Design, realization and evaluation of a side illumination finger vein patter capturing device

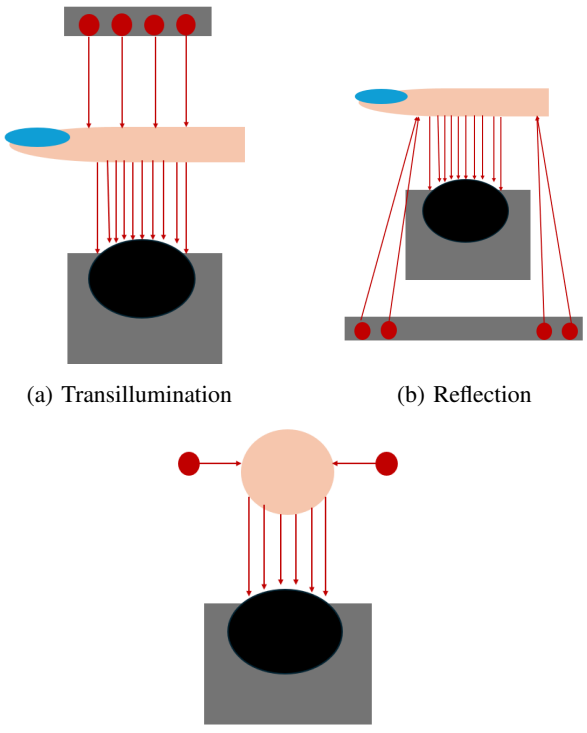
Jana Klaric s2850478

***Abstract*—The discovery that finger veins patterns can be seen as dark lines when photographed under near-infrared (NIR) light opened a door towards a new subfield of biometrics. These unique and commonly obtainable vein patterns provided an innovative solution to enhancing security systems of high-risk environments. Ideally, the finger would get illuminated from the sides to allow for open back design rather than from the top as in most existing devices. However, this optimal NIR light direction is hardly ever implemented due to the challenges it brings with it. Overexposure is most commonly seen in side illumination setups where it affects the image quality by causing the finger edges to appear overly bright. This research describes the process of designing a side illumination based finger vein pattern capturing device from the ground up by combining 3D and PCB design. The experiments also dive into the topic of identifying the best overexposure mitigation technique out of the several proposed in the related work. The device's performance was evaluated based on a small dataset collected with it. The setup was proven to provide images of similar quality as that of the University of Twente's design which operates on the principle of top transillumination.**

I. Introduction

Biometric properties of people have been used as a way of authentication since the late 19th century [1]. However, their application has drastically spread from being used only in official environments to being integrated in our everyday lives in the last two decades. Most of us have a fingerprint sensors on our phones and some newer models additionally authenticate the user with face recognition methods. Even though these two methods satisfy the basic needs of the general public, it cannot be denied that they are not without flaw. Therefore, more secure and robust methods of biometric authentication get implemented in more official and high-risk environments. Iris authentication is one such method while finger vein authentication represents a lesser-known alternative.

Finger vein recognition is a relatively new method of authentication that has proven to be more secure than the already well-known fingerprint recognition. While it is true that using fingerprints as a built-in personal key to access private or sensitive information was a revolutionary idea, these can be counterfeited or replicated using the right tools. Kraken security Labs, founded in San Francisco, California, have proven that fingerprint sensors can be fooled by a commonly available household item; glue [2]. The internal vein pattern of an individual's finger, however, cannot be easily replicated or obtained through methods such as glue use. The supposition that the arrangement of blood vessels inside human bodies is unique dates as far back as 16th century [3] but in the early 20th century it was concluded that the visible light does not show the differences between the vessels clearly enough to be used in authentication. The near infrared (NIR) light, on the other hand, was discovered to be useful in this area. The protein Hemoglobin in blood is known to absorb NIR light. This results in the images of fingers illuminated by this light to have dark paths (veins that have absorbed the NIR light) on a bright background. Depending on if the vessels are carrying oxygenated (arteries) or deoxygenated (veins) Hemoglobin they will absorb different wavelengths of the NIR light. Around 800 nm both kinds of Hemoglobin absorb the light equally [3]. Above 800 nm oxygenated Hemoglobin absorbs stronger and at 760 nm deoxygenated Hemoglobin does. Therefore, this authentication method can be called finger vein, finger vessel or finger arteries authentication based on the used wavelength.



(a) Transillumination (b) Reflection

(c) Side illumination - front view

Figure 1: Three main lighting techniques

When it comes to the recognition toolchains, most consist of similar stages such as image acquisition, image quality assessment, preprocessing, region of interest determination, feature extraction and biometric comparison [3]. The stages of greatest interest for this research are the first two as they deal with achieving a decent image quality while implementing one specific illumination direction of the finger. There are three main types of illumination that are

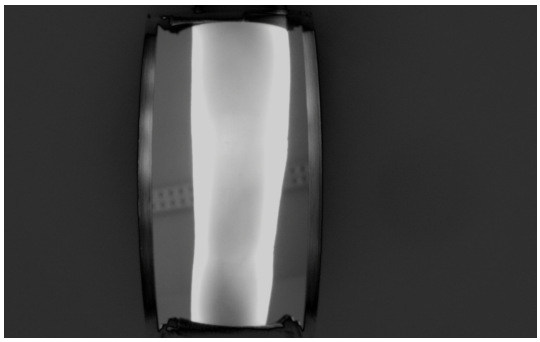


Figure 2: Overexposure

shown in Figure 1:

- Transillumination - light source and the camera are positioned on different sides of the hand (dorsal - knuckle side or palmar - palm side)
- Reflection - light source and the camera positioned are at the same side of the hand (dorsal or palmar)
- Side illumination - the light source is positioned on one or both sides of the finger and the camera is positioned beneath the finger

[3]

The University of Twente has already made remarkable progress in the finger vein authentication area by developing the University of Twente Finger Vein Pattern (UTFVP) device, which operates on the basis of transillumination from the top [4]. However, this device featured a closed design since the light source was envisioned to levitate above the hand. The question arises whether a similar yet open back design could be achieved by changing the direction of finger illumination. While it is true that repositioning the light source would lead to a more compact design, it would also introduce challenges such as overexposure of the side of the finger that is being lit. The light source inevitably diverges, meaning that not all light beams are guaranteed to hit the bone and from there illuminate the veins. A portion of light will hit the surrounding tissue and possibly get reflected from it into the camera resulting in a bright border of the finger. The more the light diverges, the greater the area of overexposure will be and the larger the risk is of not being able to capture the veins in that region (Figure 2).

Even though a part of the finger veins are visible in presence of overexposure, a negative bias would be presented in the results. The less complex of a pattern the comparison method takes as the input, the worse it will perform in spotting the differences between users.

Therefore, the main research questions are:

1) ”Is it possible to design a finger vein image acquisition device with incorporated side illumination which could measure up to the quality of pictures acquired by UTFVP while overcoming challenges of overexposure?”
2) ”Which kind of overexposure mitigation allows for the best image quality in the designed setup?”

II. Related work

In order to start making main design choices, the prior research showcases have to be explored and analysed. Hence, previous research on the topic of side illumination as well as on other building blocks relevant to the research are discussed here. The section is organized to follow the process of designing the device from the ground up.

A. Finger vein acquisition

The finger vein image acquisition has to be performed by a camera which satisfies two conditions: IR sensitivity and compatibility with a Raspberry pi 4 since it is intended to be used in the acquisition process. Research in [5] provides camera evaluation results after comparing images captured by 4 kinds of cameras. It was concluded that the monochrome camera OV9281 provided the best images while satisfying the two conditions which made it the first choice for this research.

B. Role of illumination

The positioning of the illumination source and the width of its light beams have a great impact on the image quality and, therefore, the entire system’s performance.The three primary light source configurations for finger vein authentication are:

1) Transillumination technique: The camera taking a picture of the finger and the light source are placed on the opposite sides of the hand. Since it is full of small cavities, the finger bone acts as a secondary light source and it diffracts the incoming light beam into many more which end up illuminating the veins. A downside of using this technique is that the device is often times quite tall as there has to be enough space between the light source and the finger and the finger and the camera. The research presented in [4] incorporates illumination from the top and the height of the product is successfully reduced through the use of a mirror. This allowed for the camera to be placed vertically rather than horizontally as the NIR light beams reflected off the mirror into the camera.

2) Reflection technique: This technique is characterized by the camera and the light source being positioned on the same side of the hand (palmar or dorsal side) [3]. As the light shines on the finger it is reflected from it before being captured by the camera. While this technique allows for the open back design (the finger is not completely enclosed in the box), it is susceptible to limited tissue penetration which can result in weak contrast between the veins and surrounding tissue. Partial image overexposure is also a problem relating to this technique as the light need not reflect equally off of every point of the finger [6].

3) Side illumination: Research on this topic is scarce as its implementation is accompanied by various challenges such as overexposure of finger edges. However, in paper [7] it was successfully shown that the light beam’s width is capable of reducing overexposure. Here, the bone yet again acts as a secondary light source, thereby indicating the possibility of directing a narrow beam of light into the bone and letting it illuminate the veins on the other side of said bone. The difference between using wide and narrow beams in side illumination is shown in section 3. of paper [7].

4) Beam width: Papers [6] and [7] address the challenge of producing a narrow light beam needed to minimize overexposure of the finger's edges while using side illumination. While [6] focuses more on proposing different solutions for obtaining narrow beams, [7] discusses the optimal combination of illumination directions. Some of the proposed solutions discussed in [6] are:

- Narrow beam LEDs - available NIR LEDs with a narrow transmission angle; the author suggests SFH4550 model
- Optical fibre bundle - using optical fibres to concentrate the light's direction
- Reflective tube - use of tubes coated in reflective material
- Tube guides - straight light guides
- Tapered guides - tubes which take the light in and become progressively narrower as they approach the finger

The outcome of the results have shown that the tapered guides recorded the best (the narrowest) radiation pattern. While [6] bases its results on radiation patterns, paper [7] shows what the image of the vein pattern looks like when illuminated with and without the use of pipe covers (these resemble the tube guides from [6]). The difference in overexposure from both the top and side illumination is significant when the use of narrow and wide beams is compared.

C. Finger vein recognition steps

After the images are acquired. the evaluation of the their quality is conducted. The vein pattern recognition is a complicated and involved process whose substeps are described in [3]:

1) Preprocessing: This step is comprised of different enhancement techniques as well as the image normalization. Its role in the process is to cope with the ever-changing acquisition conditions, noise, blur, poor contrast etc. [3]. Research in [4] introduces the preprocessing technique of histogram equalization applied in MATLAB. After the image is preprocessed, the contrasts between the black traces and white background becomes accentuated which later supports the clarification of the vain paths.

2) Region of Interest Determination: It is important to note that finger vein images inevitably include both the finger and its surrounding background. In usual palm vein recognition systems, the central rectangular area of the image is extracted to shorten the computation time of feature extraction algorithms. In [4] edges are determined by applying Lee's method [4] and later filling in the area between the edges with the finger contents of the image.

3) Feature extraction: After detecting the ROI, an algorithm is applied to the image in order to extract the vein patterns. According to the overview given in [3], there are two main approaches. The first one involves extracting and using skeletonised versions of the vein structures, binary representations or vein minutae. The second one relies on extracting discriminating features from the images which often requires an involvement of deep learning-based techniques. As the first approach is significantly easier to implement than the second one in given time, it was researched in more depth.

The mentioned approach includes various algorithms for finger vein patterns extraction. Some of them are Normalized cross-correlation, Maximum curvature, Repeated line tracking, Principal curvature and Wide line detector [4]. The dataset collected from [4] was tested with all of the mentioned algorithms and it was found that the Maximum curvature provided the EER (Equal Error Rate) score than the others which makes it an interesting candidate for the evaluation method used in this research. EER is an evaluation parameter which depicts the operating point at which the false acceptance rate is equal to the false rejection rate. It is measured in percentage and can more easily be explained as the error rate of the system.

Even though it is not expected to perform as well as Maximum curvature due to the presented results of [4], the Repeated line tracking algorithm adds an interesting subgoal to the research of investigating how much worse it performs and why. However, before integrating one or both methods into this project, one has to familiarize themselves with how they both function.

The Maximum curvature algorithm is based on calculating curvature behaviors of finger veins to locate the patterns in an image. It is split into two steps: extraction of positions of the vein centers and connecting the extracted centers into a vein. The first step involves finding the center points of veins by looking at the cross sectional profiles of finger vein images [8]. These profiles look like dents due to the fact that a vein is darker than the tissue which surrounds it. On an image, this shows as a dark line whose edges gradually get lighter until they achieve the color of the background. Hence, the vein centers can be found by calculating local maximum curvatures in the vein profiles [8]. The relationship between the cross sectional profile, curvature and the calculated probability score that the current center is indeed a vein center can be seen in Figure 3. The way that veins centers appear as dents on the cross sectional image profile is visible in the top graph while the calculated curvature of each found center can be seen in the middle graph. Finally, the probability scores of the spectated point being a center are presented in the lowest graph. Where the profile experiences dips, the curvature and hence the score experience a peak. It is important to state that the Figure 3 shows the relations of these three graphs for one cross sectional profile and the full algorithm is based on the idea of computing the same for each profile in the image. The matter of connecting the centers is simply based on checking the pixels which surround the determined center and establishing if they are dark enough to be considered another center. In case that the surrounding pixel is determined to also be a vein center, a line is drawn between them and the method repeats for the next center.

Unlike Maximum curvature which builds patterns from detecting the curvature behaviors of the dents, Repeated line tracking algorithm starts at a random pixel in the image and tracks the slopes of the dents. The algorithm checks the pixels surrounding the starting point and tries to find one darker. Once the darkest pixel is found, it is established as a point where the line tracking (vein building) starts just as in Maximum curvature [9].

After the vein patterns are extracted by the use of one of the algorithms, the vein pattern is binarized by comparing the image's pixels to a threshold. If the value of a pixel is

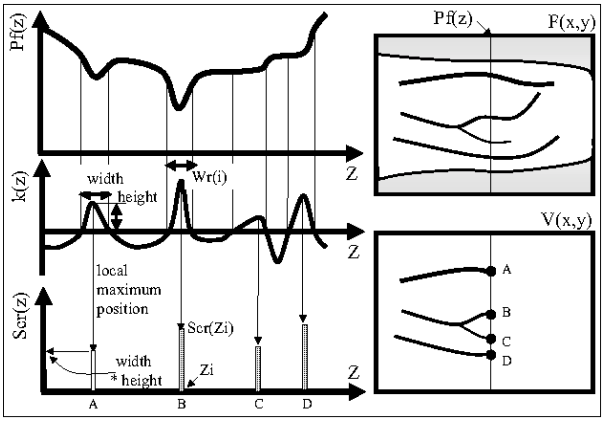


Figure 3: The relation between the cross sectional profile, the calculated curvature and the probability score [8]

Figure 4: ZKTEco side illumination device capture

higher than a threshold is assigned to the vein region and the ones with a lower value than the threshold are assigned to the background [8].

The binary template of an image is then compared both with the binary templates of its mated images and those of its non-mated images and their similarity is given out in terms of a cross-correlation score that ranges between 0 and 0.5.

4) Performance evaluation: After building the device, capturing images and running them through the algorithms, the results should get compared to some sort of a standard in order to investigate the quality of the device. The device that the side illumination setup will get compared to is the University of Twente Finger Vein Pattern (UTFVP) device.

In 2020., the UTFVP has shown impressive results when compared to some of the then leading published datasets [4]. In order to conduct the comparison the researchers first collected a dataset from 60 volunteers. A preprocessing step which increased the contrast between the veins and the surrounding tissue was incorporated by the use of the Adaptive Histogram Equalisation in MATLAB. A binary mask was used to detect the edges of the finger and the rotation and translation were estimated based on the detected edges. After the rotation and translation were applied to the image, its binary mask was compared to the dataset. The researchers computed the EER (Equal Error Rate) of 5 algorithms for their dataset and the, then leading, Peking University where it was proven that the UTFVP dataset provided better results. The results acquired by the Maximum Curvature and Principal curvature methods provided an EER as little as 0.4% when used with preprocessing.

D. Commercial scanners

An overview of commercial sensors is given in [3] where it is stated that the majority of them are produced by the Japanese companies Hitachi and Fujitsu. While the Hitachi sensors operate with the transillumination lighting technique, sensors produced by Mofiria and YannanTech have implemented side illumination in combination with the palmar image capturing [3]. At the University of Twente, one such side illumination device is available. The commercial ZKTEco device is able to capture an image of finger veins located in a small region of a finger and provides the result seen in Figure 4. As it can be seen, the veins are visible and there is almost no overexposure present even though the device uses side illumination. The only trait of this device that can be viewed as a downfall is the fact that a quite small region of the finger is captured due to the device's size. Therefore, in this research, one of the aims will be to capture as much of the finger's bottom side as possible.

III. Method

A. Setup building

Before any image quality assessment or vein recognition could be performed, the setup had to be realized. The only obvious setup requirement is that its light source must be positioned on the side of the finger. However, the device should also not be longer than an average finger and it should be black in order to absorb as much visible light as possible. Therefore, the design process required solving two main challenges. The electronics of providing a light source and a physical encasing to house the camera, the light source and the finger.

The light source designed is composed out of 8 IR LEDs, a led driver in order to control the light intensity and a microprocessor so that the light source can be controlled by a RPi 4. The source was made into a PCB using KiCad 8.0 (see section VII for details) and its physical realization can be seen in Figure 5.

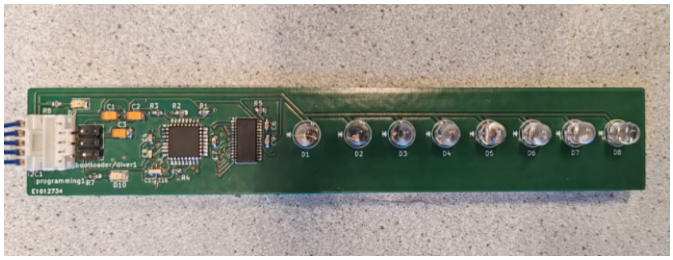


Figure 5: Light source PCB

Autodesk Fusion was used to 3D design the entire casing of the device as shown in Figure 6 and Figure 7.

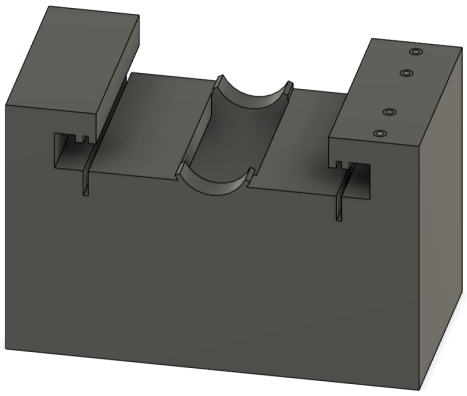


Figure 6: The body of the device

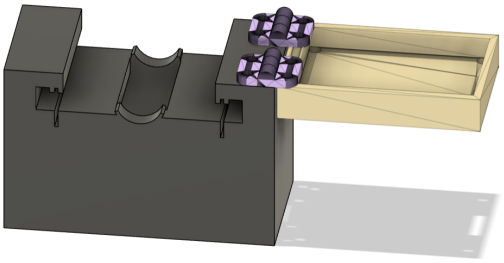


Figure 7: The body of the device with the display holder

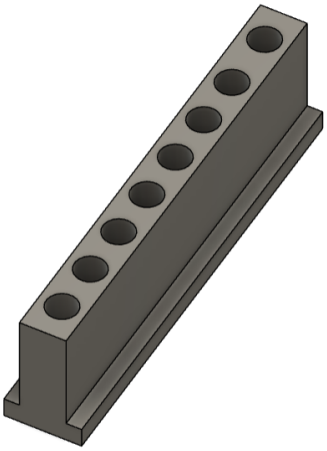


Figure 8: Straight guides

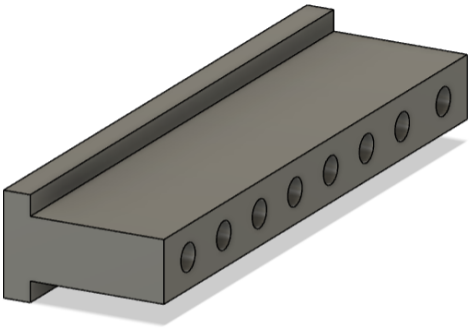


Figure 9: Tapered guides with a wider opening

As it can be seen from Figure 6, the light source PCBs are meant to slide into the slits made on the bottom and the top of the sides of the box while the semi-circled indents are meant to accommodate the finger. The camera is positioned on the bottom of the box from the inside (see section VII for more detail on the separate design parts). The box attached with hinges to the casing as shown in Figure 7 is meant to accommodate a RPi4 compatible display so that the user can see what the camera is seeing and, possibly in the future, have their name displayed after being recognized.

After the parts were printed and the PCB soldered, the communication between the RPi 4 and the PCBs was established by I2C protocol based code that was uploaded onto the ATMEGA chips on the PCBs and run on the RPi 4 (see section VII). This allowed for sending instructions about the LED intensities and channel numbers of the led driver that were meant to be in use from the RPi4 to the PCB. The interface with the camera was achieved by the use of openCV python library making a window to display what the camera is seeing on the screen (section VII). The intensity of the LEDs could be controlled via a sliding bar at the bottom of the screen and its influence can be seen in real time. Besides being able to control the intensity, the zoom and the room brightness can also be changed around.

1) Overexposure mitigation: After the setup was fully built, the finger vein images were ready to be captured. However, some sort of a solution to the overexposure problem had to be incorporated beforehand. As mentioned before, overexposure is light reflecting of the sides of the finger into the camera. Therefore, it had to be ensured that the light does not hit anything but the top of the bone (that acts as a second light source). In order to achieve this, the light beams produces by the IR LEDs have to be narrowed and directed as precisely as possible. The methods used to direct the light were:

- Using IR LEDs with the narrowest radiation pattern available
- Using straight light guides
- Using tapered light guides of two different kinds

The IR LEDs used were the ones mentioned in section II while the guides were 3D designed and printed. The guides were all designed in a way that they can be placed onto the PCB before the LEDs were soldered in order to eliminate any bending of the LEDs while soldering. In total, three types of guides were designed: straigh guides and two versions of the tapered guides. The straight guides whose opening for the LEDs and the opening for the light are the same size can be seen in Figure 8. The tapered guides, on the other hand, start out with an opening for the led and grow narrower by the end of the guide where the light is supposed to come out. The two tapered guides version can be seen in Figure 9 and Figure 10.

Their led openings, that are not visible in the figures, are just as big as those of the straight guides. Due to the fact that tapered guides were expected to give the best image quality (as mentioned mentioned in section II), a distinction was made between the two types so that this method of light direction could be investigated in more detail. The difference between

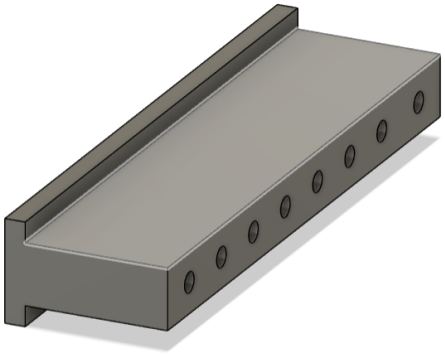


Figure 10: Tapered guides with a narrower opening

the two tapered guide versions is the length of the straight portion of the hole. In the first version, the light hole remains straight for 20 mm before becoming tapered with an angle of 30 while the second version remains straight for 18 mm before becoming tapered by the same angle. The images of tapered hole designs can be found in section VII. The difference is visible in the openings at the ending of both tapered guides. The opening for the first version is bigger than that for the second version.

After the setup is realized and the guides are printed out, a small dataset is aimed to be collected. The analysis of the dataset by the use of algorithms explained in section II shall be used to compare the device's performance to that of the UTFVP. Additionally, the results of all four light direction methods will help identify the best overexposure mitigation method.

IV. Experiments and Results

A. Dataset collection

Following the setup building, a small dataset was collected for testing. The dataset had a total of 6 participants that were all asked to sign a consent form (section VII) before participating. Image acquisition was done manually by pressing the "i" button on the keyboard every time an image was to be taken. The participants gave 2 sets of samples of either 4 or 6 fingers (2 or 3 fingers from each hand) for every light direction method - no guides, straight guides, tapered guides one and tapered guides two.

For purposes of performance comparison, the volunteers were also asked to be photographed using the UTFVP. In this case each subject gave 2 to 3 sets of 3 of their fingers. Delays regarding the availability of this device resulted in a few volunteers not getting to be photographed by it. Hence, size of the dataset of the UTFVP is 5 instead of 6 participants.

B. LED count

The goal of the first, relatively small, experiment was to investigate if less LEDs could be used to acquire images of similar quality as those acquired by 8. In case this worked, the length of the PCB could be minimized and, therefore, the size of the device would all together be smaller for the rectangular PCB design. In order test this, every odd channel of the LED driver was turned off while every even one was left on. The

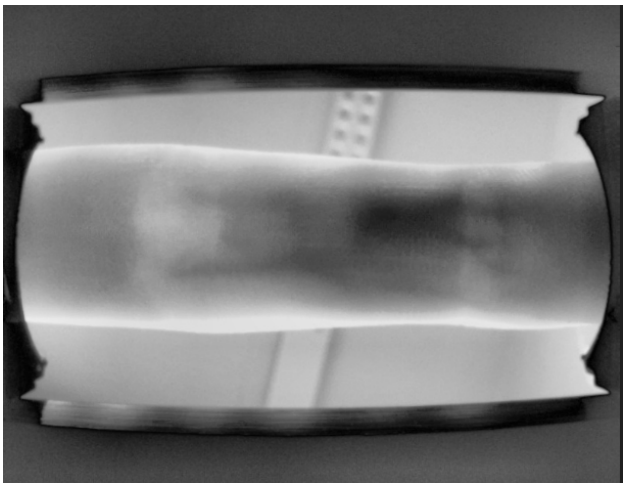


Figure 11: Image captured with all 8 LEDs on

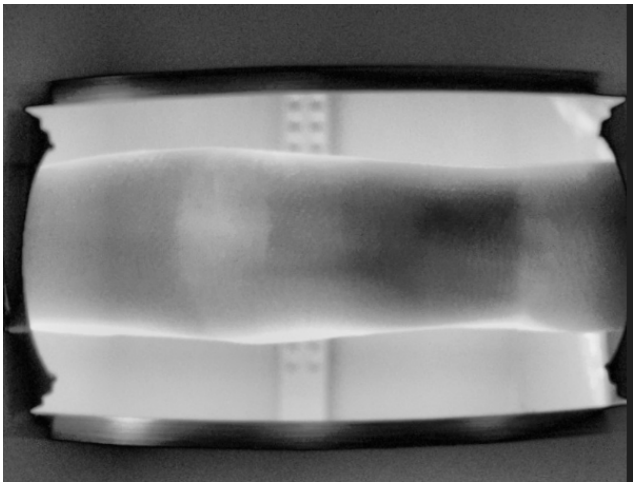


Figure 12: Image captured with 4 LEDs on

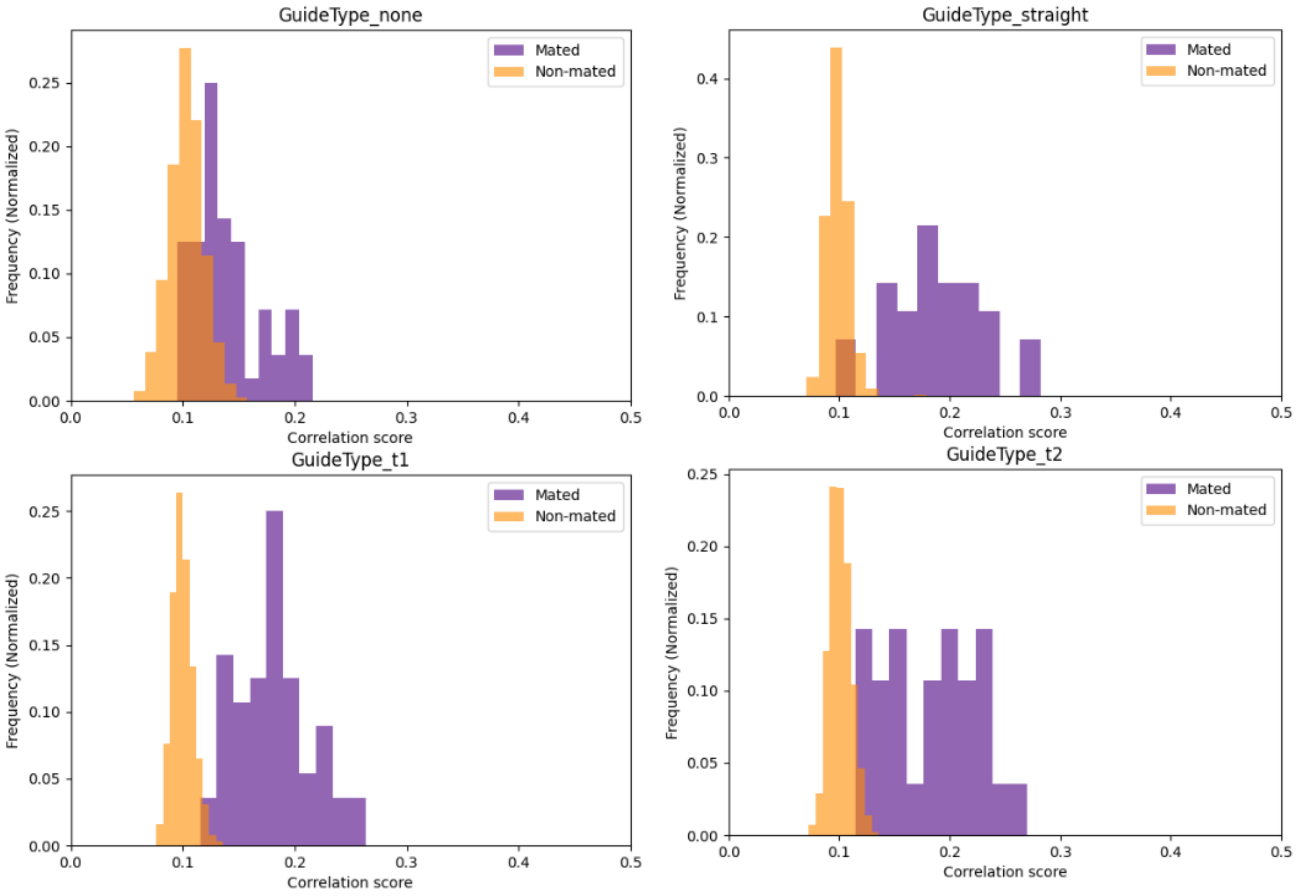
difference between preprocessed images captured with 8 vs 4 LEDs is visible in Figure 11 and Figure 12.

Even though parts of the veins are visible on both images, the one taken with 8 leds additionally shows smaller veins and creates more clear of a pattern. Therefore, it was decided to proceed with all 8 LEDs on for the remaining data acquisition time.

C. Light direction

The second conducted experiment was meant to establish the best light direction and overexposure mitigation technique out of the 4 mentioned in section III. After the dataset was acquired it was preprocessed by the use of the Adaptive Histogram Equalization (as explained in section II) and the Maximum curvature and Repeated line tracking algorithms were applied to the collected images. The results of the Maximum curvature algorithm are presented in form of histograms in Figure 13.

Each graph represents one of the 4 methods. The purple in the graph can be related to the frequency of certain correlation scores of the mated pairs while the orange is the frequency of correlation scores of the nonmated pairs. The values of the nonmated pairs do not differ by a lot per method and tend to



(a) No guides and tapered v1

(b) Straight guides and tapered v2

Figure 13: Cross correlation results of the Maximum Curvature algorithm of the side illumination setup

stagnate around 0.1 each time while the scores of the mated pairs have visibly grown from a mean of 0.14 having a mean around 0.18 by implementing light direction techniques. The results of the Repeated line tracking algorithm can be seen in Figure 14.

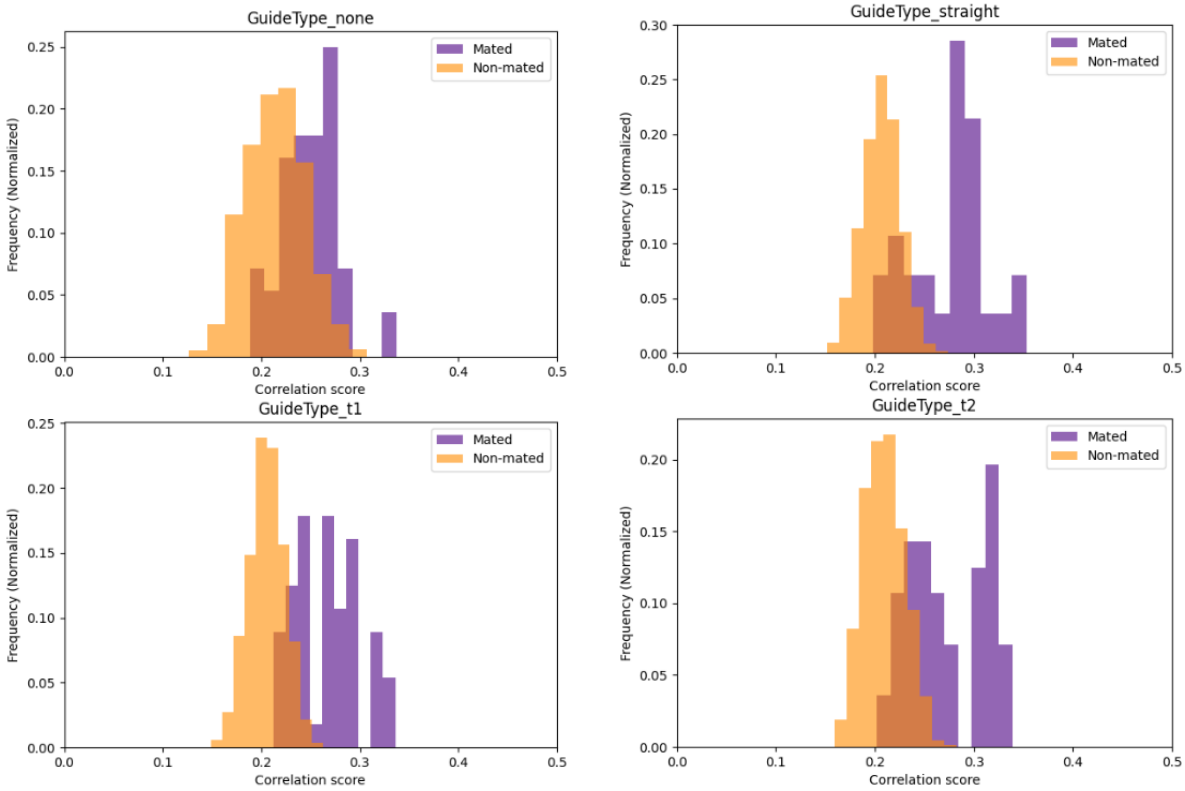
In this algorithm, the scores of the nonmated pairs have grown to stagnate around 0.2 instead of the previously seen 0.1 while the mean of the scores of the mated pairs grew to 0.27. However, a lot of overlap between the mated and nonmated scores is present in the results of this algorithm and no clear threshold can be decided upon which would almost perfectly divide the two groups of scores. Furthermore, DET (Detection error tradeoff) curves of the provided data were plotted in Figure 15. These curves are meant to show how the false non-match rate changes in relation to the false match rate while varying the separation threshold. This threshold is defined as a correlation score value which is meant to perfectly separate the nonmated and mated scores.

In Figure 15, it can be seen that for the method where no guides were implemented, the curve stagnates at 0 FMR for a while until the threshold reaches a certain value which can be supposed to be around 0.1. From there, the curve deviates from a straight line and shows that as threshold values increase the FNMR goes down as the FMR increases. After a certain threshold value is reached (around 0.16), the curve settles at 0 FNMR since at that point, there are no more nonmated pairs scores present in the histograms of Figure 13. As the guides get applied to the light source, the DET curves behaviour changes. The curves start approaching the origin, showing that for most of the picked threshold values, there is no trade off between the two rates. This could have also be concluded from the remaining Maximum Curvature histograms as the overlap between the mated and nomated bars reduces significantly as the light source gets more directed. The DET curves for the Repeated line tracking are presented in Figure 16 and can be seen deviate more from the origin.

Table I shows the nonmated and mated average scores of all four light direction methods as well as their EER for both algorithms. Figure 17 was additionally included to show the comparison between the overexposure present when no guides were mounted onto the LEDs and that present when the tapered guides v1 were used.

D. UTFVP results

Before providing evaluation results of this dataset, two images of the same finger are shown in order to introduce the difference in the image quality between the self made setup and the UTFVP. Two preprocessed images of the same finger can be found in Figure 18.



(a) No guides and tapered v1

(b) Straight guides and tapered v2

Figure 14: Cross correlation results of the Repeated line tracking of the side illumination setup

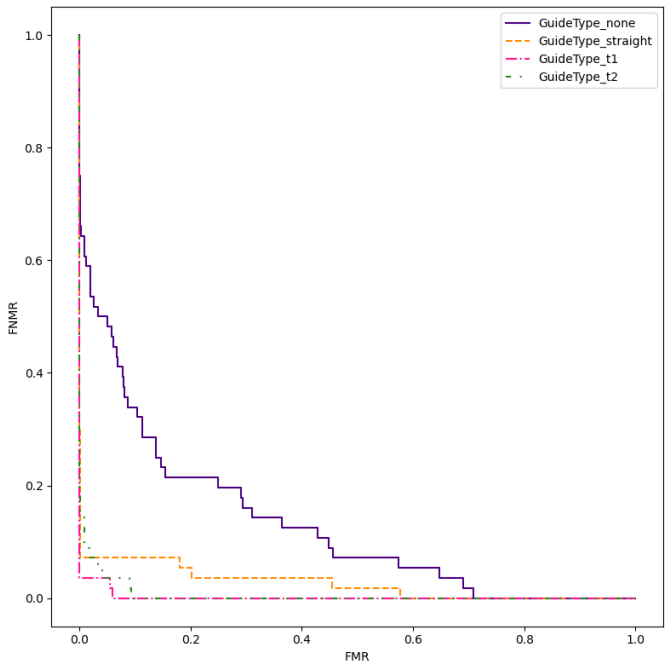


Figure 15: DET curves of all four methods - Maximum curvature

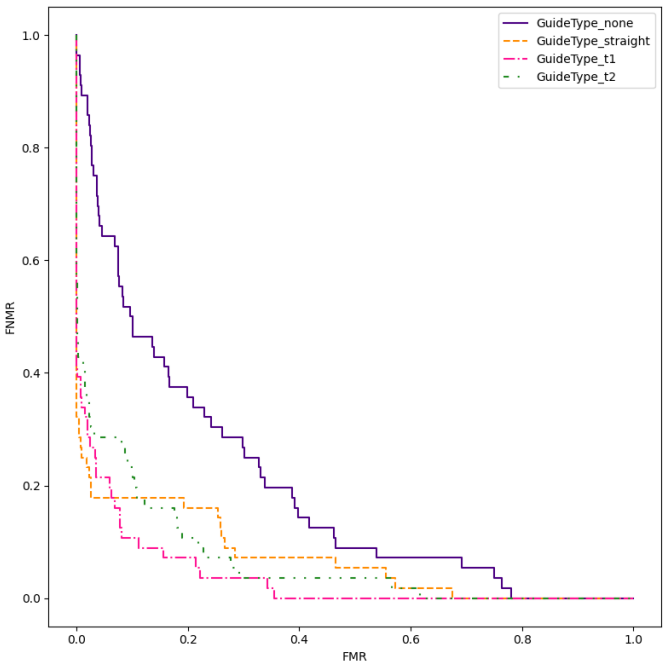
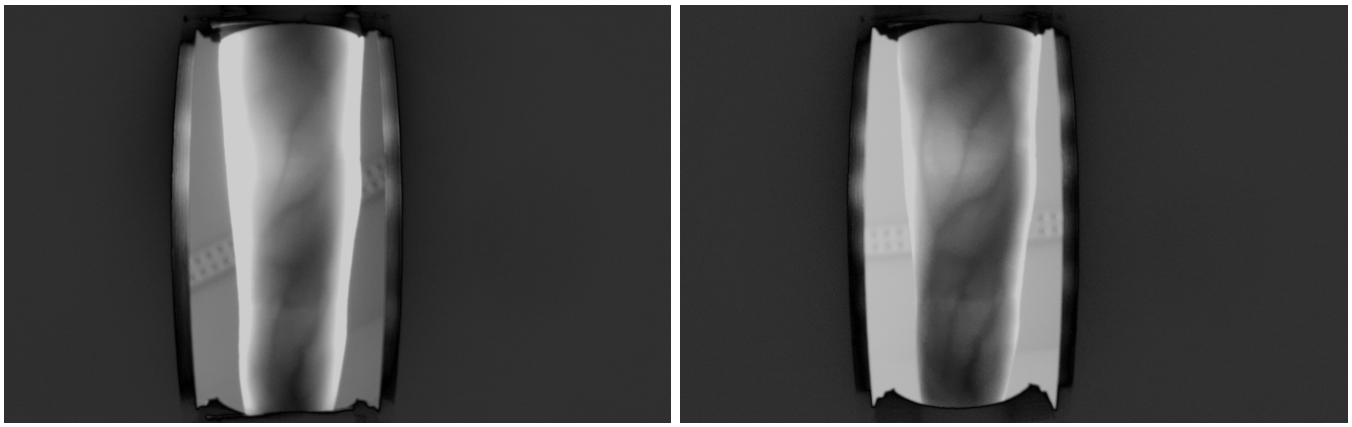
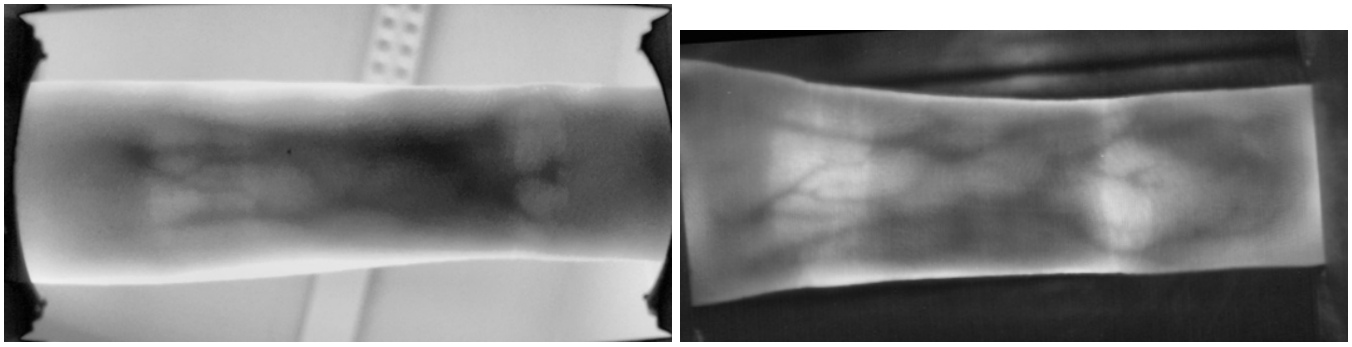


Figure 16: DET curves of all four methods - Repeated line tracking



(a) No guides (b) Tapered guides v1

Figure 17: Comparison between overexposure



(a) Side illumination device (b) UTFVP

Figure 18: Two preprocessed images of the same finger

Table I: Result summary of the four guides

	Result	MC	RLT
None	Mated pairs average score	0.14	0.251
	Nonmated pairs average score	0.103	0.222
	EER (%)	21.4	28.6
Straight	Mated pairs average score	0.189	0.277
	Nonmated pairs average score	0.0998	0.206
	EER (%)	7.1	17.9
Tapered v1	Mated pairs average scores	0.182	0.262
	Nonmated pairs average scores	0.101	0.205
	EER (%)	3.6	10.7
Tapered v2	Mated pairs average scores	0.183	0.274
	Nonmated pairs average scores	0.101	0.209
	EER (%)	4.1	16.1

The main difference between the two images lies in the details of thin vessels. While Figure 18 a) displays a pattern detailed enough to be recognized by the human eye, the Figure 18 b) enhances it by getting rid of the darker smudged area present directly below the final finger joint. The UTFVP featured separate PWM settings for each LED which allowed for a stronger illumination of the part of the finger that appears darker in Figure 18 a). Hence, the UTFVP ended up producing clearer patters than the side illumination setup and was expected to give better results.

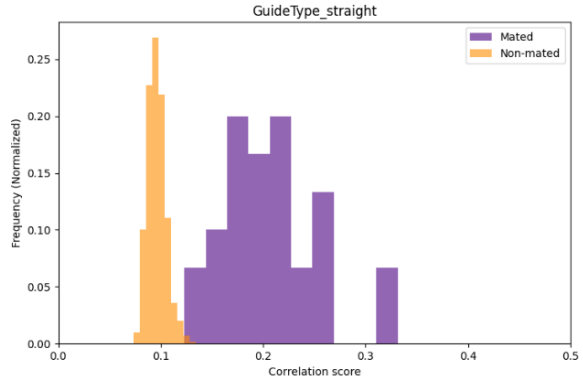
The dataset was run through the same two algorithms. The histogram results can be seen in the Figure 19 a) and Figure 19 b) while the accompanying DET curves are presented in Figure 19 c) and Figure 19 d).

Even though that in the histogram Figure 19 a) the nonmated scores group around 0.1 just like in Figure 13 a), the mated scores exhibit a higher average of about 0.202 instead of 0.189. Consequently, the scores of the mated and nonmated pairs overlap less resulting in more straight line behavior present in the DET curve in c) of Figure 19. The results of the Repeated line tracking exhibit higher average scores for the nonmated pairs just like in Figure 14 which in combination with a higher average of mated scores than those of the side illumination device produces a DET curve as shown in Figure 19 d).

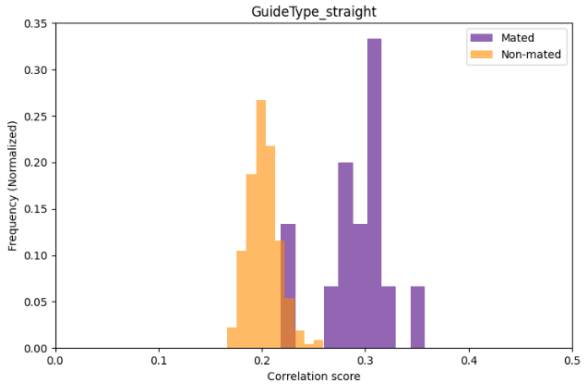
V. Discussion

A. Discussion of results

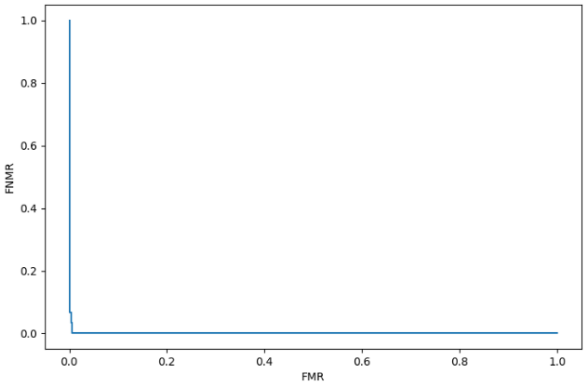
One of the most easily noticeable characteristics of the results was that the ones obtained by Repeated line tracking were always substantially worse than those of Maximum curvature. The reason for this lies in the fact that the former is known to track noise generated structures that do not always belong to a vein, especially in images where the lack of vein patterns is present. As a result, false matches and false non matches are more likely to occur in this method which reflects on the correlation scores. The nonmated scores exhibit growth while the mated scores remain the same and sometimes even



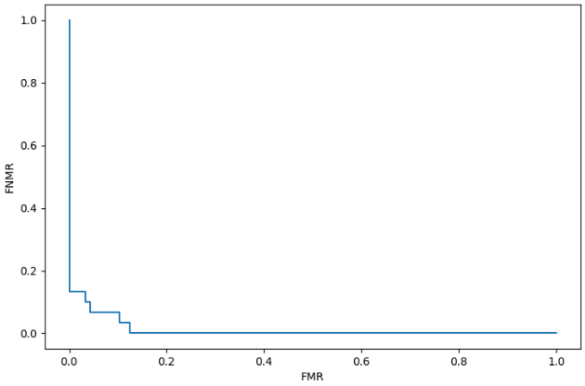
(a) Cross correlation results of the Maximum curvature algorithm



(b) Cross correlation results of the Repeated line tracking algorithm



(c) DET curve of the Maximum curvature algorithm



(d) DET curve of the Repeated line tracking

Figure 19: UTFVP results

drop. The consequences of this are also represented by higher false match rates and false non-match rates in the DET curves of the RLT method results of both the side illumination setup as well as the UTFVP. Hence, the prediction made in section II on which method would perform better was proven correct.

From the first experiment of section IV, it is not hard to conclude that the best performing guide was the first version of the tapered guide. This can be concluded both from the lowest EER values as well as the smallest overlap region in its histograms. Possible reasons why the tapered guides v2 did not work better than v1, even though they should have created the narrowest light beam, could be that not enough light was able to pass through the opening to illuminate the fingers properly or that the beam was not aimed directly into the top of the bone. Tapered v1 results are shown once again, but this time in comparison to those of the UTFVP in Table I where the side illumination device is referred to as SID.

From the comparison of the EER values of 3.6% for the SID and 0.59 % for the UTFVP, it is clearly visible that the performance of the UTFVP has not been beaten by the side illumination device. However, one must not disregard the fact that the averages of the mated scores of the two devices do not differ too drastically as they show 0.186 for the SID and 0.206 for the UTFVP. The fact that the UTFVP dataset experiences a greater difference (0.1097) between the averages of the mated and nonmated scores than SID (0.081) is the reason why

Table II: Result overview: UTFVP vs SID tapered guides v1 method

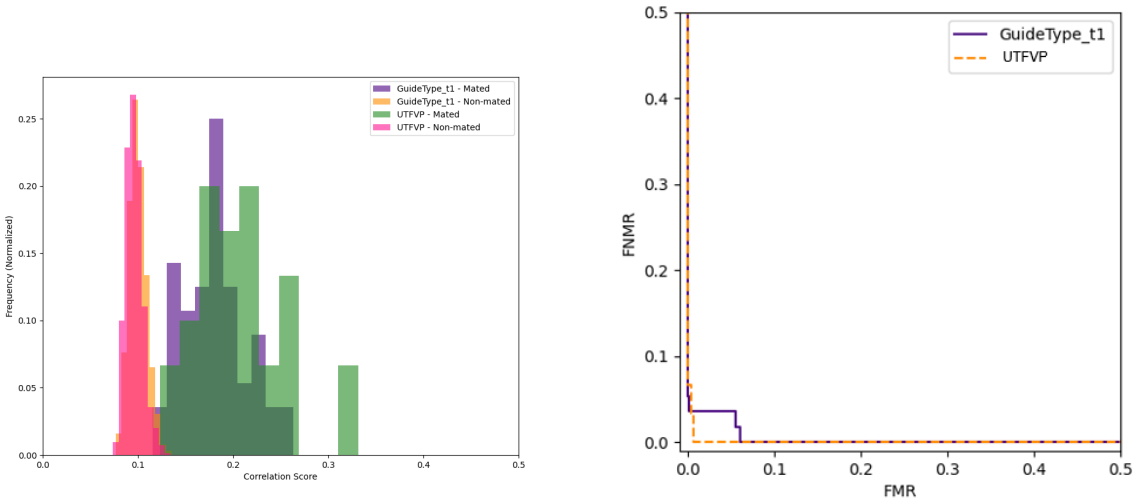
	Result	MC	RLT
SID	Mated pairs average	0.182	0.262
	Nonmated pairs average	0.101	0.205
	EER (%)	3.6	10.7
UTFVP	Mated pairs average	0.206	0.292
	Nonmated pairs average	0.0963	0.202
	EER (%)	0.59	6.7

its DETs and EER results are preferred. Nonetheless, SID exhibits remarkably good results when it comes to accuracy.

A comparison of the histograms and DET curves of best results given by the side illumination setup with the results of the UTFVP is given in Figure 20. The DET curve of the UTFVP approaches the (0,0) point on the graph almost fully while its histogram experiences minimal overlap between the mated and nonmated correlation scores. In other words, both FNMR and FMR are 0 for almost all the separation thresholds picked. A greater overlap can be seen in the histogram of the side illumination setup as well as that its DET curve exhibits greater deviation from the (0,0) point.

B. Future improvements

While the prototype with side illumination, which has proven to be quite operable, was successfully designed, there



(a) Histogram of Maximum curvature

(b) DET curves

Figure 20: Result overview: UTFVP vs SID tapered guides v1 method

is always room for improvement. For example, incorporating a mirror into the design by adding it directly below the finger to act as a light reflector into a camera could minimized the height of the design. The mirror would redirect the light into the camera which could then be mounted vertically on one of the inside walls of the box. Since the path the light has to travel in order to deliver the image of the whole finger would not change it could be split between the distance between the finger and the mirror and the the distance between the mirror and the camera lens. Furthermore, the design of the PCB has not yet been optimized as the part of the PCB containing everything but the LEDs currently extends a bit beyond the length of the device. This did not represent a problem as all the LEDs fit in the length of the device. Nonetheless, designing it in a sort of a T shape instead of a rectangle would certainly make the device more compact. The base of the T shape would then include the electronics and the top bar would hold the LEDs. The base of the T could then be pushed through the slit into the box and be hidden while the LED containing bar would remain in current placement of the PCB. Moreover, the length of the device should all together be shortened to enable for a more controlled image taking environment. This would call for taking an image of a smaller portion of the finger, but since that approach works on the UTFVP, there is no reason why it could not be incorporated here as well. A shorter device would allow for a smoother and easier finger placement which is more likely to result in a person placing their finger on the device in a similar way twice. In the case of this research the volunteer sometimes had to stretch the finger to reach the other end of the device since the length was determined based on a slightly longer set of fingers.

Lastly, collecting a greater dataset would provide an even better evaluation of the device. It would test the true robustness of the image taking procedure and verify if the device is suitable for commercial use.

VI. Conclusion

The main objective of this research was to develop a finger vein image capturing setup with incorporated side illumination that could compare to the performance of UTFVP as stated in section I. In order to achieve this, four different ways of overcoming one of the greatest challenges of side illumination were suggested. On one hand, the final results obtained from running the dataset obtained by the designed device through two different evaluation algorithms did not rival the results of the dataset captured by UTFVP. Reflecting on the first research question, it was shown that a side illumination setup which battles overexposure was possible to build, but the statement that it could measure up to the quality of UTFVP might have been too ambitious. On the other hand, a light direction method was identified which enables for most overexposure mitigation while maintaining a decent image quality. The method in question is the use of tapered guides v1 which have proven to give results that, even though worse, do not differ by a lot from those of UTFVP. Hence, the second research question stated in section I has been answered. In section V, improvements to the design and the evaluation method were given with hopes that when implemented they would result in a more accurate and possibly commercially useful design.

All in all, an original side illumination setup has been designed, produced and evaluated from the ground up. The research questions have been answered and the main goals have been achieved giving solid ground for any future work for improving a University of Twente produced finger vein image capturing device with side illumination.

References

[1] T. E. o. E. Britannica, ""alphonse bertillon". encyclopedia britannica," 2024. [Online]. Available: https://www.britannica.com/biography/Alphonse-Bertillon

[2] N. Mott, "Hacking fingerprints is actually pretty easy—and cheap," 2021. [Online]. Available: https://www.pcmag.com/news/hacking-fingerprints-is-actually-pretty-easy-and-cheap

[3] A. Uhl, "State of the art in vasuclar biometrics," *Handbook of Vascular Biometrics*, 2020. [Online]. Available: https://doi.org/10.1007/978-3-030-27731-4_1
[4] B. T. Raymond Veldhuis, Luuk Spreeuwers and S. Rozendal, "A high-quality finger vein dataset collected using a custom-designed capture device," *Handbook of Vascular Biometrics*, 2020. [Online]. Available: https://doi.org/10.1007/978-3-030-27731-4_2
[5] T. van Zonnenveld, "3d finger vein patterns: acquisition, reconstruction and recognition." [Online]. Available: https://essay.utwente.nl/88484/
[6] S. Kravchenko, "Development of a finger vein acquisition device with side illumination," 2020. [Online]. Available: https://essay.utwente.nl/97761/
[7] P. Normakristagaluh, G. Laanstra, L. Spreeuwers, and R. N. J. Veldhuis, "The impact of illumination on finger vascular pattern recognition," 2024. [Online]. Available: https://doi.org/10.1049/2024/4413655
[8] N. Miura, A. Nagasaka, and T. Miyatake, "Extraction of finger-vein patterns using maximum curvature points in image profiles," 2005.
[9] ——, "Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification," 2005.

VII. Appendix

A. PCB design

List of special components on the PCB:

- TLC 5940 - led driver
- ATMEGA328P AU - microprocessor
- SFH 4550 - IR leds with a narrow half angle
- CSTCE18MOV53 R1 - oscillator

The schematics of the PCB:

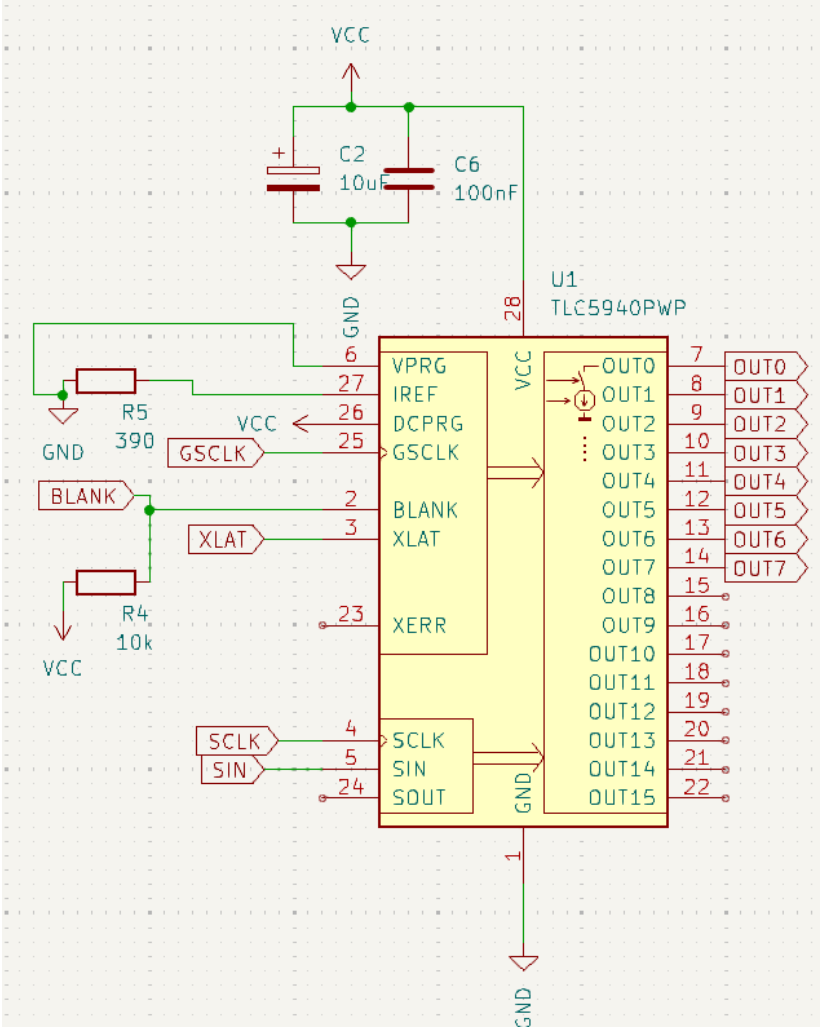


Figure 21: Connections of the led driver

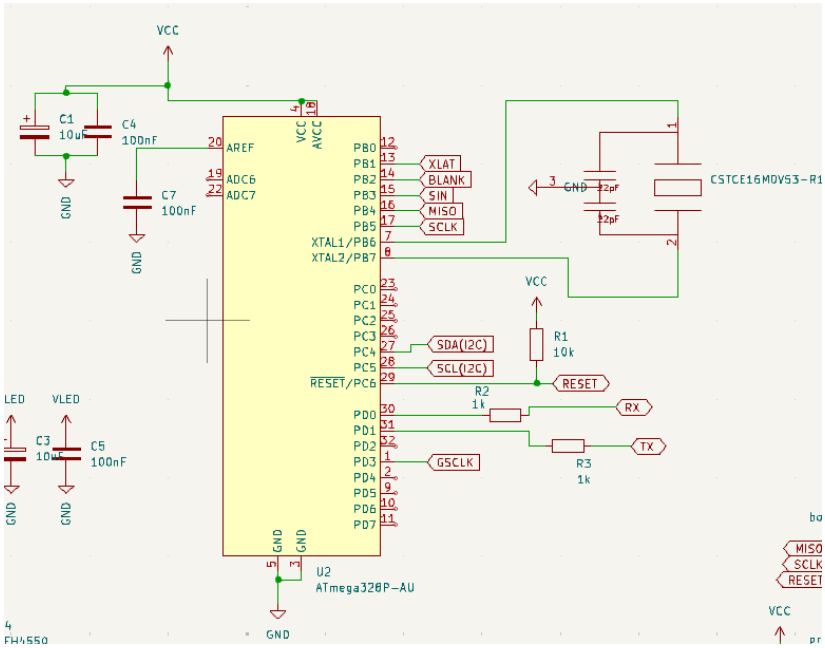


Figure 22: Connections to the microprocessor

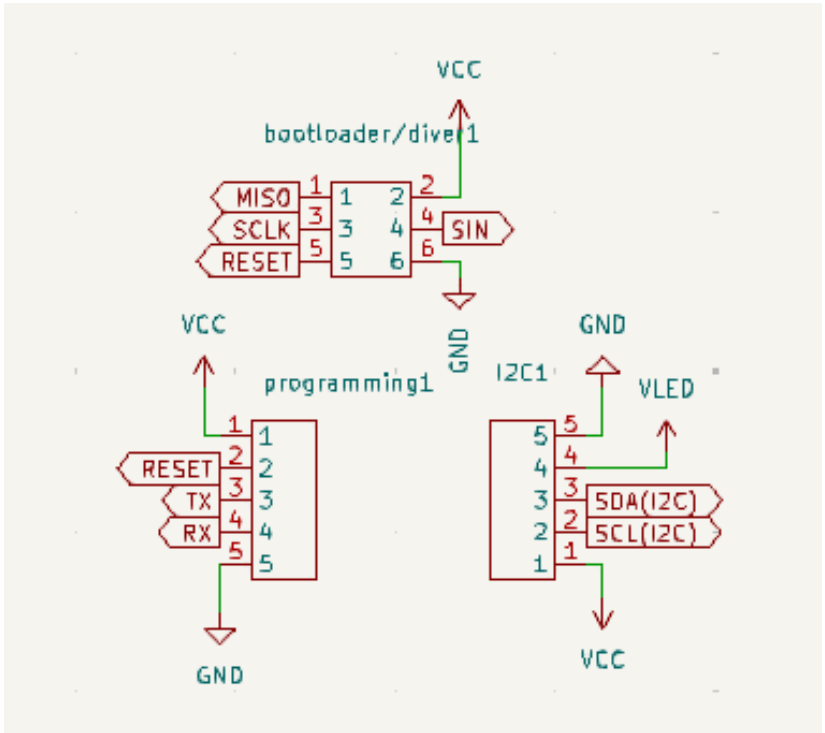


Figure 23: Header connections

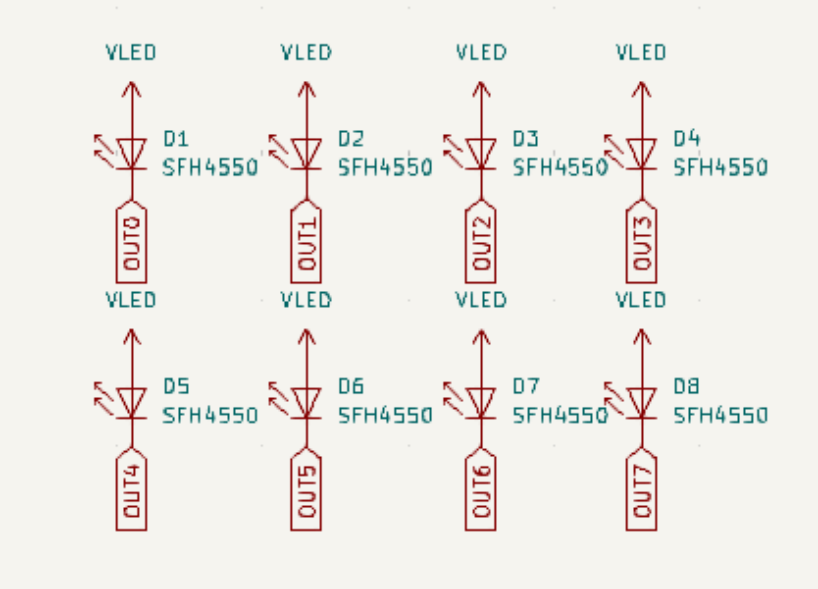


Figure 24: IR LEDs connections

B. Physical design

The body broken down into parts to asssemble:

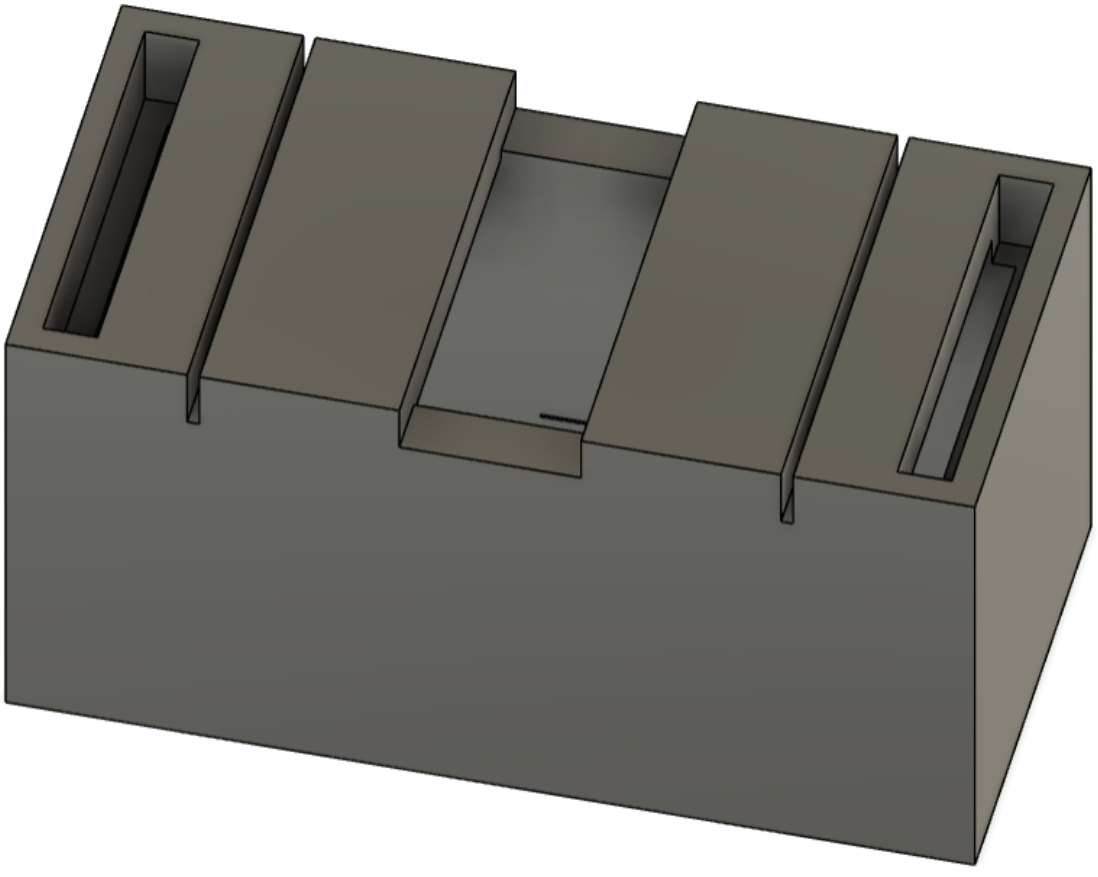


Figure 25: Base

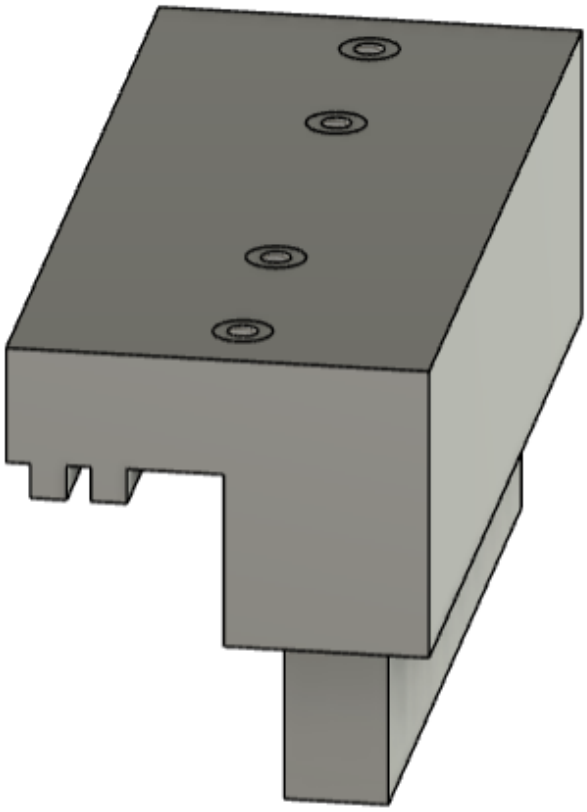


Figure 26: PCB holders

Figure 27: Display case

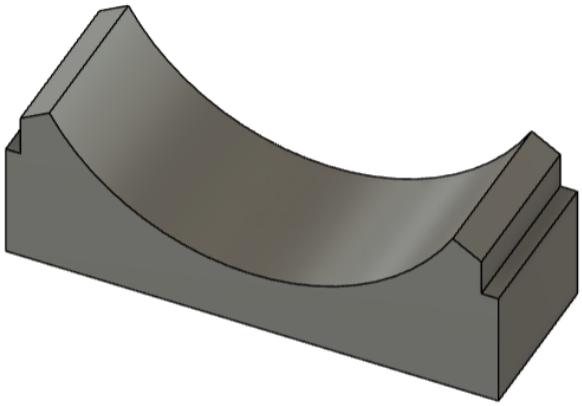


Figure 28: Finger holder

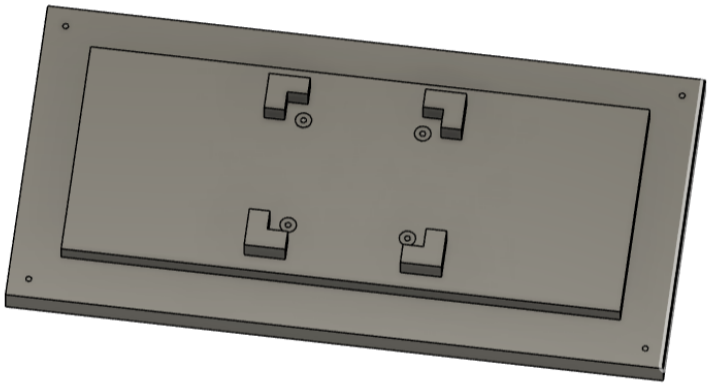


Figure 29: Camera holder at the bottom

C. Code

```
#include <Wire.h>
#include "Tlc5940.h"

#define I2C_ADDRESS 0x08
#define NUM_CHANNELS 8


uint8_t receivedChannels[NUM_CHANNELS];
uint16_t receivedPWMs[NUM_CHANNELS];
uint8_t a;

void setup() {
    Wire.begin(I2C_ADDRESS);

    Wire.onReceive(receiveEvent);
    Wire.onRequest(sendEvent);
    Serial.begin(115200);
    Tlc.init();
    Tlc.clear();



}

void loop() {
    // Main loop can be empty or contain other logic if needed


}
void setvalues(){
    for (int i = 0; i < 4; i++) {
    Tlc.set(receivedChannels[i], receivedPWMs[i]);
    Tlc.update();
  }
}

```

Figure 30: PCB code part 1

```
void receiveEvent(int howMany) {
    Serial.print("Bytes received: ");
    Serial.println(howMany);

    // Check if the first byte received is the register byte
    if (Wire.available() > 0 && Wire.read() == 0) {
        // Expecting the correct number of bytes (excluding the register byte)
        if (howMany == NUM_CHANNELS * 3 + 1) { // Each channel and PWM value is 3 bytes
            for (int i = 0; i < NUM_CHANNELS; i++) {
                receivedChannels[i] = Wire.read();
                Serial.print("read: ");
                Serial.println(receivedChannels[i]);

            }
            for (int i = 0; i < NUM_CHANNELS; i++) {

                uint8_t high=Wire.read();
                uint8_t low=Wire.read();
                receivedPWMs[i] = low << 8 | high; // Combine two bytes into one uint16_t'
                Tlc.set(receivedChannels[i], receivedPWMs[i]);


            }
            Tlc.update();
```

Figure 31: PCB code part 2

```
            Serial.print("Received channels: ");
            for (int i = 0; i < NUM_CHANNELS; i++) {
                Serial.print(receivedChannels[i]);
                Serial.print(" ");
            }
            Serial.println();

            Serial.print("Received PWMs: ");
            for (int i = 0; i < NUM_CHANNELS; i++) {
                Serial.print(receivedPWMs[i]);
                Serial.print(" ");
            }
            Serial.println();

        } else {
            // If the number of bytes received is greater or less than expected,
            // read and discard the extra bytes
            while (Wire.available()) {
                a=Wire.read();
            }
            Serial.println("Unexpected number of bytes received.");
            Serial.println(a);
        }
    } else {
        // If the first byte received is not the register byte, discard all bytes
        while (Wire.available()) {
            Wire.read();
        }
        Serial.println("Invalid register byte received.");
    }

}
```

Figure 32: PCB code part 3

```
void sendEvent() {
    // Send back the received channels and PWM values
    for (int i = 0; i < NUM_CHANNELS; i++) {
        Wire.write(receivedChannels[i]);
    }
    for (int i = 0; i < NUM_CHANNELS; i++) {
        Wire.write(receivedPWMs[i] & 0xFF); // Send the low byte of PWM value
        Wire.write(receivedPWMs[i] >> 8); // Send the high byte of PWM value

    }
}
```

Figure 33: PCB code part 4

```
import cv2
import time
import numpy as np
from picamera2 import Picamera2
import smbus
import time
import struct


ARDUINO_I2C_ADDRESS = 0x08
ARDUINO_I2C_ADDRESS2= 0x09
NUM_CHANNELS = 6
# Define the struct format for sending and receiving data
# '4B' for 4 uint8_t values followed by '4H' for 4 uint16_t values
# '<' indicates little-endian byte order
STRUCT_FORMAT = '<' + 'B' * NUM_CHANNELS + 'H' * NUM_CHANNELS
```

Figure 34: RPI4 code part 1

```python
ARDUINO_I2C_ADDRESS = 0x08
ARDUINO_I2C_ADDRESS2= 0x09
NUM_CHANNELS = 6
# Define the struct format for sending and receiving data
# '4B' for 4 uint8_t values followed by '4H' for 4 uint16_t values
# '<' indicates little-endian byte order
STRUCT_FORMAT = '<' + 'B' * NUM_CHANNELS + 'H' * NUM_CHANNELS

# Function to write data to Arduino
def write_data(data,addr):
    print("ovde sam")
    try:
        bus = smbus.SMBus(1)
        print("ovde sam 2")
        bus.write_i2c_block_data(addr, 0, list(data))
        print("ovde sam 3")
        time.sleep(0.2)  # Sleep to allow Arduino to process the data
        return True
    except Exception as e:
        print(f"Error writing data to Arduino: {e}")
        return False

```

Figure 35: RPI4 code part 2

```python
def read_data(addr):
    try:
        bus = smbus.SMBus(1)
        data = bus.read_i2c_block_data(addr, 0, struct.calcsize(STRUCT_FORMAT))
        return struct.unpack(STRUCT_FORMAT, bytes(data))
    except Exception as e:
        print(f"Error reading data from Arduino: {e}")
        return None

# Function to adjust the brightness of the image
def adjust_brightness(image, brightness):
    return cv2.convertScaleAbs(image, alpha=1, beta=brightness)

```

Figure 36: RPI4 code part 3

```python
def apply_zoom(image, zoom_factor):
    height, width = image.shape[:2]
    center_x, center_y = width // 2, height // 2
    radius_x, radius_y = int(center_x / zoom_factor), int(center_y / zoom_factor)

    min_x, max_x = center_x - radius_x, center_x + radius_x
    min_y, max_y = center_y - radius_y, center_y + radius_y

    # Ensure the coordinates are within bounds
    min_x, max_x = max(min_x, 0), min(max_x, width)
    min_y, max_y = max(min_y, 0), min(max_y, height)

    cropped = image[min_y:max_y, min_x:max_x]
    return cv2.resize(cropped, (width, height))
```

Figure 37: RPI4 code part 4

```python
def send_to_arduino(pwm):
    pwm_values = [pwm] * NUM_CHANNELS  # Example PWM values
    pwm_bytes=[pwm.to_bytes(2, byteorder='little') for pwm in pwm_values]
    data_to_send = bytes(channels) + b''.join(pwm_bytes)
    #data_to_send=channels+pwm_bytes
    print("data to send: ", list(data_to_send))
    # Write data to Arduino
    if write_data(data_to_send,ARDUINO_I2C_ADDRESS):
        print("Data sent successfully.")

        # Read back data from Arduino
        received_data = read_data(ARDUINO_I2C_ADDRESS)
        if received_data:
            print("Received data from Arduino pcb:")
            print("Channels:", received_data[:NUM_CHANNELS])
            print("PWM Values:", received_data[NUM_CHANNELS:])
    if write_data(data_to_send,ARDUINO_I2C_ADDRESS2):
        print("Data sent successfully.")

        # Read back data from Arduino
        received_data = read_data(ARDUINO_I2C_ADDRESS2)
        if received_data:
            print("Received data from Arduino on board:")
            print("Channels:", received_data[:NUM_CHANNELS])
            print("PWM Values:", received_data[NUM_CHANNELS:])
    else:
        print("Failed to send data2.")
```

Figure 38: RPI4 code part 5

```python
# Initialize Picamera2
picam2 = Picamera2()

config = picam2.create_still_configuration()  # Use still configuration
picam2.configure(config)
picam2.set_controls({"ExposureTime": 5000})
picam2.start()

# Give some time to the camera to start
time.sleep(2)

# Initial values for brightness and zoom
brightness = 0
zoom_factor = 0
image_counter = 0  # Counter for images
image_paths = []  # List to store paths of captured images

cv2.namedWindow('Camera')

# Callback functions for trackbars
def on_brightness_change(val):
    global brightness
    brightness = val - 50
    print(f"Brightness changed to: {brightness}")
```

Figure 39: RPI4 code part 6

```python
# Callback functions for trackbars
def on_brightness_change(val):
    global brightness
    brightness = val - 50
    print(f"Brightness changed to: {brightness}")

def on_zoom_change(val):
    global zoom_factor
    zoom_factor = 1 + val / 10
    print(f"Zoom factor changed to: {zoom_factor}")
def on_pwm_change(val):
    global pwm_values
    pwm_value= val*10
    print(f"PWM factor changed to: {pwm_value}")
    send_to_arduino(pwm_value)
def turn_off_leds():
    send_to_arduino(0)

# Create trackbars for brightness and zoom
cv2.createTrackbar('Brightness', 'Camera', 50, 100, on_brightness_change)
cv2.createTrackbar('Zoom', 'Camera', 2, 20, on_zoom_change)
cv2.createTrackbar('PWM', 'Camera', 100, 350, on_pwm_change)
```

Figure 40: RPI4 code part 7

```python
while True:
    # Capture frame-by-frame
    frame = picam2.capture_array()

    # Adjust brightness and zoom
    frame = adjust_brightness(frame, brightness)
    frame = apply_zoom(frame, zoom_factor)

    # Display the frame
    cv2.imshow('Camera', frame)

    # Check for key press
    key = cv2.waitKey(1)


    if key == ord('i'):
        image_counter += 1
        image_name = f'image_{image_counter}.jpg'
        cv2.imwrite(image_name, frame)
        image_paths.append(image_name)
        print(f"Image captured and saved as '{image_name}'.")

```

Figure 41: RPI4 code part 8

```
158
159
160     elif key == ord('q'):
161         print("Quitting the program.")
162         turn_off_leds()
163         break
164
165
166 # Release resources
167 cv2.destroyAllWindows()
168 picam2.stop()
169
170
```

Figure 42: RPI4 code part 9

D. Consent form

CONSENT FORM: Participation in the finger vein image acquiring process

By signing this form, I confirm that I am aware that this experiment entails providing between 8 and 48 finger vein images and that I was explained how the set up works. It has been stated to me that these images shall only be used as a part of an Electrical Engineering bachelor thesis research and that the data shall be deleted after the final presentation.

Please tick below in case you agree with the statement. In case you do not agree, leave the box empty.

- o I do not give permission for my vein images to be shown in the student's report
- o I do not give permission for my finger vein images to be shown in the final thesis presentation on 5th July 2024.

Researcher: _____ Tester: _____

Figure 43: Consent form

E. Tapered hole design

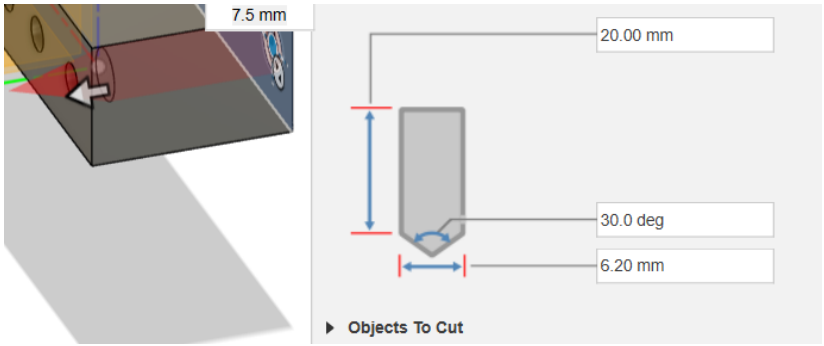


Figure 44: Hole design for the first version of the tapered guides

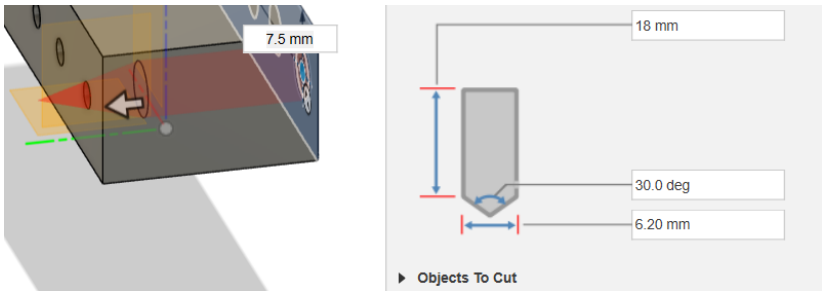


Figure 45: Hole design for the first version of the tapered guides

F. Camera vision

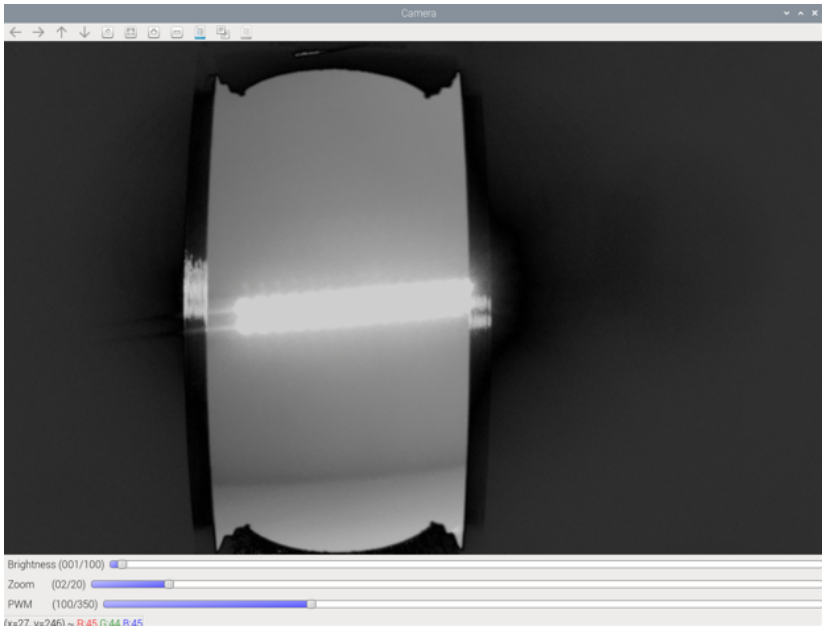
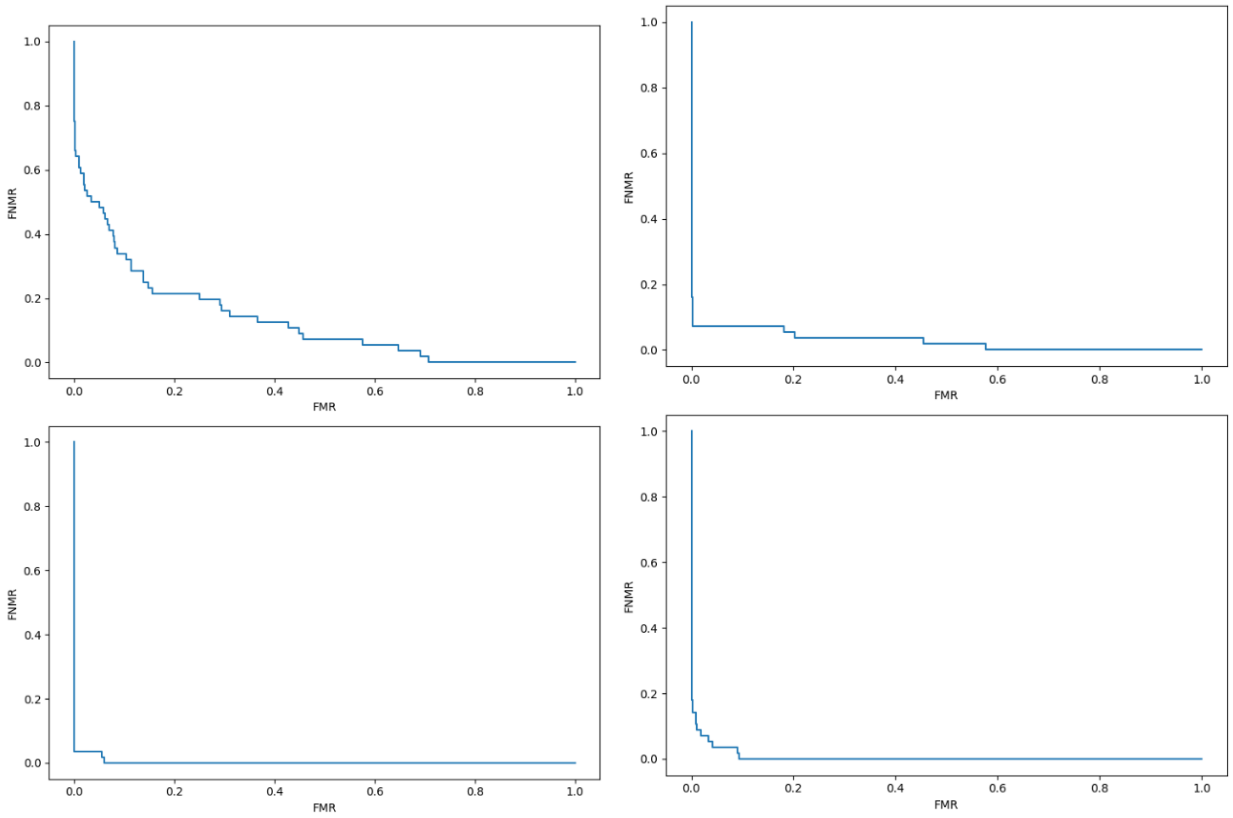


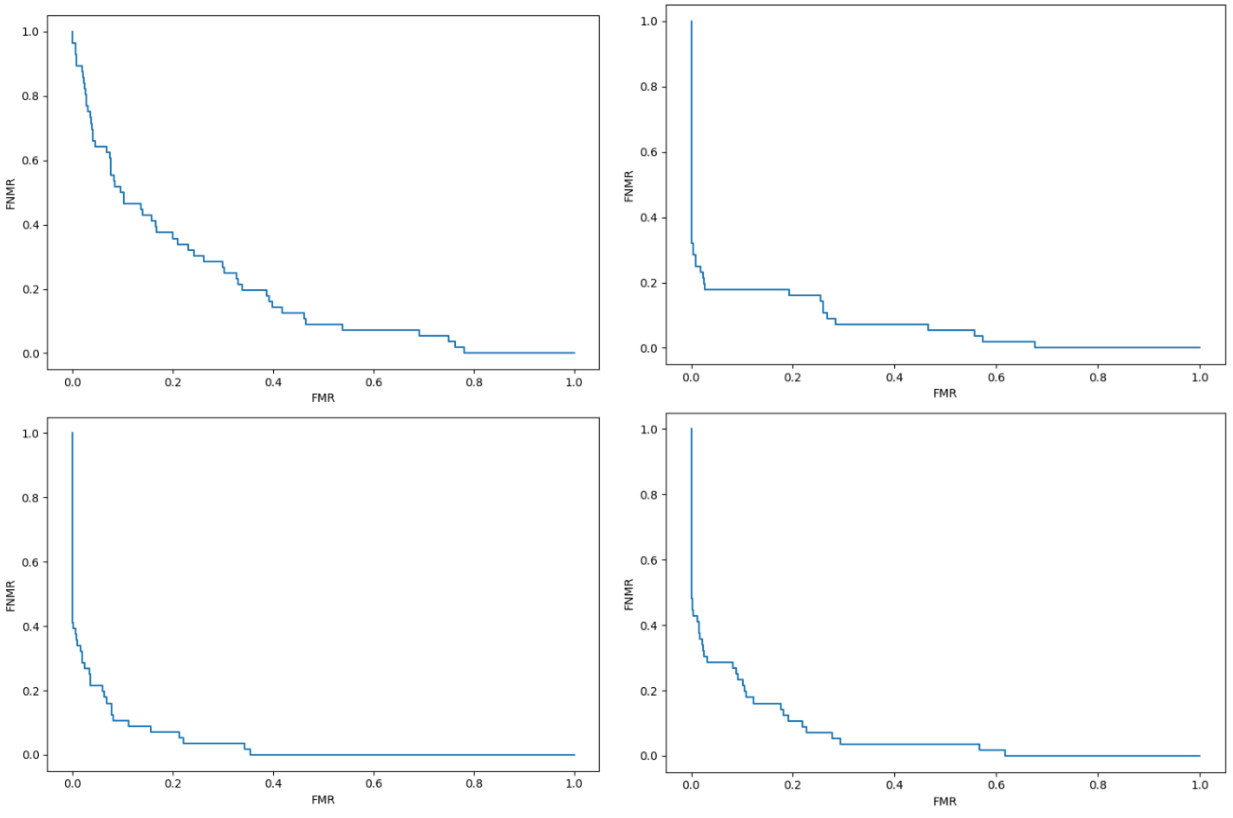
Figure 46: Camera vision with slide bars

G. Separate DET plots



(a) (b)

Figure 47: DET curve of the results of the Maximum curvature algorithm



(a) (b)

Figure 48: DET curve of the results of the Repeated line tracking algorithm