BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INTELLIGENT SYSTEMS
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

ALGORITHMIC EVALUATION OF THE QUALITY OF DACTYLOSCOPIC TRACES

ALGORITMICKÉ HODNOCENÍ KVALITY DAKTYLOSKOPICKÝCH STOP

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR **ONDŘEJ SLOUP**
AUTOR PRÁCE

SUPERVISOR **prof. Ing., Dipl.-Ing. MARTIN DRAHANSKÝ, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2022

Brno University of Technology
Faculty of Information Technology

Department of Intelligent Systems (DITS) Academic year 2021/2022

Bachelor's Thesis Specification

25028

Student: **Sloup Ondřej**
Programme: Information Technology
Title: **Algorithmic Evaluation of the Quality of Dactyloscopic Traces**
Category: Image Processing

Assignment:

1. Study the literature on the processing, recognition and evaluation of the quality of dactyloscopic traces. Familiarize yourself with the available data, including poor quality records.
2. Propose an algorithmic solution for the detection and evaluation of the quality of the dactyloscopic trace based on the input and detected parameters.
3. Create a program that will implement your solution.
4. Test your solution. Summarize the results achieved and discuss possible improvements.

Recommended literature:

- MALTONI, Davide, et al. *Handbook of fingerprint recognition*. Springer Science & Business Media, 2009.
- DRAHANSKÝ, Martin (ed.). *Hand-based biometrics: methods and technology*. Institution of Engineering and Technology, 2018.
- DRAHANSKÝ, Martin, et al. *Biometrie*, 2011.

Requirements for the first semester:

- Points 1 and 2.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor: **Drahanský Martin, prof. Ing., Dipl.-Ing., Ph.D.**
Head of Department: Hanáček Petr, doc. Dr. Ing.
Beginning of work: November 1, 2021
Submission deadline: May 11, 2022
Approval date: November 4, 2021

Abstract

Dactyloscopic traces are one of the critical aspects of biometric identification. They represent an element by which people can be authenticated and authorised. Nonetheless, it is necessary to evaluate if a given fingerprint is valid by the number of features it provides and decide if it is usable or useless. This analysis of features tells us how valuable the fingerprint is. We established a process that grades fingerprints based on contextual and statistical values using various enhancements and grading algorithms. These algorithms can determine if the fingerprint is good quality and whether it can be used for future processing or should be discarded. We divided fingerprints into groups based on the quality of their minutiae points, number of ridges, contrast, sinusoidal similarity and ridge thickness. We successfully evaluated fingerprints and grouped them similarly to the grouping in the NIST SD27 dataset. The algorithm's results allowed us to draw conclusions about graded fingerprints' quality and rate their usability.

Abstrakt

Daktyloskopické stopy jsou jedním z klíčových aspektů biometrické identifikace. Reprezentují způsob, jakým lidé mohou ověřit svou identitu a být autorizováni. Nicméně je důležité ohodnotit, jestli daný otisk je validní a jestli dokáže poskytnout dostatek informací na to, aby bylo možné určit, jestli je použitelný anebo ne. Tato analýza rysů otisku dokáže specifikovat, jak moc přínosné pro nás je se daným otiskem zabývat. Navrhli jsme proces, který hodnotí otisky prstů na základě kontextuálních a statistických výsledků. Ne vždy je otisk ve stavu, kde všechny jeho prvky jsou viditelné a je nutné odstranit rušivé elementy před samotným ohodnocením, které by mohli negativně ovlivnit výsledky algoritmů pro hodnocení kvality. Tyto algoritmy rozpoznávají, jakou kvalitu má otisk prstu a jestli bude použit na další proces anebo zahozen. Rozdělili jsme otisky do skupin pomocí jejich kvality, která se hodnotila na základě počtu markantních bodů, počtu papilárních linií, kontrastu, sinusové podobnosti a tloušťky papilárních linií. Úspěšně jsme ohodnotili otisky a rozdělili je podobně jako v NIST SD27 databázi. Z výsledků těchto algoritmů jsme schopni ohodnotit otisky na základě jejich kvality, a tak vyvodit jejich použitelnost.

Keywords

image processing, fingerprints, quality analysis, dactyloscopic traces, sinusoidal similarity, ridge thickness, contrast, minutiae points, fingerprint enhancement, fingerprint segmentation, contrast difference, SD27, number of papillary ridges, LatFigGra

Klíčová slova

zpracování obrazu, otisky prstů, analýza kvality, daktyloskopické stopy, sinusová podobnost, tloušťka papilárních linií, kontrast, markantní body, vylepšení otisků prstů, segmentace otisků prstů, SD27, počet papilární linií, LatFigGra

Reference

SLOUP, Ondřej. *Algorithmic Evaluation of the Quality of Dactyloscopic Traces.* Brno, 2022. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor prof. Ing., Dipl.-Ing. Martin Drahanský, Ph.D.

Rozšířený abstrakt

Daktyloskopické stopy jsou jedním z klíčových aspektů biometrické identifikace. Reprezentují způsob, jakým lidé mohou ověřit svou identitu a být autorizováni. Nicméně je důležité ohodnotit, jestli daný otisk je validní a jestli dokáže poskytnout dostatek informací na to, aby bylo možné určit, jestli je použitelný nebo ne. Tato analýza rysů otisku dokáže specifikovat, jak moc přínosné pro nás je se daným otiskem zabývat. Na základě tohoto rozhodnutí můžeme ušetřit výpočetní výkon nutný pro další analýzu, paměť, kde by otisk byl jinak uložen a další prostředky. Díky těmto algoritmům je možné určit, zda je nutné skenování otisku opakovat a tím signalizovat požadavek na lepší otisk prstu.

Popsali jsme jednotlivé druhy otisků a jejich vlastnosti a snažíme se přijít na hlavní rysy, které jsou důležité při práci s nimi a jakým způsobem je možné je využít. Jelikož není známo na co otisk bude použit, nemůžeme přesně definovat perfektní jednu definici, která bude fungovat pro všechny otisky a pro všechny případy použití. Existuje řada procesů, které používají různé vlastnosti otisků pro identifikaci. Navíc otisky nemusí být použité pouze pro identifikaci, ale i pro jiné záměry, které mohou být založeny na úplně jiných rysech, než bychom u identifikace předpokládali.

Navrhli jsme proces, který hodnotí otisky prstů na základě získaných kontextuálních a statistických informací, vypočítaných z jednotlivých skenů, či fotek otisků, či daktyloskopických karet, bez ohledu na jeho typ.

Abychom dosáhli lepších hodnot aplikovali jsme několik filtrů a algoritmických postupů, které ještě před analýzou jednotlivé rysy otisku zvýrazní a vyextrahují pouze pro nás důležité prvky. Ne vždy je otisk ve stavu, kde všechny jeho prvky jsou viditelné a je nutné odstranit rušivé elementy před samotným ohodnocením, které by mohly negativně ovlivnit výsledky algoritmů pro hodnocení kvality. I přestože algoritmus dokáže pracovat se všemi typy otisků, v této práci je kladen důraz na extrahování informací z latentních otisků, jenž jsou známé svou podprůměrnou kvalitou a je složité je nalézt na fotce, natož je ohodnotit. Tato práce čerpá z řady předchozích výzkumů, které se zabývají segmentací a vylepšováním otisků a které pomáhají lokalizovat otisk na skenu a odstranit veškeré neduhy, které by mohly ovlivnit naše algoritmy pro analýzu kvality.

Po odstranění všech nepříznivých elementů jsou použity všechny algoritmy pro ohodnocení kvality otisků, které dle svých definic rozpoznají a vypočítají výslednou kvalitu. Naším řešením je navrhnout více algoritmů, které hodnotí specifické vlastnosti z daných otisků podle svých vlastních definic. Tyto definice byly vytvořeny na základě předchozích výzkumů, či pozorováním výsledků z vypočtených datasetů. V navrhnutém řešení jednotlivé pod-algoritmy určují kvalitu na základě počtu markantních bodů, počtu papilárních linií, kontrastu, sinusové podobnosti a tloušťky papilárních linií. Všechny algoritmy jsou buď nově navržené nebo jsou již vylepšenou verzí publikovaných algoritmů. Některé pod-algoritmy obsahují definice, které blíže specifikují a vysvětlují, jaké vlastnosti otisk splňuje, aby jeho ohodnocení bylo více názorné. Tato práce navrhuje dva nové kontrastní algoritmy. První ohodnocuje podíl mezi průměrem světelností a tmavostí bodů otisku a druhý počítá chybu z kvadratického průměru otisku oproti jeho binární masce.

Úspěšně jsme otisky ohodnotili a rozdělili je podobně jako v NIST SD27 databázi. Bohužel, algoritmy pro výpočet sinusové podobnosti a tloušťky papilárních linií dosáhli výsledků, které neodpovídaly rozdělením do skupin podle tohoto datasetu. Nicméně, podle definice algoritmů a samotných výsledků jsou otisky správně ohodnocené. Pravděpodobně je to kvůli tomu, že pracují s jinými vlastnostmi otisku, které nebyly v rámci tohoto datasetu zkoumané a nehrají roli v NIST klasifikaci. Z výsledků těchto algoritmů jsme schopni ohodnotit otisky na základě jejich kvality, a tak vyvodit jejich použitelnost.

Algorithmic Evaluation of the Quality of Dactyloscopic Traces

Declaration

I hereby declare that this Bachelor's thesis was prepared as an original work by the author under the supervision of prof. Ing., Dipl.-Ing. Drahanský Martin, Ph.D. The dataset, fingerprint images SFinGe for illustration and supplementary information was provided by Ing. Ondřej Kanich, Ph.D. I have listed all the literary sources, publications, and other sources used during the preparation of this thesis.

.....
Ondřej Sloup
May 9, 2022

Acknowledgements

I want to thank a few people who helped me with the thesis. First, I would like to thank Viktor Rucký, who helped me interpret statistical information from the proposed algorithm to demonstrate my results. My sister, Tereza Verdoliva, LL.B., proofread the whole thesis. My friend, BSc. Justin Falkenstein, who then read the thesis from his technical point of view and gave me a few tips, which I implemented. Bc. Lukáš Piwowarski, who give me several suggestions while implementing the algorithm and writing the thesis.
I'm also greateful to MetaCentrum who provided me with computational resources of the project "e-Infrastruktura CZ" (e-INFRA CZ LM2018140) supported by the Ministry of Education, Youth and Sports of the Czech Republic.

Contents

Chapter 1

Biometrics Recognition

Biometrics recognition is a process of using a person's physiological and behavioural features for identification processes. Nowadays, it is standard for mobile and other devices to use this technology to identify their users and prove their identity. This technology is one of a few ways to identify a person reliably as it has several unique features. Identification processes help the general public be safer and help secure their information with biometric characteristics. These characteristics can validate only native users as they are the only ones who can provide them. [16, 27, 29]

Biometric features add another security layer that protects the identification process. It cannot be cheated without significant effort as the biometric elements of a user cannot be easily transferred, destroyed, or forgotten [17]. This property is a great way to combat identity thefts and other malicious activities. That is why biometric elements are used in smartphones, passports, national identification cards and other documents to strengthen their security features [17, 27].

There are many advantages and drawbacks of this technology. We can acquire a lot of information about a person, such as their fingerprint, facial features, way of walking, and others, and reliably identify them based on that data. Nonetheless, the technology also has many variables that can make the validation and identification process rather difficult. Biometrics recognition heavily depends on the technologies and standards used while sampling the user. The technologies could make an error when an older dataset is not updated or criteria are not met while scanning the user. This can create an issue where users will not be able to identify themselves, or a possible imposter can be falsely recognised. [16] There are more requirements on datasets as standards and technology advance rapidly. If the datasets are not updated or corrupted by misuse or data forgery, it can create more obstacles while validating a specific person. [17]

There is one significant drawback which is also the greatest advantage. Most people have biometric characteristics by which they can be identified; not everyone can since some people have health issues that partially or fully prevent them from being identified [16]. Therefore, we cannot use this technology as the only means of identification, and we need to assert how much of their data can be used as it is an excellent way for identification. Once the person's biometric features are damaged, for example, with an illness, we cannot create a replacement as it would directly cancel out almost all benefits of this identification. The identification is not possible if a person cannot submit their biometric data in good quality.

On the other hand, technology leads to one of the easiest and most reliable ways of identification that we currently have, and it provides us with a way to be recognised by our features if we can use them. [16]

We need a way to algorithmically evaluate given datasets and decide if they need to be updated in the database, grade if they are sufficient for our use case or if we can use them to some extent or overall say if they are valid. The datasets do not have to have all the necessary properties which we are nowadays required to identify a person beyond a reasonable doubt, but they still can be used to some extent.

Fingerprint identification is commonly used on personal devices, and each device has software that evaluates if the scan of the fingerprint is valid. It informs the user if they need to retry scanning or if the scanning is sufficient. This algorithm evaluates the fingerprint quality to ensure that it works sufficiently as part of the scanning process.

Chapter 2

Fingerprint

Fingerprints (dactyloscopic traces) are among the few biometric elements used in biometrics for authorisation and authentication. It is an imprint left by a person by which they can be identified. Still, to be able to, the fingerprint needs to be already in a database with the user's identity linked to it, and the new fingerprint needs to be of good quality to match with the one in the database beyond a reasonable doubt. [16, 29]

Papillary ridges are the main component of the fingerprint. They are located on the surface of fingers, toes, the palm, or the foot's sole. They are the surface ridges of the epidermis (outer layer of the skin) [16, 18], which gives fingerprints their characteristic look. From those ridges, it is possible to get all the information needed for identification and to decide if they are of adequate quality.

Each fingerprint has unique features which are crucial to the identification process as they all follow these rules, which were adopted from [17]:

- There are not any two people in the world who would have an identical structure of ridges – completely same fingerprint features, which would make them undifferentiable
- Ridges are immutable and perennial during a person's life, they can be damaged, but they will not change
- If the epidermis is not damaged, ridges can regenerate by skin growth, and they will not be different from the previous ridges. However, if the epidermis was damaged, the skin would not restore them, and other patterns of ridges would not grow
- The scans are not always precise, but the identification procedure can have set tolerance limits and allow systematic classification for fingerprints.

Fingerprints can be divided into categories based on their appearance, the way how they are preserved, and their visibility. Those categories are patent, plastic (impression), and latent [17, 22].

Patent fingerprints are visible fingerprints left on the surface of some object. Various liquids or oils can create those fingerprints, such as blood, dirt, ink and other similar substances. The patent fingerprint left on an object is photographed for subsequent identification and storage. [16]

Plastic fingerprints are also visible and usually left in pliable substances or forms, such as clay, wax or wet paint. It preserves the fingerprint as a three-dimensional imprint, and then it is photographed under direct light that enhances the contrast of the ridges and valleys. [37]

Latent fingerprints (i.e. hidden fingerprints) are fingerprints that cannot be seen with a naked eye. They are created with sweat/oil and other particles present on a finger. Those fingerprints can be treated with some techniques which can improve their visibility, and afterwards, they can be photographed and catalogued. [22, 29]

Exemplar fingerprints are fingerprints that were purposefully scanned and catalogued. This fingerprint category describes the deliberate collection of a fingerprint more than a particular type. An exemplar fingerprint can also be a patent fingerprint if taken voluntarily. This category is mentioned just for completion. They are usually of higher quality and are used for comparison to recognise their owner. Commonly, those fingerprints are taken when institutions need fingerprints linked to passports or national identification cards if there is no direct scan of the finger. [16]

Fingerprint as a means of recognition has few characteristic properties that we can analyse. The features of a fingerprint can be divided into three levels as was proposed in [29]:

- Level 1: Global fingerprints characteristics – we look at the fingerprint as a whole and evaluate it based on its apparent features such as patterns and ridge flow. We can also look at the fingerprint image from a statistical standpoint and grade it by those metrics.
- Level 2: Individual ridge features – defined by smaller fingerprint regions. Those regions can be small loops or ripples on the fingerprint image. The minutiae points, which are vital elements in fingerprint recognition, are detected on this level. They will be discussed in the following Section 2.3.
- Level 3: Finer ridge details, pores and ridge contours. The majority of those features are lost if the quality of the fingerprint is inadequate. It is rare to find pores on latent fingerprint images in a usable state yet to identify a person by them.

Each fingerprint has a unique combination of properties by which we can differentiate them. Those properties are examined on each level and evaluated. It heavily depends on the state of the fingerprint to gather its features. [29]

2.1 Ridges and Valleys

Papillary ridges and valleys are crucial elements for fingerprints in biometric recognition. They are protrusions on the finger created in the dermis layer (skin layer under the epidermis) and propagated through the bottom part of the epidermis as actual ridges [16, 18, 29]. Those individual propagated protrusions on the finger create ripples on the skin. These ripples form ridges, which are the taller portion of the papillary lines, and between ridges, valleys are created. The shape that is recognisable immediately is their curving and circles, which form clusters. By those clusters and other properties and features, we group them and divide them into their categories. [29]

The most prominent property is the pattern and its type. We can use the pattern to try to differentiate fingerprints with an eye and come to a reasonable conclusion if the fingerprints do not have too similar styles, but there are different approaches to comparing fingerprints which significantly outperform this type of matching. Additionally, it is nowhere near computer matching, which has reproducible results and is considerably faster. [29]

The structure of fingerprints or those curvings can be divided into a few categories – delta, core and loop or whorl (see Figure 2.1) [18, 29]. These features are not how the

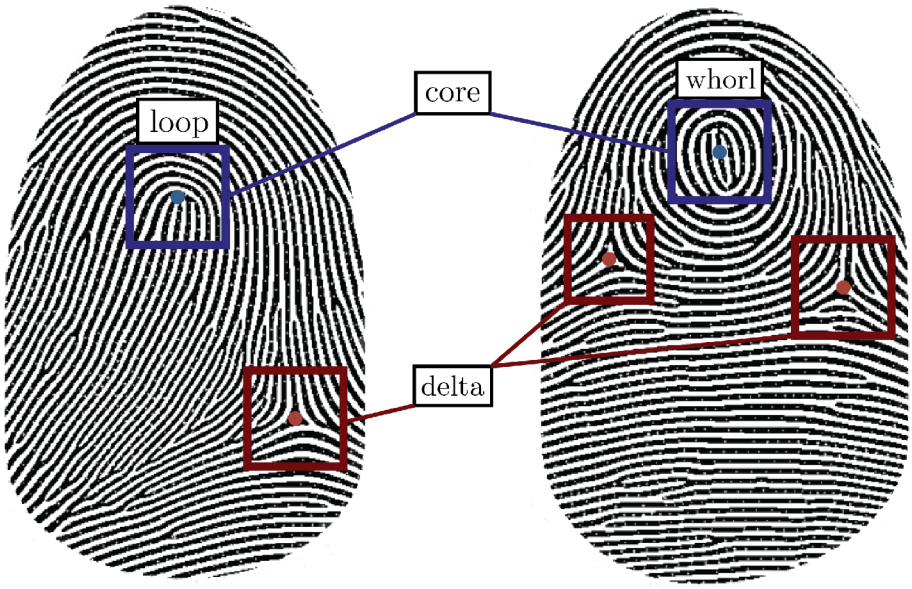


Figure 2.1: Description of regions found on fingerprints. Cores are displayed with blue boxes and deltas with red boxed. There is also an example of whorl on the right and loop on the left. Fingerprints were generated with the SFinGe tool[1].

fingerprint is structured overall but rather a description of its distinctive regions. These regions are used to overlay the fingerprint while running the matching process and help us identify their unique sections or discard fingerprints before an expensive matching algorithm is executed upon it. [29]

The core is as defined by [25, Henry, E. R. Sir, 1900] "*the north most point of the innermost ridge line*" [29, Chapter 3, page 98], which corresponds to the centre of the northmost loop type in the fingerprint. The delta is a region on a fingerprint where two ridges run side-by-side and then diverge to form a triangle. It is used with core to overlay fingerprints, an easily identifiable area. The loop or whorl describes the region which creates the pattern corresponding to its name. It can be distinctive enough that the fingerprint will be characterised only with this pattern and classified by the fingerprint style with the same name. [29]

Using these categories, we classify each fingerprint by the pattern of their clusters – arch, loop or whorl. Each class has sub-types to describe distinct fingerprints more accurately. [18, 29]

Fingerprint images consist of the valleys displayed with brighter colours and ridges shown with dark colours.

2.2 Categories of Fingerprint

Furthermore, by the defined structure of the fingerprint, we can categorise them into groups (see Figure 2.2). Those groups differ by the number of deltas and how the ridges are shaped. [17]

The Arch pattern's significant feature is that it does not contain a delta region because it does not contain any whorl or loop; therefore, there cannot be a delta by definition [17, 29].

[1] http://biolab.csr.unibo.it/sfinge.html

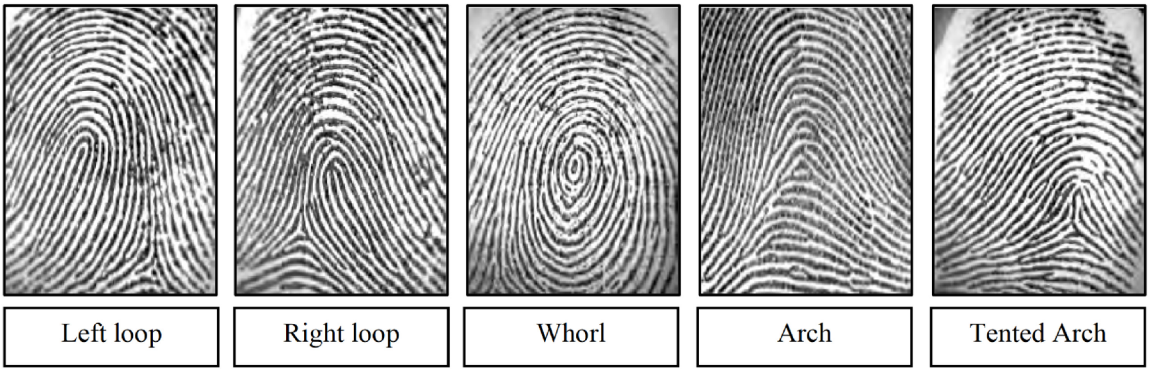


Figure 2.2: Overview of fingerprints' main classes. Image acquired from [29].

The core of the Arch type fingerprint is more challenging to identify; there is no loop, and therefore the core is selected as the point of maximum ridge line curvature, which is not always precise. The Arch pattern can be divided into two subcategories – Plain and Tented Arch. These two categories differ in the curvature of the ridge, which shapes as a "tent". [18, 29]

The Loop patterns are easier to identify. Unlike Arch or Whorl, they have precisely one delta and one core [17, 29]. This information is significant as it helps us determine fingerprint classes more effectively. The sub-type of this pattern depends on which hand the fingerprint is located. Ridge count has also been used during the classification of the Loop pattern. The ridges are counted from the core to delta with a drawn intersected with a line segment [17]. There are two subtypes – left and right loop. These subtypes correspond to the hand from which the fingerprint was taken, estimated by the loop opening and its direction. [29]

Every fingerprint with more than one delta is classified as a Whorl pattern [17, 29]. The Whorl pattern is the most difficult to divide into a specific sub-group because the identification involves a two-step process where the sub-group is identified first, then and subsequently the used tracking of the fingerprint's ridge flow. This process uniquely groups the fingerprint into its category. Three types of tracing are used and differ in the number of ridges intersecting with the line segment from the first delta to the second one. Additionally, the subtypes of the Whorl pattern are Plain Whorl and Twin Loop. They differ by the completion of a loop between fingerprints deltas and by the number of deltas found on the fingerprint, as each can create an additional loop. [29]

It is essential to distinguish the core and delta as those are key features of the fingerprint pattern and can tell a lot of information used for classification. Fingerprint categories provide us with a way to differentiate fingerprints with a naked eye and draw conclusions [29]. Some of the classes are very rare, and they can help us with identification processes. To simplify the process of dividing each fingerprint into each category, we can say that if a fingerprint has no delta, it is classified as an Arch pattern. If it has precisely one delta and one core, it is classified as a Loop pattern. If it contains more than one delta, it is a Whorl pattern. These classifications are then used for pattern matching in fingerprint matching algorithms to be more effective. [29]

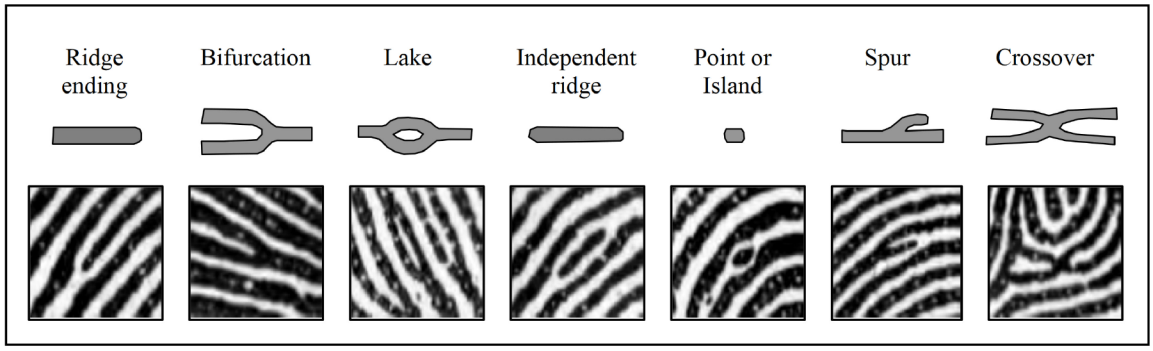


Figure 2.3: Overview of minutiae types. Image acquired from [29].

2.3 Identification Means

There are several approaches to how fingerprints can be used as a means of identification. Usually, minutiae points are used as it is the most proven process to identify people's identity, but different techniques can also be used [17, 18, 29]. This is especially true when a fingerprint is of deficient quality, and the minutiae point matching cannot be used. Fingerprints' shape, local orientation and ridge frequency and other approaches are valid for matching two fingerprints, but the correlation does not have to be adequate to match those fingerprints successfully [29].

Minutiae Points

Minutiae points are irregular shapes of ridges that are formed on fingerprints. Those regions are helpful for recognition as their position and distinctive shape create a special and unique element by which the fingerprint is identified (see Figure 2.3) [18, 29]. For example, minutiae points can be a ridge that ends abruptly, a bifurcated ridge or two ridges in a crossover, and many other patterns. The minutiae points are an essential fingerprint element in pattern matching, as there can be over one hundred minutiae points on one fingerprint. Additionally, only about twelve unique points are enough to distinguish between two fingerprints [29]. This number is referred to as twelve minutiae guidelines that most institutions recognise as a successful match. Minutiae points are the most used component in fingerprint matching because they are one of the few very stable features and can be easily documented and saved for future needs. [29]

The minutiae points are stored in the database as a triplet $min = (x, y, \alpha)$, where x and y represent its coordination, and α specifies the orientation of this point in interval from $\langle 0, 2\pi \rangle$ [29]. Other approaches are also used to store minutiae points, but this is the most common [10].

Multiple models of minutiae points can be taken into consideration while processing them. The models differ in how many minutiae types can be recognised and how they are processed and stored [29]. Using a wide variety of minutiae point types is not recommended due to unreliable computer matching and possible errors. The most common standard is the American National Standard (ANSI/NIST-ITL 1-2007) which is based on only four types of minutiae points – ridge endings, bifurcations, compound, undetermined [16, 18, 29]. Other standards are trying to build on this model by adding some conventions and improving compatibility with different formats. [29]

Figure 2.4: Showcase of visible pores on the fingerprint. The image was generated with the SFinGe tool[2].

Pores

Pores are the most delicate details which can be found on the fingerprint. They are miniature points inside the ridges, which form small circles displayed in white colour (see Figure 2.4). About 9 to 18 pores per one centimetre could be on a fingerprint, and 20 to 40 pores are sufficient to identify a person [29]. This is great because even if we have just a partial fingerprint, we can identify a person based on just this data. Nevertheless, pores cannot effectively be used on latent fingerprints as they are usually incomplete or corrupted. On top of that, the reconstruction of fingerprint images will most likely smoothen them out. Therefore to get an advantage with this feature of fingerprints, we need a high-quality scanned fingerprint as those fine details can be easily lost. NIST already mentioned the first standard in which pores appeared in CDEFFS (2008), which describes all the required scanning practices. [29]

Ridges Frequency

Ridges frequency tells us how many ridges or the density of the ridges are in a fingerprint. This value significantly differs for every fingerprint and also for their regions [29]. Ridges frequency needs to be considered while analysing and scanning them, as they should not be processed with the same spacing. Fingerprint enhancement methods usually use predetermined values for 500 PPI images, about 7.71 pixels [34]. Still, in practice, the distance between ridges on a fingerprint image varies in smaller regions throughout the entire picture. Therefore the ridge frequency should be processed and computed as it can change significantly [35]. Previously, it was thought it was unnecessary, but nowadays, we have ways and reasons for calculating this property [34]. There are different techniques used to determine and analyse this property, and the most common is to use the means of the Fourier Transform, but other more straightforward methods exist. [29]

The ridge frequency also significantly differs for part of the fingerprints [34]. The densest ridges will be around the core as the lines usually cluster there [16]. Additionally, the delta region will also have more ridges than usual parts of the fingerprints. This naturally occurs as the loops of each ridge are around those points.

[2] http://biolab.csr.unibo.it/sfinge.html

2.4 Fingerprint Acquisition

The fingerprint image acquisition is a complex process with multiple steps, which all need to be done perfectly to preserve the best quality of the fingerprint image. There are multiple ways how to obtain the fingerprint and digitalise it. The traditional way still commonly used is to use ink and dactyloscopy cards [16, 29]. The inked finger is rolled on the card, which is subsequently scanned. Nowadays, various scanners have replaced this method, delivering better quality prints while simplifying the process [16]. The main parameter for a fingerprint scanner is the resolution and the size of the scanned fingerprint area. The scanner needs to capture the best image to preserve all features given by the print. If the fingerprint capture is performed well, we can analyse all levels of details and use even pores to identify and match a person to a print. [16, 18]

Technologies for Fingerprint Acquirement

Generally, there are two ways for fingerprint acquisition. Either the finger is scanned with a scanner, which is the standard way to get the fingerprint image or with dactyloscopy cards, which still nowadays prevail, where the user applies ink on their finger and rolls it onto a card. However, if the fingerprint is plastic, patent or latent, they are usually photographed without using any of the mentioned sensors. [16, 22]

The scanners are categorised by their type of scanning and the technologies used. All of them have different features and disadvantages. A few exciting technologies described in [16, 18, 29] were adopted and outlined here:

- optical – a simple optical mechanism that reflects light on fingerprint and into the CCD / CMOS camera. The dirt which can accumulate on the sensor can significantly reduce the fingerprint quality. Also, some researchers pointed out that the skin colour of the fingerprint can influence the quality fingerprint image as the reflection of the light is changed.

- capacitive – a sensor composed of tiny conductive metal plates which send an impulse when the ridge is detected. The resulting image will combine places where the ridge touched the plate and when the valley did not. This sensor has numerous disadvantages; the biggest one is its size. The reduced area of the scanner does not have to catch the core of the fingerprint or other significant areas, which would make the fingerprint hard to evaluate. Another drawback of this technology is the coating by which the sensor is protected. Finding the right balance between the thickness is needed since the ridges would not be too visible if the coating was too thick, and if it was too thin, the sensor could get damaged.

- ultrasonic – using an ultrasonic transducer with a receiver that rotates and scans the fingerprint. It penetrates the skin's surface with all foreign elements included, such as dirt or sweat. It is one of the most accurate sensors; however, they are the most expensive.

- pressure – pressure sensors have three layers: charged, barrier and ground. If a ridge presses hard enough on the most surface charged layer, it will create an electromagnetic impulse with the furthermost layer. The problem with this sensor is that it usually does not differentiate well between valleys and ridges.

- microelectromechanical systems (MEMS) – a new technology that uses either piezo-resistive gauges or microheaters that measure the fingerprint ridges based on the heat resistance. This sensor does not handle well stray capacitance of the human finger.

Any scanner has specific issues while scanning the finger, significantly reducing the print quality and causing multiple problems. Mentioned sensors above, with their disadvantages, create new issues with fingerprint acquisition. There is a need to eliminate those disadvantages and make the most of every scanned fingerprint. Various problems arise with imperfect fingerprints, such as the difficulty to process the image and therefore increased failure to acquire (FTA) / failure to enrol (FTE) rates, rejection of a correct user – false rejection rate (FRR) / false non-match rate (FNMR), acceptance of an imposer – false accept rate (FAR) / false match rate (FMR), can make verification and identification difficult or nearly impossible. [16, 18]

Dactyloscopy Cards

The dactyloscopy cards are still, to this day, commonly used [29]. The principle is relatively simple; a person will soak their finger into ink and press it onto a dactyloscopy card, subsequently photographed or scanned. The dactyloscopy card is divided into two parts. The first half accounts for the all fingerprints that are "rolled" on the card, and the second part marks the right and left hands with the detailed print of thumbs. This method is used for manual fingerprint recognition or comparison to a fingerprint from the crime scene. [16]

2.5 Damaging Fingerprint Factors

Not only the fingerprint scanning can negatively influence the fingerprint image. There are numerous ways how a finger can be initially damaged. The mentioned features would not be for those reasons visible. Those damages can be due to skin diseases that compromise the fingerprint or an environment where the fingerprint was taken or left. [16, 18]

External factors such as dirt or other non-fingerprint elements can potentially compromise the image and destroy the feature by which the identification was possible [16, 18]. This is commonly found on latent fingerprints, where the fingerprint itself is not visible at all, and the user left them unknowingly on some item. Plastic or patent fingerprints benefit from not using the person's sweat to mark themselves on an object but using various materials or liquids. However, those fingerprints have other potential problems that may affect the identification result. Usually, exemplar fingerprints which are deliberately taken from individuals are the best quality since the conditions where the fingerprint was taken are controlled and improved if needed. Also, retaking the fingerprint is possible, which is not possible, for example, with latent fingerprints.

Skin Diseases

Skin diseases present a problem with fingerprint acquisition. If the user has damaged fingers, and scanning or submitting of fingerprints is impossible, we cannot identify them by this biometric recognition. Even though the number of people who have this condition severe enough is not high, it is still a problem that should be addressed. [19]

The sicknesses can change the structure of ridges, skin colour or both. The skin colour does not yield significant issues as the fingerprint can still be scanned with sensors which do not rely on optical scanning, or a dactyloscopy card can be used. However, if the ridge

(a) Physically damaged fingerprint

(b) Wet fingerprint

(c) Dry fingerprint

Figure 2.5: Showcase of various fingerprint damages. Images were generated with the SFinGe tool[3].

structure changes and the minutiae points or pores are no longer readable, the user cannot be identified with their fingerprints, however scanned. [16, 18]

There are other means of biometric recognition which could solve this problem, such as multimodal biometric system [16], but not every time it is possible to implement it.

Physical Damage

Similarly to skin diseases, a person can damage their fingers and devalue their fingerprints (see Figure 2.5a). This can happen either intentionally or accidentally, but the problem remains. If the damage did not influence the skin's epidermis, the same fingerprint would regrow again [16], but for that time, the fingerprint is damaged. However, if the injury is more serious, it is like a skin illness where the user cannot be identified by the print anymore.

Humidity

The humidity of the fingerprint is the problem that influences almost all recognition systems. The moisture refers to the wetness or dryness of sweat which changes the output of the scanned fingerprint [16]. This can impact skin resistance, where the sensor will have different readings as the resistance changes based on the amount of sweat found on the fingerprint. Less affected are pressure sensors and others, which do not rely on optical readings [16].

The dry fingerprint will usually not be as visible when processing latent fingerprints as there was not as much sweat to leave a significant mark (see Figure 2.5c). The wet fingerprint has a higher spread of sweat, which makes the fingerprint more visible (see Figure 2.5b); however, some features, especially ridges, can be blended. We strive to find fingerprints somewhere in between – not as wet and not as dry. The patent and plastic fingerprints are not influenced significantly as the fingerprint does not rely on the user's sweat. However, the patent fingerprint has a similar issue with the substances used for fingerprint creation. This paragraph was adapted from. [16, 18]

[3] http://biolab.csr.unibo.it/sfinge.html

Chapter 3

Quality of Fingerprint Images

As was said in [16, page 53, section 3.1] "*The exact definition of the quality of a fingerprint is difficult and nearly impossible.*", but usually, we are taking into account the fingerprint's matching potential [29] to another fingerprint in our database. We can measure certain features of fingerprints and compare them to some average which will apply to most of the fingerprints. One of the most obvious ways to grade fingerprints is to count the number of minutiae points commonly used for identification and grade the fingerprint on that value. However, a person does not have to be identified by only the minutiae points; they can be evaluated by other already mentioned features such as pores or the shape of the fingerprint. Nonetheless, those features have problems on their own, and not all types of fingerprints can use those features as they do not have to be retrievable from the print. Therefore when analysing a fingerprint, there cannot be a single definition by which the fingerprint could be graded, but rather several ways to measure each feature of a fingerprint [16]. Those algorithms can either analyse the fingerprint for us or mark the fingerprint image to indicate that it holds some value or can be discarded as there is no helpful information. [29]

Fingerprints are commonly graded on a global and local level. The local level is usually preferred because it is more descriptive and allows us to grade parts of the fingerprint rather than the whole image if the image is in such a condition. The most significant global features are the size of the fingerprint mask in contrast to its background, ridge patterns, etc. The local level features are taken from pixels of the actual fingerprint, such as contrast, variance, ridge thickness, and others. It is not that simple to extract those features, and there are various algorithms to get these values. [16, 29]

Grading of fingerprint images can use statistical values from the image, such as contrast, variance, histograms etc. [16]. This can give us an idea of whether the image is usable and how much value it can bring to us, and if we use it for fingerprint matching. Other methods are trying to utilise the fingerprint's contextual features, such as its minutiae points, ridge orientation, frequency, etc. [29]. The most deductive elements of the fingerprint are used to estimate their quality.

Results from those methods can either be displayed with a quality map, an image that shows the higher quality of fingerprint with brighter colour blocks or with a number compared with a defined one and the deviation is graded.

3.1 Quality of Minutiae Points

The minutiae points are the main point of interest while processing fingerprints nowadays. They are the critical component of fingerprint matching, and that is why they are significant when estimating the quality of fingerprints. The number of minutiae points is a crucial value of fingerprint quality since most matching algorithms work with them [16, 18, 29]. No matter how many ridges or how blurry the images are, if we can find enough minutiae points that are correctly determined, we can use the fingerprint image to identify a person. Generally, a match of twelve minutiae points is considered a match beyond a reasonable doubt. Nonetheless, fingerprints with poor quality, especially latent fingerprints, have the potential to generate more false minutiae, which significantly plummets the probability of a correct match, and we should have some overhead while grading by the minutiae points. Even missing minutiae point is better since we can correctly estimate valid information than assume the correctness of an invalid one. [29]

By all that, if we are not sure about a found minutiae point, we should not include it as a result, and we need a way to check our results and combat this or match minutiae points with a probable value of how sure we are about them. We also should enhance the image before the minutiae points algorithms are run. All algorithms do not have to be influenced as they present their enhancement, but generally, it is good to analyse the best image we can.

3.2 Image Contrast and Histograms

Image contrast is one way to describe the local pixel differences between ridges and valleys of the fingerprint. Since the contrast is used as one of the most simple steps while enhancing the fingerprint image, its value can tell us how efficient this step will be if the image contains noise and how much more we can improve the fingerprint with the histogram equalisation [16, 27]. On fingerprint images, we usually strive to look for the ratio between the intensity of the valley where we expect lower intensities and the ridge where we hope for the most distinct pixels. Then they are divided together to get the difference. While processing fingerprints, two algorithms are generally used – Michelson contrast and Weber contrast [16].

Michelson contrast is given by the equation:

$$M = \frac{\sum_{i=1}^{N} \frac{(R_{max} - R_{min})}{(R_{max} + R_{min})}}{N} * 100 \tag{3.1}$$

where the R_{max} is the maximal intensity of the line region of the image and the R_{min} is the region with the minimal intensity. The mean is calculated from those intensities multiplied by 100 to get Michelson's contrast in percentage. Weber's contrast is just normalised difference between the most significant pixel intensities of ridges and valleys. [16]

Histograms represent the number of pixels and their colour in the images. In a nutshell, the contrast makes these numbers more distinct. Furthermore, this can be analysed and used for grading since it can give us statistical information about fingerprints. One of them is the histogram's mean value, which should be in the ideal case in the middle of the histogram with a significant peak on the right side symbolising the white colours (255) and another on the left side representing the black colour (0) [16]. Normalisation can also help find the correct mean value and improve the statistical values.

3.3 Number of Ridges

The number of ridges correlates with the number of minutiae points, and also it can significantly contribute to the reliability of fingerprint analysis [16, 29]. The ridge count is determined with two selected points on the fingerprint image. An imaginary line is drawn between these points and counted intersected ridges. Usually, these two points are core and delta, but it is not required [16, 29]. Let us run lines horizontally and vertically through the fingerprint image, technically from edge to edge. We can estimate the number of lines in each row and column, and from that number, we deduce the stationary point of the fingerprint where most of the ridges are. Theoretically, it should be the core of the fingerprint since we should find the maximum number of ridges near the actual core [16]. Still, this simple algorithm is not reliable in this estimation.

With the higher amount of ridges, we can estimate that the sensor used to capture a particular fingerprint image allowed us to get more details about the fingerprint. We can put the actual scanned area or segmented fingerprint area in contrast to the ridge count. If the ridges have good density, we can grade the fingerprint quality as a better print. [16]

There are problems with ridge estimation in noisy or blurry parts of the image because the damage can hide or merge a ridge right at the point where the imaginary line should intersect them; the ridges at that place are not detected. Nonetheless, those damages should be only present in latent fingerprints and should not be counted as we cannot get any helpful information from those areas. [16, 18]

3.4 Shape of Papillary Line Crosscut

The shape of ridges on fingerprint images is interesting to analyse. Suppose we draw a line perpendicular to the ridges from the fingerprint's centre or the fingerprint's core to the end of the image and take all the values of grayscale pixels from the drawn line. We should get a curve that would display the ridges and valleys, where valleys are shown as 255 (the maximum value – white) and the ridges as zeros (with the lowest value – black). We inverted the curve to get the ridges to act as the highest value and vice versa and then displayed the curve to get various information about the fingerprint itself and to analyse its properties. The steepness and the thickness of this line are helpful since they can give us information about how well the ridge is displayed on the image [15, 16]. The curve can also be compared to a sine function, and it can provide us with some interesting conclusions about the line quality, which we can apply to the whole image. [14, 15, 16]

Thickness

It is estimated that the average thickness of the ridge is around $0.33\,mm$ [16]. We can use this number to calculate the expected thickness of our measured ridges and compare if they are thinner or thicker. To make this possible, we need to draw a line on the fingerprint ridge at a specific height and count the length of the line by counting individual pixels on it. Secondly, we need to divide it by the image's PPI (Pixels Per Inch), which solely depends on the sensor which scanned the fingerprint. Then divide it by the given average thickness to get the deviation from standard fingerprint thickness. We can describe the complete process with the following formula:

$$Th_d = \left(\frac{2.54 * P_{len}}{S_{PPI} * 0.033} - 1 \right) * 100 \tag{3.2}$$

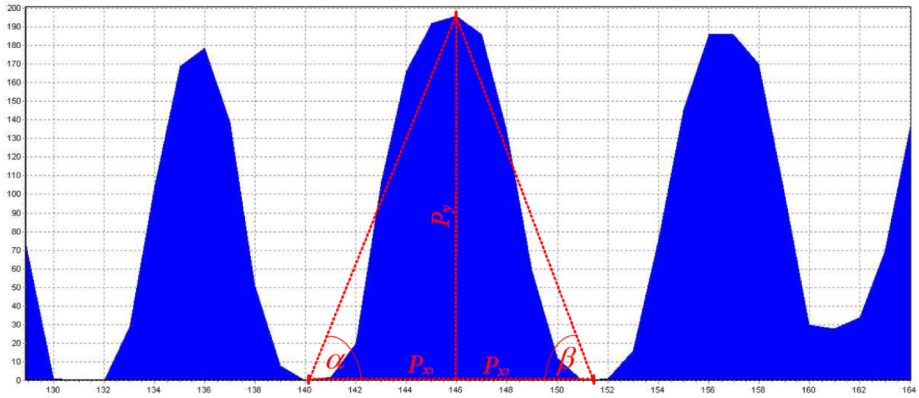


Figure 3.1: Displayed steepness of the ridge and all angles. Image acquired from [15].

where P_{len} is the number of pixels counted in a ridge, S_{PPI} is the given PPI by the sensor, and the 2.54 is for the conversion of inches to millimetres. [14, 15, 16]

If the deviation is higher than zero, we can say that the ridge is thicker than it should be and if otherwise, then thinner. Logically the more the fingerprint approaches zero, the more ideal the ridge is. [16]

Steepness

Another element that can be calculated from the crosscut is the steepness of the ridge. Ideally, the steepness should create a right-angled triangle between the ridge's local maximum and a local minimum. We can compute two angles which can bring us some gradable value. One is between the x-axis and the local maximum (α), and the second is symmetrical to the first one (β) on the other side of the ridge (see Figure 3.1). The steepness gives us information about how clean the ridge is and if there are smudges on the fingerprint image. Those ailments are displayed with a sharper angle, or the ridge can merge with others and will not be detected as a local maximum. Both found angles are then calculated and compared to the ideal case, which is 60 degrees. By the calculated deviation, we can measure the aspect of steepens and differentiate if it the higher or lower than it should be. If the divination is higher, the steepness is more significant than it should be and vice versa; if it is lower, the ridge is too horizontal. [14, 15, 16]

Sinusoidal Similarity

The acquired grayscale levels from fingerprint images should be similar to the sinusoidal curve in repeating intervals in the range $\langle \frac{-\pi}{2}, \frac{3\pi}{2} \rangle$. This similarity has been mentioned in the upcoming ISO standard (ISO/IEC SC37 N1954 WD 29794-4) but has not been thoroughly tested [16]. The sinusoidal amplitude consists of a high peak that symbolises the ridge, and the edges of the mentioned interval should correspond to a valley (see Figure 3.2). We examine the difference between the "perfect" sinusoidal ridge and our measured one, and we compute the area under each curve and count the deviation. This can be done with the following formula:

$$Sin_d = \left(\frac{\int_{\frac{-\pi}{2}}^{\frac{3\pi}{2}} f(x)\,dx}{\int_{\frac{-\pi}{2}}^{\frac{3\pi}{2}} \sin(x)\,dx} - 1 \right) * 100 \tag{3.3}$$

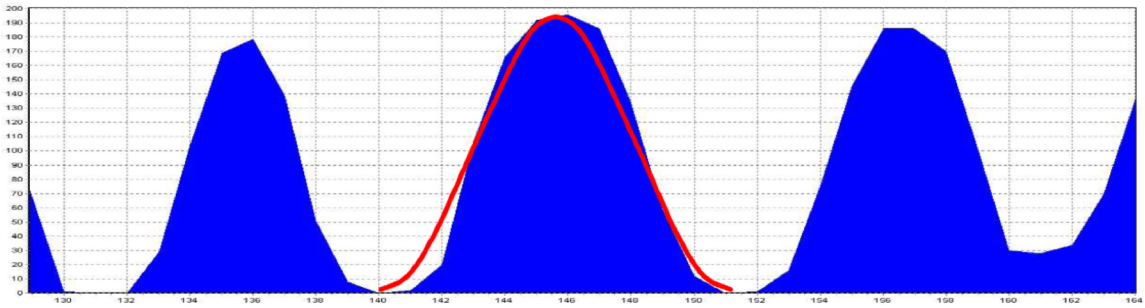


Figure 3.2: Showcase sinusoidal similarity between sinusoidal wave (red) and ridges (blue). Image acquired from [15].

where f(x) is the calculated ridge and the sin(x) is the ideal ridge made with the sinusoidal wave. [14, 15, 16]

As the local maximum of the ridge should be positioned to the local maximum of the sinusoidal, we can estimate the deviation. The deviation should be close to zero, meaning that the ridge is perfect. Still, if the deviation is higher than zero, the ridge is more elevated than the "perfect" sinusoidal and vice versa; if the sinusoidal is less than zero, the ridge is lower than the "perfect" sinusoidal. This operation assumes that the signal from the grayscale image was normalised into a signal in the interval from $\langle(0, 2\rangle)$ and to the sinusoidal signal was added one. [16]

Chapter 4

Fingerprint Enhancement

To correctly evaluate some fingerprint features and grade them accordingly, we need to enhance the fingerprint image and extract the necessary characteristics with noise or disruptive elements removed. For example, looking for minutiae points on latent fingerprints is a complex process; the algorithm used for the search could find non-existent minutiae points and extract them, which would cause misleading information. Error avoidance is one of the reasons why we need to try to eliminate the ailments before the grading. With those elements removed, we can even make our grading more reliable as random factors do not influence it.

The algorithms for improving the fingerprint images vary in their approaches, but we can simplify the general idea into the following steps as mentioned in [16, 27, 29]. First, it is needed to convert the fingerprint image to a grayscale image to simplify the picture and subsequent computations. We do not need colours when working with fingerprints since the only essential details are the ridges displayed with black and the valleys between with white colour. The grayscale will symbolise the transitions between them. Secondly, we need to remove the noise and external elements obscuring the fingerprint. Those elements can be anything significant on the background that makes ridges less readable or something on the fingerprint itself, like a skin illness that damaged the ridges or made them completely missing. We cannot rebuild those missing pieces, but we can mark them as unusable to not waste computation power and to avoid false positives [18]. The segmentation comes next as we need to separate the fingerprint from the background and estimate the ridge orientation field. The segmentation helps us to separate the ridges and binarise the image afterwards. We can apply other filters to make this process more effective or neural networks that can significantly help, but the core idea stays the same. [10, 16, 29]

4.1 Contrast

The contrast improves the visibility of details in the image, which could be lost otherwise. In fingerprint recognition, the essential factor is a contrast that enhances the difference between ridges and valleys [27]. The improvement is made by expanding the dynamic range and enhancing the variety of colours in transitions between ridges and valleys [16, 27]. There are two general approaches while computing the contrast. One is processing the image as a whole, while the second divides the image into separate blocks and processes them on their own. Finally, mean and variance are computed from each block or the overall image and then averaged. By the calculated value, the entire picture or blocks are enhanced. [29]

Contrast also has some drawbacks that need to be mentioned; one of them is the ability to enhance noise and make it more impactful on the fingerprint image [24, 27]. Nonetheless, it is still a powerful tool since we should be able to correctly identify the ridges themselves and extract them even with the noise amplified since the noise should not structurally destroy the ridges [27].

CLAHE

CLAHE (Contrast Limited Adaptive Histogram Equalisation) is a variant of Adaptive Histogram Equalisation, improving the image's contrast. It calculates the number of used colours on the image and proportionally divides them by the image colour spectrum interval [1, 27]. This method significantly improves the quality of the picture as a whole colour spectrum is used, and therefore more minor details are visible. CLAHE variant has proven excellent for image enhancement as it does not amplify the noise that standard adaptive histogram equalisation algorithms do [24].

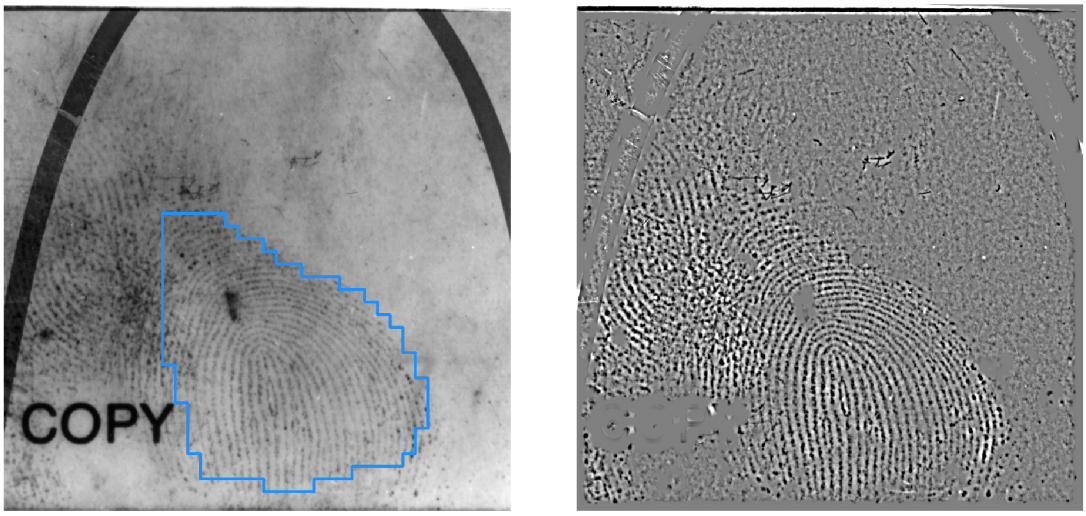
4.2 Cartoon+Texture Image Decomposition

Decomposing the image to cartoon and texture is one of the techniques for removing noise and other unfavourable elements from the fingerprint picture. The algorithm will divide the image into a texture that contains only the oscillating part and noise component of the image and a cartoon part where only contrasted shapes appear. Overall, the picture is smoothed. We will use only the texture part of the decomposition and discard the cartoon part as most external elements obscuring the fingerprint are visible on the cartooned image. The outlines on the texture are perfect for keeping only the ridges and discarding everything else from the image (see Figure 4.1). [4, 5]

The concept of this algorithm is a high total variation of the image. It will try to define the total local variation for each section of the image and then minimise it. The algorithm calculates convolution with a low pass filter in a few iterations. Fourier based filter can be used as well, but in original paper proposes a low pass filter to simplify the coding. Secondly, the Euclidian norm is computed from the gradient of the original image and the one processed with low passed filtering. Thirdly, the Gaussian filter for smoothing is applied to the picture to blur geometrical elements. Lastly, the scale parameter is used to keep the pixel intensities lower. The official proposal of this algorithm proposes a scale parameter (λ) for fingerprints to be 2.5. Finally, we get the resulting image by subtracting the original image from the calculated one, the cartoon one. [4, 5]

4.3 Short-Time Fourier Transform

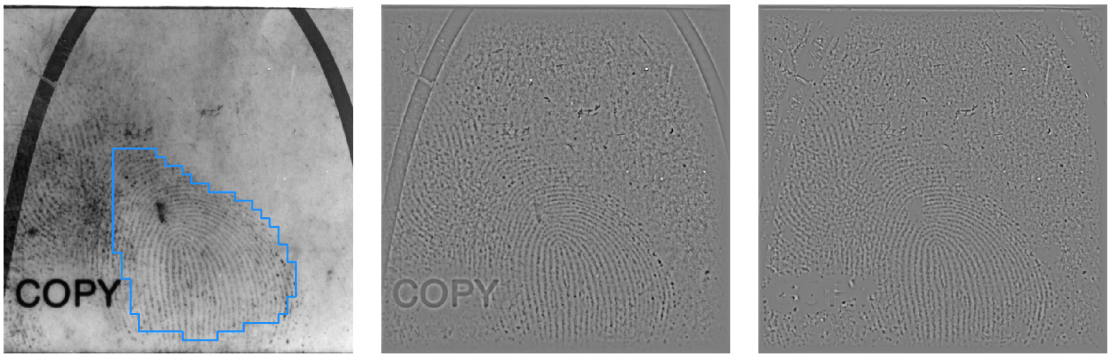
This algorithm uses Fourier transform to transform given functions the their frequency domain. The difference from the standard Fourier transform is that the function is multiplied by a window function in short periods. The algorithm was initially made for one-dimensional functions and then extended for use in two dimensions. However, it was not intended to be used on fingerprint images. Chikkerur, Cartwright and Govindaraju [12] extended the Short-Time Fourier Transform (STFT) for two-dimensional purposes with the intention to be used on fingerprint images and modified it to look for contextual and noncontextual information. The STFT analysis can estimate the ridge orientation, ridge frequency, and



(a) Source fingerprint image with a highlighted mask

(b) Generated Texture image with Cartoon+Texture decomposition

Figure 4.1: Showcase of Cartoon+Texture decomposition on a fingerprint image. Note the removed “COPY“ text.



(a) Source fingerprint image with a highlighted mask

(b) Generated fingerprint image with STFT

(c) Generated fingerprint image from Texture with STFT

Figure 4.2: Showcase of Short-Time Fourier Transform on a fingerprint image

even a mask of the fingerprint. The contextual information immensely helps with the final image enhancement (see Figure 4.2). [12]

The algorithm can be divided into two parts. The fingerprint image is divided into overlapping blocks in the first part, and then each block is analysed in the frequency domain. The algorithm probabilistically estimates each computed block’s ridge orientation and ridge frequency. Those estimations can be displayed as a frequency map used to determine the mask of the fingerprint. In the second part of the algorithm, the contextual information is used from the first part to filter the fingerprint image in each block in the frequency domain. By tiling the result of each block, the final image is obtained. [12, 26]

4.4 Filters

Filters are commonly used for the enhancement of fingerprint images. They make the fingerprint more visible, denoise the image or sharper the fingerprint ridges. While processing a fingerprint image, a combination of various filters is usually used [21]. One of the most valuable filters in fingerprint recognition is the Gabor filter which proved to be very useful while enhancing the ridges of the fingerprint thanks to its frequency detection and orientation variability [16, 29]. Another useful filter is Gaussian which is not helpful by itself, but with the support of a sharpening filter, it presents a simple way of denoising the image [21].

Gaussian Filter

Gaussian filter, or another name for Gaussian blur, is a linear filter used to smooth the image and remove unwanted noise while not having a high impact on edges [1]. This filter is another way of enhancing a fingerprint image, and it is usually combined with another filter for image sharpening. This algorithm is commonly used while denoising fingerprints since it gives surprisingly good results. [21]

Gabor Filter

Gabor filter is a linear filter that detects frequencies in a specified direction and enhances them. It is commonly used for classification or segmentation purposes. Gabor filters are used in fingerprint recognition because of their unique edge detection when processing fingerprints, especially ridges. Usually, multiple Gabor filters are used in various orientations and frequencies to determine where are the ridges [16, 29]. The ridges are then extracted and combined. The most popular setting for Gabor filters is to use eight different orientations. [29]

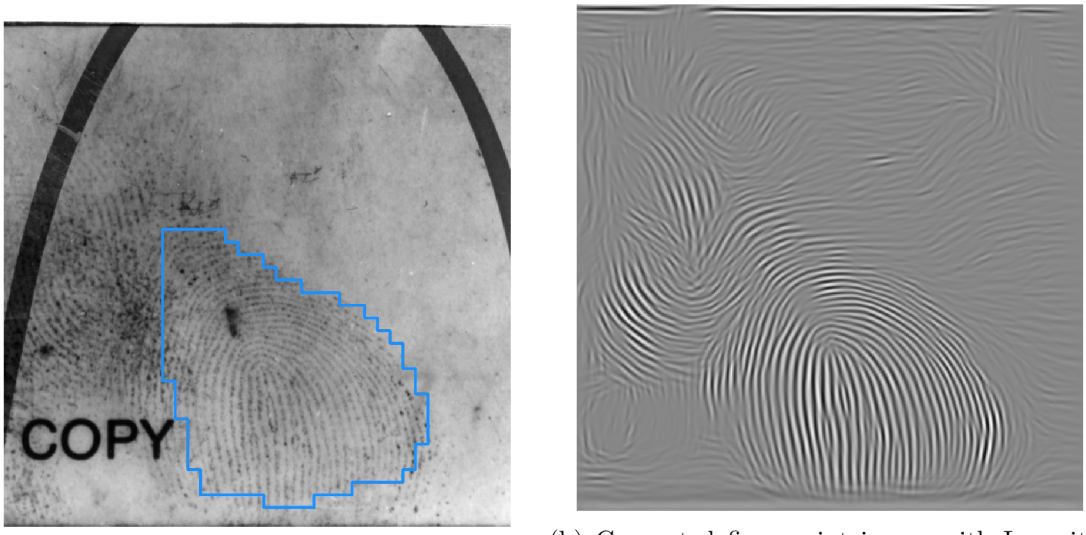
If a user wants to apply the Gabor filter to an image, he must specify the orientation, frequency, and standard deviations. The frequency and orientation should not be an issue as the ridge frequency, and ridge orientation usually determines them, but a user needs to select the standard deviation values. If a small number is selected, the filter could introduce non-existent ridges, and if the number is too large, the filter could omit some of them. [29]

Several improvements to Gabor filters were made, one of which is the use of Logarithmic Gabor filters, which improve the maximum bandwidth length and broadens the Gabor spectrum (see Figure 4.3). Also, the already mentioned Short-time Fourier transform is an improvement from the Gabor contextual filtering [29]. [38]

Sobel Filter

Sobel filter is used for edge detection and enhancement. It can be applied horizontally or vertically to enhance ridges in that particular direction. We can add both orientations together to sharpen the geometrical edges if needed. If the image is too noisy or has other non-fingerprint elements, some smoothing filters, such as a Median filter or Gaussian filter, can be applied to reduce the noise in the picture. [21]

Usually, the Sobel filter with a kernel of size $3x3$ is used, which can be approximated as first-derivative-of-Gaussian kernels [31].



(a) Source fingerprint image with a highlighted mask

(b) Generated fingerprint image with Logarithmic Gabor Filter from Texture image decomposition

Figure 4.3: Showcase of Logarithmic Gabor Filter

4.5 Estimation of Ridge Orientation

This operation tries to estimate the orientation of the ridges. This helps enhance the image as the algorithm can suppose how the ridges are structured and can contextually search for ridges under found angle. It would be impossible to estimate the ridge orientation for each pixel. Usually, the images are segmented into blocks. Each block is graded on its own and gets assigned a calculated direction in the interval from $(0, 360\rangle$. [29]

There are multiple approaches for estimating the ridge orientation, adapted from [29]:

- Gradient-based – is the simplest and the most popular method. The algorithm calculates gradients for each pixel in the vertical and horizontal directions. Then randomly adds ± 1 based on the equality of those gradients. Lastly, it estimates the local orientation of each block with the sums of those gradients and the arctangent from the summed result. [39]
- Reference orientation or slit-based approaches – is one of the oldest methods. The algorithm defines a fixed count of reference orientations and then selects the best direction based on the grayscale values for that current block in the image. The local orientation of the block is the ridge orientation. However, the gradient technique has proven to be more straightforward and reliable in most cases.
- The frequency domain – is an application of multiple directional filters in the frequency domain with local smoothing. The already mentioned Short-time FT is one of the options of how we can estimate it with this method.

4.6 Ridge Structure Dictionary

This approach of enhancement uses generated high-quality ridges stored in a dictionary. The algorithm utilising this technique will recognise and patch ridges with poor quality

on a given fingerprint and replace them with high-definition ridges. The ridge structure dictionary estimates ridge quality map, orientation and frequency fields. It uses the Cartoon+Texture decomposition, which is the first step of this algorithm. The texture is then divided into overlapping blocks, and for each block, its sparse representation and reconstructed block are calculated using the coarse level dictionary with orthogonal matching. We can estimate the coarse quality, frequency fields, and coarse orientation by calculating the structural similarity between the block and its reconstructed version.

The results obtained from blocks using the coarse level dictionary are then used with the fine-level dictionary to calculate a more precise result. The fine dictionary is used to get the ridge quality map, orientation and frequency field which are then used for segmentation. [8]

This algorithm has excellent results for enhancing the fingerprint quality and localising the fingerprint on the image.

Chapter 5

Proposed Algorithm

The quality of fingerprints is challenging to estimate only by one algorithm that grades only one fingerprint feature. The means of what makes a fingerprint good differs in our intentions and what we want to achieve with the particular fingerprint [16]. Generally, fingerprints are used for identification purposes, where they are matched against other fingerprints in a database to identify a match between two records [29].

Identification by fingerprint is commonly made with the minutiae points extracted from the fingerprint. However, to do so, the fingerprint needs to be of good quality to give us the best result that can identify a person beyond a reasonable doubt. The number of minutiae points is one example of how we can grade the fingerprints. As minutiae points are usually used in fingerprint matching, their number should be the critical element while grading them. Nevertheless, fingerprints have other means that we can use for identification. Based on these features, people can be identified as well, and therefore we should consider them. [29]

The algorithm for fingerprint grading does not have to be based on the identification values of the fingerprint. Even though the matching potential of the fingerprints is a significant factor, we could grade the fingerprint images on statistical data. This statistical grading could help determine how much or if we can use the fingerprint image. While enhancing the fingerprint, we cannot be sure that the algorithm will improve the fingerprint correctly or if will it make an error. It could even improve it in a specific way that would potentially suppress some details intended to be extracted. The grading algorithm should consider some statistical values about the image itself, which could help us differentiate results in complex analysis. [16, 29]

The proposed solution for grading a fingerprint needs to be split into multiple smaller algorithms, which each grade some properties of the fingerprint of the image. The source fingerprint images need to be processed first by transforming them into grayscale and appropriately cutted to be compatible with the following computations. Then the fingerprint needs to be enhanced as much as possible, which allows us to grade the fingerprint with all the data that the image can give us and compare it accordingly to some predefined values.

5.1 Pipeline

The designed solution to this problem is segmented into three parts. The Preprocessing, where all the initial parsing of the image happens before the fingerprint is taken further. The Enhancement, where the algorithm enhances the fingerprint and calculates additional

images, which are then used in the final part – the Grading. The last part of the algorithm is the Grading, which executes the sub-algorithms that classify the fingerprint. The structure of the proposed solution can be changed if the input and output data remain in each section of the algorithm the same.

All the information describing the quality of the fingerprints is then reported and saved in an external JSON file. Some results, such as minutiae points and the number of ridges, can be evaluated by predefined constants in the library. Those constants are based on previous research, and they are trying to determine the correct rating of calculated values.

5.2 Preprocessing

The initial process transforms the image into an object we can analyse (see Figure 5.1). The image's shape needs to be cut to be dividable into blocks. The partition into blocks is crucial for our following computation since many algorithms depend on block analysis. The Contrast Limited Adaptive Histogram Equalisation (CLAHE) highlights the ridges, making the segmentation process easier and available as soon as possible. Since this is the algorithm's initial state, those processes are relatively simple preparation for the Enhancement.

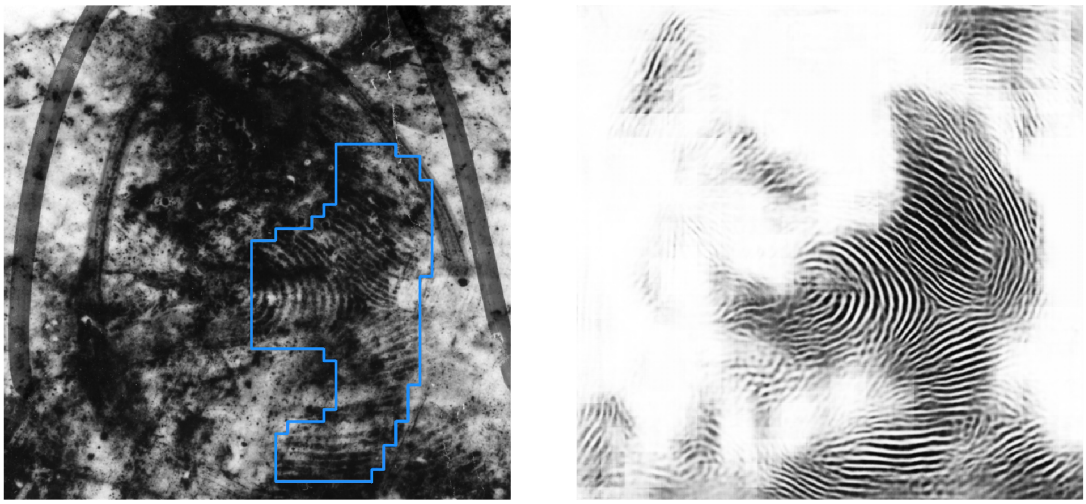
Figure 5.1: The pipeline of the Preprocessing part of the algorithm

5.3 Enhancement

This process tries to get most of the information we can get from the fingerprint image. The process can be modified or fundamentally changed as long as the output data that it should generate are preserved and passed to the last part – the Grading part of the algorithm, where the computations will allow us to grade the image based on rendered pictures. The Enhancement part of the process needs to output a binarised fingerprint image, the mask of the fingerprint (segmented area of fingerprint) and enhanced version of the fingerprint (image processed by autoencoder). Fundamentally, the process tries to remove all the artefacts and noise from the image, find the exact position of the fingerprint in the picture, mark it with a binarised mask and give us the best version of an enhanced fingerprint it can. This enhanced picture with a marked area can significantly help us grade the fingerprint as all necessary values are extracted.

After the image is processed with Preprocessing, the algorithm will use the Fast Cartoon+Texture decomposition [4, 5], which has proven to be working in fingerprint related algorithms and is relatively fast to compute and analyse. It would remove most artefacts such as letters, lines, or if present, the border from the dactyloscopy cards, which we can commonly find on rolled fingerprints where the person's fingerprint interfered with the edge of the card. It is applied to the grayscale image to generate the Texture, and the Cartoon part is discarded. We chose the Fast Cartoon+Texture image decomposition since it was used in many similar enhancing algorithms such as [28], and even in the original paper [10] it is used for fingerprint ridge enhancement and noise removal with excellent results.

After the texture image is generated, the Short-time Fourier transform [12] is applied, which analyses the ridges on the fingerprint, enhances and highlights them. As was men-



(a) Source fingerprint image with a highlighted mask

(b) Generated fingerprint image by the autoencoder

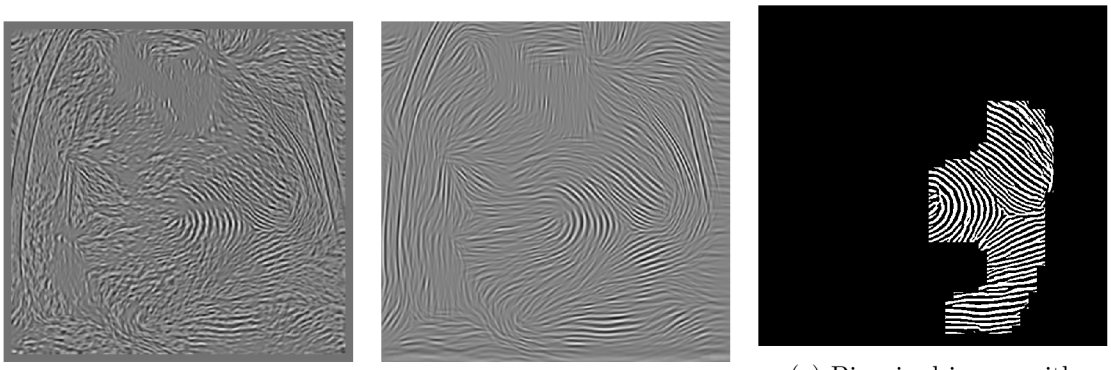
Figure 5.2: Showcase of enhanced fingerprint B176 from NIST dataset by the autoencoder

tioned in the previous Section 4.3, the Short-time Fourier transform also analyses the ridge orientation and ridge frequencies and, based on this contextual information, can help us with the enhancement [12]. The STFT algorithm works well when applied to the generated Texture image since the Texture image has all the noise and elements removed, which the STFT would otherwise sharpen. Finally, the Texture image processed with STFT is refined by ridge structure dictionary and autoencoder developed by [10]. The autoencoder is a graph neural network that consist of generated ridge patch dictionary which removes almost all of the noise and external elements which were not removed by now while enhancing the fingerprint's ridges. Thanks to those algorithms, we can estimate the mask of the fingerprint with a ridge quality map and get the best version of the fingerprint, which is crucial for our following computation and grading (see Figure 5.2). If we could not calculate the position of the fingerprint on the image, then it would be almost impossible to assert its quality as we would not know the place of the fingerprint. The [10] paper uses the dictionary-based approach for ridge flow and ridge spacing estimation, which was initially proposed in [8] and then modified to use different ridge orientations and spacings.

The parallel process creates a binarised image with a Logarithmic Gabor filter [38] applied to the Texture image generated from the Fast Cartoon+Texture algorithm [4, 5]. This particular image is used to count the ridges and estimate the stationary point on the fingerprint, which will be described in more detail in the following chapter. The Logarithmic Gabor filter has proven slightly more precise while processing the ridges than the standard Gabor filter as it yields cleaner results (see Figure 5.3).

Finally, the autoencoded image, mask and the binarised image are passed to the Grading part of the algorithm, where all images are masked to make the Grading part work just with the enhanced section of the fingerprint image (see Figure 5.4).

One issue with this algorithm is that it cannot differentiate if two fingerprints are merged and overlap. If two fingerprints are present, it will choose the one with the most information, meaning the more significant fingerprint shown. This could be solved by adding an algorithm which could split those fingerprints or analyse them gradually.



(a) Generated image with Gabor Filter

(b) Generated image with Logarithmic Gabor Filter

(c) Binarised image with applied Logarithmic Gabor Filter

Figure 5.3: Comparison of Gabor and Logarithmic Gabor filter on fingerprint image (NIST dataset fingerprint B176) with applied Gaussian filter

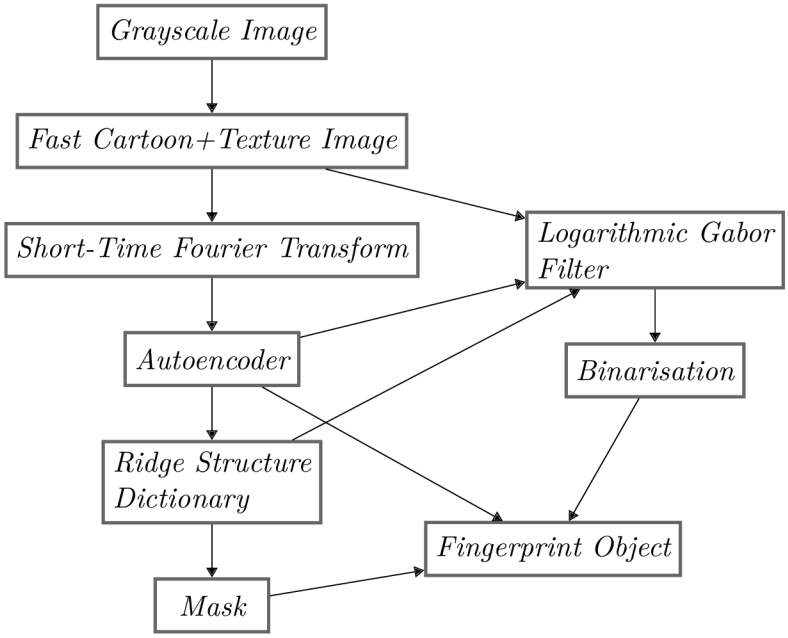


Figure 5.4: The pipeline of the Enhancement part of the algorithm

5.4 Grading

The last part of the algorithm is to grade the enhanced fingerprint images. On each fingerprint image, all sub-algorithms are run to calculate its performance. That sub-algorithms grade the number of minutiae points, number of papillary lines, contrast values, the sinusoidal similarity of papillary line crosscut and thickness of the ridges (see Figure 5.5).

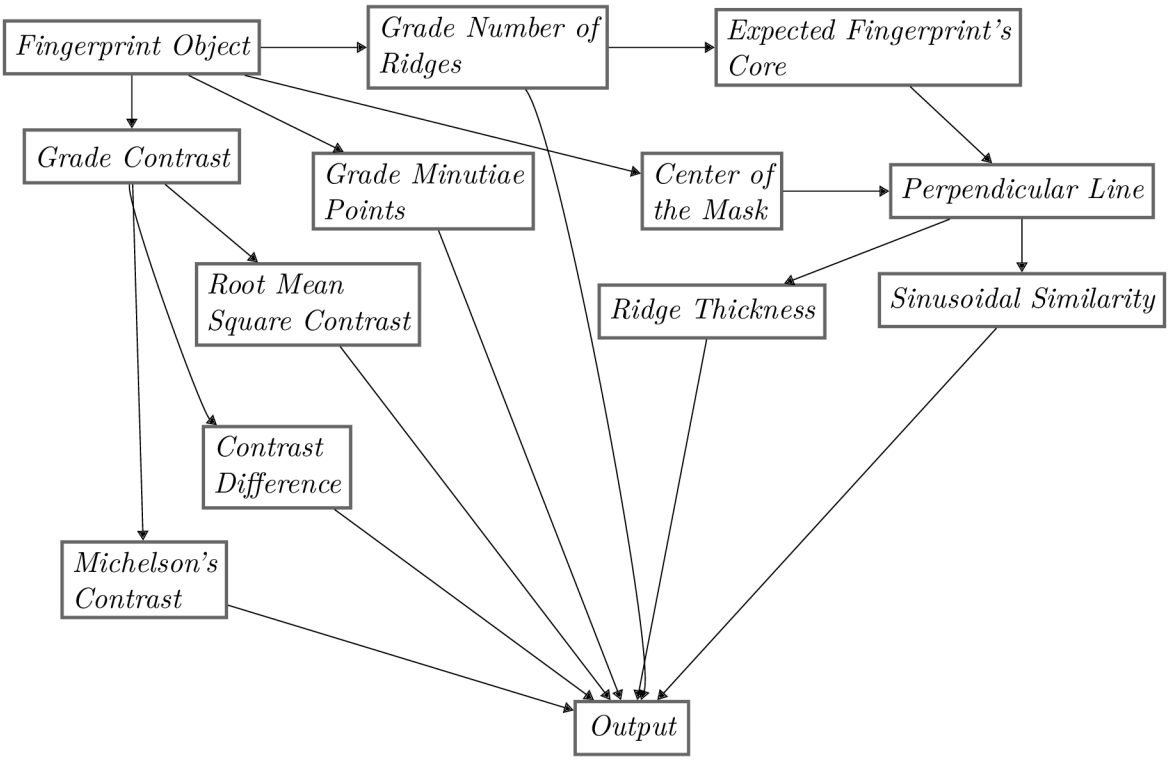


Figure 5.5: The pipeline of the Grading part of the algorithm

Number of Minutiae Points

The process of finding the minutiae points and grading their number is essential. It cannot be omitted since most fingerprint recognition depends on minutiae points and their similarity to other fingerprints in an already established database. The minutiae points are found and counted with a neural network developed by [10] that is run on multiple generated pictures. These images are created with methods already explained in sections before, such as Gabor, Gaussian and Logarithmic Gabor filter, STFT, Texture, Autoencoder, and their combinations. The graph neural network is run on those pictures, and it looks for common minutiae points across all generated fingerprint images. Those minutiae points found multiple times in the same places with some deviation acceptable by the selected threshold are counted and processed. Only minutiae points inside the already segmented mask can be counted as valid. The algorithm proposed by [10] is more complicated when detecting minutiae points as it takes multiple generated minutiae points templates from which similarities are calculated as a weighted sum and then applied. The proposed algorithm is simplified as only the common minutiae found across all images are found and taken into account. The process could potentially prove not to be as performant as it was in the original paper, but it significantly simplifies the process while still being one of the best algorithms tested.

The number of minutiae points is then compared to established definitions. Many courts and institutions accept a match of 12 and more minutiae points as a match beyond a reasonable doubt and as the suspect is successfully identified. This limit is known as the 12-guidelines [29], the probability of identification based on twelve correctly asserted minutiae points. Lowering this number makes the likelihood of a match plummet significantly

while increasing this number will not increase the possibility substantially. However, since there is a possibility of an error match in fingerprint scanning and enhancement, we raised the threshold of this match to 20, which is almost double the previous value, to ensure the correct result. As the number of minutiae is still visible in the JSON file, it can be understood as a score rating, which will guarantee that once the fingerprint exceeds the 20 minutiae point limit, it will surely match beyond a reasonable doubt [29].

Number of Ridges

The second algorithm counts the number of papillary lines in the horizontal and vertical directions. This process can tell us how dense and how big the actual fingerprint is. After calculation, we can determine the stationary point where the ridges are the densest based on those values. Theoretically, the point should be positioned right at the core of the fingerprint. However, due to the high variability of latent and other non-perfect fingerprints, this method is not accurate for core estimation. The delta of the fingerprint can confuse the algorithm as well as it is another dense ridge point on the fingerprint image.

The algorithm is simple yet effective. We draw imaginary lines throughout the image, counting each transition in the drawn line from 255 (white) to 0 (black) and then from 0 to 255. This improved approach was based on the recommended one in [29]. They They suggested counting transitions from 0 (white) to 255 (black) as 1. However, the algorithm sometimes makes an Off-by-one error. If the ridge overlaps with the fingerprint mask, it will not be counted, as the non-fingerprint background, part of the image masked out, is black as well, and we cannot differentiate the ridge from the background. The distance is then used to normalise the result to prevent small masks with a few ridges from being more performant than more extensive areas. Finally, the result is averaged throughout all imaginary lines in horizontal and vertical directions.

The number of ridges also has predefined values in the library, which were carefully selected based on the observation of good and bad fingerprints. The number serves as a rough estimate of the expected density of the fingerprint.

This algorithm can also tell us some information about the scanner itself. If the higher density of the ridges is found on the smaller area, the scanner's resolution must be higher, but since the algorithm needs PPI as an input, we do not need to measure it. [16]

Contrast Algorithms

Various algorithms were used to grade the contrast and colour difference between ridges and valleys and gather statistical information about the picture. The Michelson contrast formula calculates local differences and divides the intensities to define the contrast. It is one of the common ways to grade the image's contrast.

The contrast algorithms could try to identify the error of the neural network. It was observed that the autoencoder with the Short-Time Fourier domain and ridge structure dictionary sometimes makes an error and enhances texture which may look like a fingerprint, but it is not. The resulted difference in contrast between ridges and valleys is processed. The result could indicate some interesting values by which the algorithm could decide if the fingerprint is valid or invalid and discard it from future analysis.

Contrast Difference algorithm is proposed in this thesis. The algorithm extracts only the pixels of the ridges and the pixels of valleys, which can be obtained by combining the mask of the fingerprint and its binary image. The extracted pixel colours are averaged and normalised to $\langle 0, 1 \rangle$. In an ideal scenario, the mean for valleys would be 1 (white colour)

and for ridges 0 (black colour). The valleys are transformed to translate the differences in colours by subtracting 1 from the mean values and then counting the absolute value. If we then calculate the ratio from those mean values, we would obtain the ratio between ridges and valleys. The ideal fingerprint would have a ratio of the value of 1.0, and the ridges and valleys would have the highest possible value (which is 1.0 for both).

Suppose the ratio has a lower value than zero. In that case, it means that the fingerprint is darker and the mean of the valley is not as strong as the mean of the ridges; contrary to that, if the ratio is higher, it means that the mean of the valleys is more substantial and the image is overall whiter. This algorithm could try to divide fingerprints into dry and wet since if the algorithm recognises that the fingerprint is darker, it has a higher potential to be wet and vice versa.

A good approach for improvement would be to set interval values for the mean of ridges and mean of valleys. Since the algorithm would grade fingerprint with score 1.0 when the ridges and valley would be of a low value, the actual colour difference would not be substantial even though it would be a perfect ratio. The formula for calculating the fingerprint image contrast is as follows:

$$C_r = \frac{M_r}{M_v} \tag{5.1}$$

$$M_r = \left(\sum_{i=1}^{N} \frac{E_r}{255}\right) * \frac{1}{N} \tag{5.2}$$

$$M_v = \left(\sum_{i=1}^{N} \left|\frac{E_v}{255} - 1\right|\right) * \frac{1}{N} \tag{5.3}$$

where the E_r and E_v are extracted values from ridges and valleys from the image, respectively. The algorithm also needs to be prepared when either the mean of the valleys or ridges equals 0.0. This would not usually happen in a real-world scenario. Still, for completion, if the algorithm encounters E_v value of exactly 0.0, the result would be the highest possible value of the number in the calculated interval. If E_r would be 0.0 then the result value would be 0.0.

The following Chart 5.6 explains the ratios. The blue line symbolises the ideal mean value of the fingerprint. The closer the fingerprint gets to the blue line, the better its ratio and the better fingerprint quality. However, to be classified as a good fingerprint, it also needs to maximise its ridge and valley mean values. Therefore, the best fingerprints will be in the "ideal" section of the chart. The fingerprint with the lowest possible value will be either in the "substandard" part of the diagram or somewhere else, making them inadequate.

We propose another contrast algorithm which calculates the RMSE (Root Mean Square error) between the binary fingerprint image and the received fingerprint. The fingerprint should ideally be close to the binary image, and therefore the RMS error should be lower when the fingerprint approaches the ideal values. We add those errors together to get the final result. The final result represents the overall error from the binary image to the natural fingerprint. Similarly, as in the previous algorithm, we normalise the RMS between $\langle 0, 1\rangle$ by dividing the mean value of each error by the highest possible value in the grayscale image (255). After adding the error values together, the result is divided by 2 (for each error) and multiplied by 100 to get the final error value in the $\langle 0, 100\rangle$ interval. The lower the error rate is, the better the fingerprint image is. The formula for computing the values

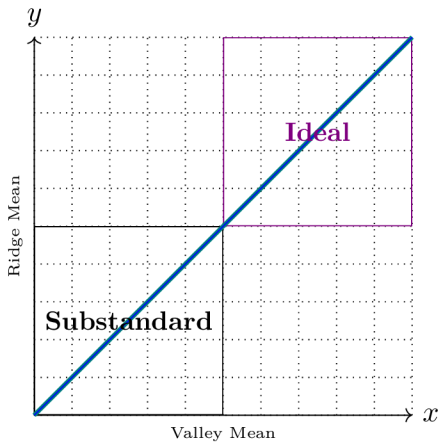


Figure 5.6: Showcase of graph which segment the results of contrast Colour difference algorithm. The perfect value should be around the blue line. Lower mean values signal substandard fingerprints, and the ideal fingerprints maximise both values.

is as follows:

$$RMS_e = \frac{(Er_r + Er_v)}{2} * 100 \tag{5.4}$$

$$Er_r = \sqrt{\left(\sum_{i=1}^{N} (E_r - B_r)^2\right) * \frac{1}{N} * \frac{1}{255}} \tag{5.5}$$

$$Er_v = \sqrt{\left(\sum_{i=1}^{N} (E_v - B_v)^2\right) * \frac{1}{N} * \frac{1}{255}} \tag{5.6}$$

where the E_r and E_v are extracted values from ridges and valleys from the fingerprint image, respectively, and B_r in B_v are extracted values from ridges and valleys from the binary image, respectively.

Sinusoidal Similarity

The fourth algorithm is examing the ridge crosscut similarity with a sinusoidal wave. This similarity can tell us how much the fingerprint's ridges deviated from the ideal ridges, which should be precisely sinusoidal wave in interval $\langle \frac{-\pi}{2}, \frac{3\pi}{2} \rangle$. Those values can estimate the overall quality of the fingerprint since tilt or thickness change would be visible. Usually, on low-quality fingerprints, ridges are damaged by smudges present on the image. These smudges can dramatically change the tilt of an otherwise perfect ridge detected by this algorithm. The algorithm is executed twice as it is computed with the fingerprint's mask centre and once with the estimated stationary point to get the most of the ridges in the perpendicular line.

Estimating a perpendicular line to the ridges is needed as we need to extract the ridges under the best possible angle. This estimation of latent fingerprints or other images with unknown quality is difficult, but getting as possible to the result is crucial. The algorithm for crosscut estimation uses the Soble filter applied to the binary images in the vertical direction to highlight ridges and estimate the best direction. Since the only Sobel filter

applied to the image will not perform well on the binary image before the Sobel filter, Gaussian blur is applied. The potential perpendicular line is drawn in all directions, and the one with the highest intensity is then selected. Blacks (0) and white (255) pixels are discarded from the drawn line.

The resulting perpendicular line is a signal with ripples created by the grayscale image. The algorithm then splits the signal by calculating local extrema, minima and maxima. The signal is, before this calculation, run through a low pass filter by [11] to minimise any disturbances. By calculating the number of local maximal extremes, we count the number of ridges while the local minimum extremes split the signal. This ensures that we extracted just ridges from the crosscut. The next step is to position the sinusoidal interval directly with the ridge. It can be achieved by calculating the square difference between the signal extracted from the grayscale image and the sinusoidal function. The biggest result has the best alignment.

After the positioning, the integral from both the sinusoidal wave and the segmented line is calculated. Integration is calculated with Simpson's rule approximation. Then the deviation between those two values is calculated as was described in 3.4. This number can then be averaged, or it can be graded per ridges.

Thickness

This algorithm is very similar to sinusoidal similarity as it uses the perpendicular line as well. The perpendicular line is estimated with the Sobel filter and the division of ridges with local extremes. It is also executed twice for the stationary point and the fingerprint's mask centre. It calculates the thickness of each ridge and then compares the calculated thickness to an estimated average.

Since each ridge can have some smudge or non-fingerprint element that would raise its grayscale values, we must determine at what height we will measure the ridge. Since we sliced the ridges, we can always have the same height in percentage as it was proposed in [16], and that is 18%. The thickness is then calculated from the received pixel length with the formula mentioned in 3.4, and the result gives us the deviation of each ridge. This number can then be averaged, or the fingerprint can be graded per ridges.

5.5 Results

The proposed algorithm with its sub-algorithms was executed on the NIST SD27 fingerprint database [23]. All 292 fingerprints were graded by NIST (National Institute of Standards and Technology) and divided into three categories – Good, Bad and Ugly. "Good" is the best classification the fingerprint can get, and "Ugly" is the worst. It is unknown by what statistic or grading algorithm the database was graded since NIST did not specify it in their documentation [23]. Nevertheless, their results can still be compared to the results of the proposed algorithm as a baseline to evaluate if the proposed algorithm with its sub-algorithms is valid. Figures of fingerprints that reflect the results of each sub-algorithm are named by the fingerprint's name determined by the NIST database and with the score calculated by the algorithm. Each fingerprint name starts with the letter 'G' (Good), 'B' (Bad) or 'U' (Ugly), which denotes their class.

Results are presented as a box-and-whisker diagram divided into three groups by classification given by NIST. Examined value is always displayed on the y-axis. Each fingerprint class is denoted with a particular colour corresponding to its category to make the diagram easier on the eyes. The chart consists of three main features: the line intersects the box, which represents the median of the given group, the edges of boxes which denote the interval between the first and third quartile, and the whiskers represent the maximum a minimum value in a group. Sometimes outliners are presented, but before the data was plotted, most were removed by the standard deviation of 2. Therefore about 5% of values were discarded in each group, and only values in intervals from 2.3% to 97,7% were kept. The standard deviation helps us see the results more clearly. Sometimes, the algorithms will merge two fingerprints because they overlap, which, especially in the "Ugly" class, makes fingerprints perform better while counting minutiae points or the number of ridges.

Support Algorithms

The sub-algorithms which will be described consist of two smaller algorithms that should be explained before we progress. The first algorithm is for finding the stationary point based on the highest number of ridges in each direction. The second tries to find the perpendicular line to analyse the crosscut. Both of those algorithms have their success variability, but overall, they have proven to be successful.

The algorithm for estimation of the stationary point was initially intended to find the core of the fingerprint, which sometimes it does correctly. However, this estimation was not used due to it not being precise for various reasons. Instead, it is used as a stationary point (Figure 5.7b) on the fingerprint image itself as it most of the time finds more ridges when used as a starting point and when looking for a crosscut.

The second algorithm is for locating the perpendicular line (Figure 5.7c). The results are not precise since estimating perpendicular lines on non-perfect curved ridges is tough. Even though the algorithm does not yield accurate results, it is usually still acceptable as the error is usually no greater than a few degrees, which does not influence the following algorithms.

(a) Source fingerprint image with a highlighted mask

(b) The calculated stationary point with highlighted lines

(c) The calculated perpendicular line on the image generated by the autoencoder

Figure 5.7: Showcase of Support Algorithms on a fingerprint G100 from the NIST dataset

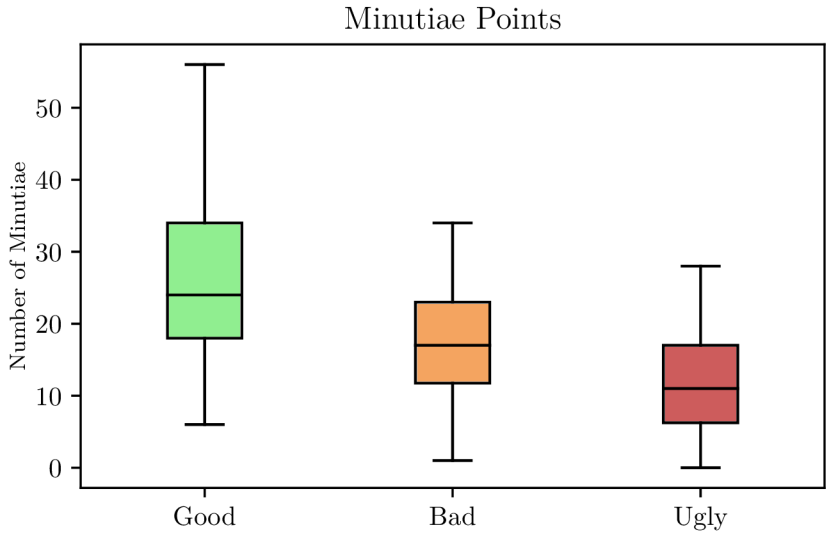


Figure 5.8: Comparison of NIST classification to the calculated Number of Minutiae Points

Minutiae Points

The algorithm for minutiae points identified that if the fingerprint was classified as a Good fingerprint, the number of minutiae points would be higher than if the fingerprint was "Ugly" or "Bad", which is expected behaviour (Figure 5.8). It is assumed that the dataset was graded based on found minutiae points, which is why we see this result. However, it proves that if the algorithm can find more minutiae points on the fingerprint, the better overall the image is classified. It should be pointed out that by 12-guidelines, users would be identified by most of the fingerprints in the dataset, but some minutiae points will be falsely identified. That is why the threshold was bumped from 12 minutiae points to 20 to evaluate fingerprints more accurately.

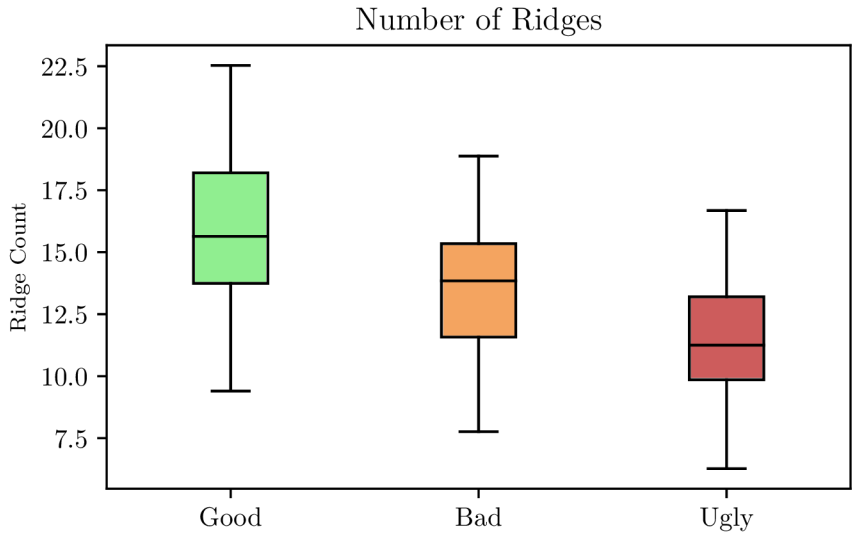


Figure 5.9: Comparison of NIST classification to the calculated Number of Ridges

Number of Ridges

Counting a number of ridges in the horizontal and vertical direction and then calculating the average proved to be a working solution for estimating the quality of the fingerprints (Figure 5.9). It scores similar to a number of minutiae points. That could be due to the overall area of the fingerprint. If the fingerprint has a high number of ridges, it is probably larger and therefore, there is more potential to find a higher amount of minutiae points.

As ridge frequency is used for identification purposes as a quick distinction between two fingerprints, it could, with slight modification, be potentially used to determine fingerprint rating and its classification.

Contrast

There are several contrast methods used in the proposed algorithm. The first algorithm commonly used on fingerprint images to get some statistical information is Michelson's contrast which did not prove to be working to divide the fingerprints into NIST fingerprint classification. The median values are almost the same (Figure 5.10) for each group. This result was expected as Michelson's algorithm was not created to grade fingerprints but as a calculation to denote the luminance of the picture, which it performs.

The following algorithms were built around the contrast information and tried to extend it. There are two developed sub-algorithms in the solution. The Colour Difference algorithm and the Root Mean Square Error Ratio work with the binary images, which helps them distinguish differences between ridges and valleys. The Color Difference does not classify the fingerprints into their categories, as we can see on the graphs (Figure 5.11), but it divides them into groups based on the darkness and lightness of the fingerprint's image.

On the chart, computed mean values for ridges and valleys are plotted, and the blue line shows the ideal ratio the fingerprint should get. The more the dot approaches this line while maintaining high values of each mean value, the better the fingerprint is. The calculated ratio cannot be taken into account on its own as it could lead to mistakes in rare circumstances, as explained in Section 5.4.

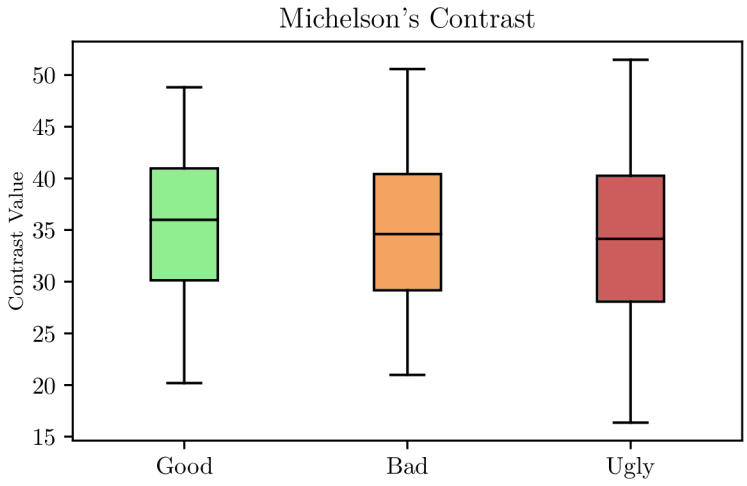
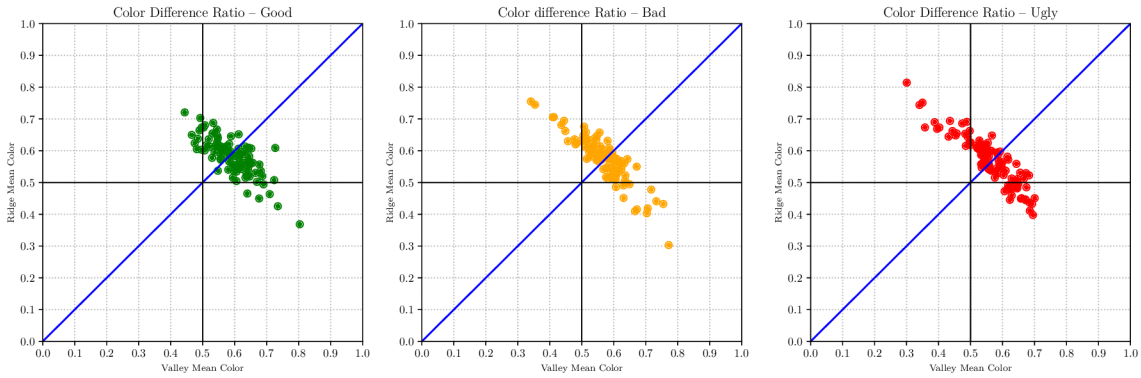
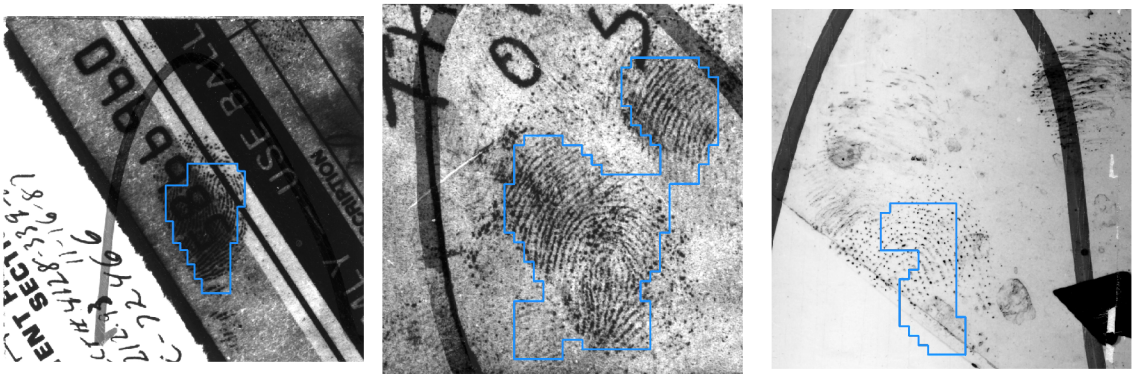


Figure 5.10: Comparison of NIST classification to the calculated Michelson's Contrast



(a) NIST "Good" classification (b) NIST "Bad" classification (c) NIST "Ugly" classification

Figure 5.11: Comparison of NIST classification to the calculated Colour Difference



(a) Fingerprint B120 with score 0.39, 0.30/0.77 (b) Fingerprint G089 with score 0.99, 0.60/0.61 (c) Fingerprint U279 with score 2.70, 0.8/0.3

Figure 5.12: Showcase of the best and worst fingerprints from the NIST SD27 dataset, which were graded by the Colour Difference algorithm

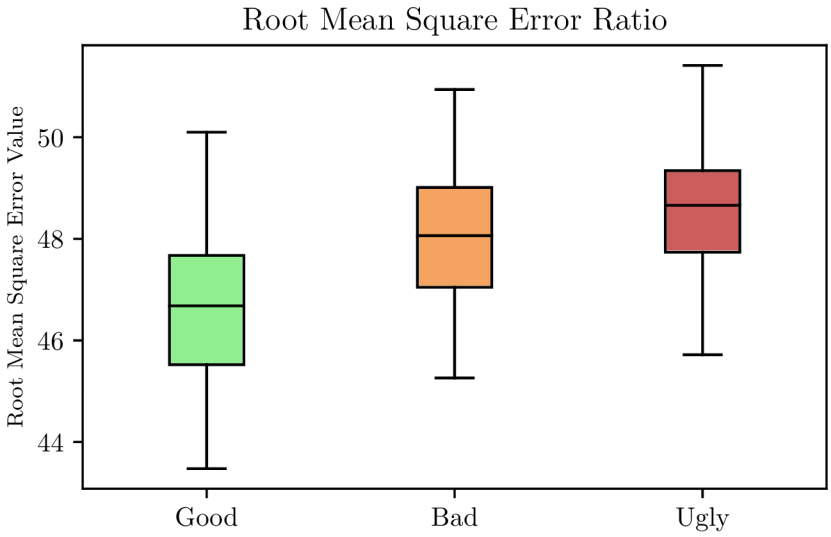


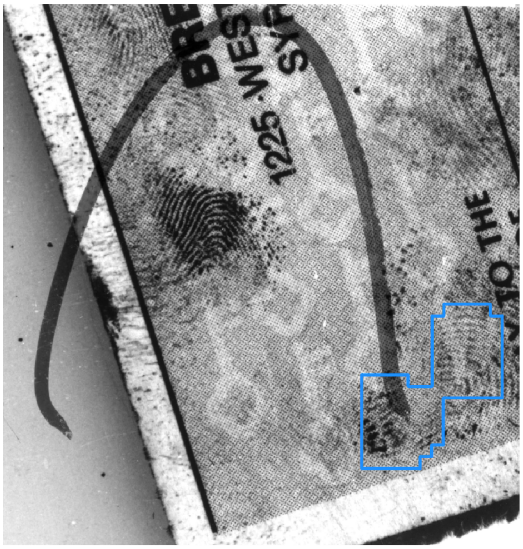
Figure 5.13: Comparison of NIST classification to the calculated Root Mean Square Error Ratio

We picked three images, one which scored the best (Figure 5.12b) and two which scored the worst from each side of the spectrum (Figure 5.12a, 5.12c). The images demonstrate that the algorithm can distinguish between fingerprints with different brightness and select where the ratio is optimal.

The last algorithm in this category is the Root Mean Square Error contrast algorithm which measures the error between the binary mask and the fingerprint. According to the NIST classification, this algorithm scored as the best contrast algorithm, dividing fingerprints by quality. The "Good" fingerprints have lower error rates than fingerprints with other classifications, which indicates that the fingerprint is closer to the ideal fingerprint – binarised fingerprint. This result can be seen in the diagram (Figure 5.13), where even though the maximum values of the "Good" class are pretty high, the median is significantly lower compared to the other classes.

We picked the worst 5.14a and the best 5.14b algorithm-rated fingerprints for comparison. It can be seen from the comparison in Figure 5.14 that the fingerprint "B118" has ridges that cannot be seen properly. Therefore, the algorithm evaluated a higher error score than the fingerprint "G051", with a much lower error score and clearer ridges. Note that the fingerprint "B118" is not the one intended to be matched on the picture.

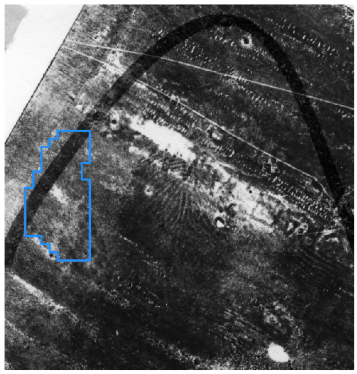
Initially, the algorithm was designed to identify false fingerprints detected by the neural network. Since the contrast from the binary image would be completely different from the grayscale image, it would be susceptible to high error rates. However, due to the high variability of this error, we could not define a perfect value that would catch all or at least the majority of false fingerprint positives, but some are still detected as they are identified as the worst in the database (Figure 5.15a, 5.15b, 5.15c).



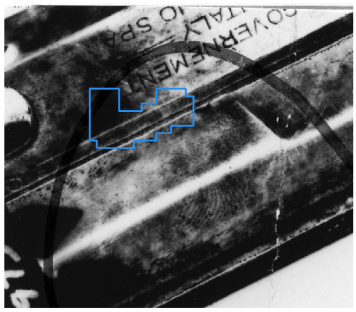
(a) Fingerprint B118 with score 51.337170

(b) Fingerprint G051 with score 40.093955

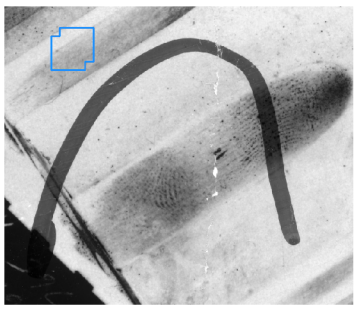
Figure 5.14: Showcase of the best and worst fingerprints from the NIST SD27 dataset, which were graded by the Root Mean Square Error algorithm omitting the false positives



(a) Fingerprint U298

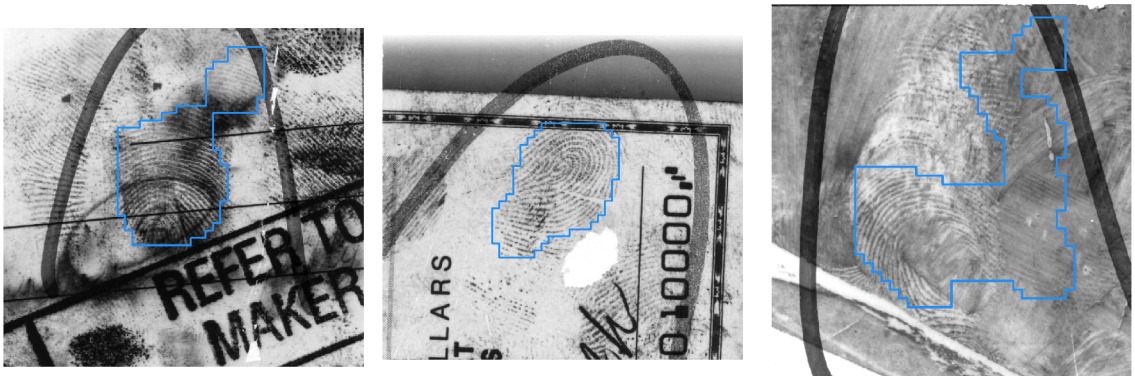


(b) Fingerprint U235



(c) Fingerprint U235

Figure 5.15: The three false-positive fingerprints from the NIST SD27 dataset detected by the Root Mean Square Error algorithm



(a) Fingerprint B111 with score 55.565425

(b) Fingerprint G093 with score −0.040597

(c) Fingerprint G090 with score −39.601382

Figure 5.16: Showcase of the best and worst fingerprints from the NIST SD27 dataset, which were graded by the averaged Sinusoidal Similarity

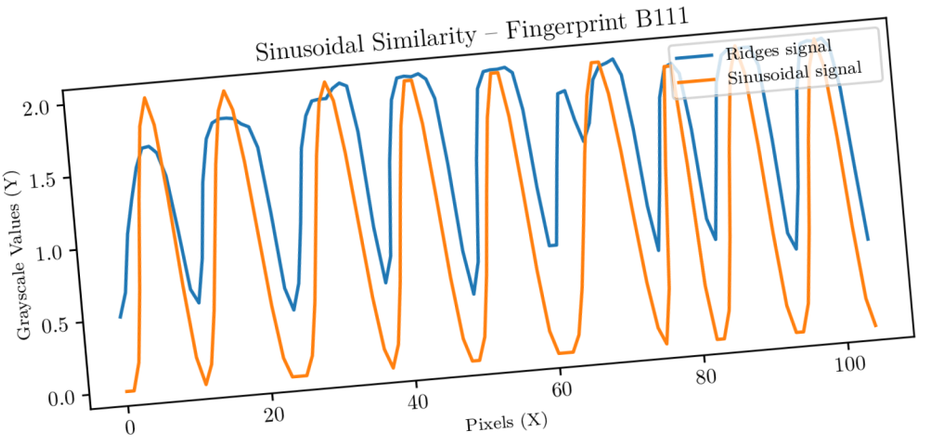
Sinusoidal Similarity

The algorithm for calculating the sinusoidal similarly was run four times in total. Once for the grayscaled image, once for the enhanced fingerprint from the autoencoder and then two more times with different starting points using an established stationary point and calculated centre of the fingerprint. The results for the sinusoidal similarity were inconclusive since the mean value differs significantly in each group, and there is no pattern for any run. However, we suspect that the database's grading cannot be used as a baseline here since a completely different process graded it. The algorithm outputs different ratings and categorises fingerprints in its class system.

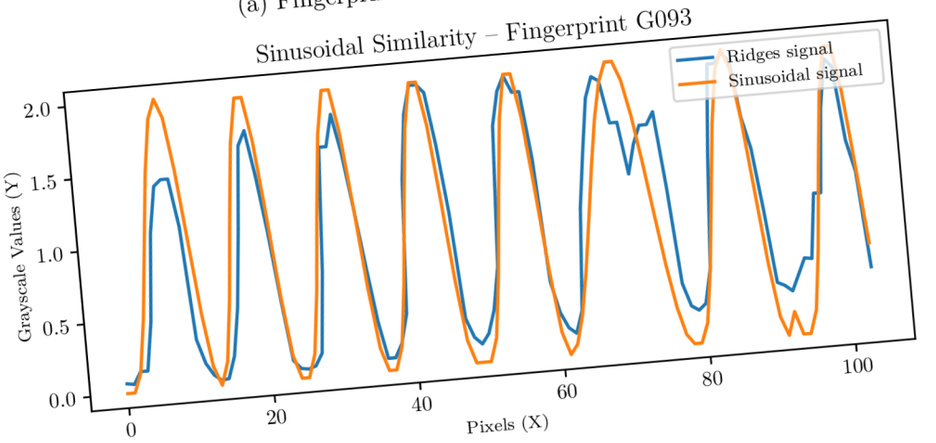
The more the fingerprint is close to 0, the higher the quality of the graded fingerprint by this algorithm is. The first two diagrams (Figure 5.18a, 5.18b) are displaying the values computed on the autoencoder image, while the other two (Figure 5.18c, 5.18d) on grayscale image. We ran the algorithm on grayscale images since this enhancement could potentially influence the results. Even though the autoencoder and grayscale results differ, it is probably due to impurities of the grayscale image and non-fingerprint patterns that are visible again. The result did not change significantly enough to assume that the autoencoder improved the sinusoidal shape of the fingerprint.

This algorithm was run on the latent fingerprint database. Even with a "Good" classification, the fingerprints will not be comparable to the sinusoidal line, and some deviation will always exist as the fingerprints are not perfect.

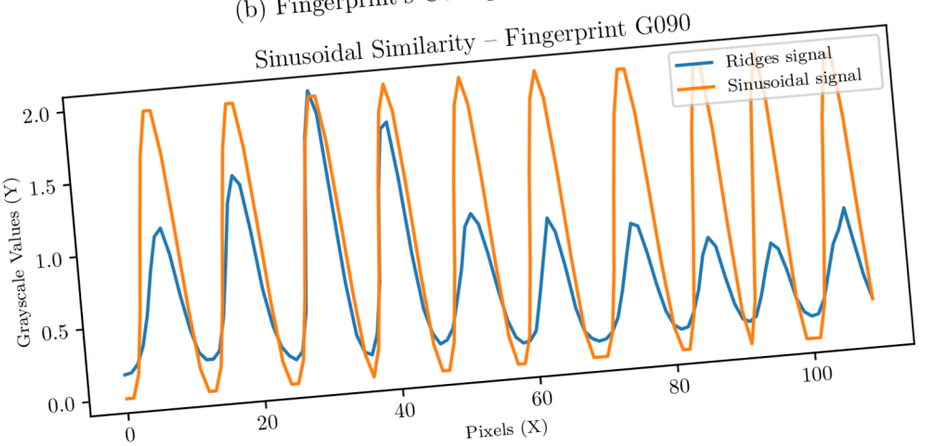
To demonstrate the working sub-algorithm, we choose three images, two fingerprints that have the highest 5.16a and lowest 5.16c values and one closest to the ideal fingerprint 5.16b. The fingerprints that did not perform well are clearly in worse shape than those that performed great, but we do not think this analysis can be compared just by fingerprint images. The corresponding crosscut of selected images extracted from fingerprints with a sinusoidal wave as a comparison in Figure 5.17 is included to describe the results more clearly. From this Figure 5.17a it can be seen that the result has thicker ridges and vice versa. The result in Figure 5.17c has slimmer ridges. Note that the fingerprint with the lowest score was classified by NIST as a "Good" fingerprint.



(a) Fingerprint's B111 perpendicular line

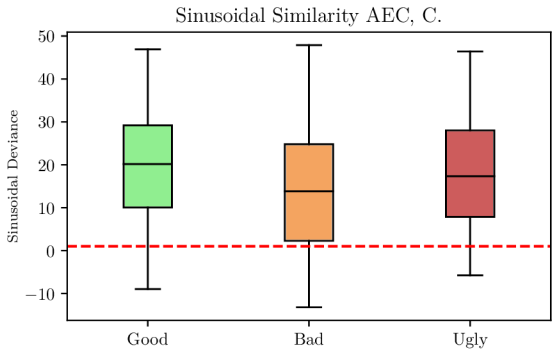


(b) Fingerprint's G093 perpendicular line

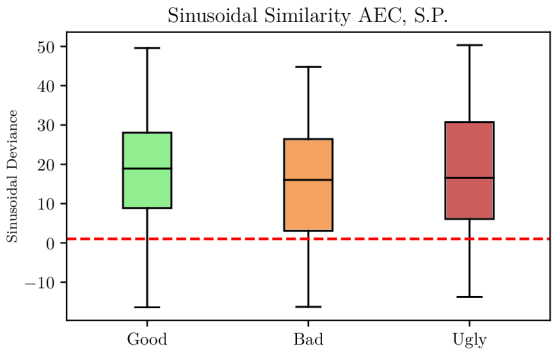


(c) Fingerprint's G090 perpendicular line

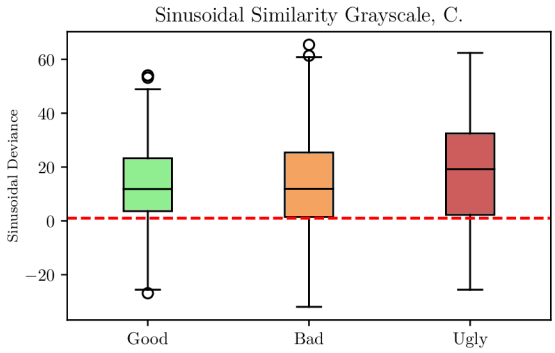
Figure 5.17: Extracted perpendicular lines from the fingerprints with sinusoidal wave in specified interval in Figure 5.16 to compare the algorithm result



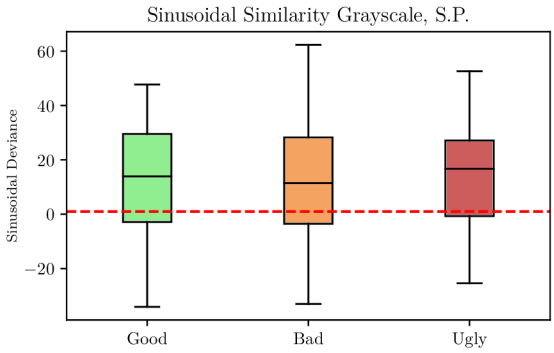
(a) Calculated score on fingerprint images generated by autoencoder with the centre of the fingerprint



(b) Calculated score on fingerprint images generated by autoencoder with the stationary point



(c) Calculated score on grayscale fingerprint images with the centre of the fingerprint



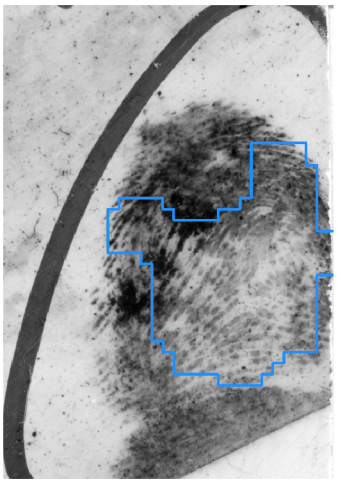
(d) Calculated score on grayscale fingerprint images with the stationary point

Figure 5.18: Comparison of NIST classification to the calculated Sinusoidal Similarity in all four variants

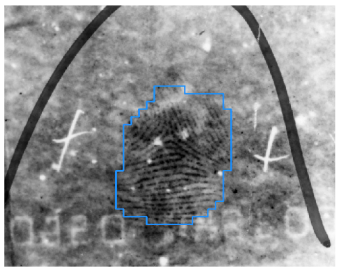
Thickness

The algorithm for calculating the thickness of the image is similar to the algorithm which explores the sinusoidal similarity. It is also run four times with different starting points and on different photos (Figures 5.20a, 5.20b, 5.20a, 5.20b). The results are inconclusive for similar reasons as they were with sinusoidal similarity. However, a few key points can be read from the diagrams. The autoencoder image with the fingerprint's centre as a starting point has slightly better results than the variants, but it is not a very significant difference. The median from the "Good" category is closer to 0, which symbolises the ideal fingerprint with perfect calculated thickness according to the formula. However, again we think that the NIST classification is not an ideal comparison to this type of algorithm and that the algorithm should grade the fingerprints with its class system.

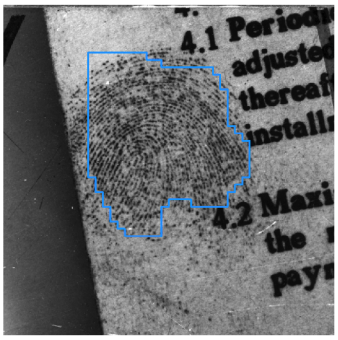
Three fingerprints are outlined to demonstrate the thickest 5.19a and thinnest 5.19b fingerprint and then ideal fingerprint with the "right" thickness 5.19c. From these results, the difference in thickness can be seen, but for completion, we included the charts of crosscut line with calculated ideal thickness and actual thickness in Figure 5.21. The ideal thickness is calculated for each ridge in 18% height as was mentioned in Section 5.4 [16]. Note again that the fingerprint with the lowest score was classified by NIST as a "Good" fingerprint.



(a) Fingerprint B105 with score 53.939394

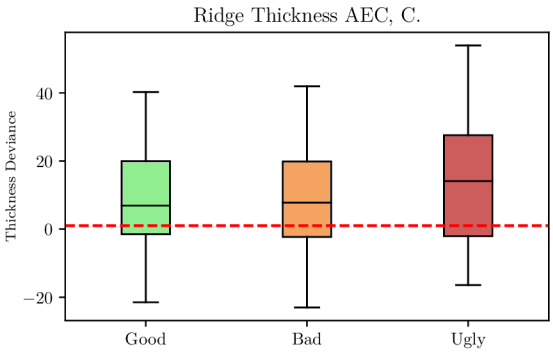


(b) Fingerprint G021 with score −0.040597

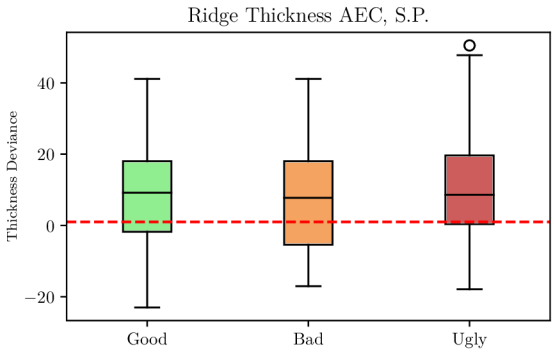


(c) Fingerprint G032 with score −25.229437

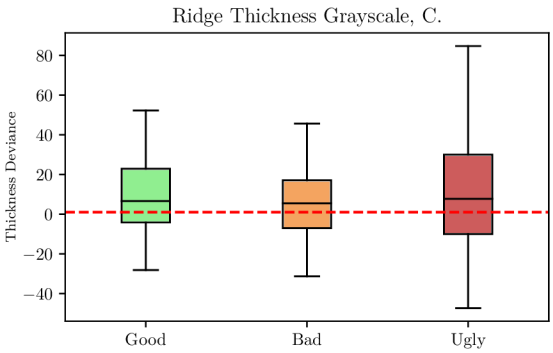
Figure 5.19: Showcase of the best and worst fingerprints from the NIST SD27 dataset, which were graded by the averaged Ridge Thickness



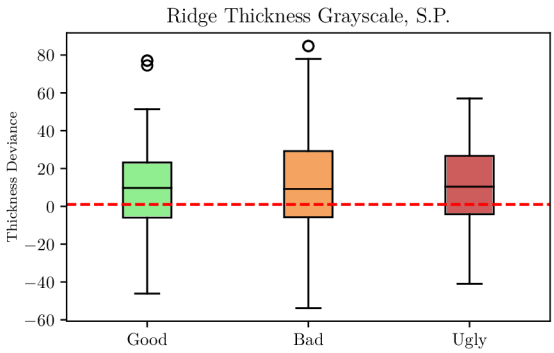
(a) Calculated score on fingerprint images generated by autoencoder with the centre of the fingerprint



(b) Calculated score on fingerprint images generated by autoencoder with the stationary point

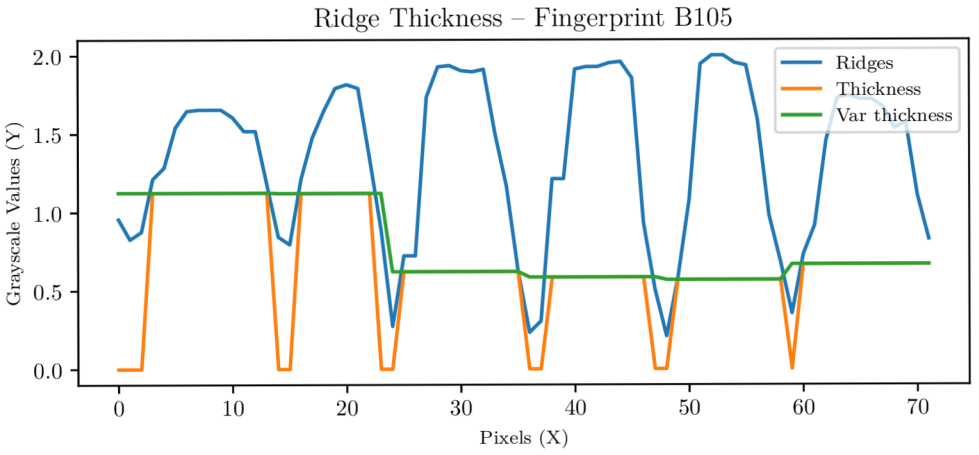


(c) Calculated score on grayscale fingerprint images with the centre of the fingerprint

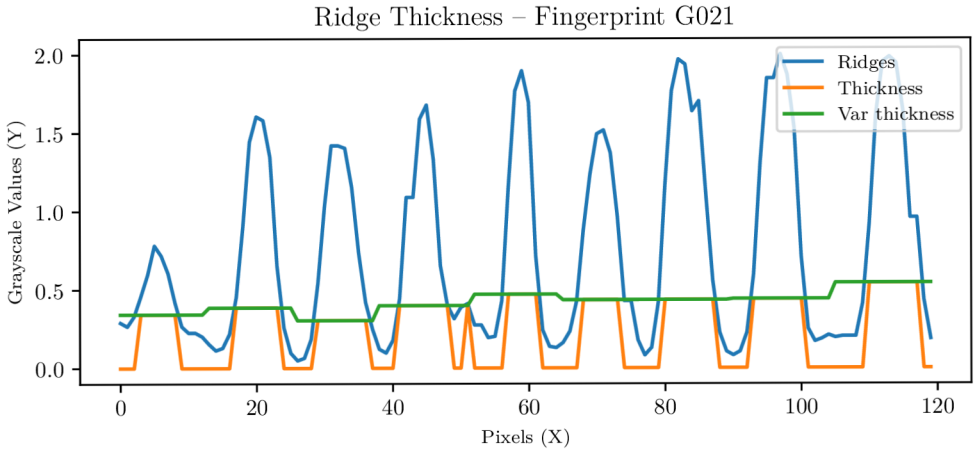


(d) Calculated score on grayscale fingerprint images with the stationary point

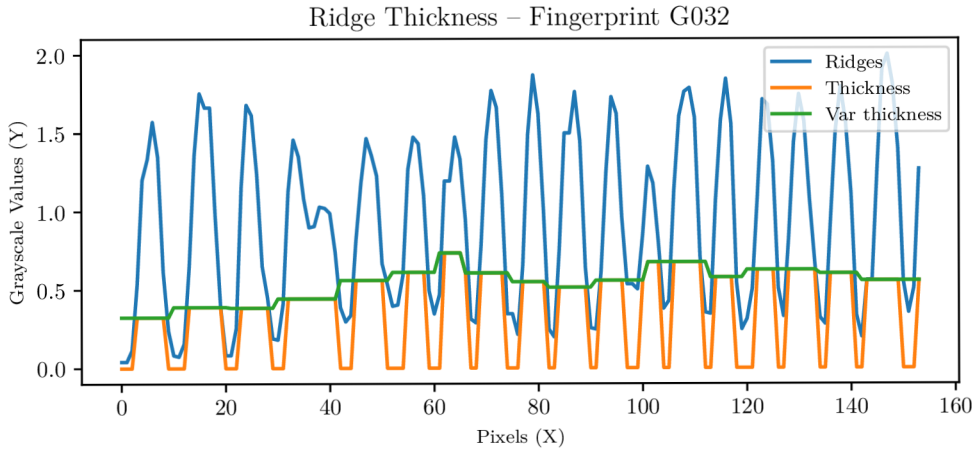
Figure 5.20: Comparison of NIST classification to the calculated Ridge Thickness in all four variants



(a) Fingerprint's B105 perpendicular line



(b) Fingerprint's G021 perpendicular line



(c) Fingerprint's G032 perpendicular line

Figure 5.21: Extracted perpendicular lines from the fingerprints with highlighted thickness in Figure 5.19 to compare the algorithm result

Chapter 6

Implementation

Implementation of this algorithm was written in Python, as many Python libraries have proven to be useful while implementing it. Also, when working with files and especially images, Python proves to be a straightforward language to approach similar data structures and problems and solve many difficulties that could happen in other languages. Writing the implementation in different languages, such as the C language, would be more time efficient. Still, the algorithms would not be easily understandable in code. There could be a lot of potential issues that could drastically slow the development of an algorithm that tries to prove the concept of the proposed algorithm.

The version of the Python language is 3.7 and was heavily determined by the Latent Automated Fingerprint Identification System from Michigan State University [10]. This algorithm was ported from Python version 2.7, and due to difficulties with TensorFlow, it was not moved to the newer version. Nevertheless, the structure was updated and the code needed to be rewritten by a significant part, so it could be possible to run on the newest Python version if the TensorFlow chart models were updated.

Used Libraries

When building the program that would incorporate the proposed algorithm, external libraries were used to support and accelerate the progress. Standard libraries were used, namely Numpy, Scipy, Skimage, OpenCV [3], Mathplotlib and TensorFlow. The Pandas package was used for the test script as it heavily simplifies the work with the JSON file and shows how the data could be used in other algorithms. Those libraries helped develop a robust environment and allowed experimenting with various versions of proposed algorithms with ease.

The proposed solution for fingerprint enhancement is based on the research done by Michigan State University designed by Anil K. Jain, Kai Cao, Dinh-Luan Nguyen, and Cori Tymoszek [10], who developed a latent automatic fingerprint identification system with minutiae point detection and an incredibly unique autoencoder. Their research had great results, and we decided to use the library for minutiae point extraction and fingerprint enhancement since many algorithms do not perform well on latent fingerprints. Their algorithm was developed as an end to end latent enhancer, analyser and matcher. Most of their code needed to be refactored and reconstructed, but the core functions were kept and applied to get the same results [9]. We created a fork of the project where the whole project is structured as an executable Python package[1]. Then this fork was restructured

[1] https://github.com/Lupphes/MSU-LatentAFIS

and simplified again to be more straightforward as it was used as a Python sub-package to develop the LatFigGra package. The original repository was not forked, but rather another fork from Manuel Aguado Martinez [30] was used. Martinez tried tried to port a significant part of the project and experimented with the Logarithmic Gabor filter [38], also used in the proposed algorithm. Nevertheless, the library he developed was still not usable in the proposed algorithm, and it had to be significantly changed as he used it for palm prints [2].

Structure

The program is written as a general Python package and can be used in any other algorithm as a dependency if imported. It has two main classes, The Fingerprint class, the object of the fingerprint image, which has saved all necessary variables and functions about the fingerprint and the Image object, the interface between the OpenCV library and the Fingerprint class. This approach significantly simplifies the usage of OpenCV library functions such as filters, kernels or simple bitwise operations on images. A fingerprint instance is supposed to be an image with all the necessary values parsed and stored. Most of the data in the Fingerprint instance is kept as Image instances, which are essentially NumPy arrays since all the filters and other functions are mainly inputs and outputs image – NumPy array. The Fingerprint object also contains all the necessary functions for launching the grading and enhancing algorithms. The Pipeline mentioned in the previous chapter describes the exact process of running the code.

The Fingerprint class contains four methods which the user can run from the Fingerprint instance. The function for fingerprint enhancement saves all the necessary data to the fingerprint image, the function for grading the fingerprints, and two functions that generate a report and final file with all received information.

The package also contains a string library and definition in separate files, which can be easily edited to extend the algorithm, and an exception-handling system was implemented as well.

6.1 Program Demo

To demonstrate the LatFigGra package, which incorporates the algorithm, we developed a demo that uses all the described processes to analyse given fingerprints. This demo takes a path to a provided folder where it expects to find fingerprint images and the PPI of the fingerprints specified, and it analyses them. It creates a new folder with an output file where all the information about given fingerprints is written. All the examples of filtered images are put into subdirectories for each fingerprint.

All fingerprint instances are saved as Python pickle files for later use, and therefore we do not need to regenerate the Fingerprint objects just for grading. Generally, only fingerprint instances can be created, and the grading process can be executed later if needed as the instance contains all necessary elements.

6.2 MSU Latent AFIS

Michigan State University Latent Automatic Fingerprint Identification System („MSU Latent AFIS“) is an algorithm for an end to end matching of rolled and latent fingerprints and a matching system for these pairs. They built a unique matching algorithm to find similarities between those pairs and identify a match in the database. This algorithm drew from

various previous researches [10, 33, 6, 32, 7] and built upon them. We took the essential part of this algorithm: the latent fingerprint enhancement and minutiae extraction. They also provided a simplified script for rolled fingerprints as well as for latent fingerprints. Still, we did not use this algorithm for grading the fingerprints as it should be uniform for all types of fingerprints, and there should not be a need to make the user select what kind of fingerprint they provided. The only difference between those algorithms is that the algorithm for exemplar fingerprints is simplified to save computation time. When the algorithm prepares the fingerprint for the autoencoder, it uses several filters and functions to prepare it for the process. One of those processes is the Cartoon+Texture image decomposition, used in this paper [10] and many others [28]. The Caroon+Texture decomposition is not the original method proposed with a low pass filter [4]. But instead, filtering in the frequency domain is used. Combining the process with the STFT and ridge structure dictionary removes almost all external elements from the image. Also, the algorithm requires TensorFlow models to function correctly, and they need to be in a specified folder by the user. Those models are graphs of the neural network used to identify minutiae points and the autoencoded image.

The whole code from previous research was restructured and made into a Python package and accessible everywhere in the code. Still, we kept the algorithm's vital parts the same, representing their excellent results.

Other Methods Tried

Before this research was found, the proposed algorithm used fingerprint enhancement that we implemented. There were numerous issues with this version that needed to be fixed. Only a few published researchers successfully developed latent fingerprints enhancer and described them in detail so we could build on their research and fix our problems. While searching for a solution to segmentation and enhancement of fingerprints, we found the Cartoon+Texture decomposition algorithm in numerous other researchers that used this algorithm as well. It was a great help, and with an algorithm developed by [28] and [13] we developed an enhancer with our implementation of the Cartoon+Textrure algorithm. Still, it was not as fast and good as the solution provided by [10], and that is why we used theirs. Nevertheless, the algorithm has proved to be an excellent solution for denoising and removing foreign elements.

We also tried to implement the segmentation algorithm based on the algorithm written here [13]. Still, it did not prove effective on unclear latent fingerprints after implementation, and other solutions were needed. The result of the segmentation created can be seen in Appendix C.

6.3 LatFigGra Package

The main package[2] (LatFigGra – an abbreviation of Latent Fingerprint Grader) uses the demo program and incorporates the proposed solution. The package creates an instance of the Fingerprint object and then executes the Grading process. The Grading process calculates and rates the fingerprints, stores images that demonstrate the results and writes all information in the JSON log file.

The Grading process is executed on the fingerprint instance. It supposes that all needed images are given and generated, and therefore it can start and launch all sub-algorithms in a sequence.

[2]The package will be publish here: https://github.com/Lupphes/LatentFingerprintGrader

Minutiae Point Grading

Since the MSU Latent AFIS package found and extracted all the minutiae points from the fingerprint and gave the final estimate with coordinates, our part of this algorithm was relatively straightforward. Based on that, we compared the number of received minutiae points with the number in the definition. Then it was decided if the number was enough to identify a person beyond a reasonable doubt.

Those final minutiae points could be listed inside the JSON log, but it would make the log unnecessary long since the algorithm could find about one hundred minutiae points on a perfect fingerprint [29]. Therefore, we decided to leave the final list in the folder with generated fingerprint images and include just the sum of those points. It is enough to deduce the quality of fingerprints from these numbers.

The library also contains definitions of intervals which define the fingerprint classifications. The classification also contains a verbal evaluation of the result. The definitions can be described as threshold values based on the research provided in [29]. It should predict the possible outcome of the trial of matching those fingerprints.

Contrast Grading

The contrast grading of the algorithm is the only statistical quality grading included. It grades the fingerprints based on the Michelson contrast, where it is looked for the intensities between ridges and valleys and the difference. The mentioned Formula 3.1 is used with the help of functions from the OpenCV library and answer on StackOverflow [20].

Overall measuring statistical values from the image is a simple approach to making conclusions about the fingerprint.

The Logarithmic Gabor filter and NumPy.ma module for masked arrays significantly help with masking and extraction for the ridge extraction. Everything else is done with simple bitwise operations. Extracted values are then processed with the formulas for RMSE 5.4 and Color difference 5.1, and the results are saved.

Sinusoidal Similarity Grading

The algorithm uses a perpendicular line, the centre of the fingerprint's mask and the stationary point determined by the ridge count. It will extract the line from the CLAHE grayscale and autoencoded image, splits it to differentiate ridges, apply a lowpass filter and calculate the max and minimal extremes. A similar process also applies to the algorithm in the next Section 6.3. The lowpass filter is used to split the ridges and calculate the sinusoidal similarity. The line is used in its initial state, just segmented as the line processed with a lowpass filter would. The lowpass filter uses the implementation done by [11], which has proven to be working well.

Initially, the line was not split by ridges, and we tried to align the sinusoidal wave with the entire ridge signal. This process did not yield good results, as can be seen in Appendix B.1, where the signal matched the line partially, but it did not prove to be working. The processes of segmenting the ridges, analysing each and then averaging those results by the Formula 3.3 have better results, and that is why they were used. We used the square difference for the alignment, which will return the sum, and the one with the highest results is used.

Ridge Thickness Grading

Similarly to the sinusoidal script, the algorithm uses a perpendicular line, the centre of the fingerprint's mask and the stationary point determined. The process is the same, and the line is parsed with the same process.

The algorithm estimates the thickness by calculating the number of pixels and then using the formula to compute the result according to the Formula 3.2. Again, similarly to the sinusoidal process, initially, the algorithm was calculated for the whole extracted line with non-variable height in 18%, but as shown in Appendix B.2. The segmentation with variable height calculated for each ridge has proven to be working better than the initial proposed solution.

6.4 Test Script

We designed another script for generating the graphs, which uses the output of the JSON file generated by the proposed algorithm. The Test script processed the data with the Pandas package and converted them into a Pandas dataframe. The script suites as an example of how the data are parsed and can be used in another algorithm. Graphs generated in Section 5.5 are developed with the help of this algorithm. It first converts the JSON file to Pandas dataframe, essentially a table, and then sorts and evaluates the data. Graphs are generated with the mathplotlib package.

Chapter 7

Conclusion

We present an algorithmic solution for evaluating the quality of dactyloscopy traces based on statistical and contextual values of an image. The proposed algorithm has sub-algorithms which evaluate different fingerprint features based on various factors and grades and classify them according. Those sub-algorithms have been used for fingerprint grading before and are modified, or they consist of new ideas that we tried and implemented.

The solution also consists of all necessary steps needed before the rating of the fingerprint can start, which is the enhancement of the fingerprint image and extraction of features required by all defined sub-algorithms.

The sub-algorithms successfully grade the fingerprints based on their definitions, and most of them have similar results to a presented SD27 NIST dataset, which divides fingerprints into three categories. The sub-algorithms where the results from NIST classification and their classification relate are those that grade fingerprints by minutiae points, number of ridges and RMSE contrast. The sub-algorithms that did not perform similarly to the NIST database were either grading different elements of the fingerprint image or features that the NIST database supposedly did not consider. Still, they have proven to be working, and they divide fingerprints by their measures into categories denoted by the computed result.

The new algorithms for grading contrast presented in the proposed solution successfully divided fingerprints, and we could conclude the quality of fingerprints from the presented results. The algorithm could divide fingerprints by their overall brightness, potentially identifying a false match made by a neural network. However, further research is needed to establish a set interval of what contrast result is sufficient to decide the fingerprint's condition. This can be precisely achieved by analysing the results of this algorithm with the more extensive dataset to evaluate the class of the given dactyloscopy trace.

This solution also presents an algorithmic solution for earlier presented algorithms that grade fingerprints based on papillary crosscuts and calculate the sinusoidal similarity and thickness of ridges. Those algorithms were originally proposed theoretically, and they were not implemented in an algorithmic solution. The proposed solution consists of this implementation with features that improved and automatised the process. Both algorithms could be improved by not using the centre of the fingerprint or calculated stationary point, but a calculated core of the fingerprint, which would increase the number of ridges that are analysed.

Bibliography

[1] *The OpenCV Reference Manual – Histogram Equalization*. 4.5.5th ed. OpenCV, April 2022.

[2] Aguado Martínez, M., Hernández Palancar, J., Castillo Rosado, K., Cupull Gómez, R., Kauba, C. et al. Document scanners for minutiae-based palmprint recognition: a feasibility study. *Pattern Analysis and Applications*. may 2021, vol. 24, p. 1–14. DOI: 10.1007/s10044-020-00923-3.

[3] Bradski, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*. 2000.

[4] Buades, A., Le, T., Morel, J.-M. and Vese, L. Cartoon+Texture Image Decomposition. *Image Processing On Line*. 2011, vol. 1, p. 200–207. https://doi.org/10.5201/ipol.2011.blmv_ct.

[5] Buades, A., Le, T. M., Morel, J.-M. and Vese, L. A. Fast Cartoon + Texture Image Filters. *IEEE transactions on image processing*. New York, NY: IEEE. 2010, vol. 19, no. 8, p. 1978–1986. ISSN 1057-7149.

[6] Cao, K. and Jain, A. K. Latent Fingerprint Recognition: Role of Texture Template. *ArXiv preprint arXiv:1804.10337*. 2018.

[7] Cao, K. and Jain, A. K. Automated latent fingerprint recognition. *IEEE transactions on pattern analysis and machine intelligence*. IEEE. 2019, vol. 41, no. 4, p. 788–800.

[8] Cao, K., Liu, E. and Jain, A. K. Segmentation and enhancement of latent fingerprints: A coarse to fine ridge structure dictionary. *IEEE transactions on pattern analysis and machine intelligence*. LOS ALAMITOS: IEEE COMPUTER SOC. 2014, vol. 36, no. 9, p. 1847–1859. ISSN 0162-8828.

[9] Cao, K., Nguyen, D.-L., Tymoszek, C. and Jain, A. *MSU Latent Automatic Fingerprint Identification System (AFIS)*. GitHub, 2022. Available at: https://github.com/prip-lab/MSU-LatentAFIS.

[10] Cao, K., Nguyen, D.-L., Tymoszek, C. and Jain, A. K. End-to-End Latent Fingerprint Search. *IEEE transactions on information forensics and security*. IEEE. 2020, vol. 15, p. 880–894. ISSN 1556-6013.

[11] Chevalier, G. *Filtering signal with a butterworth low-pass filter and plotting the STFT of it with a Hanning window and then the Laplace transform.* GitHub, 2018. Available at: https://github.com/guillaume-chevalier/filtering-stft-and-laplace-transform.

[12] Chikkerur, S., Cartwright, A. N. and Govindaraju, V. Fingerprint enhancement using STFT analysis. *Pattern recognition*. OXFORD: Elsevier Ltd. 2007, vol. 40, no. 1, p. 198–211. ISSN 0031-3203.

[13] Choi, H., Boaventura, M., Boaventura, I. A. G. and Jain, A. K. Automatic segmentation of latent fingerprints. In: *2012 IEEE Fifth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*. 2012, p. 303–310. DOI: 10.1109/BTAS.2012.6374593.

[14] Drahanský, M. *Methods for Quality Determination of Papillary Lines in Fingerprints*. Available at: https://www.nist.gov/system/files/documents/2016/12/07/drahansky_nist_bqw2007_presentation.pdf.

[15] Drahanský, M. Sinusoidal Shape of a Papillary Line Crosscut. In: *2007 ECSIS Symposium on Bio-inspired, Learning, and Intelligent Systems for Security (BLISS 2007)*. IEEE, 2007, p. 19–24. DOI: 10.1109/BLISS.2007.24. ISBN 9780769529196.

[16] Drahanský, M. *Fingerprint Recognition Technology - Related Topics*. First editionth ed. Lambert Academic Publishing, 2011. 172 p. ISBN 978-3-8443-3007-6. Available at: https://www.fit.vut.cz/research/publication/9636.

[17] Drahanský, M., Orság, F., Doležel, M. and al. et. *Biometrie*. 1. vydáníth ed. Computer Press, s.r.o, 2011. 294 p. ISBN 978-80-254-8979-6. Available at: https://www.fit.vut.cz/research/publication/9468.

[18] Drahanský, M. *Hand-based biometrics : methods and technology*. First publishedth ed. London: The Institution of Engineering and Technology, 2018. IET book series in advanced biometrics. ISBN 978-1-78561-224-4.

[19] Drahanský, M., Brezinova, E., Hejtmankova, D. and Orság, F. Fingerprint Recognition Influenced by Skin Diseases. december 2010, vol. 2.

[20] fmw42. *How extract contrast level of a photo - OpenCV*. StackOverflow, 2019. (https://stackoverflow.com/users/7355741/fmw42). Available at: https://stackoverflow.com/a/57260564/8267536.

[21] Ganchimeg, G. and Leopold., H. Fingerprint image enhancement using filtering techniques. *International Journal of Advanced Research*. 2019, vol. 7, no. 5, p. 637–645. DOI: 10.21474/ijar01/9084.

[22] Garg, R. K., Kumari, H. and Kaur, R. A new technique for visualization of latent fingerprints on various surfaces using powder from turmeric: A rhizomatous herbaceous plant (Curcuma longa). *Egyptian Journal of Forensic Sciences*. 2011, vol. 1, no. 1, p. 53–57. DOI: https://doi.org/10.1016/j.ejfs.2011.04.011. ISSN 2090-536X. Available at: https://www.sciencedirect.com/science/article/pii/S2090536X11000141.

[23] Garris, M. *Latent Fingerprint Training With NIST Special Database 27 and Universal Latent Workstation*. NIST Interagency/Internal Report (NISTIR), National Institute of Standards and Technology, Gaithersburg, MD, 2001-09-01 2001. Available at: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=151537.

[24] Hanoon, M. F. *Contrast Fingerprint Enhancement Based on Histogram Equalization Followed By Bit Reduction of Vector Quantization*. May 2011. Available at: http://paper.ijcsns.org/07_book/201105/20110517.pdf.

[25] Henry, .-. *Classification and uses of finger prints*. England: George Routledge and Sons, 1900, 1900.

[26] Journals, I., K, S. P. and B, N. *Fingerprint Image Enhancement Using STFT Analysis*. Figshare, 2015.

[27] Li, S. Z. and Jain, A. K. *Encyclopedia of Biometrics*. New York, NY: Springer, 2009. 87–87, 168–172, 474–482 p. ISBN 9780387730028.

[28] Liu, M., Liu, S. and Yan, W. Latent Fingerprint Segmentation Based on Ridge Density and Orientation Consistency. *Security and Communication Networks*. Hindawi. May 2018, vol. 2018, p. 4529652. DOI: 10.1155/2018/4529652. ISSN 1939-0114. Available at: https://doi.org/10.1155/2018/4529652.

[29] Maltoni, D., Maio, D., Jain, A. K. and Prabhakar, S. *Handbook of Fingerprint Recognition*. Second editionth ed. London: Springer London, 2009. ISBN 1848822537.

[30] Martinez, M. A. *MSU Latent Automatic Fingerprint Identification System (AFIS) – Logarithmic Gabor filter fork*. GitHub, 2021. Available at: https://github.com/manuelaguadomtz/MSU-LatentAFIS/tree/dev.

[31] Morse, B. S. *Lecture 13: Edge Detection*. Feb 2000. Available at: https://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MORSE/edges.pdf.

[32] Nguyen, D.-L., Cao, K. and Jain, A. K. Robust Minutiae Extractor: Integrating Deep Networks and Fingerprint Domain Knowledge. 2017.

[33] Nguyen, D.-L., Cao, K. and Jain, A. K. Automatic Latent Fingerprint Segmentation. 2018.

[34] Orczyk, T. and Wieclaw, L. Fingerprint ridges frequency. In: *2011 Third World Congress on Nature and Biologically Inspired Computing*. IEEE, 2011, p. 558–561. DOI: 10.1109/NaBIC.2011.6089649. ISBN 9781457711220.

[35] Porwik, P. A new fingerprint ridges frequency determination method. *Ieice Electronic Express*. february 2009, vol. 6, p. 154–160. DOI: 10.1587/elex.6.154.

[36] Saquib, Z., Soni, S. K. and Vig, R. Sweat pores-based (level 3) novel fingerprint quality estimation. In: *2010 3rd International Conference on Computer Science and Information Technology*. IEEE, 2010, vol. 8, p. 525–531. ISBN 9781424455379.

[37] Sari, S., Qalbiah, U. and Putri, I. Comparison between Latent Fingerprint Identification using Black Powder and Cyanoacrylate Glue. *Asian Journal of Chemistry*. january 2018, vol. 30, p. 2615–2620. DOI: 10.14233/ajchem.2018.21378.

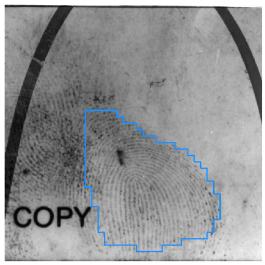
[38] Wang, W., Li, J., Huang, F. and Feng, H. Design and implementation of Log-Gabor filter in fingerprint image enhancement. *Pattern Recognition Letters*. 2008, vol. 29, no. 3, p. 301–308. DOI: https://doi.org/10.1016/j.patrec.2007.10.004. ISSN 0167-8655. Available at: https://www.sciencedirect.com/science/article/pii/S016786550700325X.

[39] Wieclaw, L. Gradient based fingerprint orientation field estimation. *Journal of Medical Informatics and Technologies*. 2013, vol. 22.

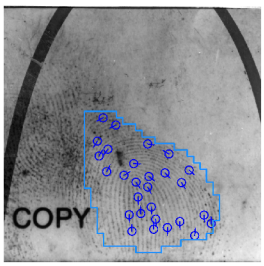
Appendix A

Summary of the Fingerprint Analysis

As an overview we outlined few pictures generated by the algorithm to demonstrate results.



(a) Source fingerprint image with a highlighted mask

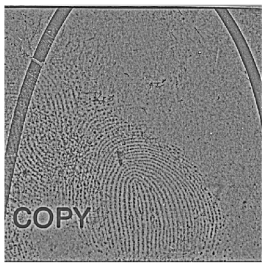


(b) Extracted Minutiae Points

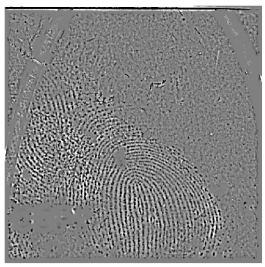
(c) Generated image with the autoencoder

(d) Removed ridges from the grayscaled image

(e) Removed valleys from the grayscaled image

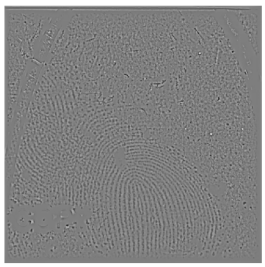


(f) Gaussian Filter



(g) Texture from image decomposition

(h) Logarithmic Gabor Filter applied on Texture image



(i) STFT applied on the Texture image

Figure A.1: Fingerprint G100 from the NIST SD27 dataset. Showcase of most filters and variants.

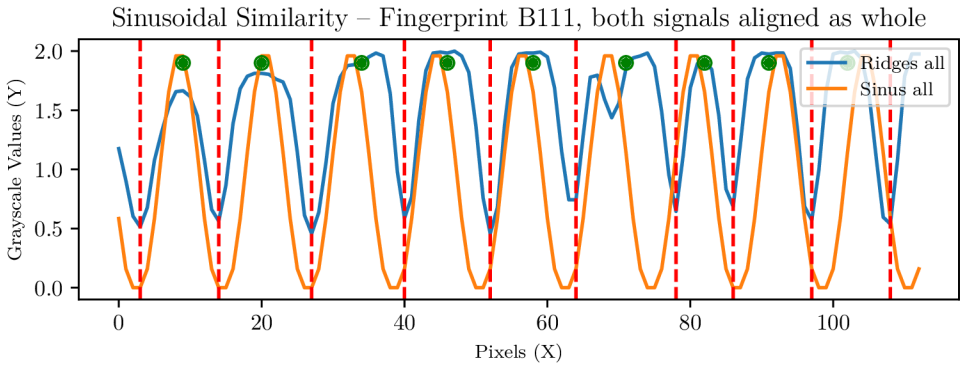
Appendix B

Results of Rejected Grading Algorithms

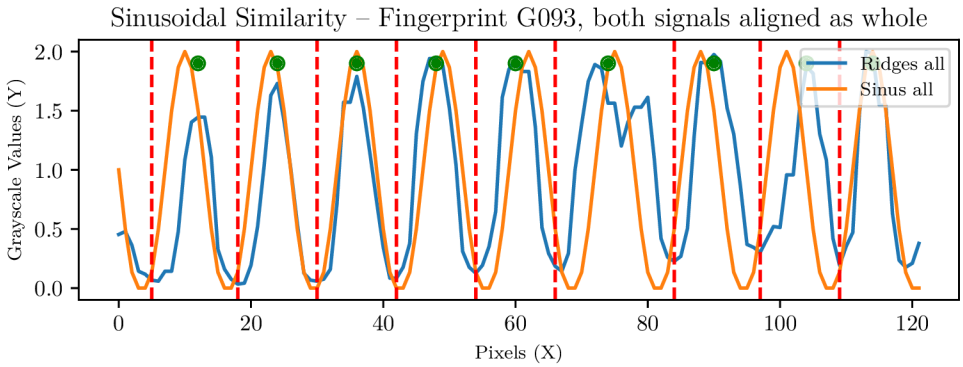
When we were experimenting with various algorithms, not each was successful. We outlined two algorithms that did not yield correct results and their reasons. These algorithms were counted on the same fingerprints, which were used in result sections on Figure 5.17 and 5.21, and they are named accordingly.

B.1 Sinusoidal Similarity

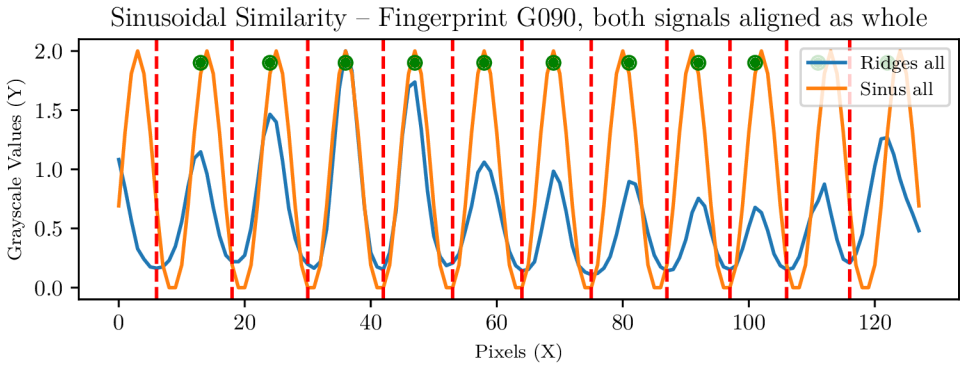
Initially, the algorithm was designed to align sinusoidal waves and extracted lines as a whole, but this approach had several issues. It can be seen from the graph that a few ridges are entirely misaligned to the sinusoidal wave, which enlarges the error. The dashed line divides lines into ridges by local extremes, and the green dots count the number of ridges by local maxima.



(a) Fingerprint's B111 perpendicular line



(b) Fingerprint's G093 perpendicular line

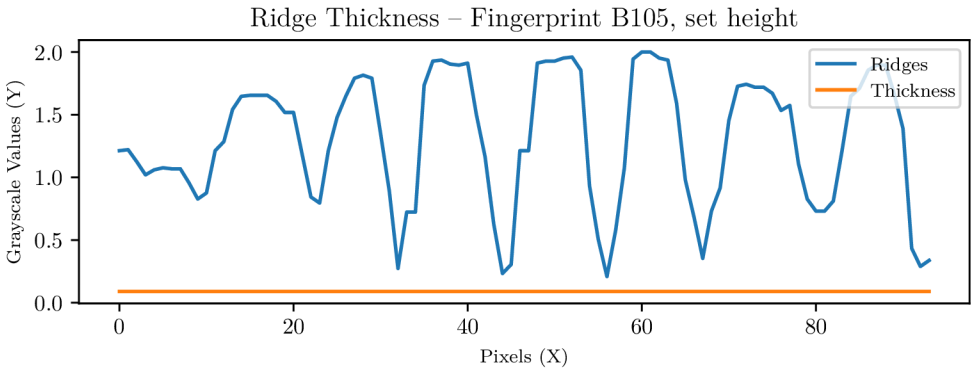


(c) Fingerprint's G090 perpendicular line

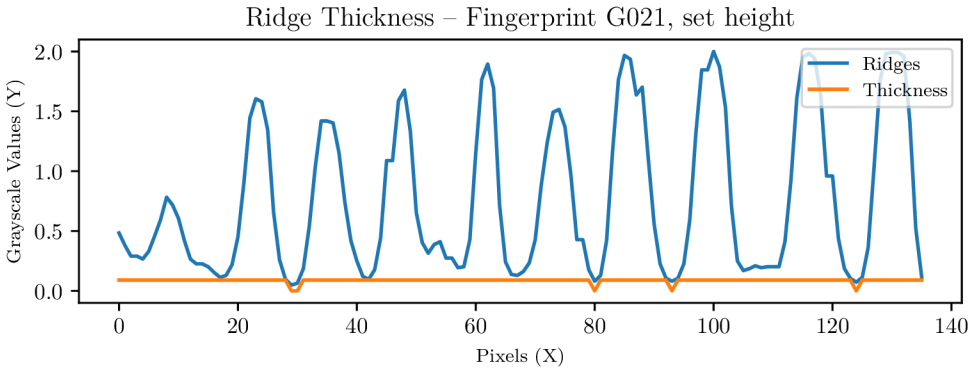
Figure B.0: The sinusoidal similarity with the extracted line aligned as a whole signal and not by individual ridges. The alignment does not have correct results as the signals are not aligned perfectly.

B.2 Ridge Thickness

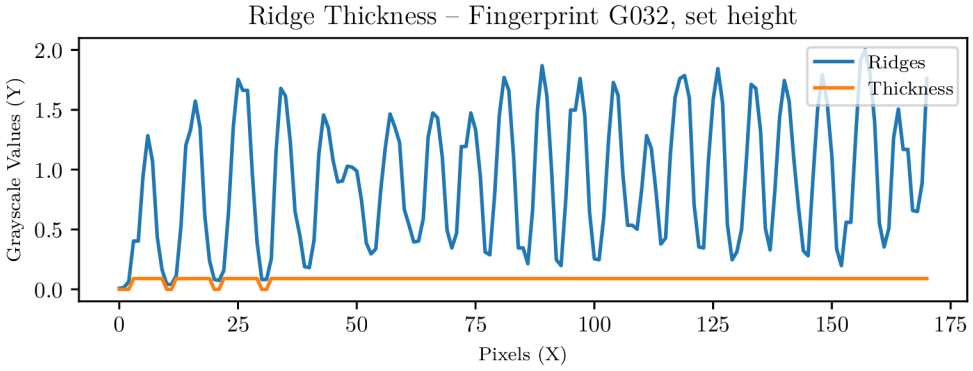
This algorithm was designed to count the pixel length of ridges. The variable height can be assumed on a perfect fingerprint but not on actual fingerprints, especially latent fingerprints. Initially, the height was 18% (45.9), which did not yield correct results.



(d) Fingerprint's B105 perpendicular line



(e) Fingerprint's G021 perpendicular line



(f) Fingerprint's G032 perpendicular line

Figure B.0: The ridge thickness with the extracted lines without variable height. The estimation method does not count almost any ridges for any selected perpendicular lines.

Appendix C

Results of Rejected Preprocessing Algorithms

Before the MSU Latent AFIS proposed solution from [10] was used, we tried to use a proposed solution in [13]. Still, it did not yield satisfactory results with the orientation feature. Therefore we moved further to look forward to another solution. It is possible that the wrong implementation or misunderstanding of some algorithms caused those bad results. However, the MSU Latent AFIS works better in our use case.

(a) Source fingerprint with applied Sobel filter (b) Generated mask from the algorithm

Figure C.1: The best-obtained result was by algorithms before the MSU Latent AFIS was used. This mask would be helpful if we worked with perfect fingerprints since it did not detect anything if we introduced latent fingerprints. Used algorithm was based on research [13]. The shown fingerprint was taken from [36].

Appendix D

Contents of the SDHC Card

```
├── LatentFingerprintGrader-2.0.3
│   ├── CHANGELOG.txt
│   ├── demo.py
│   ├── latfiggra
│   │   ├── contrast_types.py
│   │   ├── definitions.py
│   │   ├── exception.py
│   │   ├── fingerprint.py
│   │   ├── image.py
│   │   ├── __init__.py
│   │   ├── __main__.py
│   │   ├── msu_latentafis
│   │   ├── report.py
│   │   └── string_database.py
│   ├── LICENSE
│   ├── MANIFEST.in
│   ├── models
│   ├── notes.txt
│   ├── out
│   ├── README.md
│   ├── requirements.txt
│   ├── SD27-lat
│   ├── setup.py
│   └── test.py
├── LatentFingerprintGrader-2.0.3.zip
├── models.zip
├── out.zip
├── README.md
├── SD27-lat.zip
├── thesis.pdf
└── Thesis-text.zip
```

I would like to outline a few key files and folders, which would be described thoroughly:

- LatentFingerprintGrader-2.0.3 – extracted folder, which contains full program prepared for execution
  - demo.py – the demonstration file which enhances and grades the fingerprints
  - test.py – the test file, which generates charts from the JSON file
  - models – directory, where TensorFlow models are stored
  - out – folder, which contains generated result
  - requirements.txt – file, which contains requirements that need to be installed before the demo.py or test.py is ran
  - SD27-lat – folder, which contains the source fingerprints that are processed
- LatentFingerprintGrader-2.0.3.zip – zip file, which contains the source code of the proposed solution
- models.zip – folder, which contains models needed by TensorFlow
- README.md – file, which describes the process of installing and launching the program
- thesis.pdf – pdf file, which is the actual text of theses
- Thesis-text.zip – zip file, which contains the thesis source files
- out.zip – zip file, which contains the calculated results from the demo.py

I also calculated MD5 hashes for each zip file. All MD5 hashes are available at https://bachelor.lupp.es:

- LatentFingerprintGrader-2.0.3.zip – d9cd17d959cdea1fa5bfa0b4ace14a8b
- SD27-lat.zip – a433adafd9cceb38253c410352c15586
- models.zip – 487077a27e4699c0e42821a95fb7282f
- README.md – badf35a7aa19ee6ff030c8ff80d58b2c
- out.zip – 84be23d3af6a3b77d290c50096b5b53d