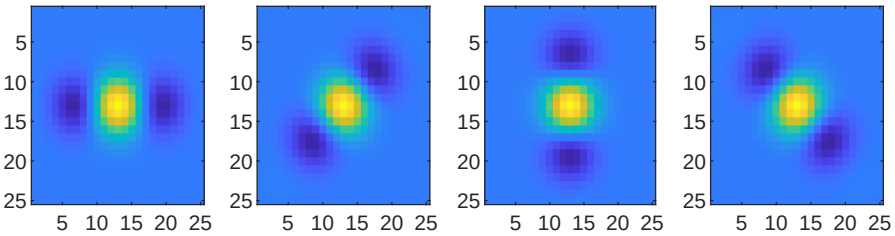

Figure 4.9: Four orientations of a Gabor filter; From left to right: $\frac{0\pi}{4}$, $\frac{\pi}{4}$, $\frac{2\pi}{4}$, $\frac{3\pi}{4}$; Parameters: γ=2,λ=16,fs=25,bw=2,N=4

For each of the N filters in this filter bank, a feature map is generated by convolution with an input image $F(x, y)$. To combine these feature images into a single image $f(x, y)$, the maximum value over all orientations at position (x, y) is taken, as given in equation 4.38. The convolution operation is denoted as $*$ and $\overline{h_{\theta_n}}(x, y)$ represents the Gabor filter with zero mean, that is, the arithmetic mean over filter is subtracted from every filter pixel position.

$$f(x, y) = \max_{\forall\, n=1,2,\ldots,N-1} \left\{ \overline{h_{\theta_n}}(x, y) * F(x, y) \right\} \tag{4.38}$$

Lastly, a morphological top-hat function and adaptive thresholding is applied to obtain the final feature image. For comparison of the generated feature images, the correlation based similarity measure described in section 4.4.1 is used.

4.4.6 Isotropic Undecimated Wavelet Transform

The Isotropic Undecimated Wavelet Transform, as described in Starck et al. [73], is a special kind of wavelet transform is given by the so called à trous algorithm, which is described below. This type of wavelet transform does not spatially shrink, i.e. it does not get decimated in terms of the number of pixels, thereby motivating the term undecimated [74].

The wavelet transform produces at each scale j a set of wavelet coefficients $w_j[x, y]$. The 2D image input signal is denoted as $c_j[x, y]$, where $j = 0$. The wavelet coefficients w_j are generated by subtracting two subsequent scaling levels of $c_j[x, y]$ as seen in equation 4.39.

$$w_{j+1}[x,y] = c_j[x,y] - c_{j+1}[x,y] \tag{4.39}$$

To generate coefficients from another scale $c_{j+1}[u,v]$, discrete convolution (denoted with symbol $*$) is applied to the coefficients from level j and a filter function $h^{(j)}[x,y]$ as given in equation 4.40.

$$c_{j+1}[x,y] = c_j[x,y] * h^{(j)} \tag{4.40}$$

The filter $h^{(j)}$ is chosen such that in 1D $h_{1D}^{(j)} = \left[\frac{1}{16}, \frac{4}{16}, \frac{6}{16}, \frac{4}{16}, \frac{1}{16}\right]$ which can be extended to 2D by calculating the dot product with itself $h_{1D} \cdot h_{1D}^T$. This yields the filter as given in equation 4.41.

$$h^{(j)} = \begin{bmatrix} \frac{1}{256} & \frac{1}{64} & \frac{3}{128} & \frac{1}{64} & \frac{1}{256} \\ \frac{1}{64} & \frac{1}{16} & \frac{3}{32} & \frac{1}{16} & \frac{1}{64} \\ \frac{3}{128} & \frac{3}{32} & \frac{9}{64} & \frac{3}{32} & \frac{3}{128} \\ \frac{1}{64} & \frac{1}{16} & \frac{3}{32} & \frac{1}{16} & \frac{1}{64} \\ \frac{1}{256} & \frac{1}{64} & \frac{3}{128} & \frac{1}{64} & \frac{1}{256} \end{bmatrix} \tag{4.41}$$

For each scaling level j, 2^j zeros get inserted in between every filter coefficient in $h_{1D}^{(j)}$. On border cases, the signal $c_j[x,y]$ gets mirrored such that $c_{j+1}[x,y]$ remains the same size after filter convolution. The original image signal $c_0[x,y]$ can be reconstructed as shown in equation 4.42 where J is the highest used scale factor, however the implementation in the OpenVein toolkit uses only the wavelet coefficients w_j from the scale levels 2 and 3. The final feature image $f[x,y]$ is obtained by addition of those coefficients from the two scale levels as seen in eq. 4.43 and applying thresholding afterwards.

$$c_0[x,y] = c_J[x,y] + \sum_{j=1}^{J} w_j[x,y] \tag{4.42}$$

$$f[x,y] = w_2 + w_3; \tag{4.43}$$

For comparison of the generated feature images, the correlation based similarity measure described in section 4.4.1 is used.

4.4.7 Anatomy Structure Analysis-Based Vein Extraction

One vein network based recognition scheme that extracts two different vein structures (a vein network V and a vein backbone B) from the same input image was described by Yang et al. [94]. This is the only vein network based

recognition algorithm used in this thesis that comes with its own comparison scheme. Therefore, first the vein extraction process is described followed by the comparison approach.

For the first step, a curvature image $C(x, y)$ is created by looking in local cross sectional profiles of an input image $I(x, y)$. It is determined, which angle θ_{xy} the cross sectional profile needs to have, to achieve a perpendicular direction to the vein structure at a given pixel position. This is done by estimating a so called orientation map using least mean square orientation estimation as described in [25]. With every pixel having an estimated orientation θ_{xy}, local cross sectional profiles perpendicular to the angle θ_{xy} are extracted that are of length $2*w+1$, where w is a parameter that is supposed to be approximately half the average vein thickness, measured in pixel. Let (x_0, y_0) be the point around which the cross sectional profile is extracted and θ_{xy} its orientation from the orientation map. The local cross sectional profile $g(i)$ is the constructed as seen in equation 4.44 where $i = -w, ..., 0, ..., w$.

$$g(i) = I(x_0 - i\,cos(\theta_{xy}), y_0 + i\,sin(\theta_{xy})) \tag{4.44}$$

Next the curvature of the local cross sectional profile $g(i)$, κ_g is computed. To create a curvature image $C(x, y)$, this is done for every pixel position. The curvature at a point $C(x_0, y_0)$ corresponds to the curvature κ_g at the position $w + 1$.

$$C(x_0, y_0) = \kappa_g(w + 1) = \frac{g(w + 1)''}{[1 + g(w + 1)'^2]^{\frac{3}{2}}} \tag{4.45}$$

For generation of the vein network V, morphological preprocessing is applied to the curvature image C that includes filling (i.e. fill holes with an area smaller than a predefined threshold), thinning (also known as skeletonization), denoising, and connecting sekeltonized gaps. V therefore is a binarized image that includes vein structures with a thickness of one pixel. The second feature image that is generated is called vein backbone B. The idea of the vein backbone image is to use vein patches with large contrast to the background since they are more likely to appear in different imaging acquisitions of the same subject. Due to the fact that vein structures with more contrast also have large curvature, the vein backbone feature image$B$ is created by thresholding the curvature image $C(x, y)$ by a statistic value, namely its average curvature value C_m. The binarization process can therefore be described as seen in equation 4.46.

$$B(x, y) = \begin{cases} 1 & \text{if } C(x, y) \geq C_m \\ 0 & \text{if } C(x, y) < C_m \end{cases} \tag{4.46}$$

To reduce intra-subject misalignments such as translation, a shift correction is applied based on the maximal matched pixel ratio (MPR). The MPR is defined as the ratio of the number of matching pixel to the total number of pixel in the matching patterns [72]. Effectively this is a correlation measure that is well suited for comparing binarized images. The MPR is calculated between the backbone images of an enrolled B^e and a probe B^p sample for multiple horizontal and vertical shifts. The offset where MPR has its maximum is then used in the following step, which is to perform comparison of the vein networks V_p and V_e, denoting the vein network for the probe and enrollment respectively. Yang et al. named this step in the comparison approach *elastic matching*, because the points in the skeletonized probe vein network are not strictly compared with their corresponding pixel at the same location in the enrollment vein network, but a square neighbourhood $N(x_0, y_0)$ with side length $2r$ is defined around the point in question (x_0, y_0).

$$N(x_0, y_0) = \{(x, y) | x_0 - r \leq x \leq x_0 + r, y_0 - r \leq y \leq y_0 + r\} \tag{4.47}$$

The elastic matching score E_m of two vein networks is defined by the number of matched points P_m^p divided by the number of all vein points P^p in V^p as can be seen in equation 4.48.

$$E_m(V^p, V^e) = \frac{P_m^p}{P^p} \tag{4.48}$$

Due to non symmetry of comparison scores $E_m(V^p, V^e)$ and $E_m(V^e, V^p)$, the similarity score S_e is built by taking the maximum of both cases as given by equation 4.49.

$$S_e = max\left(E_m(V^p, V^e), E_m(V^e, V^p)\right) = max\left(\frac{N_m^p}{P^p}, \frac{P_m^e}{P^e}\right) \tag{4.49}$$

To receive the final similarity score S between two input images, the elastic matching score S_e is further combined with a measure of overlap λ between the two backbones B^p and B^e. Equation 4.50 shows the computation of this overlap measure, where n^p and n^e denote the number of vein points in the backbone feature images and n_0 is the number of overlapped points that is computed as shown in equation 4.51. The operator $\bullet$ denotes point wise multiplication of the two backbone images.

$$\lambda = \frac{2n_0}{n^p + n^e} \tag{4.50}$$

$$n_0 = \sum_{xy} B^p \bullet B^e \tag{4.51}$$

The final score S, as seen in eq. 4.52, therefore weights the elastic matching score by the degree of overlap in the backbone image. The square root is meant to make the score distribution more consistent.

$$S = \sqrt{\lambda S_e} \tag{4.52}$$

4.4.8 Scale Invariant Feature Transform

The algorithm described by David Lowe [46] was designed to find and describe keypoints. To use such keypoints for vein recognition, first keypoints need to be found and afterwards a description of that point is generated. For comparison, the point descriptors, which are essentially feature vectors of dimension 128, are compared using Euclidean distance. The feature extraction part is done in four major stages as described hereafter.

(i) Scale space extrema detection: First, the so called scale space of an image is computed. The scale space of an image $I(x, y)$ is defined as the function $L(x, y, \sigma)$ which is created by convolution of a Gaussian function $G(x, y, \sigma)$ (eq. 4.53) in various scales (increasing values of σ).

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\left(x^2 + y^2\right)/2\sigma^2\right) \tag{4.53}$$

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{4.54}$$

Next difference-of-Gaussian (DoG) images are computed by effectively subtracting the nearby scales separated by a factor k as seen in equation 4.55. Keypoint candidates are found by looking for local extrema in three dimensions including 8 surrounding pixel from the same DoG image and 9 pixel from the two neighbouring DoG images. This procedure is done for multiple resolutions of the initial input image, which is consecutively downsampled by a factor of two for each octave level. Parameters in [46] such as number of octaves and images per octave are determined empirically.

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \tag{4.55}$$

(ii) Keypoint localization
After finding keypoint candidates, for every candidate a 3D quadratic (order 2) function is fitted using Taylor series expansion of the scale space function

D to get a more accurate interpolated location of the extremum. Let the candidate point be denoted as $\mathbf{x_0} = (x_0, y_0, \sigma_0)$ in equation 4.56 and $\mathbf{x} = (x, y, \sigma)$ is used to describe the offset to a point in the surrounding neighbourhood of $\mathbf{x_0}$.

$$T(\mathbf{x}) \approx D(\mathbf{x_0}) + \frac{\partial D(\mathbf{x_0})}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 D(\mathbf{x_0})}{\partial \mathbf{x}^2}\mathbf{x} \tag{4.56}$$

The extremum of $\mathbf{x}$, denoted as $\hat{\mathbf{x}} = (\hat{x}, \hat{y}, \hat{\sigma})$, can be determined by setting the derivative of function $T(\mathbf{x})$ with respect to $\mathbf{x} = (x, y, \sigma)$ equal to zero.

$$\hat{\mathbf{x}} = -\frac{\partial^2 D(\mathbf{x_0})}{\partial \mathbf{x}^2}^{-1} \frac{\partial D(\mathbf{x_0})}{\partial \mathbf{x}} \tag{4.57}$$

Note that $\frac{\partial^2 D(\mathbf{x_0})}{\partial \mathbf{x}^2}$ corresponds to the Hessian matrix and $\frac{\partial D(\mathbf{x_0})}{\partial \mathbf{x}}$ is often referred to as Jacobian. Both are computed using neighbouring function values of the scale space function around the candidate point (i.e. standard discrete derivative filter). This results in a 3x3 system of equations. The offset $\hat{\mathbf{x}}$ is added to the position of the candidate point to get the sub-pixel estimate of the extremum. Also the function value of the extremum, $D(\hat{\mathbf{x}})$ is evaluated and if this value is below a threshold of 0.03, the candidate point is discarded due to low contrast.

Further eliminations of candidates are made for points along edges. The idea is that candidate points along edges will have a large principal curvature across the edge in the DoG image, but have a relatively small principal curvature in the perpendicular direction. Therefore, at the location and scale of the candidate point in question, a Hessian matrix is computed, since its eigenvalues are proportional to the principal curvatures. Note that only the one DoG image D from the scale space is used for analysis on which the candidate point lies. The Hessian therefore is computed as seen in equation 4.58.

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix} \tag{4.58}$$

As described, the goal is to sort out candidates were one eigenvalue is much larger than the other, i.e. have a high ratio $r = \alpha/\beta$. Let α be the bigger eigenvalue and β the other eigenvalue of the Hessian matrix. Using the connection of the trace and the determinant to the eigenvalues as seen in equations 4.59 and 4.60 it is avoided to explicitly compute the eigenvalues.

$$Tr(H) = D_{xx} + D_{yy} = \alpha + \beta \tag{4.59}$$

$$Det(H) = D_{xx}D_{yy} - D_{xy}D_{yx} = \alpha \cdot \beta \tag{4.60}$$

Instead, the following ratio of trace and determinant is used:

$$\frac{Tr(H)^2}{Det(H)} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r} \tag{4.61}$$

Doing so allows for checking whether the ratio of principal curvatures is below a certain threshold defined by r. Otherwise the candidate point is discarded. Therefore equation 4.62 needs to be fulfilled. The ratio r is chosen to be 10.

$$\frac{Tr(H)^2}{Det(H)} < \frac{(r+1)^2}{r} \tag{4.62}$$

(iii) Orientation assignment
Every remaining candidate is now a treated as a keypoint, whose orientation is determined as a next step. For this, the Gaussian smoothed image L with the closest scale to the interpolated scale from the keypoint is used. The parameter σ is therefore omitted in the notation. The gradient magnitudes (eq. 4.63) and orientations (eq. 4.64) are computed for every position as given in the following equations.

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{4.63}$$

$$\theta(x,y) = tan^{-1}\left(\frac{L(x+1,y) - L(x-1,y)}{L(x,y+1) - L(x,y-1)}\right) \tag{4.64}$$

With this information, an orientation histogram is built using a neighbourhood around the keypoint. The histogram divides 360 degrees of orientation into steps of 10, resulting in 36 bins. Each orientation that contributes to the histogram is weighted by its corresponding gradient magnitude and by a Gaussian-weighted circular window with a $\sigma = 1.5$ times the current scale around the keypoint. The bin with the highest count, and also any bin with count $\geq 80\%$ of the highest count, is assigned as the orientation of the current keypoint. Note that for any additional bin where the count lies within the 80% of the maximum count, a new keypoint is created that only differs in orientation.

(iv) Keypoint descriptor
Now that every keypoint has a location, a scale and an orientation, a descriptor of the local area around the point is created that describes the local

neighbourhood. This resulting descriptor will be the feature vector that can be used for comparison. On the Gaussian smoothed image L, where the scale of the keypoint $\hat{\sigma}$ is used to select the amount of Gaussian blur, a 16x16 pixel grid is considered around the location of the estimated keypoint position $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$. Similar to step (iii), the gradient magnitude and orientation needs to be calculated and gradient orientations are rotated relative to the determined keypoint orientation. This 16x16 field of gradients is further weighted using a Gaussian function with a $\sigma = 0.5$ times the width of the descriptor window such that less weight is assigned to the magnitudes that are far away from the estimated keypoint location.

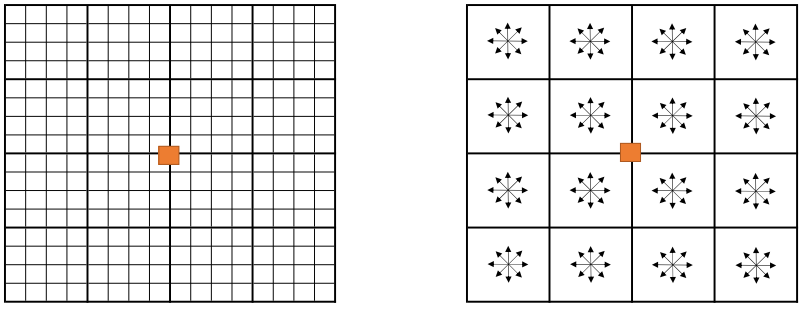

Figure 4.10: Left: 16x16 grid around the estimated location of the keypoint (centered orange square). Right: Orientation histogram for every 4x4 sub block, having 8 bins each.

The grid is further divided into patches of 4x4 pixel that build an orientation histogram with 8 bins (steps of 45 degree). Similar to the previous step, the contribution of each orientation is given by the gradient magnitude times the value of the Gaussian function at this location. Figure 4.10 depicts the idea of having 4x4 pixel patches where every patch generates an orientation histogram. The length of the arrows, although of similar length in the figure, indicate the bin height in the histogram. The final feature vector for one keypoint is then constructed by concatenation of the 16 histograms, yielding in total a feature vector of 4x4x8=128 dimensions. The feature vector is normalized to unit length. To achieve a sort of illumination invariance, every entry larger than 0.2 is set to 0.2 before the vector is renormalized again to unit length.

Keypoints descending from a reference image I_R and a test image I_T, are compared using Euclidean distance of the 128-dimensional feature vectors. The keypoints that have the smallest distance, if smaller than some threshold, are considered as a match. However, if the ratio of distances from the first-best match to the second-best match is greater than 80%, the keypoint from

I_T is rejected.

The similarity score is generated by the ratio of the number of compared points m to the number of maximal possible compared points, which is either the number of keypoints that are enrolled k_e or the number of keypoints from the probe vein image k_p, as seen in eq. 4.65.

$$S = \frac{m}{min(k_e, k_p)} \tag{4.65}$$

4.4.9 Speeded Up Robust Features

Inspired by the SIFT keypoint detection and description method (section 4.4.8), Bay et al. [7] developed a slightly different keypoint algorithm named SURF (short for Speeded Up Robust Features). Similar to other keypoint schemes, first suitable candidate points are detected and afterwards a high dimensional feature vector is created for every such point that describes the local surrounding area around that point.

(i) Interest point detection

Analogous to the SIFT algorithm, a scale space is created that is divided into octaves where each octave contains a series of filter response maps of same resolution but increasing scale, caused by increasing Gaussian σ. While in SIFT the scale space is built from differences of Gaussian filtered images, SURF uses the determinant of the Hessian matrix that is created from the filter responses of convolving an image $I(x, y)$ with Gaussian second order derivatives. The Hessian matrix $H(x, y, \sigma)$ can therefore be defined as

$$H(x, y, \sigma) = \begin{bmatrix} L_{xx}(x, y, \sigma)\, L_{xy}(x, y, \sigma) \\ L_{yx}(x, y, \sigma)\, L_{yy}(x, y, \sigma) \end{bmatrix} \tag{4.66}$$

where $L_{xx/xy/yy}(x, y, \sigma)$ are the responses from convolving Gaussian second order derivatives with an image $I(x, y)$. The Gaussian second order derivatives are approximated by upright filter boxes (see figure 4.11) to reduce computation time.

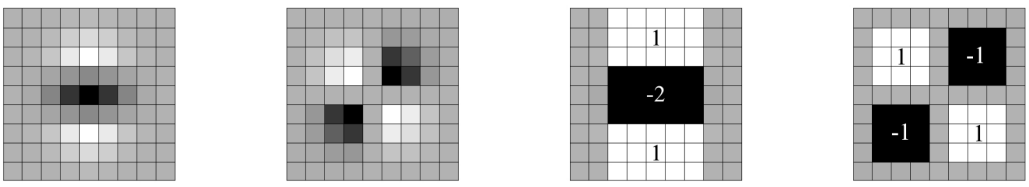

Figure 4.11: Left side: Discrete second order derivatives of Gaussian function. Right side: Approximations using rectangular boxes. Taken from [7].

The responses to the approximated filters are obtained in an efficient manner by using a so called integral image [84] $I_\Sigma(x, y)$ that is initially created from a given input image $I(x, y)$ as seen in equation 4.67. With such integral images, only four additions are needed to compute the sum of any pixel intensities in a rectangular shaped area.

$$I_\Sigma(x, y) = \sum_{j=0}^{j \leq x} \sum_{i=0}^{i \leq y} I(i, j) \tag{4.67}$$

The determinant is computed as given in equation 4.68, where D denote the approximated L using the simplified boxes.

$$det(H) = D_{xx} D_{yy} - \left(0.9\, D_{xy}\right)^2 \tag{4.68}$$

Keypoints are found quite similar as in the SIFT algorithm. Maxima in 3D space of Hessian determinants (considering neighbouring pixel on the same scale as well as in neighbouring scales) are considered as interest points. In order to achieve more accurate results, scale and image space interpolation is done by 3D Tailor series expansion.

(ii) Descriptor generation

For every found interest point, a descriptor is created that describes the local area around that point. To achieve rotation invariance, a reproducible orientation needs to be found as a first step. To do so, Haar wavelet responses are computed in x and y direction within a circular neighbourhood with radius $6s$ around the keypoint, where s is the scale at which the keypoint was detected. The two Haar wavelets are depicted in figure 4.12. Every point in this circular region has now a Haar wavelet response for both directions that are further weighted with a Gaussian function centered at the keypoint location. The two responses for every point are now viewed as a 2-dimensional vector that can be represented on a 2D grid. For estimation of an orientation vector, a sliding orientation window of size $\frac{\pi}{3}$ is rotated as depicted on figure 4.12 and all the responses within this window are summed up. The orientation where the summed up response vector has the highest magnitude is chosen as the keypoint orientation.

Next, a square region with a window size of $20s$ is created around the keypoint which is rotated using the orientation that was assigned before. This region is then split up into smaller 4x4 sub-blocks. In each of the 16 sub-blocks, a 5x5 regularly spaced grid of sample points is interpolated. Let now dx and dy be Haar wavelet filter responses using sample points over the whole square region, where the filters are rotated in the same direction as the square. These responses are then weighted by a Gaussian ($\sigma = 3.3$s) centered

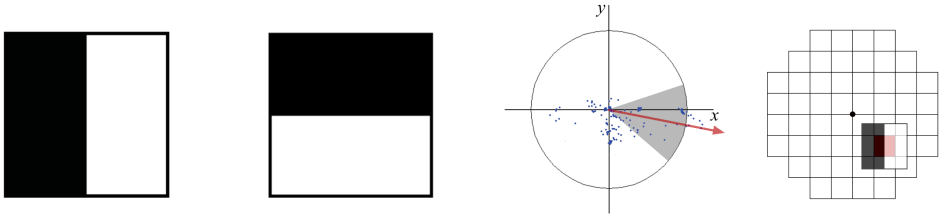

Figure 4.12: Left to center: Haar wavelet filters in x and y direction where black parts are weighted -1 and white parts +1. Right to center: Circular area around keypoint and sliding orientation window (shadow in circle) of size $\frac{\pi}{3}$ with summed orientation. Images taken from [6].

at the keypoint location. Afterwards the 5x5 responses dx and dy per sub-block are summed up for each of the 16 blocks. Additionally, the absolute values of the responses, $|dx|$ and $|dy|$ are also summed up separately. These four sums per sub block build a feature vector $(\sum dx, \sum |dx|, \sum dy, \sum |dy|)$. Each sub-block is now described by a 4-dimensional feature vector. For final description of the keypoint, the 4-dimensional feature vectors from all 16 blocks are concatenated, resulting in a 64-dimensional feature vector for every keypoint. Finally this feature vector is normalized to unit length.

Comparison and similarity score computation is done in the same way as described for the SIFT algorithm.

4.4.10 Deformation-Tolerant Feature-Point Matching

Another keypoint-based feature extraction and comparison method that was tailored for finger vein recognition task was proposed by Matsuda et al. [48]. This method defines its own inherent preprocessing step, which includes the creation of an even symmetric Gabor filter bank as described in section 4.4.5. To create a single preprocessed image from a given input image, the pixel intensities at every pixel position get ordered based on their value. At every pixel position (x, y) the average of the third and fourth largest pixel intensity is computed to generate the preprocessed image $F(x, y)$.

Next, keypoint candidates are found. This is achieved via eigenvalue analysis of the Hessian matrix (see equation 4.69, where the indices indicate second order derivatives) at every pixel position in $F(x, y)$.

$$H(x, y) = \begin{pmatrix} \partial F_{xx} & F_{xy} \\ \partial F_{xy} & F_{yy} \end{pmatrix} \tag{4.69}$$

The idea is that in order to find distinctive points on veins such as turning

points of bifurcations, points are desired where the curvature is high in all directions. Since the eigenvalues of the Hessian matrix yields the principal curvature λ_1 and the minimum curvature λ_2 (where $\lambda_1 > \lambda_2$), a minimum curvature map MCM can be created by using the smaller eigenvalue at every pixel position as seen in 4.70.

$$MCM(x, y) = max(\lambda_2, 0) \tag{4.70}$$

Candidate keypoint positions are then found at the positions where the MCM has local maxima. To describe every keypoint with a feature vector, first a vein pattern map VPM is created that contains the principal curvatures at every pixel position as given in equation 4.71. This is done using the larger eigenvalue.

$$VPM(x, y) = max(\lambda_1, 0) \tag{4.71}$$

A square area around the keypoint location in the VPM (named the descriptor area) is divided into WxH blocks. For each block, a histogram consisting of N bins, representing $\frac{180}{N}$ degrees, is created. The pixel value of VPM is added to the bin that contains the direction of the eigenvector corresponding to the larger eigenvalue λ_1. The final feature vector hence consists of $WxHxN$ entries and is normalized to unit length. Due to the possibility of longitudinal finger rotations in sample captures, the descriptor area is normalized in size depending on how far the keypoint is located from the finger line of center. The process is visualized in figure 4.13.

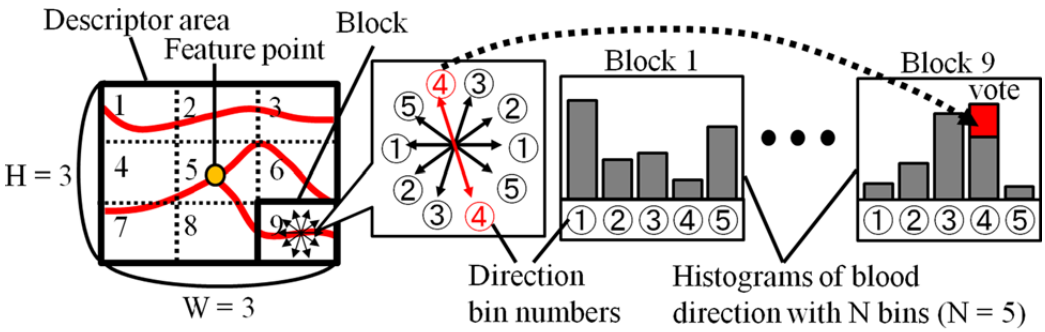

Figure 4.13: Keypoint descriptor generation. Taken from [48].

Matsuda et al. also propose a comparison scheme for the just generated keypoints, that consists of two steps: (i) First, for both the enrolled image (denoted E) and the input image (denoted I) a non-rigid transformation is estimated. To do so, a feature distance FD_{ij} is computed for every keypoint

$i = 1, ..., n_1$ in the enrolled image with every keypoint $j = 1, ..., n_2$ in the input image using Euclidean distance.

$$FD_{ij} = \sqrt{\sum_{d=1}^{dim} \left(vi_{i,d} - ve_{j,d}\right)^2} \tag{4.72}$$

Equation 4.72 shows the formula for feature distance calculation, where $dim = WxHxN$ and vi and ve denote feature vectors from input and enrollment images that are of dimension dim. Each point i is assumed to correspond to the point j that has minimum feature distance FD.

For every point pair at locations (x_e, y_e) for the point in the enrolled image and (x_i, y_i) in the input image, a displacement vector (dx, dy) is calculated by simple subtraction, i.e. $dx = x_e - x_i$ and $dy = y_e - y_i$. The displacement vectors are then sorted into a 2D histogram with bin-axes dx and dy. Since the displacement is expected to happen more or less uniformly on a global level, keypoint correspondences whose displacement vector is outside a certain radius from the mode (i.e. most occurrence) are discarded as errors. This kind of histogram filtering is done once on a global level, using all displacement vectors, and once on a local basis, where for every remaining (after global filtering) keypoint correspondence only a subset consisting of correspondences within a defined radius are considered. A transformation from enrolled image to input image is estimated using the thin-plate spline model as described in [8].

(ii) After registration, each transformed enrollment point is assigned to an input point that has minimum feature distance. However, to count as a matched point, two conditions (eq. 4.73 & 4.74) need to be met:

$$ED_{ij} \leq R_{ED} \tag{4.73}$$

Where ED_{ij} represents the Euclidean distance between the coordinates of the transformed enrollment point and the coordinates of the input point. R_{ED} is a parameter and symbolizes a radius for a disk within both, enrolled and input point need to be for condition 1.

$$FD_{ij} < T_{FDi} \tag{4.74}$$

$$T_{FDi} = FD_{av_i} - \alpha FD_{\sigma_i} \tag{4.75}$$

Where FD_{av_i} is the average and FD_{σ_i} the variance over all FD_{ij} made for a fixed enrollment image i. The variable α is a parameter. If both conditions are met, the corresponding points count as a match, i.e. a counter of matched points m is incremented by one.

The final similarity score is obtained by dividing the number of matched points m by the sum of enrollment image keypoints n_1 and input image keypoints n_2.

$$S = \frac{m}{n_1 + n_2} \tag{4.76}$$

4.4.11 Local Binary Patterns

One popular texture descriptor is Local Binary Patterns (originally described by Ojala et al. [56]) which has been used in several variations in the domain of finger vein recognition [43] as well as finger vein presentation attack detection [44, 39]. The basic idea of LBP is that every pixel in the feature image $F(x_c, y_c)$ is obtained by comparison of a pixel in the input image $I(x_c, y_c)$ with its local neighbourhood. Usually, the neighbourhood is chosen to be the surrounding eight pixels of a center pixel $I(x_c, y_c)$ as shown in figure 4.14. $F(x_c, y_c)$ is now calculated by comparing every surrounding pixel to the center pixel and temporary storing a binary value depending on whether the difference is positive or negative. This results in an ordered set of binary values that encodes the difference of the pixel in question to its local neighbourhood. This can be formulated as seen in equation 4.77

$$F(x_c, y_c) = \sum_{n=0}^{7} s\left(I(x_n, y_n) - I(x_c, y_c)\right) \cdot 2^n \tag{4.77}$$

where (x_n, y_n) represent the gray scale intensity at certain position around the center pixel as seen in figure 4.14 and the function $s(x)$ is defined as given in equation 4.78.

$$s(x) = \begin{cases} 1 & \text{if} \quad x \geq 0 \\ 0 & \text{if} \quad x < 0 \end{cases} \tag{4.78}$$

The implementation used in this thesis uses LBP encapsulated in a three step feature extraction pipeline: (i) Creation of a filter bank that uses Gabor filters of size $s_f * s_f$ in various orientations θ_j and scales σ_k with subsequent filtering. This results in multiple $(j*k)$ of images in Gabor feature space. (ii) All resulting images are then processed using the LBP algorithm described above. (iii) As a third step, every LBP processed image is divided into blocks of size $s_b * s_b$. The image intensities from every block are represented in a normalised histogram. Lastly, every histogram from every feature image is concatenated to build a final feature vector whose dimensionality depends on the number of resulting images in Gabor feature space $j * k$, the block size

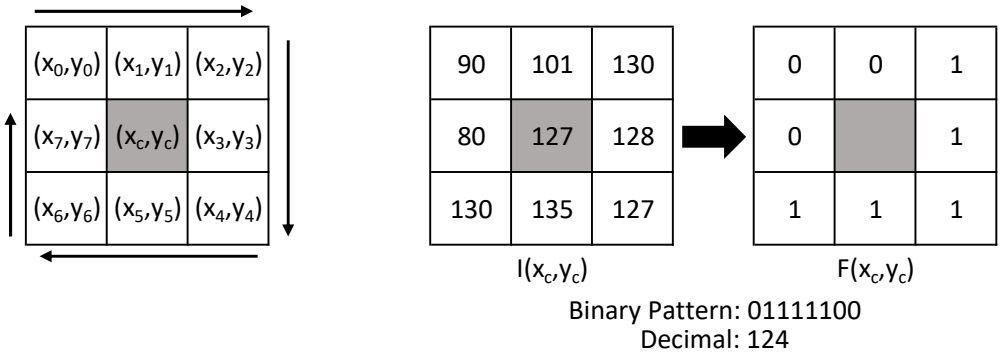

Figure 4.14: Left: fixed index assignment around a center pixel $I(x_c, y_c)$; Right: comparison of the center pixel to the neighbouring pixels with resulting binary code that defines the resulting intensity at pixel (x_c, y_c) in the feature image F.

s_b and the number of blocks that a single image can be divided into, i.e. the resolution of the input image.

Gabor Filters are created by multiplying two orthogonal spatial sinusoidals (i.e. sine and cosine) with an Gaussian envelope as explained in previous sections (4.3.3 & 4.4.5). Multiple scales are obtained by variation of the σ in the Gaussian part, rotations by application of a rotation matrix as done in section 4.4.5. Figure 4.16 shows components of Gabor filters in four rotations and three scales as well as an example for the resulting feature images after filtering and applying the LBP algorithm.

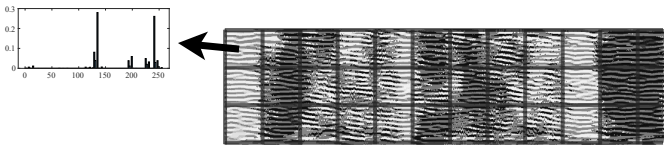

Figure 4.15: The block histogram from the upper left block.

Figure 4.15 shows the block histogram of one block from one feature image. Choosing a block size that results in fractions of full blocks (as seen on the right) does not pose a problem as long as the whole image data set is of same resolution. For comparison of two given vein images, the respective concatenated histograms of the enrolled h^e and probe h^p vein image are evaluated on their similarity using histogram intersection. The formula for histogram intersection is stated in equation 4.79 where n is the length of the histogram feature vector.

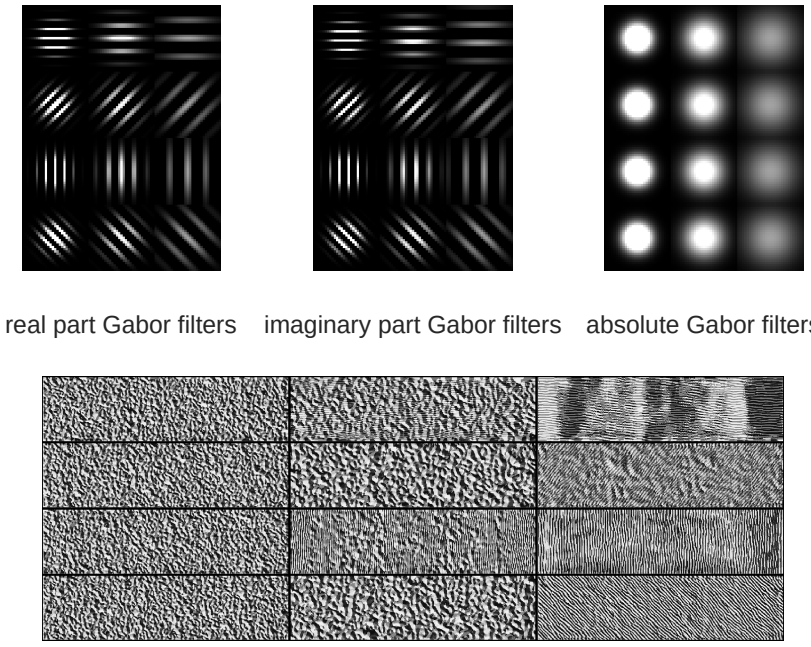

Figure 4.16: Upper row: components of a Gabor filter bank in four orientations and three scales. Bottom images: Feature images after Gabor filtering and LBP algorithm.

$$HI = 1 - \frac{\sum_{i=1}^{n} min\left(h_i^e, h_i^p\right)}{\sum_{i=1}^{n} h_i^e} \tag{4.79}$$

To arrive at a final similarity score S, the calculated value for histogram intersection HI is inverted ($S = \frac{1}{HI}$) to ensure that more similarity results in a higher similarity score to be consistent with the other recognition schemes.

4.4.12 Convolutional Neural Network

The only non hand-crafted approach for vein recognition that is included in the experiments of this thesis is given by a convolutional neural network that uses triplet loss as loss function. The task of a loss function is to guide the net during the training procedure on how to update its internal parameters such that it learns a desired behaviour. The network in use was described and implemented by Wimmer et al. [92].

While for many existing applications CNNs are trained to classify a given input into a set of pre-defined classes, this approach is not an ideal solution for biometric recognition where subjects (classes) may be added in the future. By using a concept known as triplet loss this problem can be avoided, because here the net learns to quantify the similarity between images. For doing so, always three images (triplets) are needed as input: a so called anchor image A, a positive P which is from the same class as the anchor image, and a negative N which is an image descending from a different class. The net is now trained to minimize the distance between images A and P, while maximizing the distance between A and N. The idea is illustrated in figure 4.17.

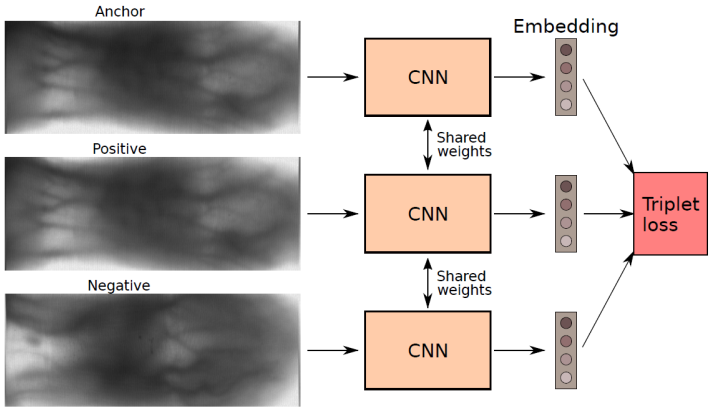

Figure 4.17: Training the CNN using triplet loss. Taken from [92].

The triplet loss function can be formalized using squared Euclidean distance, i.e. the square root is omitted, as seen in equation 4.80.

$$L(A, P, N) = max\left(\|f(A) - f(P)\|^2 - \|f(A) - f(N)\|^2 + \alpha, 0\right) \quad (4.80)$$

The variable α is a margin between positive and negative pairs and its goal is to prevent that images of a single class get projected to a single point in the embedding space, i.e. allow inter-class variability. The function $f(x)$ denotes the embedding that maps a vein image from image space to the output space R^{256}.

To avoid that, by random selection, triplets are chosen where the classes (subjects) are naturally different enough to fulfill the constraint in equation 4.80, only so called hard triplets are chosen. Only triplets that also fulfill equation 4.81 are therefore allowed for training. Doing so ensures that every calculation contributes to the training procedure.

$$\|f(A) - f(P)\|^2 + \alpha > \|f(A) - f(N)\|^2 \tag{4.81}$$

The network architecture is a SqueezeNet [27], which is a small pre-trained network, specifically created to have few parameters and small memory requirements. The input images are resized to a fixed size of 3x244x244, where each color channel is set as the gray scale value. The size of the last convolutional layer is adapted such that a 256-dimensional embedding (output vector) is produced.

The similarity score S between two vein images can therefore be obtained by computing the inverse Euclidean distance between their 256-dimensional embedding as given in equation 4.82, where e_i and p_i stand for embeddings of an enrolled and a probe image, respectively.

$$S = \frac{1}{\sum_{i=1}^{256} (e_i - p_i)^2} \tag{4.82}$$

The inversion causes similar vectors to have a higher score and vice versa.

5 Databases

In this chapter all the databases used in this work are described. Section 5.1 covers the two video databases for the presentation attack detection experiments to answer *RQ3*. In section 5.2, two publicly available still image finger vein presentation attack data sets are described that are used to obtain reference results for the threat analysis (*RQ2*), i.e. to compare the "effectiveness" of the attack samples with ones that are already well established in the research community. Unfortunately no such presentation attack databases are available for dorsal hand vein images that could be used as a reference.

5.1 Video databases for presentation attack detection

The presentation attack detection methodologies from chapter 3 are evaluated on two video data sets that were captured using near infrared illumination. Both data sets descend from the Multimedia Signal Processing and Security Lab at the University of Salzburg and are described in this section hereafter.

5.1.1 Palmar Finger Vein Data Set (PLUS-FV)

This palmar finger vein video data set was captured in summer 2020 using a subset of the *PLUSVein-FV3*[35] as bona fide samples. The subset used for collecting this presentation attack database comprises of 6 fingers (i.e. index, middle and ring finger of both hands) from 22 subjects that were captured 3 times. Every used sample was captured in palmar perspective and with two types of light sources, namely *LED* and *Laser*. Therefore, presentation attacks were also created for both illumination variants. The imaging sensor in use is the *PLUS OpenVein finger vein sensor*, which is same as the one that has been used to collect the *PLUSVein-FV3* database. The presentation attacks are created for sensors that operate in transillumination mode, i.e.

imaging sensor and source of illumination are placed on opposite sites of the finger. The design for the presentation attack instrument was inspired by a talk that was given by german hackers on a hacking congress [40]: An extracted vein pattern is printed in a white paper using a 'HP LaserJet 500 colour M551' laser printer which is then sandwiched in between a top and bottom part made of beeswax. Both parts can be seen on figure figure 5.1 (e). The goal is that the beeswax mimics the tissues inside a human finger. The bottom part, which is presented towards the illumination source, is of elliptic shape. The idea is that it diffuses the penetrating light and therefore ensuring uniform illumination. The task for the rectangular top part is to blur the vein pattern that lies in between the two beeswax parts.

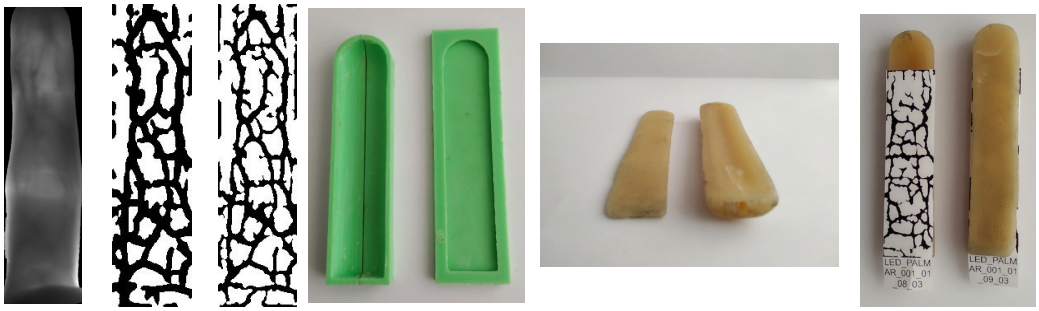

Figure 5.1: (a) original image from *PLUSVein-FV3*; (b) & (c) extracted vein patterns *thick* and *thin*; (d) 3D printed moulds; (e) cast top and bottom made from beeswax; (f) presentation attack instrument usage.

For acquisition, the elliptic part is facing towards the imaging sensor. Both parts are cast of yellow beeswax using the moulds shown in figure 5.1 (d). In figure 5.1 (f) one can see the final presentation attack instrument. The vein patterns are extracted using principal curvature (PC - described in section 4.4.2) feature extraction in two thicknesses, denoted *thick* and *thin* as can be seen in figure 5.1 (b) & (c).

Presentation attack video sequences were captured over a duration of 10 seconds each with a frame rate of 15 fps. Since real fingers sometimes tend to not be perfectly still during video acquisition, two different types of presentation attack video sequences were created: *Still*, where the wax presentation attack instruments lie motionless in the sensor over the duration and *Trembling*, where small movements were applied to the attack instruments with to goal of creating a natural trembling effect. Such video sequences were created for all 396 (22 subjects x 6 fingers x 3 sessions) used samples from the reference database in LED and Laser illumination, with both thicknesses

of the extracted vessel patterns. In addition to every video sequence, a single still image was acquired that can be used for the threat analysis to answer *RQ2*. Since original *PLUSVein-FV3* has no videos, new bona fide acquisitions had to be made as well. Unfortunately only for 16 subjects, new bona fide acquisitions (in LED and Laser variant, together with a still image) could be made. Table 5.1 lists the final numbers of this database.

	Presentation Attacks				Bona Fide	
	Thick		Thin		*PLUSVein-FV3*	New
	Still	Trembling	Still	Trembling		
Image	22x6x3	22x6x3	22x6x3	22x6x3	22x6x5	16x6x5
Video	22x6x3	22x6x3	22x6x3	22x6x3	/	16x6x5

Table 5.1: Overview of the scale of the PLUS finger vein data sets. Note that every acquisition was made in LED and Laser illumination.

Image 5.2 shows examples of both illumination variants of the original *PLUSVein-FV3* dataset (left column), presentation attacks using thick vein patterns (center column) and newly acquired bona fide samples (right column).

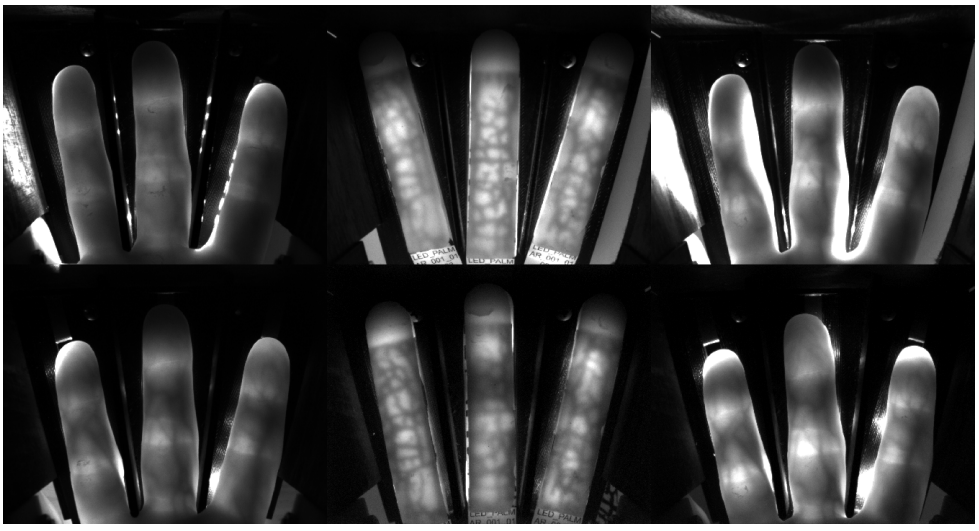

Figure 5.2: Examples; Top row: LED illumination; Bottom row: Laser illumination; Left column: *PLUSVein-FV3*; Center column: Presentation attack using thick vein patterns; Right column: Newly acquired bona fide.

5.1.2 Dorsal Hand Vein Data Set (PLUS-HV)

The dorsal hand vein presentation attack data set used in this thesis was originally created for a publication by Herzog and Uhl [24]. The imaging installation used to capture the video sequences is the one that was also used in [23]. It contains a Canon EOS 5D MarkII DSLR with removed IR blocking filter ans additional 850nm pass-through filter as imaging sensor, which is placed at the ceiling of a wooden box. The capturing device was designed for being able to capture both transillumination and reflected light illumination. For reflected light illumination, 6 950nm LEDs are attached next to the imaging sensor at the top side of the box. On the bottom of the capturing device, a near-infrared surveillance lamp including 50 940nm LEDs is responsible for transmissive illumination. The database contains both hands from 13 participants, that were captured with both lighting versions. Therefore the bona fide basis of this database consists of (13 attendees x 2 hands) 26 genuine video sequences for both illumination variants.

For every genuine video sequence, attack counterparts were created using five presentation attack instruments:

- *Paper:* A single frame where the dorsal hand vein area was cropped as a region of interest, printed using a laser printer and put on a cardboard tray for easier insertion to the imaging sensor
- *Paper Moving:* the presentation attack sample from *Paper* moving back and forth to simulate heartbeat, using a pace of approximately 70 to 90 beats per minute
- *Smartphone:* The dorsal hand vein region of interest, displayed in a smartphone (Samsung Galaxy S8) display
- *Smartphone Moving:* The presentation attack sample from *Smartphone* with a programmed sinusoidal translation oscillation along the x axis
- *Smartphone Zooming:* The presentation attack sample from *Smartphone* with a programmed sinusoidal scaling oscillation in every direction (i.e. a slight zooming effect)

To achieve the rhythmic moving and zooming on the smartphone, [24] created an Android application that loops these operations. Both variants of motion, i.e. translation and scaling, are meant to simulate a heart-beat-like variation of illumination on the dorsum of the hand. In total, the whole database consists of (13x2x2 bona fide + 13x2x2x5 presentation attack) 312 video sequences, all of resolution 1920x1080. Figure 5.3 shows examples of

bona fide and presentation attack frames in both illumination variants. The sequences are of varying length. Bona fide videos range from 14.75 to 25.29 seconds, presentation attack sequences from 10.41 to 21.49 seconds. Every sample was captured with a constant frame rate of 30 frames per second.

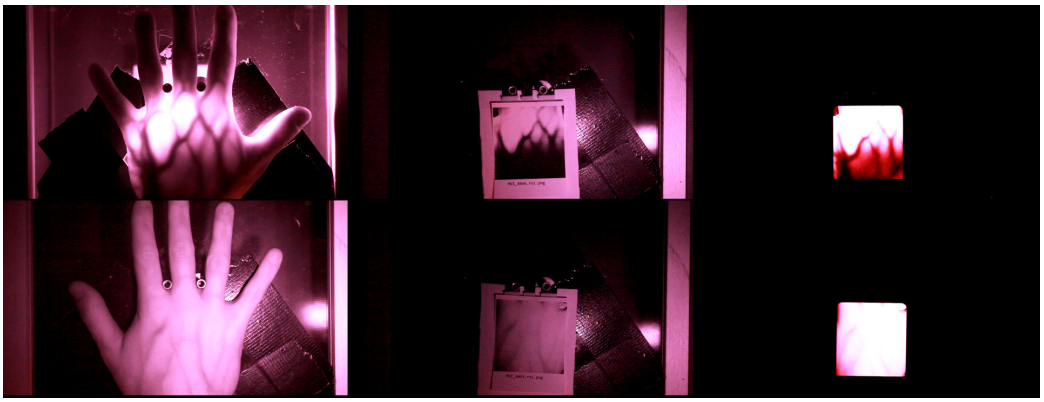

Figure 5.3: Example frames from dorsal hand vein video sequences; Top row: transillumination; Bottom row: reflected light; Left column: bona fide video frame; Center column: paper presentation attack; Right column: smartphone presentation attack

5.2 Public finger vein image databases for threat analysis comparison

In this section, two publicly available finger vein still image presentation attack databases are described: The IDIAP VERA FingerVein database [78] and the South University of China Spoofing Finger Vein Database [60]. These two data sets are well established in the finger vein research community and therefore serve as a benchmark for the threat analysis experiments to answer *RQ2*. Consequently, the same threat evaluation experiments that are executed on the finger vein database from section 5.1.1 are also carried out on the databases from this section. Doing so allows a more meaningful assessment on how hazardous the attacks used for this thesis actually are.

5.2.1 VERA

The VERA FingerVein Database for fingervein recognition [5] was the first larger presentation attack database available for research purposes. It con-

[5] Available here: `https://www.idiap.ch/en/dataset/vera-fingervein`

sists of left and right hand index fingers of 110 subjects that were captured in 2 acquisition sessions. This makes a total of 440 images from 220 unique fingers, where each finger has two samples. Every sample has one presentation attack counterpart. These Presentation attacks were created by first applying a histogram equalisation method (similar to one as described in section 4.3.1) and then printing the preprocessed samples on quality paper using a laser printer. As a next step, the vein contours were enhanced using a black whiteboard marker after which the samples were again provided to the biometric sensor to finally acquire the presentation attack samples. Every of the 880 (bona fide + presentation attack) samples is provided in a *full* version and in a *cropped* version. The full set consists of the raw images captured with size 250x665. For the cropped set, images were further truncated by removing 50pixel margin from the border, resulting in images of size 150x565. This cropping procedure is meant to cut the finger contours since otherwise the geometric shape of the finger would be also considered and would therefore introduce a bias in the results for finger vein recognition.

5.2.2 SCUT

The South University of China - Spoofing Finger Vein Database [6] was collected from 100 persons that scanned 6 fingers (namely index, middle and ring finger from left and right hand) in 6 acquisition sessions, making a total of 3600 bona fide samples. Presentation attacks were generated by printing each finger vein image on two overhead projector films that are aligned and stacked. In order to reduce overexposure, a strong white paper ($200g/m^2$) is sandwiched in-between the two overhead projector films. With similar intentions as with the VERA database, namely reducing the impact of the finger contours, the SCUT is available in a *full* version and *ROI* version. While in the full set every image sample has a resolution of 640x288 pixel, the samples from the ROI set are of variable size. Since certain recognition algorithms from chapter 4 can not be evaluated on image samples that do not have a fixed size, samples have been resized to 474x156 which corresponds to the median of all heights and widths from the ROI set.

[6] Available here: `https://github.com/BIP-Lab/SCUT-SFVD`

6 Experimental Setup and Results

This chapter contains the experiments in this thesis and describes the corresponding experimental setup. The experiments are divided into the following main parts: Section 6.1 explains how the data from the Multimedia Signal Processing and Security Lab (section 5.1) is preprocessed such that only the region of interest remains. An extensive evaluation on the attack potential of the data sets is given in section 6.2. Experiments for performing presentation attack detection are carried out in section 6.3.

6.1 Attack Database Preparation

Due to the circumstance that the raw hand vein data contain a lot of background from the imaging installations and also the finger vein data always contains batches of three fingers from the same hand, some preprocessing is required as a first step such that only the vein regions remain.

6.1.1 Finger Vein

ROI extraction for the finger vein data is done in a similar way as described in [35] and is visualized in figure 6.1: First, images are split into three parts to have index, middle and ring finger in separate images. This is done using fixed image coordinates since the imaging installation provides a certain template where each finger must be. Next, the finger outline is detected using edge detection and consecutive morphological processing techniques. With the help of the finger outline, a center line is calculated which is used to rotate the finger image accordingly such that as a final step the finger ROI can be cropped using a defined box as seen in sub image (d). Every finger sample is of size 192x736 pixel.

When cropping the ROI in videos, rotation parameters are only determined for the first frame and then used for all consecutive frames in the

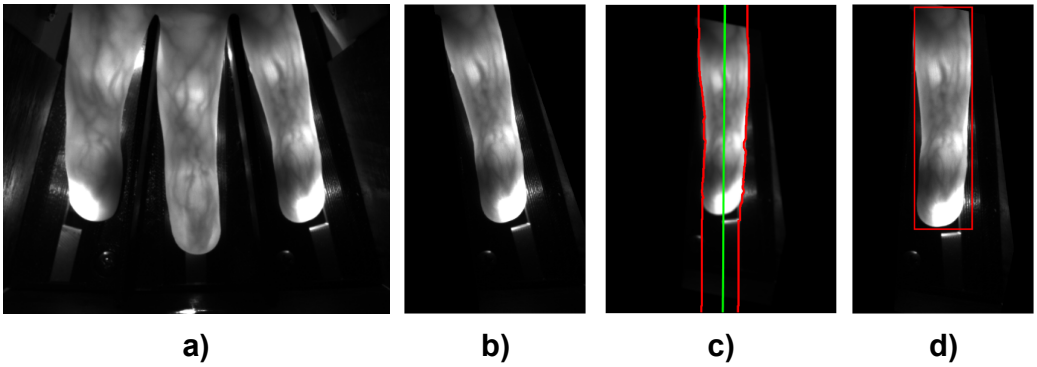

Figure 6.1: Cropped finger vein attack sample. a) Raw acquisition from capturing device. b) single finger isolated c) already aligned finger, rotated such that the center line is aligned with the pixel grid d) fixed ROI.

same video sequence. This ensures that possible finger motion such as trembling is preserved while it prevents ROI jumping artefacts.

6.1.2 Hand Vein

The imaging installation used for capturing the dorsal hand vein data has two screwed pins where, during the imaging process, the middle finger of the subject is placed inbetween. Hence every hand and every attack sample is placed on the same predetermined position. The pins can be seen in figure 6.2. While in [68] a fixed 600x600 ROI for both bona fide and attack sample was used, a slightly different approach was used for the experiments in this thesis because it has been found that the attack samples are of different scale.

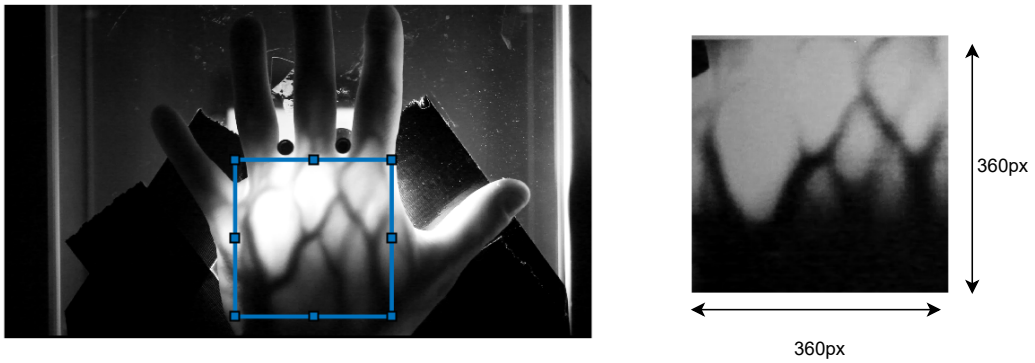

Figure 6.2: Right: Cropped attack sample. Left: Corresponding original frame, where the rectangle was found by cross correlation using various scales.

Therefore, as a first step, one exemplary frame from a paper attack video

sequence was manually cropped such that the maximum area can be used while ensuring that only little to no background is visible as can be seen in right image in figure 6.2. Using 2D cross correlation on this manually cropped sample in various resolutions and a frame from its origin video, a scaling factor of 1.4225 together with coordinates for cutting the bona fide video could be determined. The box in the left image in figure 6.2 shows the found best fit for the scaled attack sample. The determined parameters were then used for cropping every video sequence from the hand vein data set. The final size of every sample is now 513x513 pixel ($\lceil 360 \cdot 1.4225 \rceil = 513$).

6.2 Attack Database Threat Evaluation

It is essential to test the actual functionality of given presentation attacks before developing countermeasures against them. To do so, experiments in this section employ twelve vein recognition methodologies introduced in section 4.4 together with a common threat analysis protocol described in section 4.1 to evaluate whether the attack data sets used in this thesis would be able to deceive a real system, therefore providing an answer to *RQ2*.

6.2.1 OpenVein Toolkit Settings

The recognition experiments using the OpenVein toolkit can be configured in a modular way. Prior to feature extraction, preprocessing schemes described in 4.3 can be chosen to be applied to every biometric sample. Table 6.1 gives an overview of what preprocessing schemes are used in the experiments.

Preprocessing Settings OpenVein Toolkit											
Method	MC	PC	GF	IUWT	RLT	WLD	ASAVE	DTFPM	SIFT	SURF	LBP
Gaussfilt							✓				
HFE		✓	✓	✓	✓	✓			✓	✓	✓
CLAHE	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓
CGF	✓	✓	✓	✓		✓					✓
Resize	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓

Table 6.1: Preprocessing settings OpenVein toolkit

When dealing with feature extraction schemes that generate a feature image that depicts extracted binarized vessel networks, one can choose from a variety of morphological postprocessing algorithms to be applied on the generated feature image. Table 6.2 lists which postprocessing steps are applied after feature extraction in the experiments. The toolkit uses the standard Matlab functions for the postprocessing steps.

Postprocessing Settings OpenVein Toolkit							
Method	MC	PC	GF	IUWT	RLT	WLD	ASAVE
AreaOpen		✓	✓	✓	✓		
InverseAreaOpen		✓	✓	✓	✓		
Thinning			✓				
Spur			✓				
Dilation			✓				

Table 6.2: Postprocessing settings OpenVein toolkit for feature extraction schemes that extract a binarized vessel structure from the vein image.

Note that the order of pre- and postprocessing steps is indicated by the ordering inside the tables. Since configuration files for all of the recognition schemes were already existent from members of the Multimedia Signal Processing and Security Lab, the files have been adopted for the experiments in this thesis with no further modifications.

6.2.2 Finger Vein

This section contains the threat evaluation of the finger vein attack samples. As main metrics, the equal error rate (EER) and impostor attack presentation match rate (IAPMR) from the two scenario protocol as described in section 4.1 are reported. For every finger vein video sequence, an additional image was captured as mentioned in section 5.1.1. The experiments in this section include the images that correspond to the video sequences where the attack fingers remain still. Comparison is done using the Full protocol as described in section 4.2. As first experiment, the samples from the original *PLUSVein-FV3* database are used as bona fide samples. The results are reported in table 6.3. For the training of the CNN, samples from the PROTECT [20] data set are used since they descend from the same sensor as the *PLUSVein-FV3* capturings.

One can see that methods that use a binarized vessel structures as features images for comparison (i.e. the first seven methods) have IAPMRs that are with the exception of ASAVE and RLT always above 30% which means that every third attack sample would be wrongly accepted. About 9 out of 10 attack samples would be accepted by MC and IUWT for the case of the thin LED attack instrument, posing the overall highest false accept rate for this experiment. The keypoint and texture based methods seem to be fairly unaffected by the attack samples, having IAMPRs that are often around zero.

The *PLUSVein-FV3* database does not include any video sequences and therefore new acquisitions had to be made that are used as bona fide samples

EER & IAMPR for PLUS-FV database using *PLUSVein-FV3* counterparts as bona fide [%]												
	MC	PC	GF	IUWT	RLT	WLD	ASAVE	DTFPM	SIFT	SURF	LBP	CNN
LED												
EER	0.61	0.62	1.06	0.53	4.91	1.13	2.35	2.20	0.96	3.43	3.79	2.89
IAPMR thick	72.29	71.24	37.78	79.35	43.40	69.28	24.31	16.99	0.00	0.00	0.00	0.57
IAPMR thin	89.52	80.93	60.98	90.03	36.49	84.22	19.07	16.16	0.00	0.00	0.38	0.35
Laser												
EER	1.29	1.90	2.65	1.97	6.59	2.80	2.59	2.64	0.91	3.49	4.24	6.85
IAPMR thick	58.37	55.17	31.80	79.82	23.75	57.73	8.81	5.62	0.00	0.00	0.00	0.00
IAPMR thin	75.00	64.27	53.41	84.34	17.30	78.66	1.89	6.31	0.13	0.00	0.00	0.05

Table 6.3: Results PLUS-FV threat evaluation using *PLUSVein-FV3* samples as bona fide.

for a second threat evaluation experiment. Again for every video sequence a corresponding still image is available that is used for the experiments in this section. Results for this experiment are reported in table 6.4. A significant drop in EER can be observed which can be explained by the fact that the bona fide samples have been acquired without supervision. This causes a variety of lightning artefacts which ultimately leads to less recognition accuracy. However, the overall trend for IAPMRs remains similar.

EER & IAMPR for PLUS-FV database using newly acquired samples as bona fide [%]												
	MC	PC	GF	IUWT	RLT	WLD	ASAVE	DTFPM	SIFT	SURF	LBP	CNN
LED												
EER	4.24	4.68	5.84	3.86	11.98	4.48	7.84	6.97	2.42	6.57	10.18	6.36
IAPMR thick	43.03	43.74	32.10	69.84	46.74	51.50	4.41	23.99	0.35	0.35	0.88	0.71
IAPMR thin	54.86	59.72	47.92	74.83	33.16	70.31	2.26	21.35	0.17	0.00	0.69	0.83
Laser												
EER	13.81	17.38	15.92	14.38	26.34	15.21	19.16	11.99	11.19	15.21	15.33	10.62
IAPMR thick	48.50	48.68	30.16	67.55	47.80	40.21	21.69	35.98	0.71	5.11	3.88	0.48
IAPMR thin	56.77	56.08	46.35	67.36	39.93	58.85	13.19	37.15	1.04	3.47	6.42	0.62

Table 6.4: Results PLUS-FV threat evaluation using newly acquired samples as bona fide.

It can be concluded that the PLUS-FV attack samples would pose a threat to a real system using recognition algorithms from the category that finally

create a binarized vessel image as extracted features. The keypoint- and texture-based recognition methods remain, with the exception to DTFPM to some extent, relatively unaffected by the attack samples.

Comparison to other FV Attack Databases

In order to compare the functionality of the above evaluated finger vein attack samples (i.e. PLUS-FV database) with already existing finger vein attack databases, the same experiments for threat evaluation are applied on the databases described in sections 5.2.1 and 5.2.2. While the PLUS-FV data sets are evaluated with the full protocol, the two additional databases are evaluated using the slimmer FVC protocol. The reason for this is to reduce calculation time for the recognition experiments since the SCUT database is relatively large. The CNN training and evaluation for these databases is done using a 2-fold cross validation. Since such learning is non deterministic, the feature vectors from different splits can not be compared and must be evaluated separately. Thus, the reported EERs and IAPMRs are the arithmetic mean of both folds.

VERA The threat analysis results for the VERA database are reported in table 6.5. It can be observed that for the ROI samples, the EERs are mostly higher by a significant amount for the vein pattern based approaches. This can be interpreted as that the contour of the finger plays an important role for recognition.

EER & IAMPR for VERA database [%]												
	MC	PC	GF	IUWT	RLT	WLD	ASAVE	DTFPM	SIFT	SURF	LBP	CNN
Full												
EER	2.66	2.73	6.83	4.95	30.00	6.03	9.11	10.45	4.54	11.39	7.38	6.35
IAPMR	93.18	90.45	85.76	93.18	41.67	93.48	72.58	26.21	14.24	0.91	26.82	17.57
ROI												
EER	17.72	20.91	24.55	13.30	27.19	13.18	19.10	6.69	5.43	11.62	6.91	10.18
IAPMR	53.79	49.24	53.94	64.24	38.64	63.48	68.79	81.97	44.55	14.24	73.33	8.63

Table 6.5: Threat evaluation VERA database.

While these kind of algorithms show similar results as for the PLUS database by having IAPMRs over 90%, keypoint- and texture-based approaches appear no longer to be unaffected by the attacks. The only recognition algorithms that does not get fooled by these attacks appears to be the SURF based approach when using the full version of the attack samples. Another interesting observation is that the EER for the RLT recognition

scheme is as high as 30%. Since in an publication [4] by authors from the same research institute that created the VERA set, RLT was reported with EERs from 11-19%, the results can still be considered sensible.

SCUT-SFVD The threat analysis results for the SCUT database are reported in table 6.5. Similarly to the VERA database, ROI versions of the samples yield increased EERs for the vein pattern based schemes which indicates that here the contour of the finger is a considerable factor as well. Altogether the evaluation for the VERA and the SCUT datasets is, with few exceptions, analogous.

EER & IAMPR for SCUT database [%]												
	MC	PC	GF	IUWT	RLT	WLD	ASAVE	DTFPM	SIFT	SURF	LBP	CNN
Full												
EER	4.01	4.79	9.40	6.29	14.01	7.60	11.56	8.90	2.37	5.49	8.78	0.74
IAPMR	86.33	84.67	54.90	74.06	40.36	74.21	74.98	73.75	30.43	4.18	45.43	69.80
ROI												
EER	22.59	23.41	25.05	11.87	2.94	16.96	6.03	5.63	1.92	9.41	3.51	0.83
IAPMR	27.14	26.08	23.96	16.51	9.46	18.83	59.88	55.08	34.93	7.42	55.36	55.59

Table 6.6: Threat evaluation SCUT database.

When comparing the creation processes for the attack samples from different datasets, it is important to highlight that by reprinting already captured vein images, important parts of the finger texture get preserved, while by using an already binarized vein image (i.e. veins extracted using PC), parts from the actual finger are not modelled sufficiently and are therefore apparently not able to deceive texture or keypoint based methods.

6.2.3 Hand Vein

To evaluate the attacks for the hand vein data set, similar experiments as for the finger vein data are conducted. Since exactly one video sequence is available for each setting, 10 linearly spaced frames throughout the video are used as different sessions. Doing so however is somewhat biased in terms of reducing the intra-class variability. Application of the two scenario protocol, as can be exemplary seen in figure 6.3, does often not yield satisfying results. Here, the genuine and impostor scores are perfectly separated such that a whole range of decision thresholds (for the example in the figure that uses maximum curvature feature extraction somewhere in the interval between 0.11 and 0.18) would be eligible options. The dash-dotted lines represent the two extreme cases for the decision threshold where the equal error rate is

still zero, i.e. perfect separation of genuine and zero effort impostor scores. One can now see that, depending on the decision threshold, the IAPMR varies from 0.00% to 93.15%. As mentioned, it can not be cancelled out that this perfect separation is solely due to only having one video sequence for each setting. To avoid this problem, the threat evaluation for the hand vein samples is reported using two IAPMR values that correspond to the the two extreme cases as given in table 6.7 for reflected light and transillumination respectively.

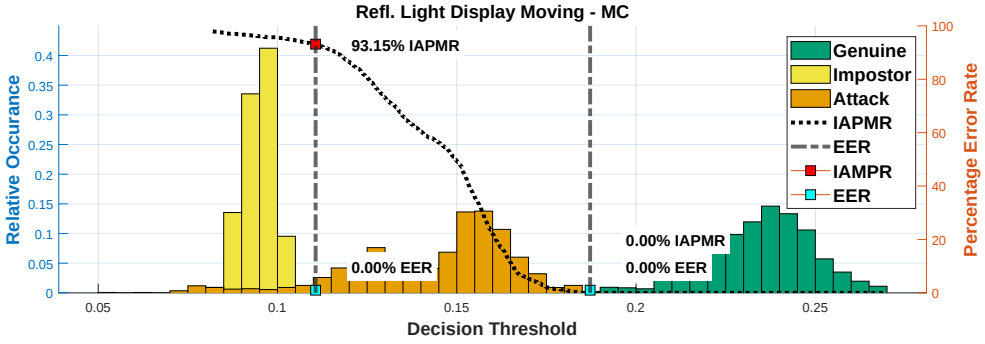

Figure 6.3: Exemplary visualization of the two scenario protocol evaluation applied on hand vein data. Maximum Curvature was used on attacks shown on a smartphone display with added translational movement. The attacks descend from data captured in reflected light illumination.

The rows with the arrow pointing right therefore contains IAPMR values for the last decision threshold before a the false non match rate would increase and the rows with the arrow pointing left contains IAPMR values for the last decision threshold before the false match rate would increase. For cases where the EER is not 0.00% the IAPMR values are the same since the upper and lower threshold coincide. For most feature extraction and comparison schemes, a disparity in IAPMR between the paper and the smartphone display attacks can be observed. The display attack samples tend to pose more threat to the algorithms than the paper attacks. A look at one sample from each attack type descending from the same subject, as shown in figure 6.4, reveals the differences in brightness between the modes. The reason why moving and zooming attacks are also included in the experiments that use only still images is to test whether subtle transformations between the used frames would make any difference. The experiments show however that in most cases the variance in IAPMR for the two paper attacks and the three

display attacks is reasonably small such that it can be concluded that it does not make much of a difference what motion type to use for threat evaluation. It can be further observed that while MC has an overall high potential to get tricked by those attacks, other binary vessel methods such as PC, RLT and WLD have rather diverse amount of vulnerability depending on the type of attack used. DTFPM, although tailored for finger vein data, appears to be overall pretty vulnerable as well. The two general purpose keypoint schemes SIFT and SURF as well as LBP seem to be overall very unimpressed by the attack samples. The CNN method was not used for hand vein samples

EER and IAPMRs Reflected Light [%]												
		MC	PC	GF	IUWT	RLT	WLD	ASAVE	DTFPM	SIFT	SURF	LBP
EER		0.00	0.00	0.10	0.00	3.25	0.00	0.00	0.00	0.00	0.18	0.00
Attacks												
Paper	→	0.00	0.00	2.66	0.00	0.00	13.50	0.00	0.00	0.00	0.00	0.00
Still	←	92.80	0.00	2.66	0.07	0.00	13.50	17.90	8.18	0.00	0.00	0.00
Paper	→	0.00	0.00	2.52	0.00	0.00	10.56	0.00	0.00	0.00	0.00	0.00
Moving	←	91.96	0.00	2.52	0.63	0.00	10.56	19.65	23.43	0.00	0.00	0.00
Display	→	0.00	8.46	12.24	2.45	64.83	0.00	0.00	0.00	0.07	3.08	0.00
Still	←	94.97	67.90	12.24	30.07	64.83	0.00	13.15	54.62	3.57	3.08	0.00
Display	→	0.00	7.97	12.38	2.59	65.38	0.00	0.00	0.00	0.00	2.45	0.00
Moving	←	93.15	71.33	12.38	37.34	65.38	0.00	19.86	64.13	3.43	2.45	0.00
Display	→	0.49	9.58	11.96	2.73	59.16	0.00	0.00	0.00	0.00	2.38	0.00
Zoom	←	95.31	73.43	11.96	36.92	59.16	0.00	12.66	58.39	3.36	2.38	0.00

EER and IAPMRs Transillumination [%]												
		MC	PC	GF	IUWT	RLT	WLD	ASAVE	DTFPM	SIFT	SURF	LBP
EER		0.00	0.00	0.00	0.00	2.66	0.00	0.00	0.00	0.00	0.00	0.00
Attacks												
Paper	→	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Still	←	94.90	5.03	0.28	6.01	0.00	11.75	38.95	85.52	0.35	0.00	6.78
Paper	→	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moving	←	92.38	4.97	0.42	5.87	0.00	10.77	32.94	79.86	0.00	0.00	3.85
Display	→	1.40	0.49	24.20	0.56	8.46	10.77	0.00	0.00	0.00	0.00	0.00
Still	←	99.65	33.15	32.87	42.24	8.46	61.82	2.17	96.22	3.64	0.00	0.00
Display	→	0.63	0.28	23.01	1.19	7.13	10.84	0.00	0.00	0.00	0.00	0.00
Moving	←	98.95	34.83	29.93	40.70	7.13	54.69	5.66	95.03	3.43	0.00	0.00
Display	→	0.14	0.21	22.87	0.21	9.02	10.56	0.00	0.00	0.00	0.00	0.00
Zoom	←	99.65	33.22	31.47	43.29	9.02	57.55	2.87	96.50	3.43	0.00	0.00

Table 6.7: Results of the threat analysis for the hand vein attacks.

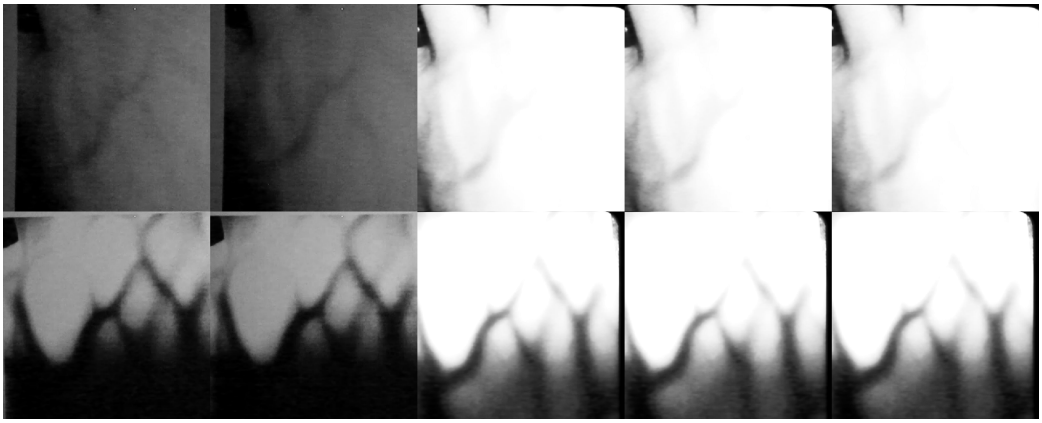

Figure 6.4: Example attack frames. Top row: Reflected light. Bottom Row: Transillumination. Columns from left to right: Paper Still, Paper Moving, Display Still, Display Moving, Display Zooming.

6.3 Attack Detection using Video Sequences

The experiments within this section test the functionality of the attack detection methods described in chapter 3 on the video databases introduced in section 5.1. Therefore, this section aims to give an answer to *RQ3*. The ISO/IEC 30107-3:2017 [2] defines metrics for presentation attack detection such as Attack Presentation Classification Error Rate (APCER) and Bona Fide Presentation Classification Error Rate (BPCER):

- *Attack Presentation Classification Error Rate (APCER)*: Proportion of attack presentations incorrectly classified as bona fide presentations in a specific scenario
- *Bona Fide Presentation Classification Error Rate (BPCER)*: Proportion of bona fide presentations incorrectly classified as presentation attacks in a specific scenario

Note that both error rates are functions parameterized by a the decision threshold. Results for the attack detection within this section are reported as the Detection Equal Error Rate (D-EER), that is, similar to the normal EER, the point where APCER = BPCER. It is worth noting that especially for the hand vein data, where only limited samples are available, both classification error rates do not necessarily overlap. Hence, the reported error rate is the interpolated intersection of APCER and BPCER as can be seen in the zoomed subplot in figure 6.5 where the greenish square demonstrates the D-EER.

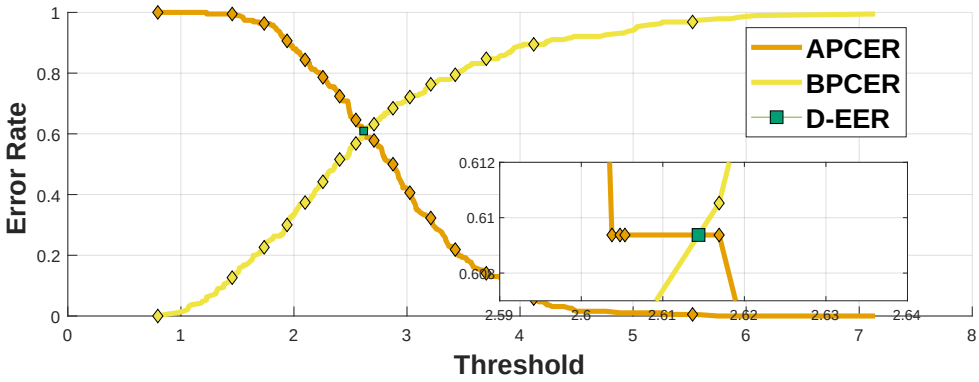

Figure 6.5: Calculation of D-EER from APCER and BPCER.

Additionally results are shown using receiver operating characteristic (ROC) curves. In general, ROC curves plot the false accept rate (which is the equivalent of the APCER) on the x axis and the true positive rate (which is the equivalent of 1-BPCER) on the y axis. In the plots in figures 6.7, 6.8, 6.9 and 6.10, the dashed orange line indicates the D-EER. The plots always show the whole diagram in the smaller window in the right corner to give depict the overall trend and a zoomed version in the bigger window. The zoom always shows the upper left corner of the whole diagram.

With the exception of the EVM approach, a Support Vector Machine is employed as the final classification step. While the authors in Bok et al. [11] proposed using a radial basis function kernel, for the experiments in this thesis only linear kernels are used. The time series signals used for the PPG methods have zero mean, i.e. the average value was subtracted from every data point per time series.

For the Bok et al. approach, the corresponding authors captured finger vein videos of variable length with a frame rate of 30 frames per second (FPS). They then split the sequences into smaller parts of length 150 frames with an overlap of 100 frames. Consequently the same is done for the hand vein videos used in this thesis. For the case of remaining frames at the end of a video, which do not fit into another video split, they are simply ignored. The finger vein videos in this work are used as is since they already consist of 150 frames, although only at a rate of 15 FPS.

The authors that initially used EVM for finger vein attack detection [62] captured video sequences over a duration of 1.67 seconds at a rate of 15 FPS, resulting in 25 frames for each video. Because the finger vein videos used in this thesis are 10 seconds long at frame rate of 15 FPS, they were split into

six smaller video sequences with 25 frames each. A similar procedure was undertaken for the hand vein videos which are of variable length. To preserve the length of 1.67 seconds for each video piece, the initial videos were cut into frame chunks consisting of 50 frames, since the hand vein videos have a frame rate of 30 FPS.

EVM settings in this thesis are adopted from [24] for all experiments. The threshold for the final classification of the motion magnitudes is chosen such that more motion (i.e. a higher motion magnitude) would indicate bona fide video samples. However, one can see for example in figure 6.7 that sometimes the classification is inverted. This implies that bona fide and attack samples could be separated but the bona fide hands are the ones with the smaller motion. Thus the numbers in brackets in the following tables show the complementary error rate for values that are above 60%.

6.3.1 Finger Vein

The results for the attack detection experiments using finger vein videos are reported in table 6.8. Comparing the two vein thicknesses on the attack samples, it can be concluded that this difference does not have much of an effect for the attack detection using video sequences.

D-EER Attack Detection Finger Vein [%]					
		Eulerian Video Magnification	PPG Bok et al.	PPG Schuiki & Uhl 1	PPG Schuiki & Uhl 2
LED	Thick Still	3.57	4.49	3.74	0.52
	Thick Trembling	58.51	9.62	11.75	7.05
	Thin Still	3.31	1.85	6.60	0.43
	Thin Trembling	62.92 (37.08)	23.38	23.38	10.90
Laser	Thick Still	6.52	12.12	1.05	1.94
	Thick Trembling	72.78 (27.22)	26.48	16.84	24.62
	Thin Still	7.70	4.80	0.58	0.51
	Thin Trembling	73.48 (26.52)	24.97	29.85	22.42

Table 6.8: Results attack detection experiments using finger vein videos. Values in brackets for the EVM approach indicate the complementary error rate that would apply if one would assign the label *attack sample* to higher motion magnitudes instead of *bona fide*.

The approaches from sections 3.3 and 3.4 (denoted in the table as *PPG Schuiki & Uhl 1* and *2*) were initially developed for the hand vein video data set. Consequently, slightly modified parameters are used for the analysis

of the time series descending from finger vein videos. The window size for both methods is set to 50. Because the maximum frequency obtainable with 15 FPS is 7.5 Hertz, as given by Shannon's theorem $f_{max} \leq \frac{f_{sample}}{2}$, three harmonics are considered for the method 2.

While the EVM approach seems to work against attacks with no additional movement applied with D-EERs in the range of 3-8%, the video sequences with applied movement (Trembling) seem to cause difficulties with error rates up to 58% which is close to guessing. Interestingly a jump in error rates can be observed for the PPG method Schuiki 2 when comparing moved LED and Laser video sequences. Another interesting observation can be made when averaging the frequency spectra from every time series as seen in figure 6.6. Peaks at certain frequencies that seem to be inherent to the illumination source used appear in every video sequence regardless of its content (i.e. real or fake finger). A possible explanation could be that this is some kind of aliasing from the control of the respective illumination source like pulse width modulation. Summarized it can be said that sound attack detection could be achieved for the videos without extra motion with all four methods. However, for the case of videos with trembling effects, the overall best achieved D-EER is 7.05% which indicates that these attacks are harder to detect.

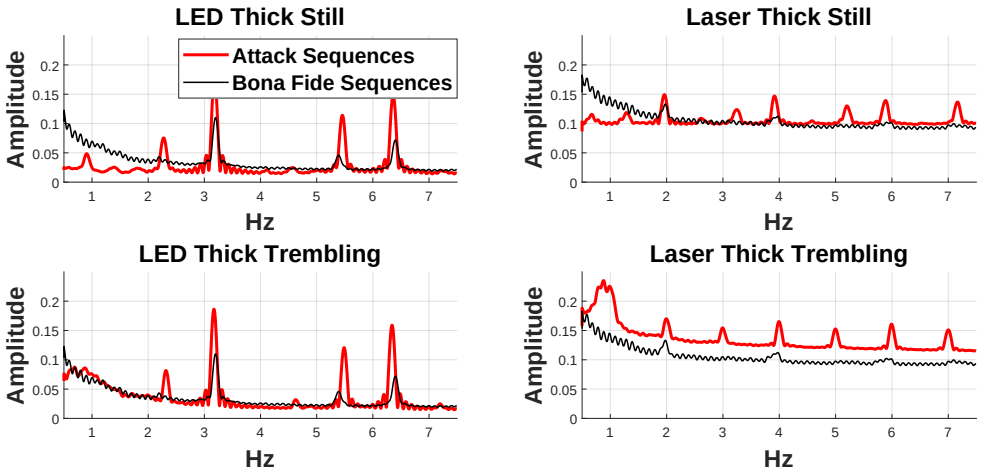

Figure 6.6: Averaged frequency spectra of time series generated finger vein video sequences. Only thick attack vein patterns are used in the plots since figures for thin vein pattern are quite similar.

Figures 6.7 and 6.8 contain ROC curves for all finger vein experiments.

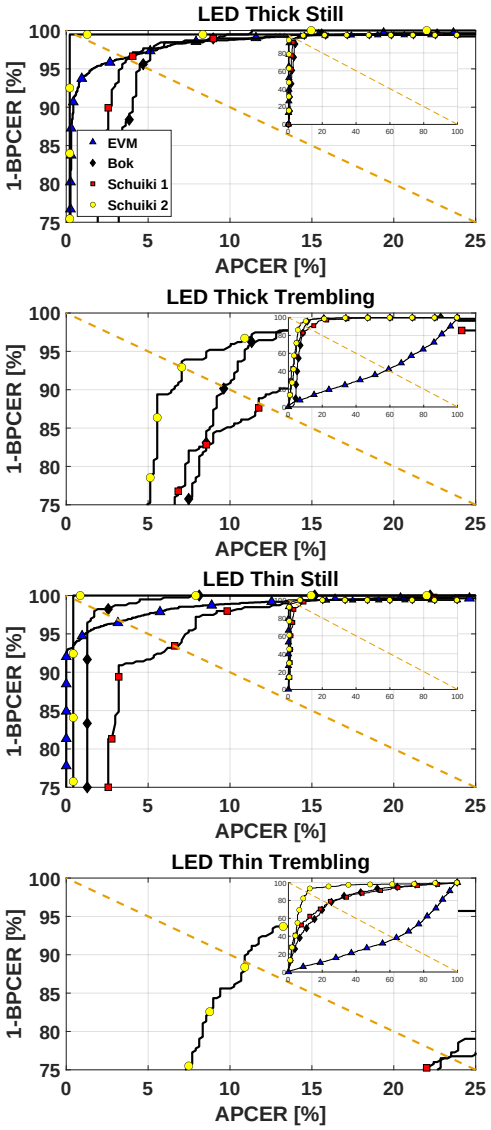

Figure 6.7: Finger vein attack detection ROC plots LED.

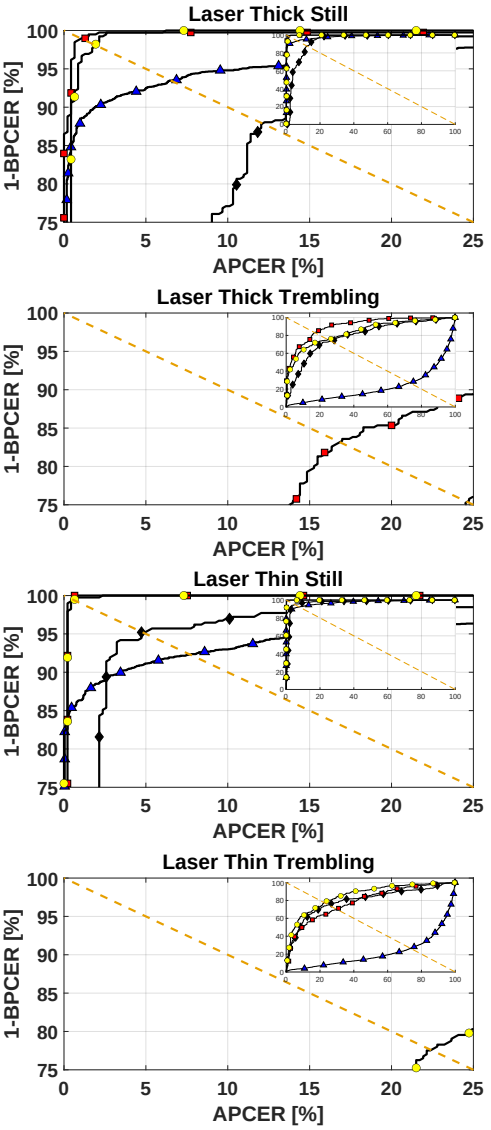

Figure 6.8: Finger vein attack detection ROC plots Laser.

6.3.2 Hand Vein

As mentioned earlier in this section, for the EVM approach and the PPG method from Bok et al., the hand vein videos were split into smaller pieces. Previous to splitting, due to various step and peak artefacts at the beginning of generated time series, the first three seconds from each video are ignored for all attack detection methods similar to the experiments in [24]. The settings for the Schuiki 1 & 2 approach are adopted from [68], meaning that a window size of 150 frames is used per video and five harmonics are considered for the method 2. The results for the attack detection experiments using hand vein videos are reported in table 6.9.

D-EER Attack Detection Hand Vein [%]					
		Eulerian Video Magnification	PPG Bok et al.	PPG Schuiki & Uhl 1	PPG Schuiki & Uhl 2
Refl. Light	Paper Still	60.94 (39.06)	9.75	23.08	7.69
	Paper Moving	87.10 (12.90)	1.46	0.00	0.00
	Display Still	8.06	16.81	11.54	3.85
	Display Moving	41.02	7.63	3.85	7.69
	Display Zooming	53.08	0.37	0.00	0.00
Transill.	Paper Still	65.44 (34.56)	15.66	15.38	3.85
	Paper Moving	86.81 (13.19)	0.00	19.23	0.00
	Display Still	22.01	31.54	0.00	0.00
	Display Moving	74.18 (25.82)	19.26	0.00	3.85
	Display Zooming	73.63 (26.37)	7.60	0.00	0.00

Table 6.9: Results attack detection experiments using finger vein videos. Values in brackets for the EVM approach indicate the complementary error rate.

The results from the EVM approach suggest that this method is not perfectly suited for attack detection of the hand vein dataset with only one error rate being below 10% (8.06% Reflected Light Display Still). The Bok et al. approach seem to work better on moving and zooming attacks as compared to the still attacks. Since for the Schuiki 1 & 2 approaches exactly 26 bona fide and 26 attack sequences are available one should consider that 3.85% error rate corresponds to one misclassification $\frac{1}{26} = 3.85\%$ in both classes. While method 1 (windowed majority voting) sometimes yields perfect separation and sometimes has error rates at around 20%, method 2 (windowed analysis of harmonics) seems to be overall fairly robust.

Figures 6.9 and 6.10 contain ROC curves for the hand vein experiments.

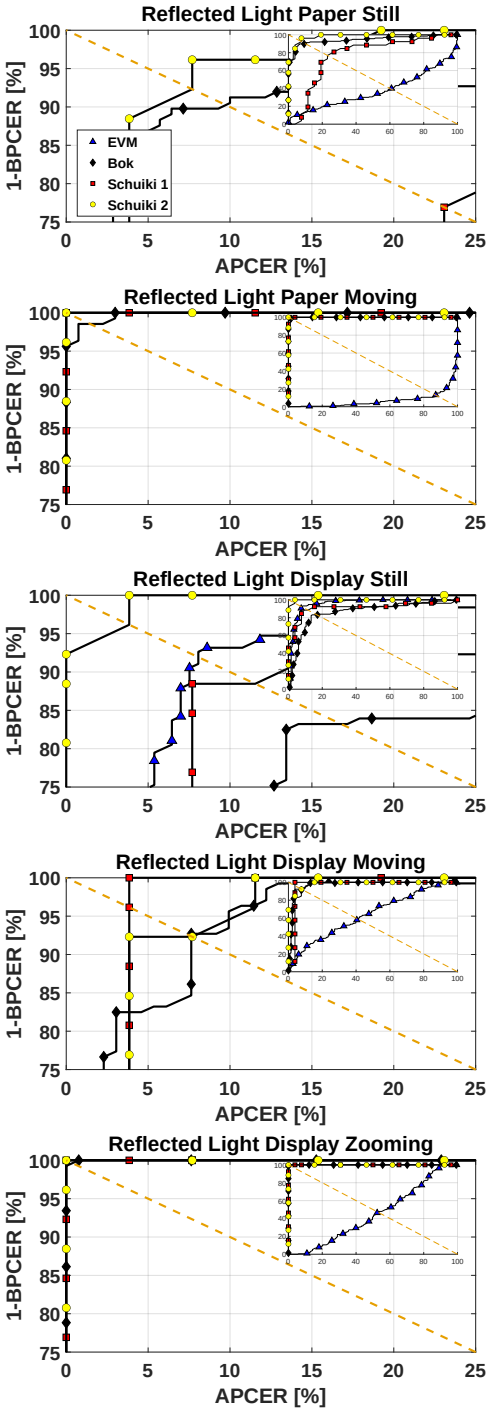

Figure 6.9: Hand vein attack detection ROC plots reflected light.

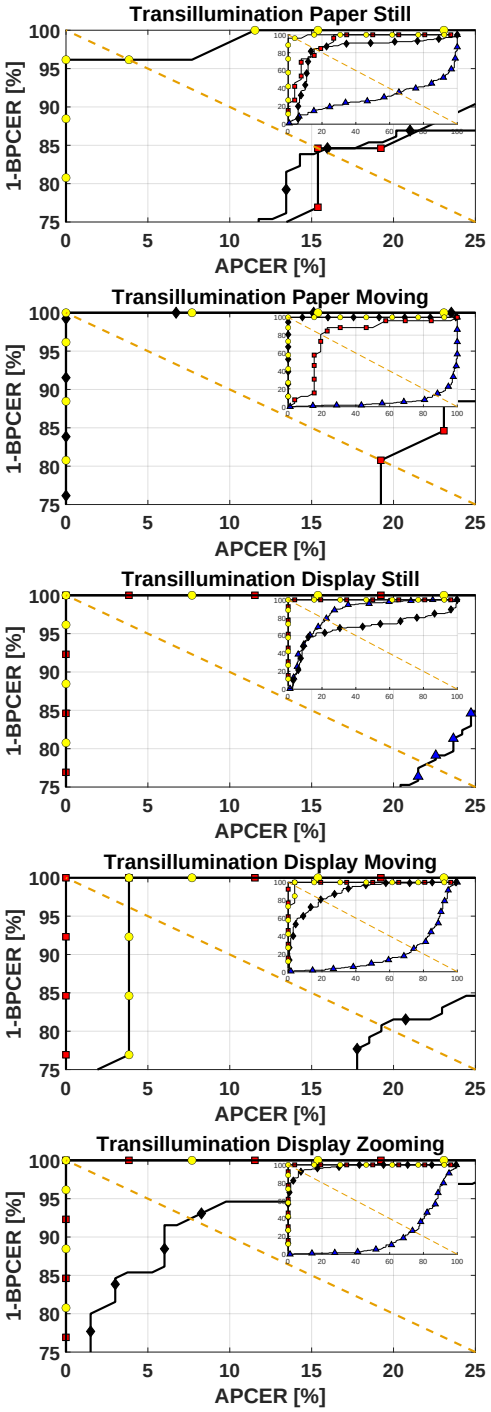

Figure 6.10: Hand vein attack detection ROC plots transillumination.

7 Summary

The research objective of the present master thesis was to to investigate the distinguishability of videos containing real and fake vein patterns the hand region by using methods that extract information from consecutive video frames to classify a given video. Experimental results should be obtained by using one finger vein and one hand vein video data set captured by the the Multimedia Signal Processing and Security Lab at the University of Salzburg. This task was further divided into smaller consecutive steps:

In total, four methods were found to be suitable candidates for the attack detection experiments including two methods that were developed in the course of this thesis. One method aims to amplify tiny motions which are often too small to be seen with the naked eye. Employing such an artificial microscope, in theory, appears especially useful when looking for small motion artefacts in the hand region that was created by the human blood flow. The three other methods build upon a common preprocessing step. A time series is built by averaging the brightness value from every pixel in a frame. After doing so, the generated time series is then transformed into frequency space using Fourier transform. Essentially the three methods differ in the approach how a final feature vector is created from the frequency space.

Prior to conducting experiments on the performance of the attack detection methods on the two video data sets, it was essential to evaluate whether the attack samples would actually be able to deceive a real recognition system. To do so, twelve state of the art vein recognition methodologies that can be categorized into three categories of algorithms were tested on their vulnerability to those attacks. The recognition algorithms include texture based ones, keypoint based ones and ones that try to segment vessel structures to create a binary image containing only the vein networks. To evaluate the threat of a given attack set, first a suitable decision threshold is found that would work well when only considering captures from real fingers. As a second step, the ratio of wrongly accepted attack samples is calculated using

the decision threshold from the first step, thus denoted two scenario protocol. Additionally, the same threat analysis was undertaken for two already existing finger vein datasets in order to be able to make a comparison. It can be concluded that the two data sets under test in this thesis pose a threat to at least some degree, however mainly to the algorithms that create binarized vessel structures as a feature image.

Finally, the four attack detection methods were applied on the two video datasets. Experimental results for the finger vein data suggest that the attack sequences where additional motion was applied to the attack instruments are more challenging than the unmoved ones. For the dorsal hand vein data, no such clear trend regarding type of attack can be identified. The method that uses harmonic analysis seems to achieve sound hand vein attack detection overall, considering that 7.69% corresponds to a misclassification of 4 video sequences out of 52 for this method. Although in any case one of the methods that were developed in the course of this thesis appears to outperform the other two methods under test, the achieved results demand further research in this area.

Bibliography

[1] ISO/IEC JTC 1/SC 37 Biometrics. Information technology — biometric presentation attack detection — part 1: Framework. Standard ISO/IEC 30107-1:2016, International Organization for Standardization, Geneva, CH, 2016.

[2] ISO/IEC JTC 1/SC 37 Biometrics. Information technology — biometric presentation attack detection — part 3: Testing and reporting. Standard ISO/IEC 30107-3:2017, International Organization for Standardization, Geneva, CH, 2017.

[3] ISO/IEC JTC 1/SC 37 Biometrics. Information technology — vocabulary — part 37: Biometrics. Standard ISO/IEC 2382-37:2017, International Organization for Standardization, Geneva, CH, 2017.

[4] André Anjos, Pedro Tome, and Sébastien Marcel. An introduction to vein presentation attacks and detection. In Sébastien Marcel, Mark S. Nixon, Julian Fierrez, and Nicholas Evans, editors, *Handbook of Biometric Anti-Spoofing: Presentation Attack Detection*, pages 419–438. Springer International Publishing, Cham, 2019.

[5] Nurul Nabihah Ashari, J. H. Teng, T. S. Ong, and S. M. A. Kalaiarasi. Finger vein presentation attack detection based on texture analysis. In Rayner Alfred, Hiroyuki Iida, Haviluddin Haviluddin, and Patricia Anthony, editors, *Computational Science and Technology*, pages 427–436, Singapore, 2021. Springer Singapore.

[6] Herbert Bay. *From wide-baseline point and line correspondences to 3D*. PhD thesis, Eidgenössische Technische Hochschule ETH Zürich, 2006. https://doi.org/10.3929/ethz-a-005212689.

[7] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[8] Asker M. Bazen and Sabih H. Gerez. Fingerprint matching by thin-plate spline modelling of elastic deformations. *Pattern Recognition*, 36(8):1859–1867, 2003.

[9] Shruti Bhilare, Vivek Kanhangad, and Narendra Chaudhari. Histogram of oriented gradients based presentation attack detection in dorsal hand-vein biometric system. In *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*, pages 39–42, 2017.

[10] Amrit Pal Singh Bhogal, Dominik Söllinger, Pauline Trung, Jutta Hämmerle-Uhl, and Andreas Uhl. Non-reference image quality assessment for fingervein presentation attack detection. In *Proceedings of 20th Scandinavian Conference on Image Analysis (SCIA'17)*, volume 10269 of *Springer Lecture Notes on Computer Science*, pages 184–196, 2017.

[11] Jin Bok, Kun Suh, and Eui Chul Lee. Detecting fake finger-vein data using remote photoplethysmography. *Electronics*, 8:1016, 09 2019.

[12] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. In Tomás Pajdla and Jiří Matas, editors, *Computer Vision - ECCV 2004*, pages 25–36, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[13] Ivana Chingovska, Amir Mohammadi, André Anjos, and Sébastien Marcel. Evaluation methodologies for biometric presentation attack detection. In Sébastien Marcel, Mark S. Nixon, Julian Fierrez, and Nicholas Evans, editors, *Handbook of Biometric Anti-Spoofing: Presentation Attack Detection*, pages 457–480. Springer International Publishing, Cham, 2019.

[14] Joon Hwan Choi, Wonseok Song, Taejeong Kim, Seung-Rae Lee, and Hee Chan Kim. Finger vein extraction using gradient normalization and principal curvature. In *Image Processing: Machine Vision Applications II*, volume 7251, pages 7251 – 7251 – 9, 2009.

[15] David A. Connell, George Koulouris, Duncan A. Thorn, and Hollis G. Potter. Contrast-enhanced MR angiography of the hand. *RadioGraphics*, 22(3):583–599, May 2002.

[16] Adam Czajka. Lecture notes on biometrics cse 40537/60537, University of Notre Dame, spring 2019, 2019.

[17] Michele De Santis, Sandro Agnelli, Donatella Nardiello, and Antonio Iula. 3d ultrasound palm vein recognition through the centroid method for biometric purposes. In *2017 IEEE International Ultrasonics Symposium (IUS)*, pages 1–4, 2017.

[18] Luca Debiasi, Christof Kauba, Heinz Hofbauer, Bernhard Prommegger, and Andreas Uhl. Presentation attacks and detection in finger- and hand-vein recognition. In *Proceedings of the Joint Austrian Computer Vision and Robotics Workshop (ACVRW'20)*, pages 65–70, Graz, Austria, 2020.

[19] Henley Ding. Anti-spoofing a finger vascular recognition device with pulse detection. In *24th Twente Student Conference on IT*, Enschede, The Netherlands, 01 2015. University of Twente.

[20] Chiara Galdi, Jonathan Boyle, Lulu Chen, Valeria Chiesa, Luca Debiasi, Jean-Luc Dugelay, James Ferryman, Artur Grudzień, Christof Kauba, Simon Kirchgasser, Marcin Kowalski, Michael Linortner, Patryk Maik, Kacper Michoń, Luis Patino, Bernhard Prommegger, Ana F. Sequeira, Łukasz Szklarski, and Andreas Uhl. Protect: Pervasive and user focused biometrics border project – a case study. *IET Biometrics*, 9(6):297–308, 2020.

[21] Zeno Geradts. Forensic implications of identity systems. *Datenschutz und Datensicherheit - DuD*, 30(9):557–559, Sep 2006.

[22] Zeno Geradts and Peter Sommer. D6.1: Forensic implications of identity management systems. In Zeno Geradts and Peter Sommer, editors, *FIDIS Deliverables*, 2006. `http://www.fidis.net/fileadmin/fidis/deliverables/fidis-wp6-del6.1.forensic_implications_of_identity_management_systems.pdf`.

[23] Alexander Gruschina. Veinplus: A transillumination and reflection-based hand vein database. In *Proceedings of the 39th annual workshop of the Austrian association for pattern recognition (OAGM'15)*, 2015.

[24] Thomas Herzog and Andreas Uhl. Analysing a vein liveness detection scheme. In *Proceedings of the 8th International Workshop on Biometrics and Forensics (IWBF'20)*, pages 1–6, Porto, Portugal, 2020.

[25] Lin Hong, Yifei Wan, and A. Jain. Fingerprint image enhancement: algorithm and performance evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):777–789, 1998.

[26] Beining Huang, Yanggang Dai, Rongfeng Li, Darun Tang, and Wenxin Li. Finger-vein authentication based on wide line detector and pattern normalization. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 1269–1272. IEEE, 2010.

[27] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size, 2016.

[28] Antonio Iula. Optimization and evaluation of a biometric recognition technique based on 3d ultrasound palm vein. In *2020 IEEE International Ultrasonics Symposium (IUS)*, pages 1–4, 2020.

[29] Antonio Iula, Alessandro Savoia, and Giosuà Caliano. 3d ultrasound palm vein pattern for biometric recognition. In *2012 IEEE International Ultrasonics Symposium*, pages 1–4, 2012.

[30] A. K. Jain, Ruud M. Bolle, and Sharath Pankanti. Introduction to Biometrics. In A. K. Jain, Ruud M. Bolle, and Sharath Pankanti, editors, *Biometrics: Personal Identification in Networked Society*, pages 1–41. Springer US, 1999.

[31] A.K. Jain, A. Ross, and S. Pankanti. Biometrics: a tool for information security. *IEEE Transactions on Information Forensics and Security*, 1(2):125–143, 2006.

[32] A.K. Jain, A. Ross, and S. Prabhakar. An introduction to biometric recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):4–20, 2004.

[33] Anil K. Jain, Arun A. Ross, and Karthik Nandakumar. *Introduction to Biometrics*. Springer US, 2011.

[34] Shou-kun JIANG, Fu Liu, Bing KANG, Zi-yue YOU, and Yu-xuan ZONG. Dorsal hand vein enhancement and fake vein detection. *DEStech Transactions on Computer Science and Engineering*, 11 2017.

[35] Christof Kauba, Bernhard Prommegger, and Andreas Uhl. Focussing the beam - a new laser illumination based data set providing insights to finger-vein recognition. In *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–9, Los Angeles, California, USA, 2018.

[36] Christof Kauba, Bernhard Prommegger, and Andreas Uhl. The two sides of the finger - an evaluation on the recognition performance of dorsal vs. palmar finger-veins. In *Proceedings of the International Conference of the Biometrics Special Interest Group (BIOSIG'18)*, pages 1–8, Darmstadt, Germany, 2018.

[37] Christof Kauba, Bernhard Prommegger, and Andreas Uhl. Openvein - an open-source modular multipurpose finger vein scanner design. In Andreas Uhl, Christoph Busch, Sebastien Marcel, and Raymond Veldhuis, editors, *Handbook of Vascular Biometrics*, chapter 3, pages 77–111. Springer Nature Switzerland AG, Cham, Switzerland, 2019.

[38] Christof Kauba and Andreas Uhl. An available open-source vein recognition framework. In Andreas Uhl, Christoph Busch, Sebastien Marcel, and Raymond Veldhuis, editors, *Handbook of Vascular Biometrics*, chapter 4, pages 113–142. Springer Nature Switzerland AG, Cham, Switzerland, 2019.

[39] Daniel Kocher, Stefan Schwarz, and Andreas Uhl. Empirical evaluation of lbp-extension features for finger vein spoofing detection. In *Proceedings of the International Conference of the Biometrics Special Interest Group (BIOSIG'16)*, page 8, Darmstadt, Germany, 2016.

[40] Jan Krissler and Julian. Venenerkennung hacken - Vom Fall der letzten Bastion biometrischer Systeme. Chaos Computer Club e.V., 2018. `https://doi.org/10.5446/39201`, last accessed: 18 Jun 2021.

[41] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 1097–1105, Red Hook, NY, USA, 2012. Curran Associates Inc.

[42] A. Kumar and Y. Zhou. Human identification using finger images. *IEEE Transactions on Image Processing*, 21(4):2228–2244, 2012.

[43] Eui Chul Lee, Hyeon Chang Lee, and Kang Ryoung Park. Finger vein recognition using minutia-based alignment and local binary pattern-based feature extraction. *Int. J. Imaging Syst. Technol.*, 19(3):179–186, September 2009.

[44] W. Q. Janie Lee, Thian Song Ong, Tee Connie, and H. T. Jackson. Finger vein presentation attack detection with optimized lbp variants.

In Mohammed Anbar, Nibras Abdullah, and Selvakumar Manickam, editors, *Advances in Cyber Security*, pages 468–478, Singapore, 2021. Springer Singapore.

[45] Chih-Lung Lin and Kuo-Chin Fan. Biometric verification using thermal images of palm-dorsa vein patterns. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(2):199–213, 2004.

[46] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov 2004.

[47] Babak Maser, Dominik Söllinger, and Andreas Uhl. Prnu-based detection of finger vein presentation attacks. In *Proceedings of the 7th International Workshop on Biometrics and Forensics (IWBF'19)*, pages 1–6, Cancun, Mexico, 2019.

[48] Yusuke Matsuda, Naoto Miura, Akio Nagasaka, Harumi Kiyomizu, and Takafumi Miyatake. Finger-vein authentication based on deformation-tolerant feature-point matching. *Machine Vision and Applications*, 27, 02 2016.

[49] Y. Matsumoto, Y. Asao, A. Yoshikawa, H. Sekiguchi, M. Takada, M. Furu, S. Saito, M. Kataoka, H. Abe, T. Yagi, K. Togashi, and M. Toi. Label-free photoacoustic imaging of human palmar vessels: a structural morphological analysis. *Scientific Reports*, 8(1):786, Jan 2018.

[50] N. Miura, Akio Nagasaka, and T. Miyatake. Extraction of finger-vein patterns using maximum curvature points in image profiles. *IEICE Trans. Inf. Syst.*, 90-D:1185–1194, 2005.

[51] Naoto Miura, Akio Nagasaka, and Takafumi Miyatake. Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification. *Machine Vision and Applications*, 15:194–203, 10 2004.

[52] Dat Nguyen, Hyo Yoon, Tuyen Pham, and Kang Park. Spoof detection for finger-vein recognition system using nir camera. *Sensors*, 17:2261, 10 2017.

[53] Dat Tien Nguyen, Young Ho Park, Kwang Yong Shin, Seung Yong Kwon, Hyeon Chang Lee, and Kang Ryoung Park. Fake finger-vein

image detection based on fourier and wavelet transforms. *Digital Signal Processing*, 23(5):1401–1413, 2013.

[54] J. A. Nijboer, J. C. Dorlas, and Hans F. Mahieu. Photoelectric plethysmography–some fundamental aspects of the reflection and transmission method. *Clinical physics and physiological measurement : an official journal of the Hospital Physicists' Association, Deutsche Gesellschaft fur Medizinische Physik and the European Federation of Organisations for Medical Physics*, 2 3:205–15, 1981.

[55] P. Normakristagaluh, L.J. Spreeuwers, and R.N.J. Veldhuis. A prototype of finger-vein phantom. In Luuk Spreeuwers and Jasper Goseling, editors, *Proceedings of the 2018 Symposium on Information Theory and Signal Processing in the Benelux*, pages 163–166, Netherlands, May 2018. Werkgemeenschap voor Informatie- en Communicatietheorie (WIC).

[56] T. Ojala, M. Pietikainen, and D. Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 582–585 vol.1, 1994.

[57] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[58] Akira Otsuka, Tetsushi Ohki, Ryogo Morita, Manabu Inuma, and Hideki Imai. Security evaluation of a finger vein authentication algorithm against wolf attack. 37th IEEE Symposium on Security and Privacy, San Jose, CA, may 2016.

[59] I. Patil, S. Bhilare, and V. Kanhangad. Assessing vulnerability of dorsal hand-vein verification system to spoofing attacks using smartphone camera. In *2016 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA)*, pages 1–6, 2016.

[60] X. Qiu, W. Kang, S. Tian, W. Jia, and Z. Huang. Finger vein presentation attack detection using total variation decomposition. *IEEE Transactions on Information Forensics and Security*, 13(2):465–477, 2018.

[61] Xinwei Qiu, Senping Tian, Wenxiong Kang, Wei Jia, and Qiuxia Wu. Finger vein presentation attack detection using convolutional neural

networks. In Jie Zhou, Yunhong Wang, Zhenan Sun, Yong Xu, Linlin Shen, Jianjiang Feng, Shiguang Shan, Yu Qiao, Zhenhua Guo, and Shiqi Yu, editors, *Biometric Recognition*, pages 296–305, Cham, 2017. Springer International Publishing.

[62] R. Raghavendra, M. Avinash, S. Marcel, and C. Busch. Finger vein liveness detection using motion magnification. In *2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–7, 2015.

[63] R. Raghavendra and C. Busch. Presentation attack detection algorithms for finger vein biometrics: A comprehensive study. In *2015 11th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, pages 628–632, 2015.

[64] R. Raghavendra, S. Venkatesh, K. B. Raja, and C. Busch. Transferable deep convolutional neural network features for fingervein presentation attack detection. In *2017 5th International Workshop on Biometrics and Forensics (IWBF)*, pages 1–5, 2017.

[65] Ramachandra Raghavendra, Kiran B. Raja, Sushma Venkatesh, and Christoph Busch. Fingervein presentation attack detection using transferable features from deep convolution neural networks. In Angshul Majumdar Mayank Vatsa, Richa Singh, editor, *Deep Learning in Biometrics*, chapter 12, pages 295–395. CRC Press, mar 2018.

[66] N. K. Ratha, J. H. Connell, and R. M. Bolle. Enhancing security and privacy in biometrics-based authentication systems. *IBM Systems Journal*, 40(3):614–634, 2001.

[67] Johannes Schuiki, Bernhard Prommegger, and Andreas Uhl. Confronting a variety of finger vein recognition algorithms with wax presentation attack artefacts. In *Proceedings of the 9th IEEE International Workshop on Biometrics and Forensics (IWBF'21)*, pages 1–6, Rome, Italy (moved to virtual), 2021.

[68] Johannes Schuiki and Andreas Uhl. Improved liveness detection in dorsal hand vein videos using photoplethysmography. In *Proceedings of the IEEE 19th International Conference of the Biometrics Special Interest Group (BIOSIG 2020)*, pages 57–65, Darmstadt, Germany, 2020.

[69] Johannes Schuiki, Georg Wimmer, and Andreas Uhl. Vulnerability assessment and presentation attack detection using a set of distinct finger

vein recognition algorithms. In *Proceedings of the 2021 International Joint Conference on Biometrics (IJCB'21)*, pages 1–7, Shenzen, China, 2021.

[70] Soumyadip Sengupta, Angjoo Kanazawa, Carlos D. Castillo, and David W. Jacobs. Sfsnet: Learning shape, reflectance and illuminance of faces 'in the wild'. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6296–6305, 2018.

[71] J. M. Singh, S. Venkatesh, K. B. Raja, R. Ramachandra, and C. Busch. Detecting finger-vein presentation attacks using 3d shape diffuse reflectance decomposition. In *2019 15th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, pages 8–14, 2019.

[72] Wonseok Song, Taejeong Kim, Hee Chan Kim, Joon Hwan Choi, Hyoun-Joong Kong, and Seung-Rae Lee. A finger-vein verification system using mean curvature. *Pattern Recognition Letters*, 32(11):1541–1547, August 2011.

[73] J. Starck, J. Fadili, and F. Murtagh. The undecimated wavelet decomposition and its reconstruction. *IEEE Transactions on Image Processing*, 16(2):297–309, 2007.

[74] Jean-Luc Starck and Fionn Murtagh. *Astronomical Image and Data Analysis*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[75] George Stockman and Linda G. Shapiro. *Computer Vision*. Prentice Hall PTR, USA, 1st edition, 2001.

[76] Dominik Söllinger, Pauline Trung, and Andreas Uhl. Non-reference image quality assessment and natural scene statistics to counter biometric sensor spoofing. *IET Biometrics*, 7(4):314–324, 2018.

[77] S. Tirunagari, N. Poh, M. Bober, and D. Windridge. Windowed dmd as a microtexture descriptor for finger vein counter-spoofing in biometrics. In *2015 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6, 2015.

[78] P. Tome, R. Raghavendra, C. Busch, S. Tirunagari, N. Poh, B. H. Shekar, D. Gragnaniello, C. Sansone, L. Verdoliva, and S. Marcel. The 1st competition on counter measures to finger vein spoofing attacks. In *2015 International Conference on Biometrics (ICB)*, pages 513–518, 2015.

[79] P. Tome, M. Vanoni, and S. Marcel. On the vulnerability of finger vein recognition to spoofing. In *2014 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–10, 2014.

[80] Andreas Uhl. State of the art in vascular biometrics. In Andreas Uhl, Christoph Busch, Sébastien Marcel, and Raymond Veldhuis, editors, *Handbook of Vascular Biometrics*, pages 3–61, Cham, 2020. Springer International Publishing.

[81] Andreas Uhl. Eye-based vascular patterns. In Sushil Jajodia, Pierangela Samarati, and Moti Yung, editors, *Encyclopedia of Cryptography, Security and Privacy*, pages 1–4. Springer Berlin Heidelberg, Berlin, Heidelberg, Germany, 2021.

[82] Andreas Uhl. Hand-based vascular patterns. In Sushil Jajodia, Pierangela Samarati, and Moti Yung, editors, *Encyclopedia of Cryptography, Security and Privacy*, pages 1–5. Springer Berlin Heidelberg, Berlin, Heidelberg, Germany, 2021.

[83] Hemant Vallabh. Authentication using finger-vein recognition. Master's thesis, University of Johannesburg, 2012.

[84] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001.

[85] Yiding Wang, Qi Qi, and Kefeng Li. Liveness detection of dorsal hand vein based on autoregressive model. In *2014 IEEE Computers, Communications and IT Applications Conference*, pages 206–210, 2014.

[86] Yiding Wang, Di Zhang, and Qi Qi. Liveness detection for dorsal hand vein recognition. *Personal Ubiquitous Comput.*, 20(3):447–455, June 2016.

[87] Yiding Wang and Zhanyong Zhao. Liveness detection of dorsal hand vein based on the analysis of fourier spectral. In Zhenan Sun, Shiguan Shan, Gongping Yang, Jie Zhou, Yunhong Wang, and YiLong Yin, editors, *Biometric Recognition*, pages 322–329, Cham, 2013. Springer International Publishing.

[88] Yuehang Wang, Zhengxiong Li, Tri Vu, Nikhila Nyayapathi, Kwang W. Oh, Wenyao Xu, and Jun Xia. A robust and secure palm vessel biometric sensing system based on photoacoustics. *IEEE Sensors Journal*, 18(14):5993–6000, 2018.

[89] J Wayman. A definition of biometrics. *National Biometric Test Center Collected Works*, 1(2):21–23, 2000.

[90] James L. Wayman. Biometric verification / identification / authentication / recognition: The terminology. In Stan Z. Li and Anil Jain, editors, *Encyclopedia of Biometrics*, pages 153–157. Springer US, Boston, MA, 2009.

[91] Ching-Chuan Wei, Chin-Ming Huang, and Yin-Tzu Liao. The exponential decay characteristic of the spectral distribution of blood pressure wave in radial artery. *Computers in Biology and Medicine*, 39(5):453 – 459, 2009.

[92] Georg Wimmer, Bernhard Prommegger, and Andreas Uhl. Finger vein recognition and intra-subject similarity evaluation of finger veins using the cnn triplet loss. In *Proceedings of the 25th International Conference on Pattern Recognition (ICPR)*, pages 400–406, 2020.

[93] Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, and William T. Freeman. Eulerian video magnification for revealing subtle changes in the world. *ACM Transactions on Graphics (Proc. SIGGRAPH 2012)*, 31(4), 2012.

[94] L. Yang, G. Yang, Y. Yin, and X. Xi. Finger vein recognition with anatomy structure analysis. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(8):1892–1905, 2018.

[95] W. Yang, W. Luo, W. Kang, Z. Huang, and Q. Wu. Fvras-net: An embedded finger-vein recognition and antispoofing system using a unified cnn. *IEEE Transactions on Instrumentation and Measurement*, 69(11):8690–8701, 2020.

[96] Yapeng Ye, He Zheng, Liao Ni, Shilei Liu, and Wenxin Li. A study on the individuality of finger vein based on statistical analysis. In *2016 International Conference on Biometrics (ICB)*, pages 1–5, 2016.

[97] Y. Zhan, A. Singh Rathore, G. Milione, Y. Wang, W. Zheng, W. Xu, and J. Xia. 3D finger vein biometric authentication with photoacoustic tomography. *Appl Opt*, 59(28):8751–8758, Oct 2020.

[98] Jing Zhang and Jinfeng Yang. Finger-vein image enhancement based on combination of gray-level grouping and circular gabor filter. In *2009 International Conference on Information Engineering and Computer Science*, pages 1–4, 2009.

[99] Jianjun Zhao, Hogliang Tian, Weixing Xu, and Xin Li. A new approach to hand vein image enhancement. In *2009 Second International Conference on Intelligent Computation Technology and Automation*, volume 1, pages 499–501, 2009.

[100] Karel Zuiderveld. *Contrast Limited Adaptive Histogram Equalization*, page 474–485. Academic Press Professional, Inc., USA, 1994.