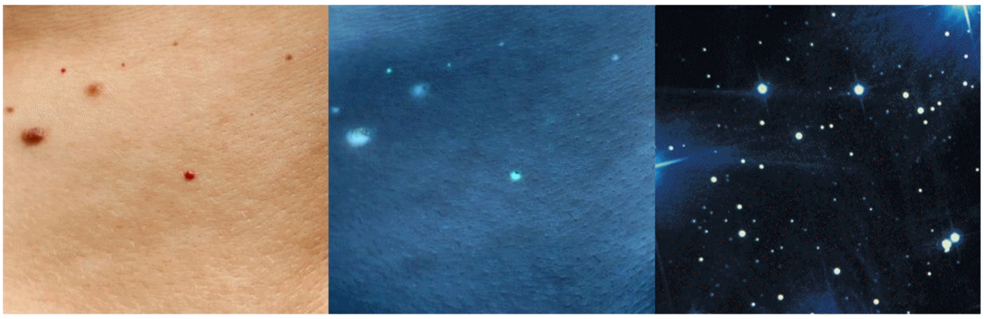
STARPeople

Schinkelshoek, Laurens

STARPeople



THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE
in
RESEARCH IN PHYSICS, CLASSICAL/QUANTUM
INFORMATION

Author :	LJ Schinkelshoek
Student ID :	S0332755
Supervisor :	Dr. Matthieu Schaller
2nd corrector :	Dr. Elena Sellentin

Leiden, The Netherlands, August 2, 2023

STARPeople

LJ Schinkelshoek

Huygens-Kamerlingh Onnes Laboratory, Leiden University
P.O. Box 9500, 2300 RA Leiden, The Netherlands

August 2, 2023

Abstract

The past decades have shown a rise in skin cancer. This creates the need for prevention and efficient treatment. The most common skin cancer (melanoma) can only be treated when detected early. In this thesis we propose a method of increasing awareness for people with a high risk of skin cancer as well as allowing for early detection.

Skin cancer is hard to detect, even for experiences healthcare professionals. One of the signals of potential harm full lesions is change over time. We propose to develop an application with which changes in skin lesions can be identified early. By allowing patients to film their body with a mobile phone camera we aim to track the development of lesions. If a patient films their body regularly changes can be detected and the application can urge the patient to consult a dermatologist.

In this thesis we explore the possibility of combining the frames of these films into an overview displaying the patients complete back or arm. Combining frames is called stitching. Different stitching techniques found in literature are explored and tested for effectiveness. The optimizations performed are reported and the final result is presented.

The location of the different lesions on an overview of the body is needed to show the patient and the healthcare professional where potential harmful lesions are located on the body. This allows for further inspection at the dermatology department.

Contents

Chapter 1

Introduction

Skin cancer is the most common form of cancer in the United States [1]. Also in Europe its incidence is high (around 16 in 100.000 people) and increasing by (on average) 3% per year [2]; with the Netherlands as the 5th highest incidence in Europe [3]. Out of all skin cancer Melanoma is the most aggressive and lethal. Especially here early detection and start of treatment is necessary to increase the chance of a positive patient outcome. Unfortunately diagnosing Melanoma proved difficult. With the naked eye in Dutch dermatology clinics a sensitivity of .79 and a specificity of 0.96 is achieved. And with the help of a dermoscope this is increased to a sensitity and specivicity of resp. .86 and .98 [4]. Here sensitivity means: the probability that a positive diagnose is correct when melanoma is present. And specificity means that a negative diagnose is correct when melanoma is not present.

To assist in a quick and accurate diagnose automated systems are proposed [5] and developed [6] to support dermatologists. The german company FotoFinder even has a system on the market that performs comparably to physicians in diagnosing melanoma from dermoscope images (a sensivity and specificity of resp. 0.95 and 0.77) [7]. But the assessment of a skin lesion by a dermatologist is only undertaken when a patient becomes worried about his or her skin and visits their general practitioner (gp) and secondly the gp refers the case to a specialist. While, as stated, early detection and a quick treatment is very important. To allow for an earlier detection self monitoring is proposed.

To aid the general public in self monitoring smartphone apps are developed, of which the most famous example is SkinVision [8]. This app allows people to take photos of skin lesions and based on the result of an AI algorithm the app might suggest to contact a specialist. The performance of SkinVision is suggested to be comparable to FotoFinder (sens: 0.95 and spec: 0.78) [8], although it is suggested that this performance might be overestimated [9].

In this thesis we propose an new way of self monitoring: STARPeople. We combine image recognition techniques with analysis algorithms from astronomy to allow users to map skin lesions on their body and signal changes. We motivate this approach in section 1.2, which gives an introduction into skin lesions and dermatology. Then we introduce source extractor in section 1.3. This software is used in astronomy and can help segment skin lesions. Consequently we explain which image processing techniques will be used in section 1.4. Chapter 2 explains the methods used in the proposed solution. Namely: data collection 2.1, image mosaicing 2.2 and sharpness detection **??**. The results are presented in chapter 3 and discussed in chapter 4. Finally we draw conclusions in chapter 5.

This project is a collaboration between Leiden Observatory [10] and the Dermatology department of the LUMC [11]. Patients have volunteered to collaborate to the study. As such all data is strictly confidential and will remain within the LUMC. The images shown in this thesis are for illustrative usage and not actual patient data.

1.1 Research objective

STARPeople users will scan their body by filming their arms and back with their mobile phone. This will result in a sequence of pictures (the frames in the movie) capturing an entire body part. From these pictures a catalogue of detected lesions will be created. It is essential to present the location of the detected lesions on the body to the user and the dermatologist. One way to achieve this would be to combine the frames in the movie into one common reference frame. Lesions found in different frames will overlap, so doubles can be removed and all frames combined will be a picture of the entire body part.

The aim of this thesis is to combine multiple pictures from a movie scanning a patients back or arm into one common reference frame. Therefor multiple techniques are explored. A literature search is performed to find possible solutions and the different solutions are tested to finally propose the technique to implement.

1.2 Skin lesions

Skin lesions are parts of the skin that have abnormal growth or appearance compared to the skin around it. Most lesions are harmless, for instance birthmarks, moles, acne, freckles, skin tags (or: acrochordons), cherry Angioma's (small red bumps that commonly appear after age 30). But they can be malignant. In that case they are called skin cancer. Skin cancer is the most common type of cancer. There are three types:

- **squamous cell carcinoma (scc)** is overproduction of squamous cells in the top layer of the skin (the epidermis). It can appear anywhere on the body, it usually develops on parts of the skin that have endured prolonged sun exposure. It's visual marking are diverse and might be a bump or growth which might crust over, a growth that's higher than the skin around it but sinks down in the middle, a wound or sore that won't heal or an area of skin that is flat, scaly and red and larger then about 2.5 cm. It grows slowly and is easily curable especially if caught early. If left untreated it can spread to other area's of the body and be lethal, but this is very rare.

- **basal cell carcinoma (bcc)** causes a lump, bump or lesion to form on the epidermis. Again this happens on the parts of skin that endure sun exposure. It looks like a small bump or scaly flat patch on the skin that slowly grows over time. Bcc rarely spreads to other regions. Though if left untreated it can grow into the body or develop to become more aggressive.

- **melanoma** is by far the most dangerous form of skin cancer. It grows quickly and has the ability to spread to any organ. The cancer develops from skin cells that produce dark pigment (melanocytes). Most melanoma's are black or brown, but they can also be pink, red, purple or even skin-colored. 30% of melanoma's develop from existing moles. But in the other 70% of cases the melanoma has started in normal skin. They appear as moles, scaly patches, open sores or raised bumps.

All skin cancer is most commonly seen in sun-exposed area's. This is because the suns UV-light damages the skin. The risk factors for developing skin cancer are (among others):

- Spend a considerable amount of time in the sun
- Get easily sunburned; have a history of sunburns
- Live in a sunny or high-altitude climate
- Tan or use tanning beds
- Have light-colored eyes, blond or red hair and fair or freckled skin
- Have many moles or irregular-shaped moles
- Have actinic keratosis (skin growths that are rough, scaly, dark pink-to-brown patches

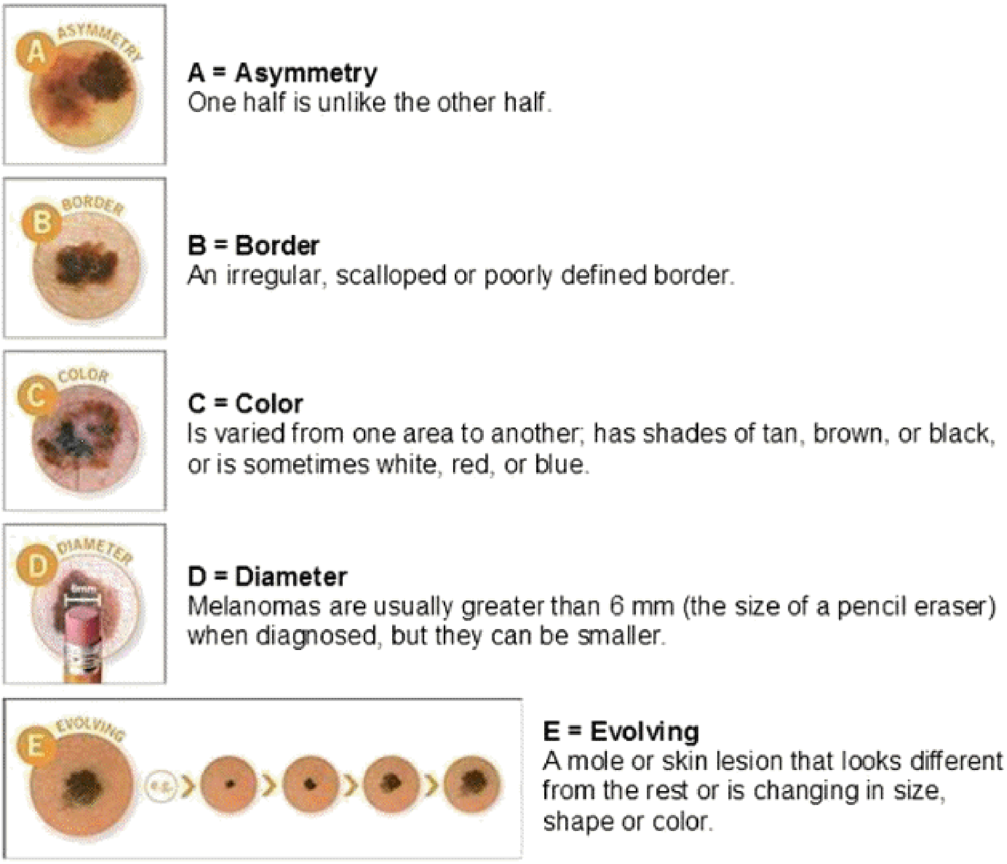
All forms of skin cancer are very diverse in appearance and, as might be clear from the above descriptions, can look very similar to each other and to benign skin conditions. This is what makes it difficult for healthcare professionals (and automated systems) to make the correct diagnose and start the right treatment. To make the distinction between benign or malignant a mnemonic method is used: the ABCDE shown in figure 1.1 [12] .

A healthcare professional uses the list shown in figure 1.1 to assess the risk a lesion has on being malignant, but for a definite diagnose a biopsy is needed. As the list shows one of the risk factors is that the lesion changes over time and an other is that the lesion differs from other lesions present on the patients body. The two systems that are mentioned earlier (FotoFinder and SkinVision) are not able to take this information in account. SkinVision processes a picture of a single lesion, so there is no comparison to other lesions or assessment of change. FotoFinder might compare a lesion to other lesions on the body, as the software works with a complete body image, but it is not able to compare to pictures taken earlier in time.

This is how we come to propose an alternative automated system: STARPeople.

1.3 Analogy between dermatology and astronomy images

Lesions are (at least on white skin) dark features on a light background. The inverse of a photograph of skin with lesions is a dark background with bright features. As shown in figure 1.2 such an image is very similar



A = Asymmetry
One half is unlike the other half.

B = Border
An irregular, scalloped or poorly defined border.

C = Color
Is varied from one area to another; has shades of tan, brown, or black, or is sometimes white, red, or blue.

D = Diameter
Melanomas are usually greater than 6 mm (the size of a pencil eraser) when diagnosed, but they can be smaller.

E = Evolving
A mole or skin lesion that looks different from the rest or is changing in size, shape or color.

Figure 1.1: *ABCDE used to classify skin lesions*

to an image of stars. As the images are visually comparable it is sensible to explore whether techniques used in astronomy can be applied to dermatology images.

Astronomy analyses digital images captured by the CCDs in telescopes. These sensors produce a high amount of data that needs processing. So efficient software has been developed to automate this. A commonly used application is Source Extractor or SExtractor developed in 1996 [13]. The software is able to automatically perform object detection, segmentation and photometry and is primarily used for large scale galaxy-survey data images. SExtractor works in 6 steps. Firstly the background and image noise is measured and subtracted from the image. Then the resulting image is filtered to smooth out small perturbations or distortions. Thirdly objects are detected with a thresholding algorithm. Because SExtractor is used for object detection of light sources each pixel is assumed to be the

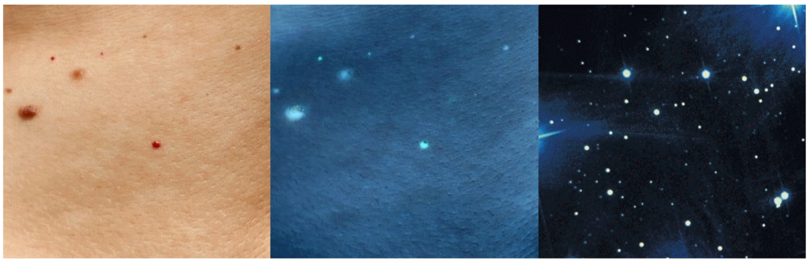


Figure 1.2: *Visualization of comparability between skin images and sky images.*

sum of the background noise and the different objects found in its vicinity. So deblending is applied to separate objects that are close together and might be detected as one object. Then the photometry is calculated to describe all found objects. For each object characteristics like magnitude, ellipse shape, size and angle are computed. Consequently each object is classified as either a star or a galaxy. This is performed by a pre-trained neural network. Finally all objects are catalogued.

In this project SExtractor will be applied to efficiently process skin photos. We aim to develop software with which people can scan their body for features with their mobile phone. Our software will detect lesions, map and catalogue them and store them. SExtractors background detection will be used to detect the skin and remove it from the image. The features that are left will be lesions. They will be detected and catalogued. The next time the user scans their body the detected lesions can be compared to the catalogue. So newly appeared lesions or lesions that changed in color or shape can be detected. This way our software would be able to give an early warning to contact a dermatologist for a consult.

1.4 Image mosaicing

As stated in paragraph 1.1 the research objective of this thesis is to explore techniques for combining different images. This process is called image mosaicing or image stitching in literature and Pandey [14] has written a complete overview of the current field in 2019. The process consists of 3 steps:

1. **Image registration** defines the projections that can transform the images from their local coordinate system to common global coordinates.

2. **Warping** or reprojection projects the images to the global coordinate

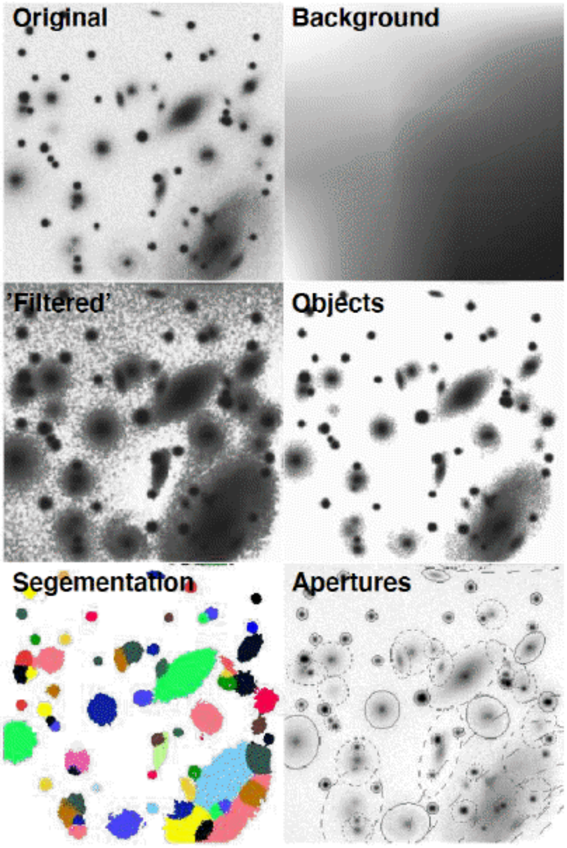


Figure 1.3: *Source extractor pipeline visualizing the 6 steps: Background Subtraction, Filtering, Object detection, Segmentation and Photometry*

system. When two images are projected to a common surface, perceptible edges might appear.

3. **Blending** or seam smoothing is the process that aims to smooth out those edges to produce a visually appealing mosaic.

The last few decades a lot of techniques that can perform these steps have been developed and applied in different areas. Even so not all problems have been solved and not one technique fits all use cases. Still existing problems are for instance illumination variation, camera rotation and zoom and moving objects in the scene [14]. This means different applications require different mosaicing algorithms depending on the specific challenges involved.

Mosaicing has many different fields of application. For example it is available in almost any smartphone to create panoramic pictures, it is used for aerial and satellite images to combine different images into a single map or terrain overview, and it is used in astronomy for instance for alligning and combining different images from the Hubble Space Telescope [15]. In the medical and biometric field image mosaicing is used for instance for confocal microscopes to extend the field of view [16].

1.4.1 Projections

A very detailed overview of techniques used for registration and stitching is given in this paper by Szeliski [17]. Figure 1.4 and table 1.1 are taken from this publication. As stated during the registration step the projections that can transform images from their local coordinate systems to common coordinates are defined. Figure 1.4 shows the names and examples for different projections between the coordinate systems of two 2D images. Each consequent projection allows for more mutations and thus more degrees of freedom (D.O.F.). In computer vision coordinates are often expressed as homogeneous coordinates (also called projective coordinates). A point on the Euclidean plane defined by: $X = (x, y)$ is represented in homogeneous coordinates as: $\tilde{X} = (zx, zy, z)$ for any non zero real number z. This coordinate system is used in computer vision to more easily represent projective transformations by matrices.

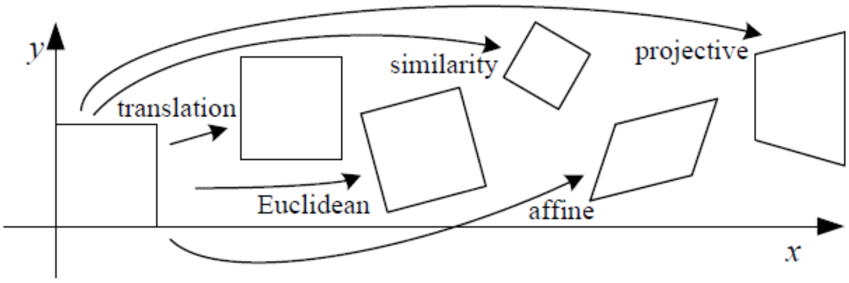


Figure 1.4: *Basic set of 2D planar transformations. Taken from [17]*

1. **Translation** is a pure translation of the image. The projection can be written as: $x' = x + t$ or:

$$x' = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \tilde{x}$$

   where $\tilde{x}$ is expressed in the homogeneous coordinate system.
   This projection has 2 D.O.F, as is clearly shown by the matrix.

2. **rigid or Euclidean** is a combination of translation and rotation. It can be written as: $x' = Rx + t$ or:

$$x' = \begin{bmatrix} cos\theta & -sin\theta & t_x \\ sin\theta & cos\theta & t_y \end{bmatrix} \tilde{x}$$

   The rotation introduces an extra variation, so 3 D.O.F.

3. **similarity** introduces one extra projection: scaling. It can be expressed as: $x' = sRx + t$ or:

$$x' = \begin{bmatrix} s \cdot cos\theta & -s \cdot sin\theta & t_x \\ s \cdot sin\theta & s \cdot cos\theta & t_y \end{bmatrix} \tilde{x}$$

   D.O.F = 4.

4. **affine** allows for all of the above plus skewing in two directions, thus adding 2 D.O.F. The mathematical representation is straightforward: $x' = A\tilde{x}$ or:

$$x' = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \tilde{x}$$

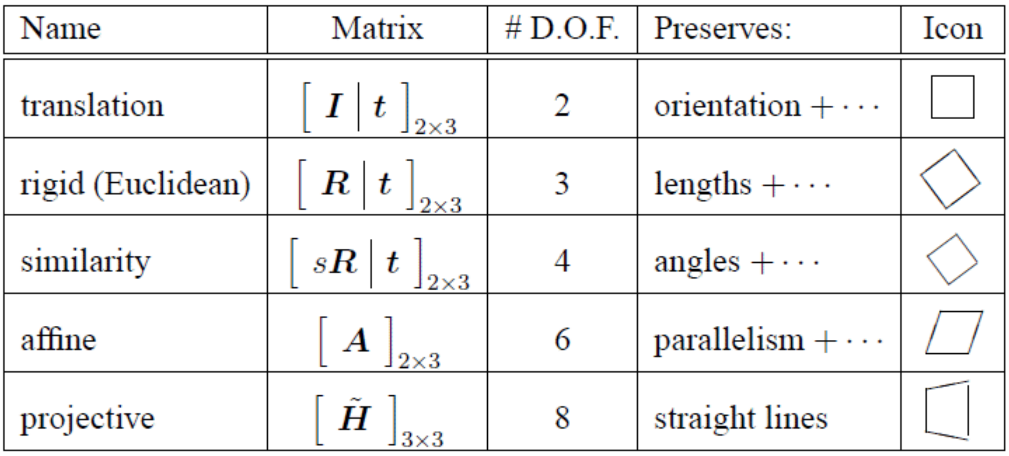
   With the limitation that A has to be invertible so that $A^{-1}A = 1$. (The projection is reversable). Under affine transformation parallel lines are conserved.

5. **projective** also known as *perspective transform* or *homography*. This transformation additionally allows for changing the angle between two non parallel lines. The angle for a line can be altered between both the X and the Y axes, so this this transformation adds 2 D.O.F. and so can be represented by $\tilde{x}' \sim \tilde{H}\tilde{x}$. Where both coordinates are homogeneous and even H is. The scale of H is fixed by setting the $9^t h$ element to 1:

$$\tilde{x}' = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix} \tilde{x}$$

A summary of these projections is given in figure 1.1.

There are multiple approaches for registration. A complete overview is given by Pandey [14]. Here we will present two methods in more detail. Firstly the direct method and secondly the feature based method.



Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\left[\, \boldsymbol{I} \mid \boldsymbol{t} \,\right]_{2\times 3}$	2	orientation $+ \cdots$	□
rigid (Euclidean)	$\left[\, \boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	3	lengths $+ \cdots$	◇
similarity	$\left[\, s\boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	4	angles $+ \cdots$	◇
affine	$\left[\, \boldsymbol{A} \,\right]_{2\times 3}$	6	parallelism $+ \cdots$	▱
projective	$\left[\, \tilde{\boldsymbol{H}} \,\right]_{3\times 3}$	8	straight lines	⏢

Table 1.1: *Projections for 2D coordinate transformations. Taken from [17]*

1.4.2 Direct method

The direct method aims to find the registration of two images (I_1 and I_2) by minimizing the pixel intensity discrepancies. This technique is commonly used in healthcare for instance for combining imaging data from different systems (MRI, CT-scans). A widely used application has been developed by Marius Staring (LUMC) and Stefan Klein (ErasmusMC): Elastix [18]. Finding the translation projection between two images would be done by moving I_1 over I_2 such that the following equation is minimized:

$$E^2 = \sum_{x,y} [I_1(u,v) - I_2(x,y)]^2 \tag{1.1}$$

where $(u,v) = P[(x,y)]$ and P is the projection that maps the coordinate system of I_1 to I_2. As this is an iterative process the technique is best used for images that are fairly similar, e.g.: only rigid transformation or, for transformations with more D.O.F., a projection that is close to the identity matrix. The challenge of using the mean square error 1.1 as error function is that difference in illumination between two images will lead to an incorrectly large error. This can be prevented by using the (normalized) mutual information as an error function; which will be introduced in section 2.3.

1.4.3 Feature based method

Where the direct approach uses all pixels to register two images, the feature based method calculates the projection matrix ($\tilde{H}$) by finding 4 common points in the two images. As projective transformation has 8 degrees of freedom and each point is described by two coordinates (x, y) so 4 common points lead to 8 linear equations that can be solved to calculate $\tilde{H}$. Selecting 4 common points can be done manually. But to process large amount of images, or in our use case frames from a movie, an automated method for finding and describing features is required. The challenge here is that the features need to be compared across different images, so different coordinate systems, this means the detection and description of the features needs to be independent of the chosen coordinate system.

Chapter 2

Methods

2.1 Data collection

For this project the data was collected in 2020-2021. This was done by filming volunteering patients at the LUMC, with a then popular phone; the iPhone7, Samsing Galaxy a71 and Huawei P30 Lite where used. The patients body parts were scanned, either the back, arms or legs. Stickers have been applied on the body to help the camera focus, this prevented the camera from focusing on the background and encouraged to keep the skin area in focus. Eight patients volunteered to be filmed and all are filmed with two cameras. The photos shown as examples in this thesis are not patient images.

The data is stored and processed on a virtual server at the LUMC to assure the data is secure and private. On the server Python version 3.8.10 is used and most processing is done with OpenCV 4.5.3. This python implementation of OpenCV offers the algorithms introduced in paragraph 1.4 like SIFT and calculating and projecting homographies.

The fact that the data was captured with a handheld mobile phone gives challenges that need to be solved in the mosaicing algorithm:

- The camera corrects focus often and not always focuses on the skin (even with the stickers applied). When the skin is out of focus there might not be enough clear features.

- It happens often that a part of the frame is in focus, while other parts are out of focus. This can even be the case if the frame has only skin in view. The camera might focus on a particular feature on the skin while other parts are blurred.

- The distance from the camera to the skin changes drastically. This is by intention, the camera is moved close on skin patches to bring details clearly on screen. However this requires feature descriptors to be independent of zoom and robust for large changes.

- The camera tilts and rotates in any direction. This means feature descriptors have to be independent of tilt and rotation and the projection matrix must be able to correct these difference between frames.

- Features can also be found in the photo area that shows background (wall / floor) by the parallax effect these features would distort the calculation of the projection. So masking (based on color) has been applied to only select areas of the frames that show skin.

- Most of the time the camera moves slow, but sometimes a lot of distance is covered. This makes it that the pipeline should be robust for both situations. If the camera travels a lot while close to the skin it might be impossible to calculate a proper projection.

- To define the projection between two frames four matched features are needed, as explained in section 1.4.1. But if one (or more) of those features is an outlier (mismatch) the projection will be very distorted. So both a sanity check on the projection matrix is needed and more then 4 matched features are needed to be robust for outliers.

- The frames in the movie are high resolution (1920 x 1080 pixels), which means the processing scripts can need a lot of computer memory. For instance loading a 1 minute movie into memory uses about 10GB.

- If all frames are projected into the frame of reference for the first frame, the orientation of that reference frame decides what the point of view for the entire image is. If the first frame of the movie is tilted, this makes the entire image tilted.

2.2 Mosaicing methods

As introduced in paragraph 1.4 building a mosaic takes three steps: registration, warping, and blending. In this section I will explain the methods to realize these steps.

2.2.1 Feature detection

In this thesis registration is mainly realized by feature selection and mapping. To allow for feature matching across coordinate systems Lowe introduced a (by now widely used) algorithm called Scale Invariant Feature Transform (SIFT) [19]. The reasoning behind the algorithm is that in order to find scale invariant features, features on different scales should be selected and described in a method independent of scale. This is done by building a set of scale space images, $L(x, y, \sigma)$ by convolving the input image $I(x, y)$ with a variable-scale Gaussion $G(x, y, \sigma)$:

$$L((x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{2.1}$$

where $*$ is the convolution operation in x,y and

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{2.2}$$

This is done for increasing σ and the set of Gaussians, produced is shown as the left stack in figure 2.1. So in essence the first Gaussian convolution removes fine grain features and with each increasing σ coarser grained features are removed. Each layer has less fine grained features. Now by subtracting L_{n+1} from L_n a difference-of-Gaussian (DOG) $D(x, y, \sigma)$ is calculated.

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{2.3}$$

This way the Gaussian convolution has removed fine grain features and the subtraction has removed the coarse grained features. Now features are shown per scale. Consequently extrema are selected in the space build up from the DOG. If a pixel is a local minimum or maximum compared to all 26 neighbors (8 in the same scale and 9 each in the scales above and below) the pixel is selected as a possible feature. On these pixels filtering methods are applied to remove features that are sensitive for noise or features that are on edges. These methods are described in detail in Lowe's paper [19].

2.2.2 Feature description

Now that we have detected features in all different length scales we need to describe them independent of scale (and orientation) to allow for comparison of features found in different images. This is done by firstly defining them in the scale in which the extrema was found ($L(x, y)$). In this

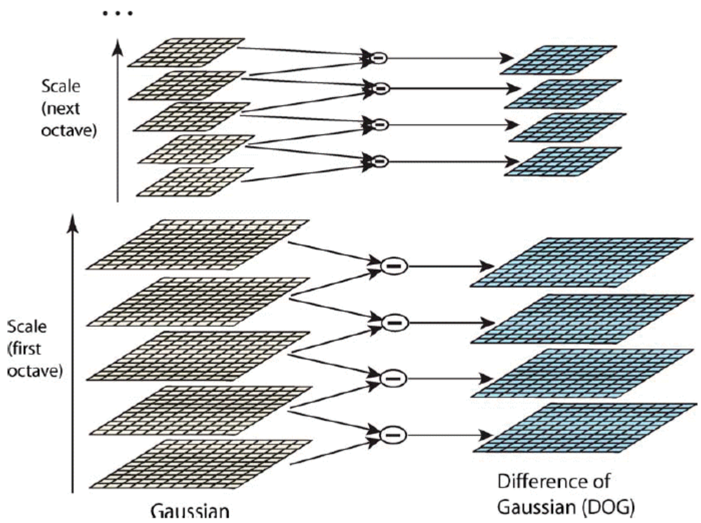


Figure 2.1: *For each scale octave gausian filtering is applied with an increasing kernel size producing a stack of 'Gaussians'. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right, fig. 1 from [19]*

.

scale for the feature a gradient magnitude and orientation are calculated:

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

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

With the location (x,y), the magnitude m and the orientation θ the features can be localized in the original image. Now the descriptor can be created. Around the location all gradients magnitudes and orientations are calculated, using the scale of the feature to select the level of Gaussian blur for the image. In order to achieve orientation invariance, the coordinates of the descriptor and the gradient orientations are rotated relative to the feature orientation θ. Now the different gradients per 4 x 4 square are added together as shown in figure 2.2. The figure shows eight directions for each orientation histogram, with the length of each arrow corresponding to the magnitude of that histogram entry. The magnitudes are combined into a vector as a feature descriptor. A 128 element feature vector is created by combining 8 of these 4 x 4 direction histograms.

After the introduction of SIFT other feature detection methods have been developed and implemented. In this thesis also FAST [20] and ORB [21] will be covered. FAST is an implementation of the Harris corner detection method [22] with a focus on computational speed (hence the name)

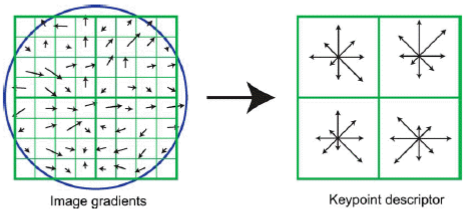


Figure 2.2: *Schematic representation of the calculation of the features descriptor used by SIFT. Fig. 7 from [19]*

and ORB is a later improvement on FAST, adding a feature descriptor, which was not part of the FAST algorithm and more optimization on computation time, in order to make it possible to run feature detection on a live camera feed. A lot of publications have compared these (and other) techniques in different use cases, for instance [23], [24] and [25]. Showing that different techniques excel in different scenario's. Although most publications conclude that SIFT is the most accurate and robust technique, but comes with an higher computational burden on the system.

2.2.3 Warping

Features found in consequent frames are matched to each other with nearest neighbor matching; the euclidean distance is taken between the vectors describing the features. With this method a match for each feature is selected; even if there is no corresponding feature in the other frame. To filter the wrongfully matched features Lowe introduced a step called Lowe's ratio [19]. This step selects only the features for which the distance between the nearest neighbour is smaller then 80% of the distance between the second neighbour. This forces the difference between the best neighbor and the second best to be large. The reasoning is that the distance between two wrongfully matched features is, in general, more similar then the distance between a rightfully matched set of features and a wrongfully matched set. Examples demonstrating the effectiveness of this filter are shown in appendix **??** in figures 1, 2 and 3.

With the features in two frames matched we can calculate the projection. As described in paragraph 2.1 the camera has free movement, which means translation, rotation, tilt and zoom can change between frames. It follows from paragraph 1.4.1 that the projection between two frames is projective and is described by a $3x3$ homography matrix. To calculate the projections we require 4 matched points. If more then 4 matches are available the RANSAC algorithm can be used to make the projection calculation robust for outliers (eg. wrongly matched features) [26]. RANSAC stands for Random sample consensus, and works by repeatably sampling 4 random points multiple projections are calculated. The best one is selected by using each projection to project all points from frame 1 onto the reference frame of frame 2. The projection that leads to the least outliers is selected as best.

A digital image is a discreet representation of a view on a continuous world. The view is encoded as a grid where each cell (pixel) represents either a color intensity or gray scale value on a scale from 0 to 255. This grid has an origin $(0, 0)$ and a X- and Y-axis, stretching from 0 to 1920 (for the X-axis) and 0 to 1080 (for the Y-axis). When an image is projected into a new reference frame the pixels will not exactly line up with the new grid, so interpolation is necessary. In this thesis we used linear interpolation.

Another phenomena that has to be corrected for is that the origin and outer border will not align. As the camera shifts in any direction the next frame will often be projected outside of the original grid. To make grid available for the new pixels, the grid has to be extended along the X and or Y axis. And, as negative axes do not exist, if the next frame is lower and/or to the left of the original image, the original image has to be translated.

This is done by first projecting the 4 outer corners onto the new grid. Based on their location the size and translation of the new grid is calculated and, if required, a translation projection matrix is created to shift the original image. This procedure is represented in algorithm 1

2.2.4 Blending

As described in paragraph 1.4 blending aims to smooth edges that appear in the resulting image from combining multiple images. This is a delicate procedure; as the human eye is very sensitive for contrast difference along lines. Robust blending techniques are part of ongoing research. In this thesis image edge smoothing is left for future research.

In this thesis we blend images by either copying the second image over the first, or by selecting the sharpest of the two images first and then adding the parts from the less sharp image that do not appear in the sharpest.

2.3 Projection quality

As mentioned in section 1.4.2 a metric to quantify how good two images have been mapped onto each other is the mean squared intensity differ-

Algorithm 1 Find image dimensions, with H as the homography matrix

```
procedure FINDDIMENSIONS($image, homography, base\_img$)
    Determine image dimensions $y, x$
    Initialize image corners $base\_p1, base\_p2, base\_p3, base\_p4$
    Initialize $max\_x, max\_y, min\_x, min\_y$ as $None$
    for $pt$ in $[base\_p1, base\_p2, base\_p3, base\_p4]$ do
        Project point $hp \leftarrow$H $\cdot pt$
        Update $max\_x, max\_y, min\_x, min\_y$ if necessary
    end for
    Set $min\_x, min\_y$ to 0 or minimum values
    Adjust $max\_x$ and $max\_y$ by base image size
    Initialize translation matrix $move\_H$ as identity matrix
    if $min\_x < 0$ then
        Update $move\_H[0, 2]$ and $max\_x$
    end if
    if $min\_y < 0$ then
        Update $move\_H[1, 2]$ and $max\_y$
    end if
    return $(min\_x, min\_y, max\_x, max\_y, move\_H)$
end procedure
```

ence or mean squared error. However it is sensitive to illumination differences between the images. For a more robust metric we can use the property mutual information, defined in information theory.

Information theory is the field of science that studies ways of quantifying, storing and communicating of information. The field was introduced by Claude Shannon in his groundbreaking paper ”A Mathematical Theory of Communication” in 1948. One of the key concepts is entropy, which measures the amount of uncertainty or randomness in a system. One way of quantifying entropy in a set of variables X in terms of bits is Shannon entropy:

$$H(X) = -\sum_{x \in X} P_X(x) log_2(P_X(x))$$

Where P_X is the probability distribution for collection X. An other way of phrasing this is that the entropy is the amount of information that is stored in the collection of variables and Shanons entropy shows the minimum amount of bits required to store that information. The mutual information (MI) between two sets of variables (X, Y) quantifies the amount of information obtained about one set if the variables of the other set are known. It is defined as:

$$MI(X.Y) = \sum_{y \in Y} \sum_{x \in X} P_{(X,Y)}(x, y) log \left(\frac{P_{(X,Y)}(x, y)}{P_X(x) P_Y(y)} \right)$$

Where $P_{(X,Y)}$ is the joint probability distribution. MI is symmetric so: $I(X, Y) = I(Y, X)$ and it is non negative. $I(X, Y) = 0$ means no information is gain about Y by knowing X. So X and Y are completely independent. With this definition MI is unbound: $MI \in [0... \text{inf})$, which makes it hard to assess when the registration is good. To solve this normalized mutual information (NMI) is often used[27] [28]. To normalize the MI is divided by the individual entropy's:

$$NMI(X.Y) = \frac{MI(X, Y)}{H(X)H(Y)}$$

We can use this metric for the area of overlap between two images. If the images are projected correctly the NMI between the two images is close to 1.

2.4 Implementation

2.4.1 Image registration

For registration we have tested the direct implementation with Elastix [18] and we have compared different feature based approaches. SIFT (as introduced in section 2.2.1) is compared to newer and faster methods: FAST [20] and ORB [21]. The different feature selection methods are compared on robustness and speed and this is done quantitatively (average amount of features found, minimum amount of features found and frames with less then 4, 10 or 20 features found, processing time) and qualitatively (visual inspection of selected features).

Subsequently the features are described and matched. This too is reported up on: the average amount of matches before and after Lowe's ratio is reported, and again the number of frames with less then 4 and 8 matches. Matching is done with a FLANN (Fast Library for Approximate Nearest Neighbors) algorithm for accelerated nearest neighbors matching in high dimensional spaces [29].

2.4.2 Blending

Initially the projected image was added completely over the base image. However this meant that if the second frame was out of focus, while the first was sharp, the resulting image would lose details. This is solved by calculating the sharpness in the area of overlap for both images. As a measurement for sharpness the gradient is calculated. An image with a higher gradient for the same skin surface will be sharper than an image with a lower gradient. Eventually, to accord for the fact that often images were sharp for parts of the frame, I divided the area of overlap in nine pieces as shown in image 2.3 and selected for each subsection the sharpest section. As a measurement for sharpness I have used edge detection with a Sobel operator in the x- and y-direction. The total sharpness of a patch is calculated by taking the mean of the sum of the absolute values in each direction.

2.4.3 Pipeline

With all elements described we can combine them into a mosaicing pipeline. The algorithm is shown in algorithm 2. We start with initializing the FLANN feature matcher and by reading first frame. The first frame is named 'result' because all subsequent frames will be added to this one.

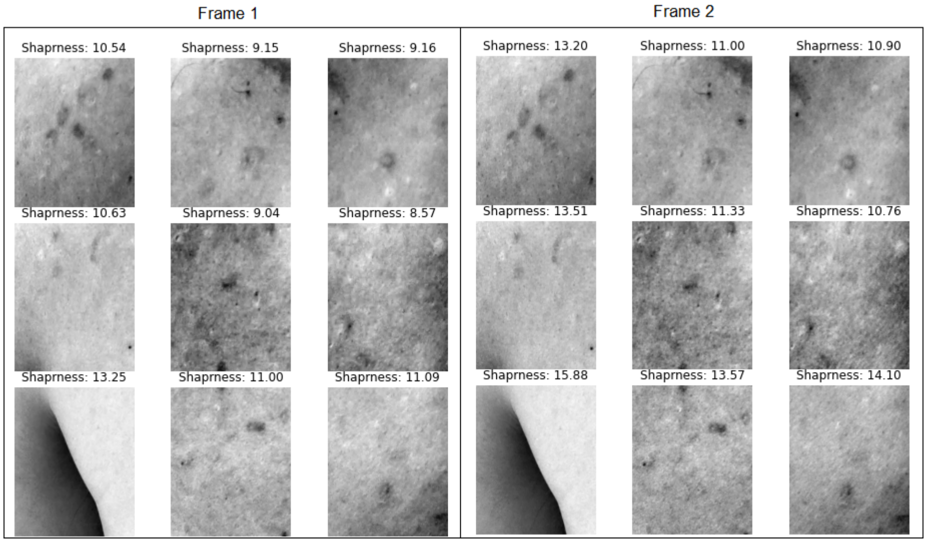


Figure 2.3: *Example of dividing the overlap between two frames in 9 patches and selecting the sharpest patch to add to the result image.*

A mask is created based on color to select the area that contains skin and discard background. The mask is calculated in the YCbCr color space with fine tuning by trail and error. Values between $[0, 133, 77]$ and $[235, 173, 127]$ are regarded as skin. After selecting these colors the holes are filled, as some lesions and shadow area's are to dark to be detected as skin.

Subsequently in the while loop each step the next frame is read, masked and the projection from this frame to the previous is calculated. With the inverse projection the resolution of the new image is calculated, as this is required for warping both frames. The new frame is combined with the existing result and this combination is renamed as result. If no homography could be found all next steps are skipped and the next frame is read.

2.5 Different experiments undertaken for robust mosaicing

To successfully produce one image out of a movie of between 500 to 1500 frames, each of the frames has to be projected correctly to a common reference frame. And as mapping the frames is a recurrent process each mismatched frame might lead to either a accumulating error in the final image or a complete break in the mapping process. A lot of experiments are performed to come to a robust process. Tested mosaicing variations:

Algorithm 2 Video Mosaicing Pipeline

```
procedure PIPELINE($VideoPath$)
    $flannIndex \leftarrow$ createFlannMatcher$()$
    $video \leftarrow$ readVideo$(videoPath)$
    $frame \leftarrow$ readFrame$(video)$
    $mask \leftarrow$ getMask$(frame)$
    $result \leftarrow$ applyMask$(frame, mask)$
    while video hasNextFrame do
        $frame2 \leftarrow$ readFrame$(video)$
        $mask2 \leftarrow$ getMask$(frame2)$
        $H, success \leftarrow$ findHomography$(result, frame2, mask, mask2, flannIndex)$
        if $success$ then
            $H^{-1} \leftarrow$ computeInverseProjection$(H)$
            $(X_{min}, Y_{min}, X_{max}, Y_{max}, moveH) \leftarrow$ findDimensions$(H^{-1})$
            $resultWarp \leftarrow$ warpPerspective$(result, moveH, (imgW, imgH))$
            $resultMask \leftarrow$ warpPerspective$(mask, moveH, (imgW, imgH))$
            $frame2Warp \leftarrow$ warpPerspective$(frame2, H^{-1}, (imgW, imgH))$
            $mask2 \leftarrow$ warpPerspective$(mask2, H^{-1}, (imgW, imgH))$
            $overlap \leftarrow resultMask \cdot mask2$
            $result \leftarrow$ combine$(resultWarp, frame2Warp \cdot mask2, overlap)$
        end if
    end while
    return $result$
end procedure
```

- Minimizing pixel intensity discrepancies using s-ITK and Elastix as introduced in section 1.4.2.
- Different feature selection matching methods: SIFT, FAST and ORB are compared to select the most robust.
- Initially for each frame the projection to the previous frame was calculated. First I looped over all frames and calculated all projections from $frame_{N+1}$ to $frame_N$. Then I warped $frame_{N+1}$ to $frame_N$, the result would be warped to $frame_{N-1}$, this result to $frame_{N-2}$, etc. However this led to an accumulating error. All the imperfections in each projection added up and the final result was very distorted.
- I solved this by combine each frame to the complete image. So I calculate the projection from $frame_2$ to $frame_1$ and combine them. Then I calculate the projection from $frame_3$ to the result, and so for each following frame.
- I improved the pipeline by skipping frames that have less then 10 features, as they are often out of focus.
- I tried to further improve by skipping frames that give only a small translation, the idea was that combining multiple largely overlapping frames would add more noise then data. But this did not improve stability.
- I tried to refine the masking of the background (non skin parts of the image) based on color detection. I extracted the 10 most occurring colors and only included areas of the image that where close enough to these colors (with a euclidean distance threshold). After filtering I filled the holes in the mask to include skin areas, because the non skin areas are always on the edge. It did work, but was computational heavy so added an hour on average of processing time.
- some even with RANSAC and enough features, sometimes the calculated projection would be way off. To prevent using these projections the homography matrix is checked. If zoom is more then a factor 1.2 or less then a factor 0.8 I drop the frame.
- from two matched frames select sharpest area of overlap.
- from two matched frames select sharpest area of overlap per patch.

The results of these experiments will be shown and discussed in the chapters Results and Discussion.

2.6 Lesion catalogue

As described in section 1.3 the software Source Extractor will be used for lesion detection. The steps for building the lesion catalogue are as follows:

1. For each frame mask the none skin elements.
2. Invert the image in all three color channels.
3. Apply Source Extractor background subtraction to remove the skin from the image.
4. Remove hairs with canny edge detection as they give a falls detection.
5. Source Extractor object detection in each color channel.
6. Source Extractor object description.
7. Remove overlapping findings from the different channels and keep the largest.
8. Warp all lesions for every frame to the common reference frame.
9. Remove overlapping lesions and keep the largest again.

With these steps a catalogue of all lesions found on the patients body part can be build. Each lesion is described by source extractor with characteristics used for describing galaxies for instance among others: flux as a measurement for brightness and size; x and y coordinates (centre, min and max); a,b as the major and minor axes, as the found object is approximated to be an ellipse and theta as the angle between the major axis and y-axis. These characteristics can be used along with thumbnails from one or multiple frames as shown in figure 2.4.

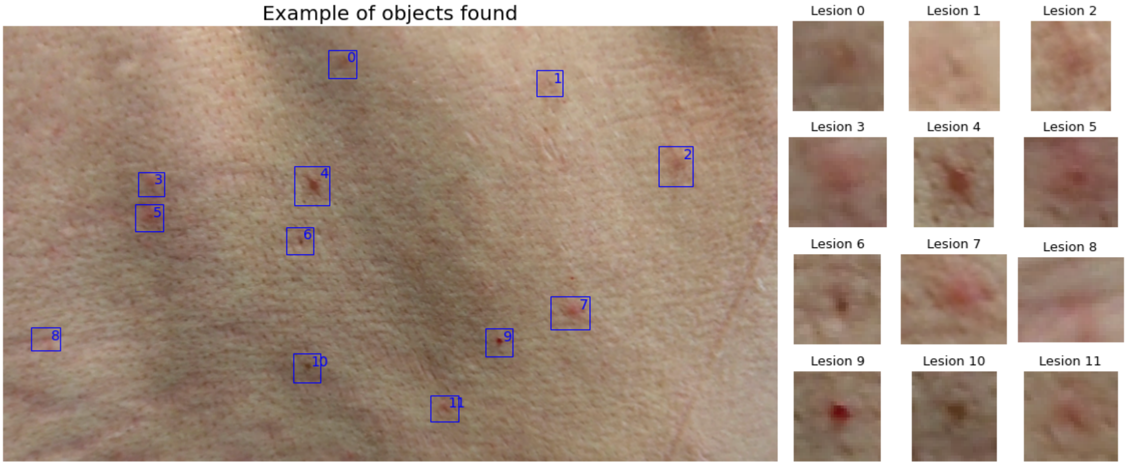


Figure 2.4: *Figure displaying an example of the lesions found by source extractor on 1 frame.*

Chapter 3

Results

As addressed in section 2.1 the data collected for this thesis is patient data and as such sensitive. To be able to support the explanations and results with examples, I have recorded a movie of my own back in the same manner as the actual data was collected. Figures showing body images will be taken from this example movie. Figure 3.1 shows frames taken from this movie as to give a clear picture of the material. Every 40 frames a thumbnail is shown, with the frame number in the top left. The sequence of images shows how the camera moved along the body area. Something to notice is that the white balance can change during recording (for instance as is visible in the last 4 thumbnails). It is clear that in this example every frame greatly overlaps with the previous one. This is not the case in all data points, as mentioned in the list of challenges in section 2.1.

In this chapter I will present the results. First I will present the result from the comparison of the different feature detection techniques, then I will evaluate the matching process and consequently the mosaicing process. And finally I will elaborate on further improvements made during the process.

3.1 Feature detection

As explained in section 2.4.1 there are different feature extraction methods. To compare these methods I have selected three promising algorithms mentioned in literature and available in OpenCV. I have analysed all movies with each method (SIFT, FAST and ORB). The results are shown in figure 3.2. The amount of features found differs greatly, SIFT and FAST can find thousands of features but also just 2. As we are interested in find-

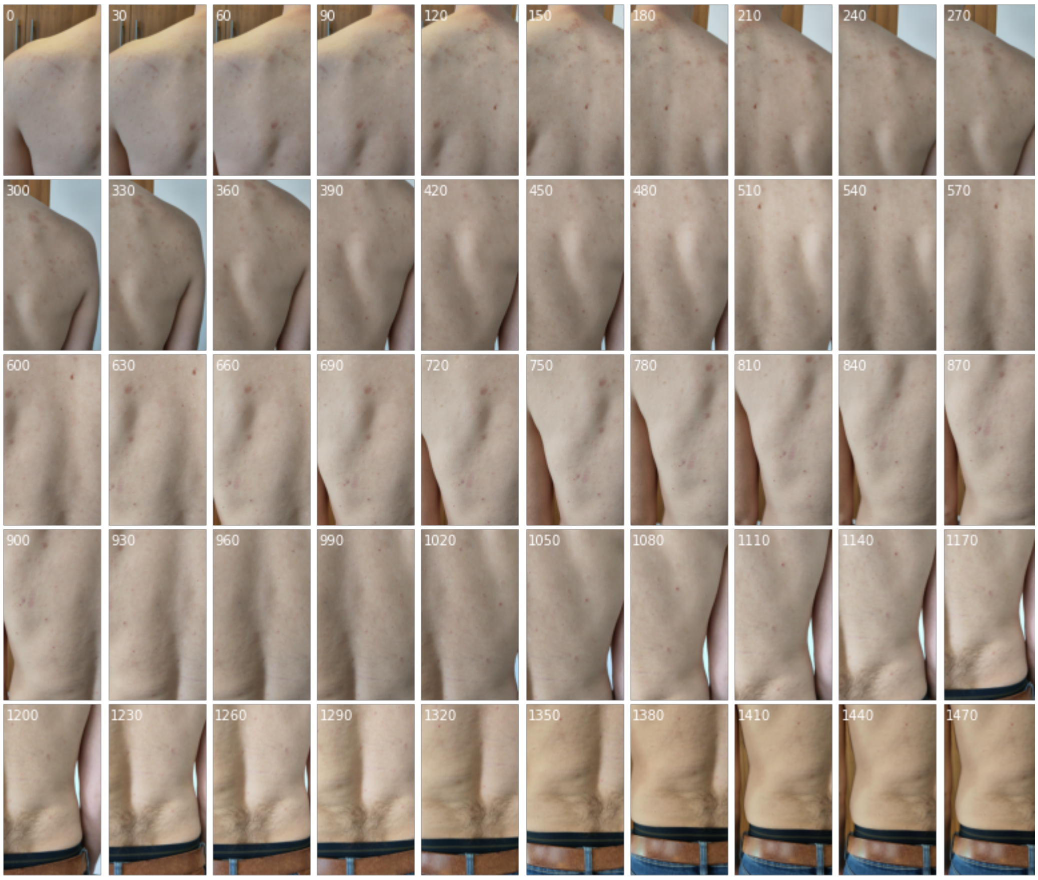


Figure 3.1: *Figure showing a thumbnail for every 40 frames from the movie used as example material. The frame number is shown in the top left.*

ing at least enough features to calculate a good homography between two overlapping frames, the plots are limited to frames where 200 or less features have been found. For how many frames this is the case differs per method. In each graph the amount of frames incorporated in the plot is shown in the text box, along with the amount of movies.

The results have been separated per phone type and per body part to be able to detect differences between these. In each graph the continues lines represent the density plots and the dashed lines give the mean amount of features found over the frames in the graph. The density plot is the amount of features found in each frame, normalized to set the surface below the graph to 1.

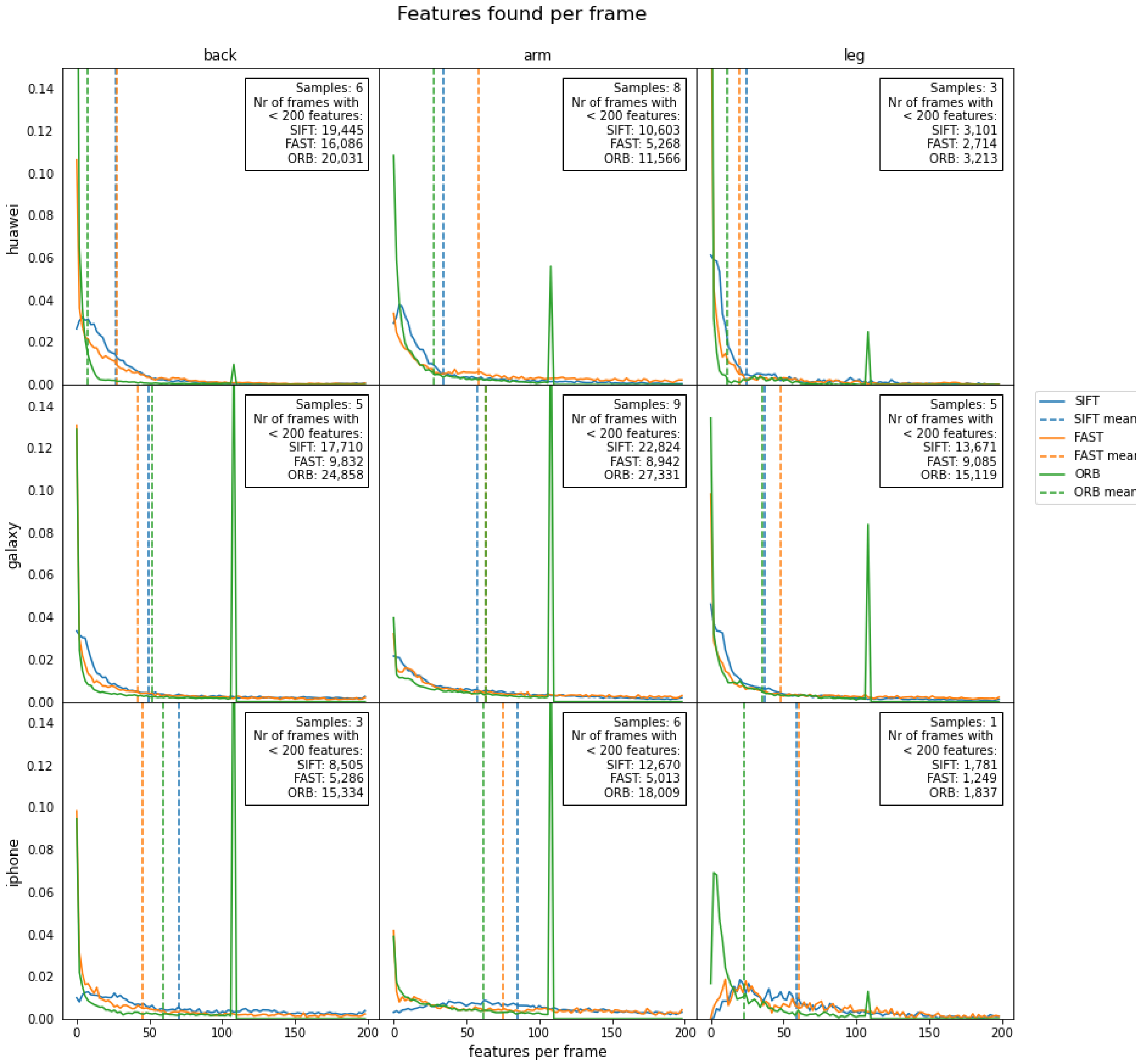


Figure 3.2: *Density plots showing the amount of features found in each frame for different body locations and different phones. Only frames where less then 200 features where found are shown, as we are selecting a feature detection method that finds enough features in frames were features might be hard to detect.*

The ORB graphs always show a peak at 109 features because the method is limited to detecting 109 features per default, to optimize for computational speed. The plots and means for movies recorded with the Huawei phone have a significantly lower mean amount of features found. The reason for this is that Huawei has a skin smoothing filter build into the camera to remove lesions from pictures. We were unable to disable this mechanism and as the project revolves around detecting and labeling lesions this phone seems not suitable for data acquisition, movies from the Huawei phone will be left out further analysis. Something else to notice is that there is only 1 movie taken with the iPhone for legs, so this graph has less data then the others and thus a less smooth distribution.

Looking at means in the different plots it stands out that there is no clear 'best' method. All methods sometimes have lowest and in other graphs highest mean. Anytime we do not find at least 4 features in common between two frames the homography will break, and with exactly 4 there is no outlier detection. This can lead to the wrong homography and will lead to a distorted result. It does stand out that ORB often has the highest peak for frames with 0 features, so this method most often cannot find any features or at least 4.

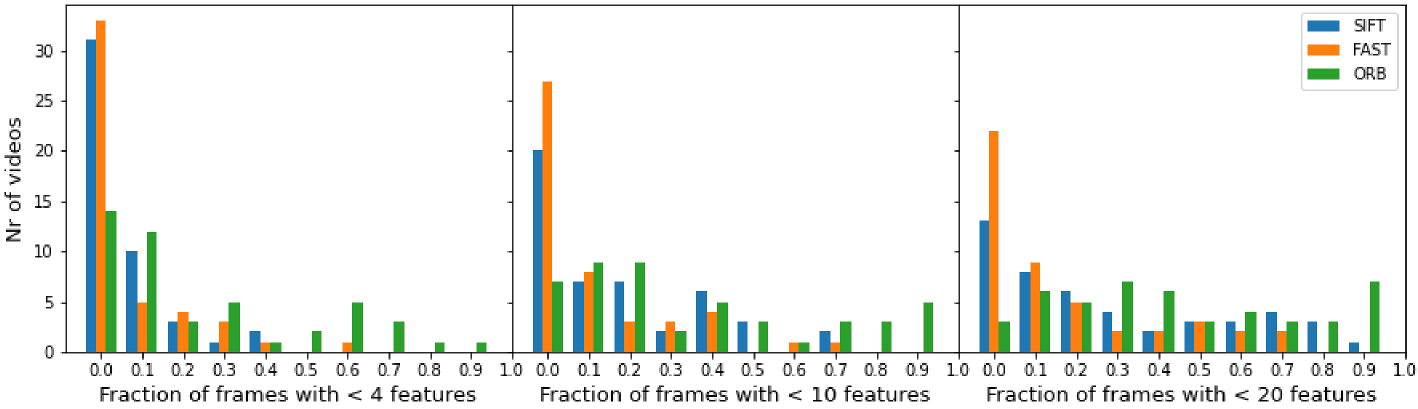


Figure 3.3: *Histograms showing the fraction of frames for each movie with less then a certain value of features found.*

Figure 3.3 shows histograms counting the amount of movies for different fractions of frames with less then a certain value of features found. For instance the graph to the left shows that around 30 movies have a fraction between 0 and 0.1 with less then 4 features found for SIFT and FAST, while ORB has only 15 movies with that same fraction. We see the amount of movies with this fraction between 0 and 0.1 quickly drops if we raise the amount of features. To make a correct mosaic with all frames, all frames would need to have an adequate amount of features. This graph shows this will be challenging (if not impossible) for the collected data. The graph clearly shows FAST and SIFT outperform ORB in detecting the minimum required amount of features.

Figures 3.4 and 3.5 show 2 examples of images and the detected features. The features are marked with a circle and a line. For SIFT and ORB the circle size represents the size of the feature and the line represents the orientation. The orientation is defined by the axis along which the gradient is sharpest. The colors of the different features are appointed at random.
It is visible that as soon as clothing with structure appears in view the amount of features explodes, especially for the FAST method. The figures with SIFT features demonstrated that SIFT detects features on different length scales.

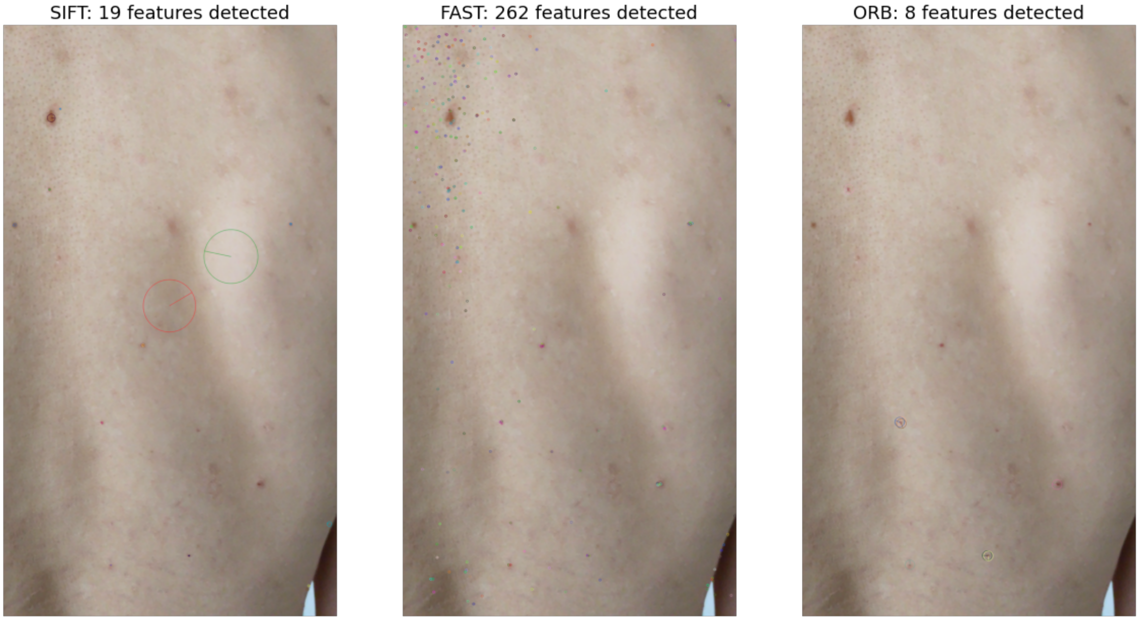


Figure 3.4: *Example frames with the difference in detected features for the different methods (SIFT, FAST and ORB).*

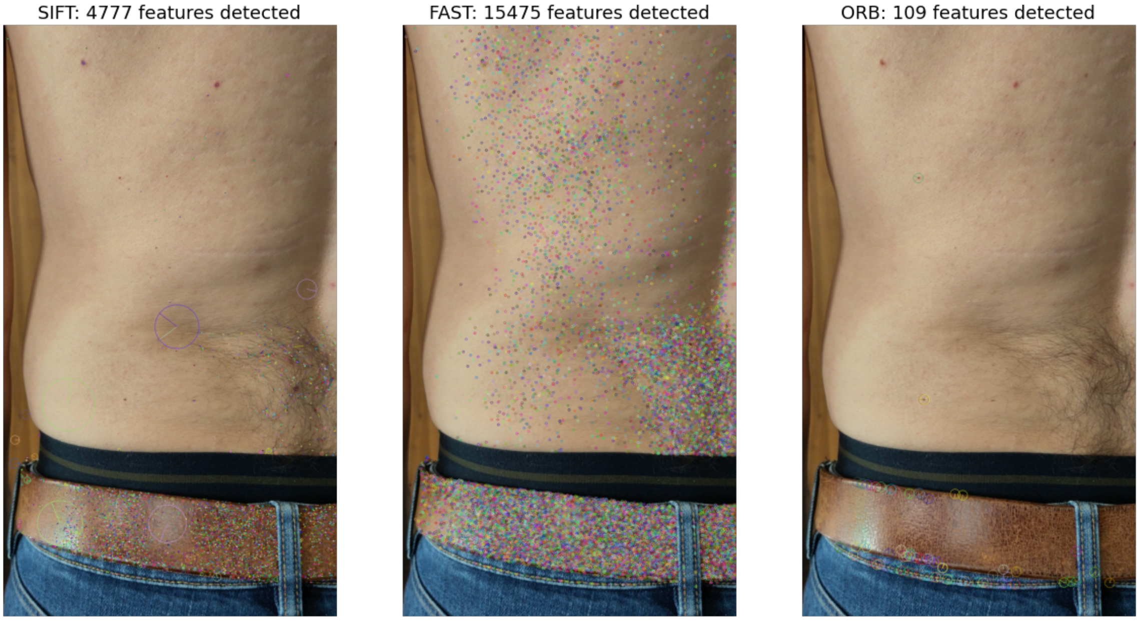

SIFT: 4777 features detected
FAST: 15475 features detected
ORB: 109 features detected

Figure 3.5: *Second set of example frames with the difference in detected features for the different methods (SIFT, FAST and ORB).*

3.2 Feature matching

In this section I show the results for feature matching done with both the SIFT feature detection and description and the ORB detection and description. I have opted to compare SIFT and ORB because they are very different. SIFT is optimized for quality at the cost of computation time. While ORB is designed to run at live camera feeds, it is an implementation of the same detection algorithm that FAST uses, but slightly enhanced and optimized for speed. It also comes with it's own descriptor mechanism [23].

Graph 3.6 shows the result of feature matching between two sequential frames. The large peak at zero far exceeds the y-axis, so the value for zero is given in the text boxes. If less then 9 features were found the amount of features was set to zero and no homography is calculated. My reason for choosing 9 as minimum is as follows: to allow for outlier detection more then half of the features need to be correct. So we chose 7 as a minimum. Next assume that 80% of the frames overlap then these 7 features need to appear in the overlap area, which means that both frames would need $7/.8 = 8.75$ features evenly spread out.

The plot unfortunately demonstrates that very often we will not be able to calculate the homography from one frame to the next. An example of a frame without enough features is shown in 4 in the appendix 5.2 . It shows that if both the skin is smooth and the camera is out of focus the feature detection algorithms do not find enough features. In the patient videos the camera moves closer to the skin regularly, so there are more frames that are out of focus. This explains the high peak. The graph extends a long way along the x-axis to a maximum of 7184 (features found in 1 frame), but our region of interest lies in the frames were features are sparse. So the plot is limited to 100 features. Two things stand out:

- SIFT finds features more often then ORB, this matches with the findings in section 3.2 and confirms SIFT is a better feature selection mechanism then ORB for the research question at hand. As such I will use SIFT for the mosaicing process, results are presented in the next section.
- In movies recorded with the iPhone 9 or more features are more often found. This shows that the iPhone camera produces movies in which features are better detectable. Features are in essence detected by gradient, so this suggests that the iPhone movies are sharper in general.

3.3 Mosaicing

From the performed analysis it is evident that not all frames can be used for the mosaicing process, as for some frames (around 10% on average) SIFT does not detect enough features and for others not all features can be matched to the features found in the frames before or afterwards. As described in paragraph 2.2 I solved this by ignoring frames with less then 9 frames. But between the remaining frames there still were not always enough matching features between sequential frames. The difficulty of a linear stitching process (combing frame N to $N-1$, and $N-1$ with $N-2$) is that a break between frames would cause the and of the process. I created a more robust process by looking further for frames with feature that could be matched. If no homography could be calculated between frame N and $N-1$, we would try again with the homography of $N+1$ to $N-1$, and kept trying until we reached $N+20$.

If still no projection could be established, we would try between $N+1$ and $N-2$, and then $N+2$ and $N-2$, and so forth until eventually $N+50$

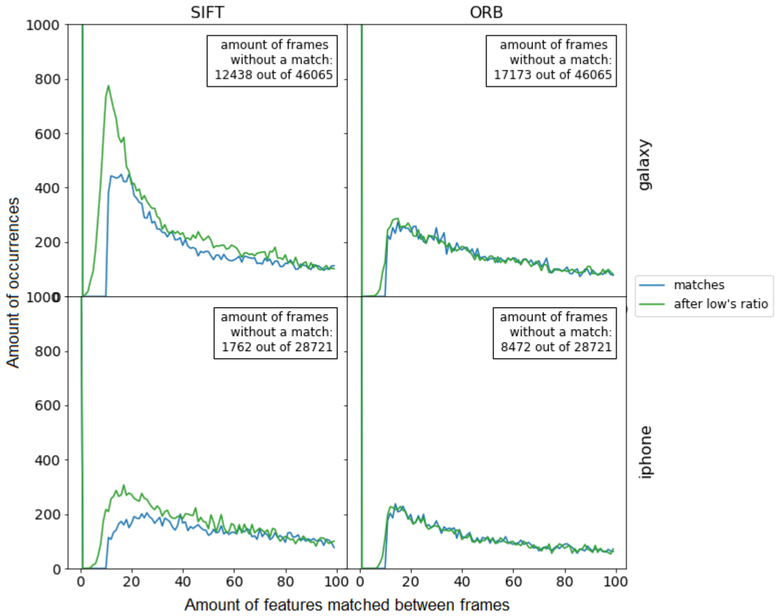


Figure 3.6: *Graphs showing the amount of times a certain amount of features is matched. The amount of matched features is on the x-axis. The amount of times it occurs is on the y-axis.*

would be match to $N - 50$. And if even then no match could be found or if the homography found for the combination with the highest amount of matches was incorrect I started a new mosaic with the next frame. Homographies are tested for unrealistic amount of translation (more then 500 pixels) or to much skewing (more than a factor 2.5).

The result of this process on our example movie is shown in figure 3.7. It shows that from the movie 5 images have been constructed. The first is a combination of 83 frames, leading into what seems to be proper reconstruction of the patients shoulder. Then there is a break of apparently some frames that are to much out of focus. The next constructed images is the result of 166 combined frames and this picture shows the weakness of combining each frame to its predecessor. There is no correction for small errors in the homography and so they can start to add up. In this example it is visible that the frames start to shear and this increases with each next frame being add on.

The combination in the bottom right is the last one made before OpenCV crashes without an exit code or error message. The last frame combined is the result of a miscalculated homography with to much zoom and rotation. I suspect that to make room for the next frame (which will be even more deformed, and thus hugely spread out) the image canvas is increased by so much that it exceeds the maximum image size that Opencv allows for.

From these results we can learn 2 things:

1. Combining the $N + 1^{th}$ frame to the N^{th} frame comes with the disadvantage of accumulating calculation errors.

2. When two frames have, in theory, enough features the calculated homography sometimes still is completely off. In the patient data I have seen more extreme examples.

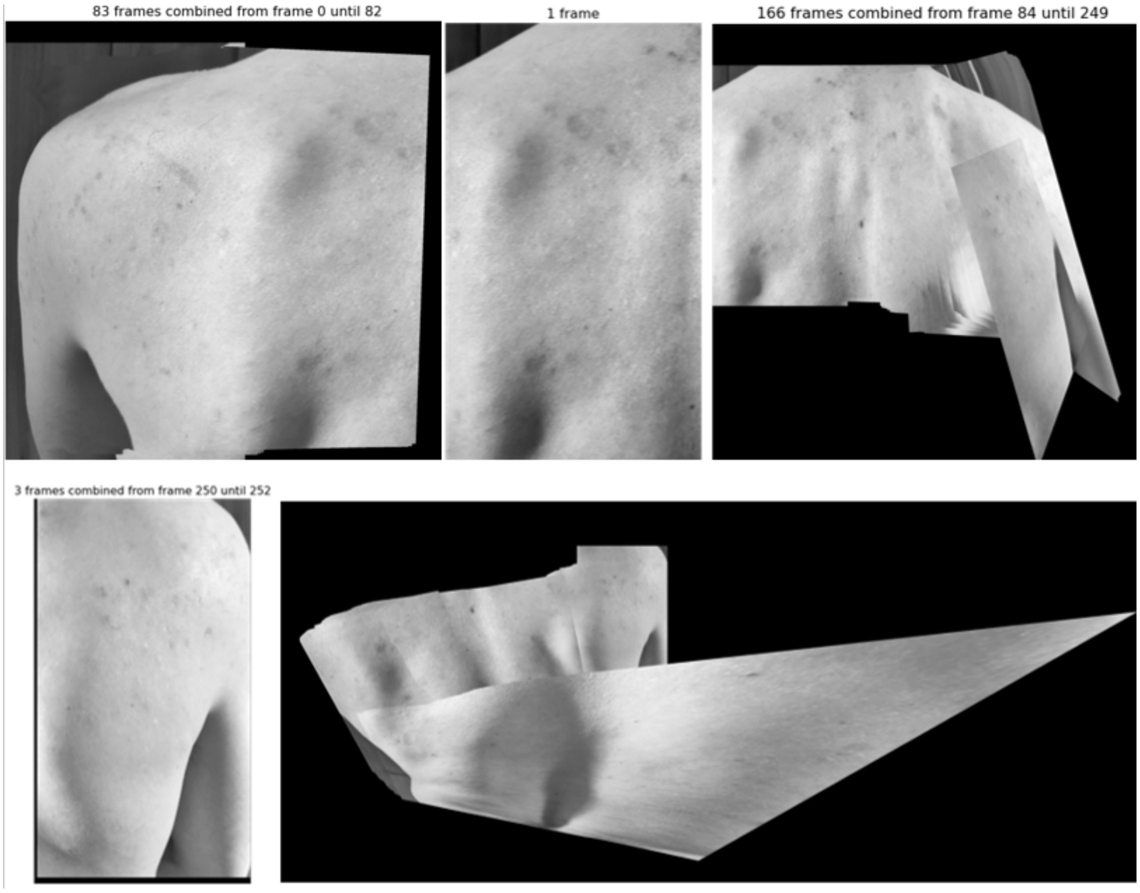

83 frames combined from frame 0 until 82
1 frame
166 frames combined from frame 84 until 249
3 frames combined from frame 250 until 252

Figure 3.7: *Results of warping each frame to the reference frame of the previous and combining all frames into the reference frame of the first.*

3.4 Mosaicing improvements

To avoid the accumulating error we have switched to calculating the homography between the result image (with all previously combined frames) and the next frame. This has an additional advantage: if the next frame overlaps with earlier frames, that information will be in the image and so can be used for feature matching.

One thing I noticed was that features found in the background cause deformation. This follows from the theory discussed in 1.4. Homography projects a plane onto plane in another reference frame. So if the features used for calculating the projection are not in a single plane the parallax effect between the features deforms the projection. To prevent this from happening we created a masking filter. Based on color we selected the parts of the image that have skin for feature detection and prevent features from being detected in the background.

As described in section 2.1 not all frames are in focus and often only parts of the frame are in focus. In the previous approach the next frame was always added on top, so overwriting all information that was already in the picture. To correct for out of focus image parts we have calculated the sharpness of the image in 9 parts as described in section 2.4.2. The example is shown in figure 2.3. Only these parts that are sharper than the existing image are added to the mosaic. The parts that are less sharp are ignored.

In the previous section I showed that a erroneous projected image interrupts the stitching process. To avoid erroneous projected frames from being added to the result image, I aimed to verify the validity of the calculated homography. I hypothesised that the matrix giving the final projection in figure 3.7 will be significantly different from normal projections, as normally the differences between two frames will be very small.
I have analysed how to recognise incorrect homography matrices and created a function to check the matrix and reject it if certain thresholds are exceeded. To analyse the homographies I have referred back to the different projection matrices explained in section 1.4.1. Using these relations I have calculated for a projection matrix the zoom, angles and the translation:

$$H = \begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} \\ x_{1,0} & x_{1,1} & x_{1,2} \\ x_{2,0} & x_{2,1} & x_{2,2} \end{bmatrix}$$

- $Zoom = (x_{0,0} * x_{1,1}) + (x_{0,1} * x_{0,1})$
- $Angle_1, Angle_2 = cos(x_{0,0}/Zoom), cos(x_{1,1}/Zoom)$

- $Angle_3, Angle_4 = sin(x_{1,0}/Zoom), -sin(x_{1,1}/Zoom)$
- $Translation : dx, dy = x_{0,2}, x_{1,2}$

If we plot these for every projection we see a smooth trajectory for correct projections and sudden deviations for errors. As shown in figure 3.8

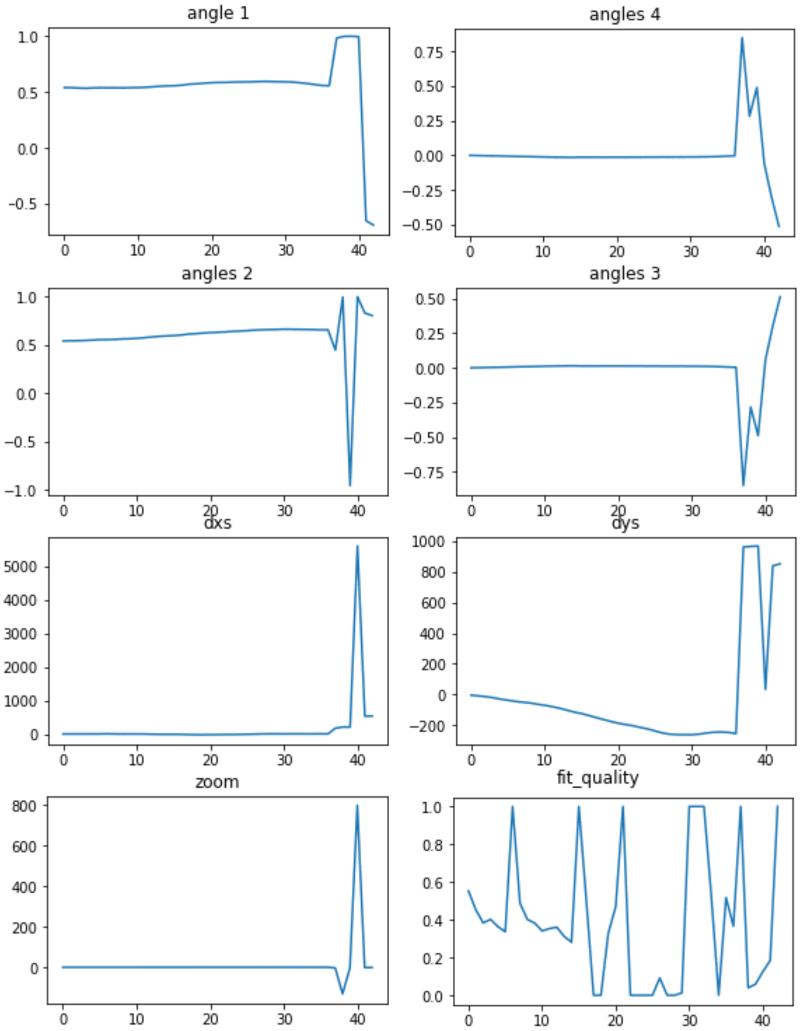


Figure 3.8: *The homography characteristics plotted for every frame in a movie. The frames are on the X-axis.*

From the outliers in the graph we can make safe estimations for tolerance values above which homogrpahy matrices should be rejected. After analysing multiple movies I came to these values:

- $Zoom > 0.5$ or $Zoom < 1.5$

- $abs(dx - dx_{previous}) > 500$
- $abs(dy - dy_{previous}) > 500$
- fit quality < 0.3

The result of the mosaicing process with all improvements implemented is shown in figure 3.9. It shows that even with these improvements the process does not run successful for an entire movie. Even so the result is improved. Three images show combinations of more then 100 frames that seem to be reasonable correct. The first (a combination of 267 frames) looks very well combine and one wonders why the mosaic could not be build further. I will show what happened in the next frames in the discussion. The first result is followed by a period of images with not enough features. They result in a lot of dropped frames and some combinations of a smaller amount of frames, this is the case after each break. Examples of shorter combinations are shown in the right column of figure 3.9. In general around 100 frames are dropped between to successful results. After some time the algorithm picks up again, in a combination of 85 frames, that becomes very distorted. Once the image is so distorted that the next match cannot be made the process breaks again.

3.5 Projection quality

As introduced in section 2.3 we aim to use normalized mutual information as a measurement for projection quality. We hope to see that correct projections score high in mutual information (close to 1) and erroneous projections low (close to 0). Further more we would expect that if a lot of features can be matched between frames, the homography calculation is robust, so the resulting projection should score high.

In figure 3.10 the NMI and RMSE are plotted as functions of the log of the amount of matched features between frames. The graph is very spread out and there is no clear relation between the two entities; especially for the Samsung Galaxy phone. For the iPhone there is a slight correlation, but it is negative for the NMI and positive for the RMSE. Reversed from what we expected. So are expectations were incorrect and it is questionable if the NMI (or RMSE) can be a usefull measurement for projection quality.

In figure 3.11 the graphs for the amount of matches and NMI are plotted for each frame that is added to the mosaic. It shows, for instance for result 2 and 3, that a normal NMI lies between 0.2 and 0.8. And that values both below and above are projections gone wrong.

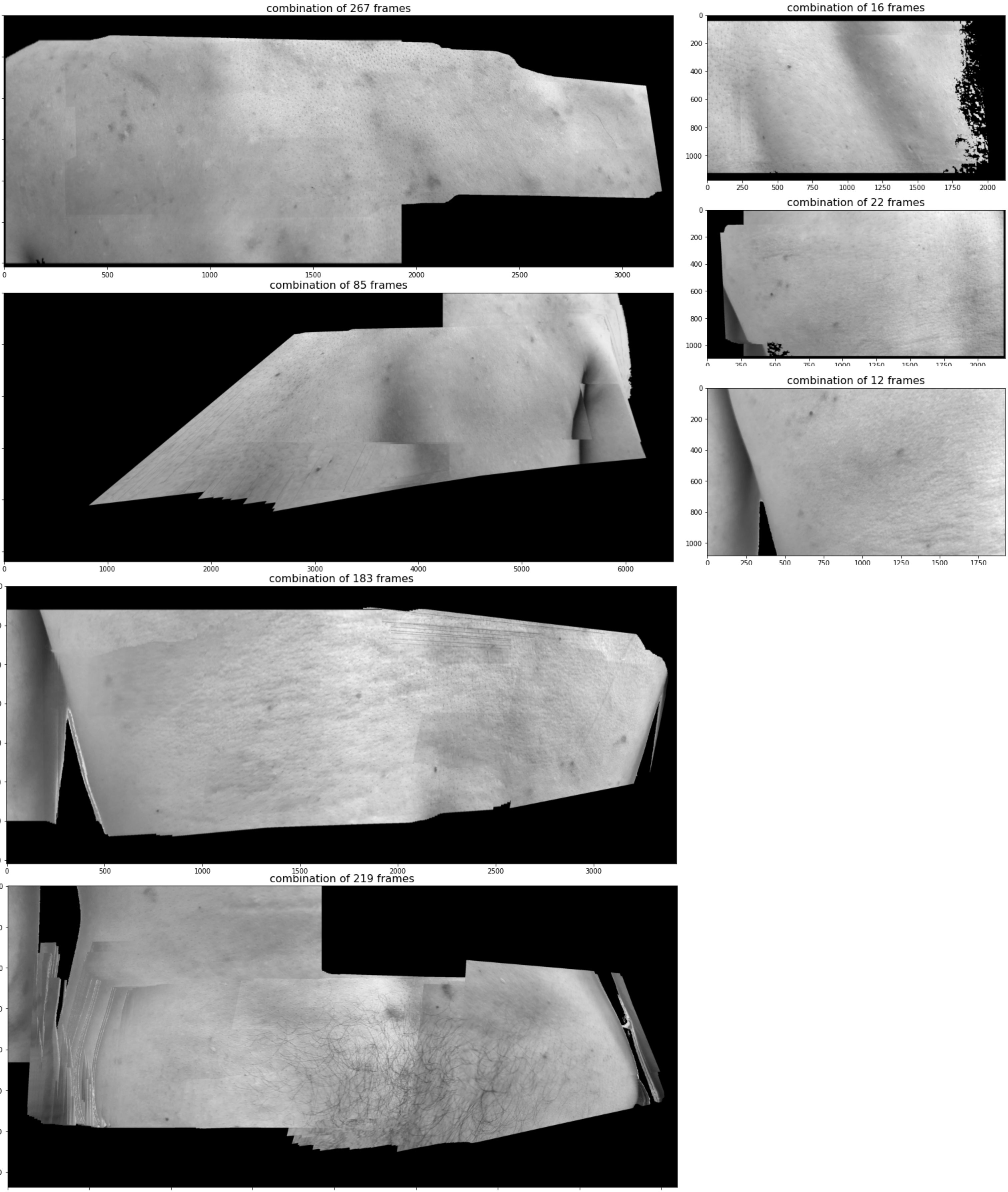


Figure 3.9: *The result of adding improvements to the mosaicing pipeline.*

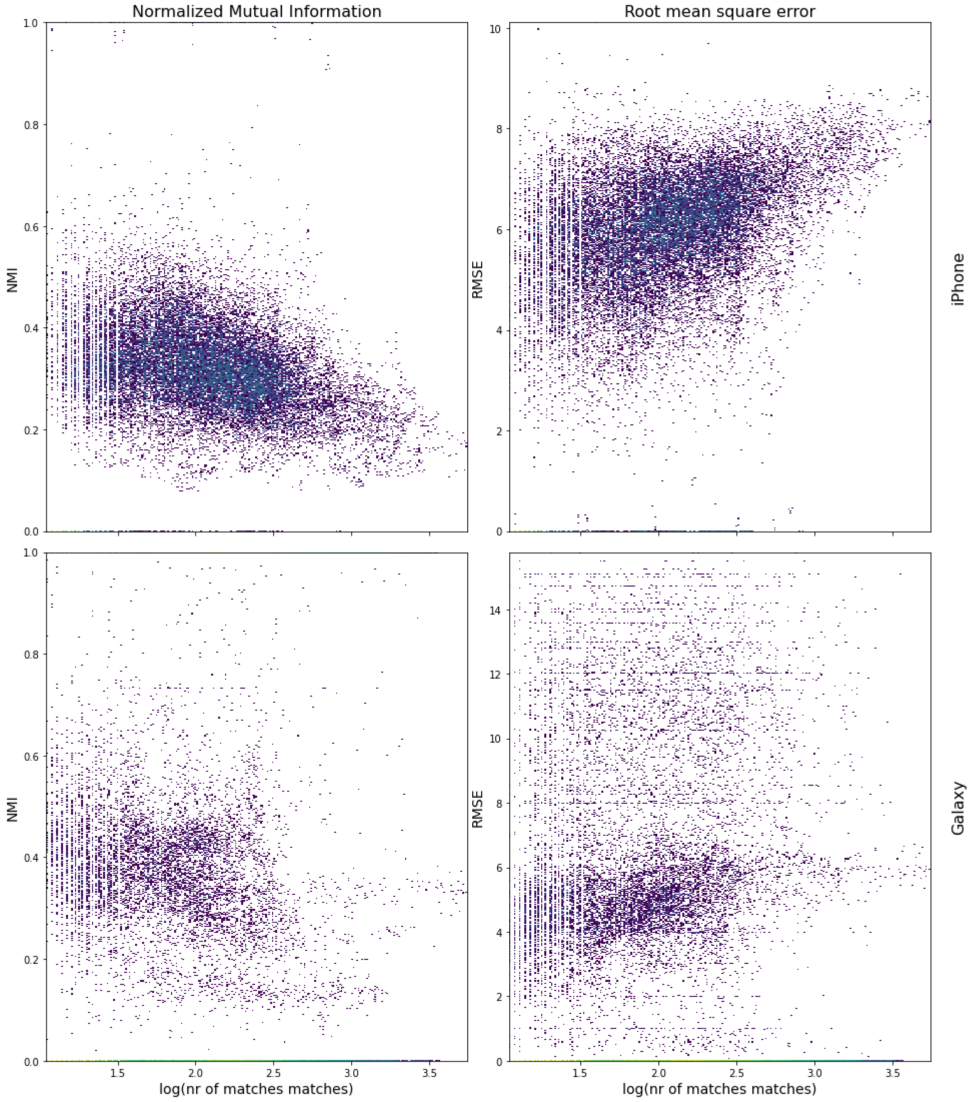


Figure 3.10: *The normalized mutual information and root mean squared error plotted versus the log of the amount of matched features between two frames. If no match between the features could be found, both the NMI and the RMSE were set to 0.*

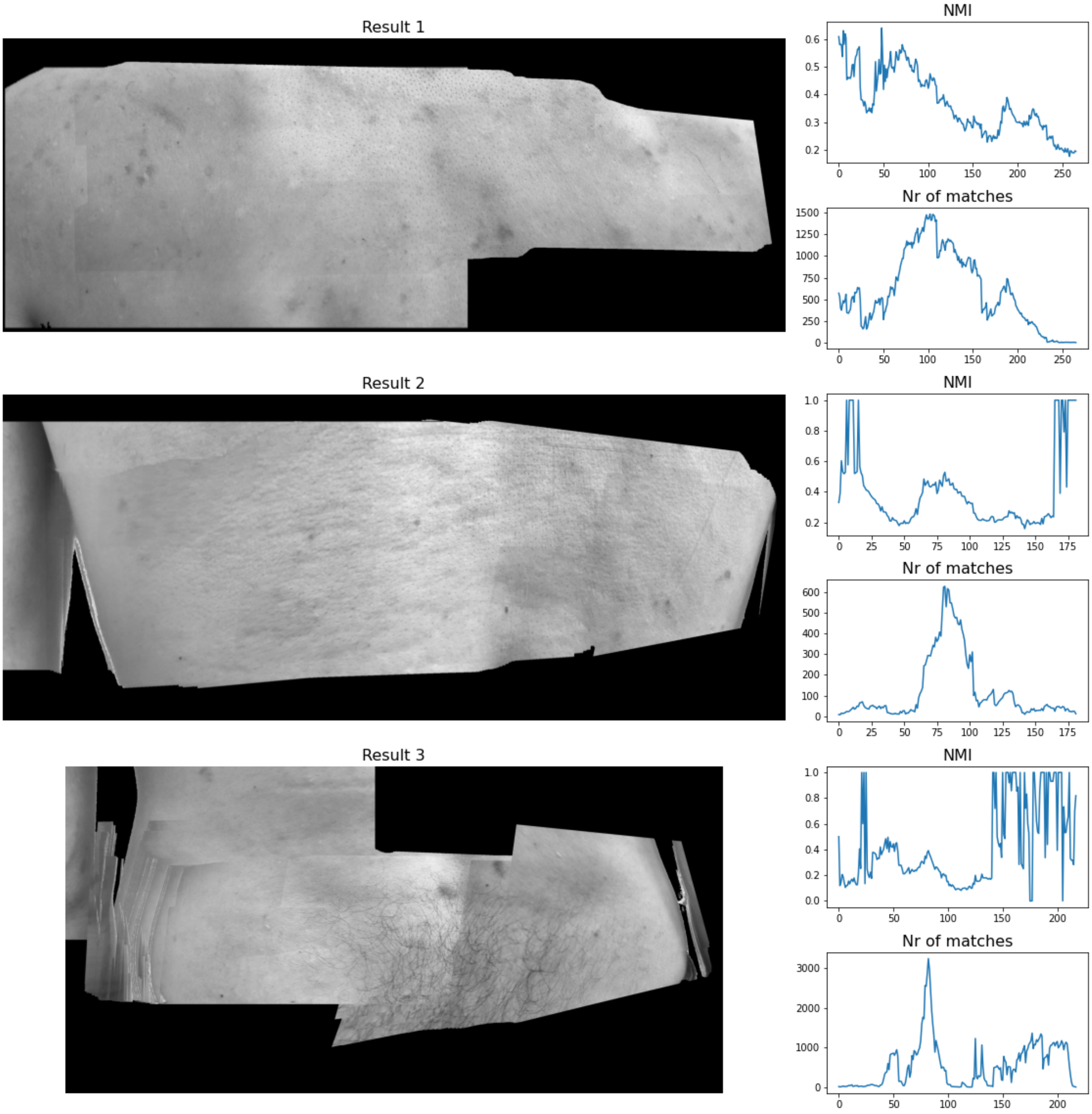


Figure 3.11: *Mosaic results with graphs showing the NMI and amount of matches for each frame in the mosaic.*

Chapter 4

Discussion

In this chapter I will reflect on the results presented in chapter 3. I will analyse the results per step in the mosaicing process. I will start with the feature detection step where I select the most promising detection algorithm. Then the learnings from feature matching will be evaluated and finally the mosaicing process will be discussed. I close of with a reflection on possible options to further improve on the results presented in this thesis.

4.1 Feature detection

In section 3.1 I have compared different techniques for automated feature detection and shown that there are large differences in quality and computation time. By analysing all recorded movies we have seen that with the techniques found in literature it will be impossible to combine all frames in the movies into one image.

This might not be needed, as the time between frames is around 33 ms (the cameras record with 30 fps). This means that the frames are very similar and we can freely discard frames that are to blurred or uniform. Even so there are moments during the recording process were the camera is moved quickly and very close to the surface, these moments might cause a break in the process.

The methods popular in literature (SIFT, FAST and ORB) are compared and the results shown in figures 3.2 and 3.3. From the results we conclude that FAST and SIFT are most promising, while ORB under performs. We select SIFT as detection mechanism because it detects features on different length scales, evenly spread over the image and because it comes with a good descriptor. Which means we can use the results for feature matching. FAST is only a detection mechanism and does not have a descriptor mechanism and no orientation for features as mentioned in [23]. ORB is a implementation of FAST were orientation of the feature and a descriptor are added, both optimized to be light in computation time at the cost of quality. Our analysis show that in our use case we need the rigor of SIFT and cannot afford to use the faster mechanisms. The choice for SIFT as most robust is confirmed by papers covering a more elaborate comparison like Tareen et al. [25] and Karami et al. [23] which both conclude that SIFT is the most accurate and robust, at the cost of computation time.

4.2 Feature matching

To test different feature matching possibilities the features found with both SIFT and ORB were used to find the matches between sequential frames. Figure 3.6 shows the amount of features matched between two frames both before and after applying Lowe's ratio, as mentioned in section 2.2. The graphs show a significant difference in quality between the iPhone and the Samsung Galaxy. For the Galaxy phone twice as many frames do not have enough matched features between them compared to the iPhone. Even so, even for the iPhone with SIFT 6% of frames do not have enough matched features (versus 27% for the Galaxy). With ORB these numbers are even higher: 29% with the iPhone versus 37% with the Samsung Galaxy. From this we conclude that movies recorded with the iPhone have higher potential to be mosaiced correctly and that SIFT is the only feature detection and description technique that has potential to successfully calculate the homography in 94% of frames.

4.3 Mosaicing

In the previous results and by care full visual assessment of the movie material (findings listed in section 2.1) we already concluded that creating one mosaic per movie will be challenging. The first approach we explored is to calculate the homography from each frame to the previous. By iterative warping $frame_N$ to the reference frame of $frame_{N-1}$, and the combination to frame $frame_{N-2}$, etc. until we reach $frame_1$ we would combine all frames into the same reference frame and could have combined them all in one image.

The result of performing this process on our example movie is shown in figure 3.7. As discussed in section 3.3 this approach comes with the problem of building up errors. Additionally we notice that even though we check the homography for certain quality norms, still the result image can become very deformed.

4.4 Experiments to improve mosaicing

We identified several improvements that were listed in section 3.4 and helped improve the mosaicing proces. Not all experiments improved the process though. For instance we tried to skip 10 frames each time, this helped speed up the process and was no problem for parts of the movie with sharp frames and slow movement. But it would break at the same points, where frames were out of focus and movement was faster. Plus there was no guarantee that the 10% of frames that were used were the sharp images. To avoid breaks in area's of faster camera movement we calculated the translation and only added frames that were translated more then 500 pixels. This way we hoped to prevent blurring by blending a lot of frames and adding significant parts of the image with each merge of frames. But again, it did not lead to less breaks in the mosaicing process and also it seems that, if the homography calculation is correct, the blending of two frames does not add a lot of distortion to the resulting image.

I tried to calculate the mask dynamicaly for each frame (instead of based on a one time set skin color), by selecting the 10 majority colors for the frame and adding every pixel that had roughly that color to the image. For the resulting mask I would fill the holes to come to a complete mask. This gave rough edges, but further more worked very well. The disadvantage though was computation time. it tripled the processing time for each movie, which already was significant.

I tried if there was a difference between features found in color or gray scale images. But the found feature were identical. I also varied the use of padding. In most online examples i found, images where masked so that the first and last rows and columns where not used (for instance the first and last 100 rows/columns). I tried to leave this masking out, as to have more area to search for features, but this gave lines around the image where pixel intensity was to low or to high. I think this is the result of interpolation. When the image is warped the new pixel grid does not align with the original grid. So pixel intensities have to be interpolated. For the pixels at the edge of the image no information is present for half of the surrounding area. OpenCV uses nearest neighbor interpolation. This will benefit from neighbors at all sides of the pixel. I assume that, given a masked area, the neighboring pixels are used for interpolation at the mask edge. It thus is important to add the padding mask before warping, so it can be used during the interpolation calculations projection.

4.5 Improved mosaicing

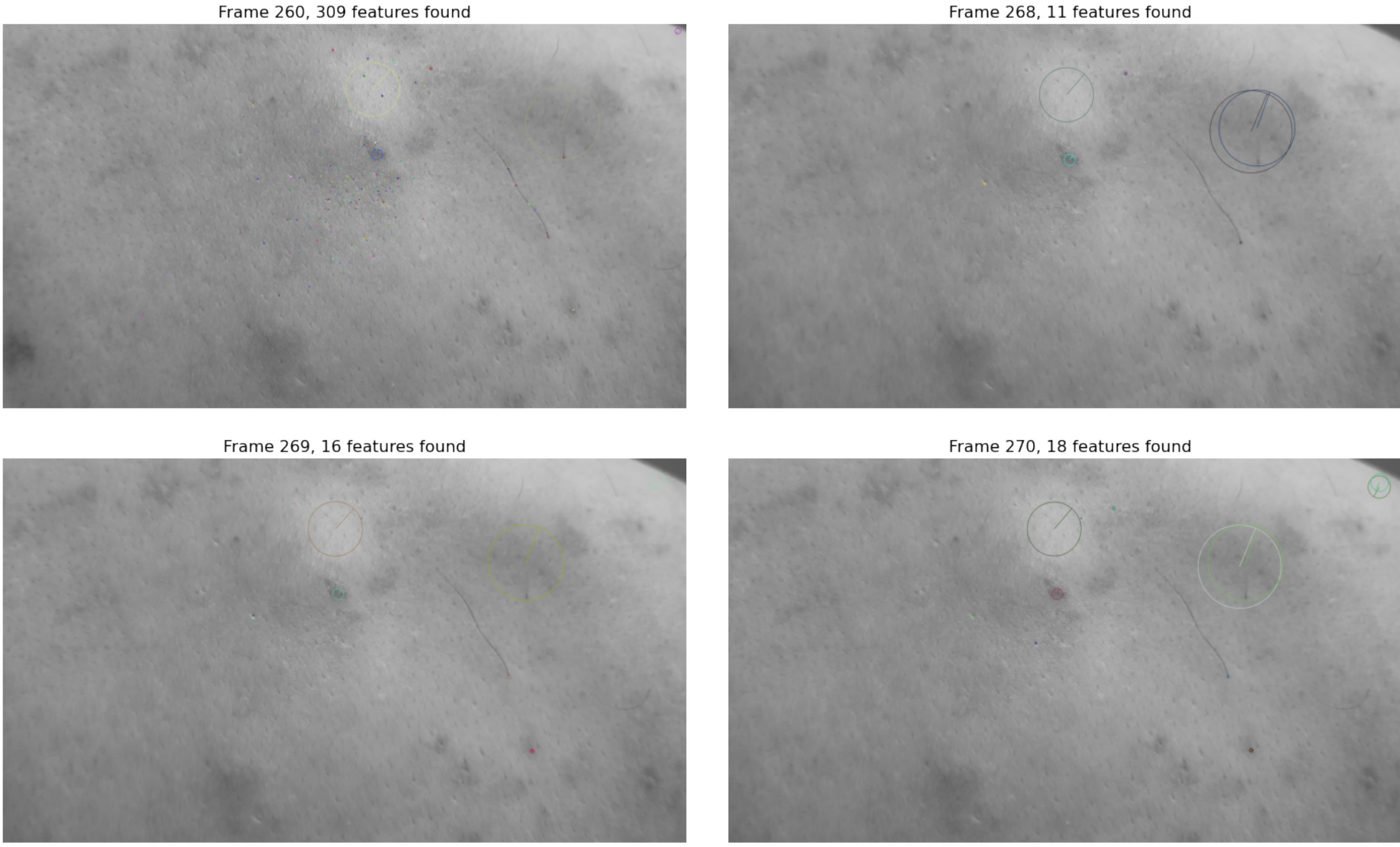

Frame 260, 309 features found
Frame 268, 11 features found
Frame 269, 16 features found
Frame 270, 18 features found

Figure 4.1: *4 examples of dropped frames in the mosaicing process.*

The improved process still does not lead to a successful mosaic. As mentioned in section 3.4 sequences of successful combined frames are followed by sequences of dropped frames. For instance the first result in figure 3.9, combining 267 frames, combines frames 0 until 267 (frame nr 260 is dropped because of a miscalculated homography). The detected features in the dropped frames are shown in figure 4.1. They show a sudden drop in amount of features, which is explained by the fact that the frames after

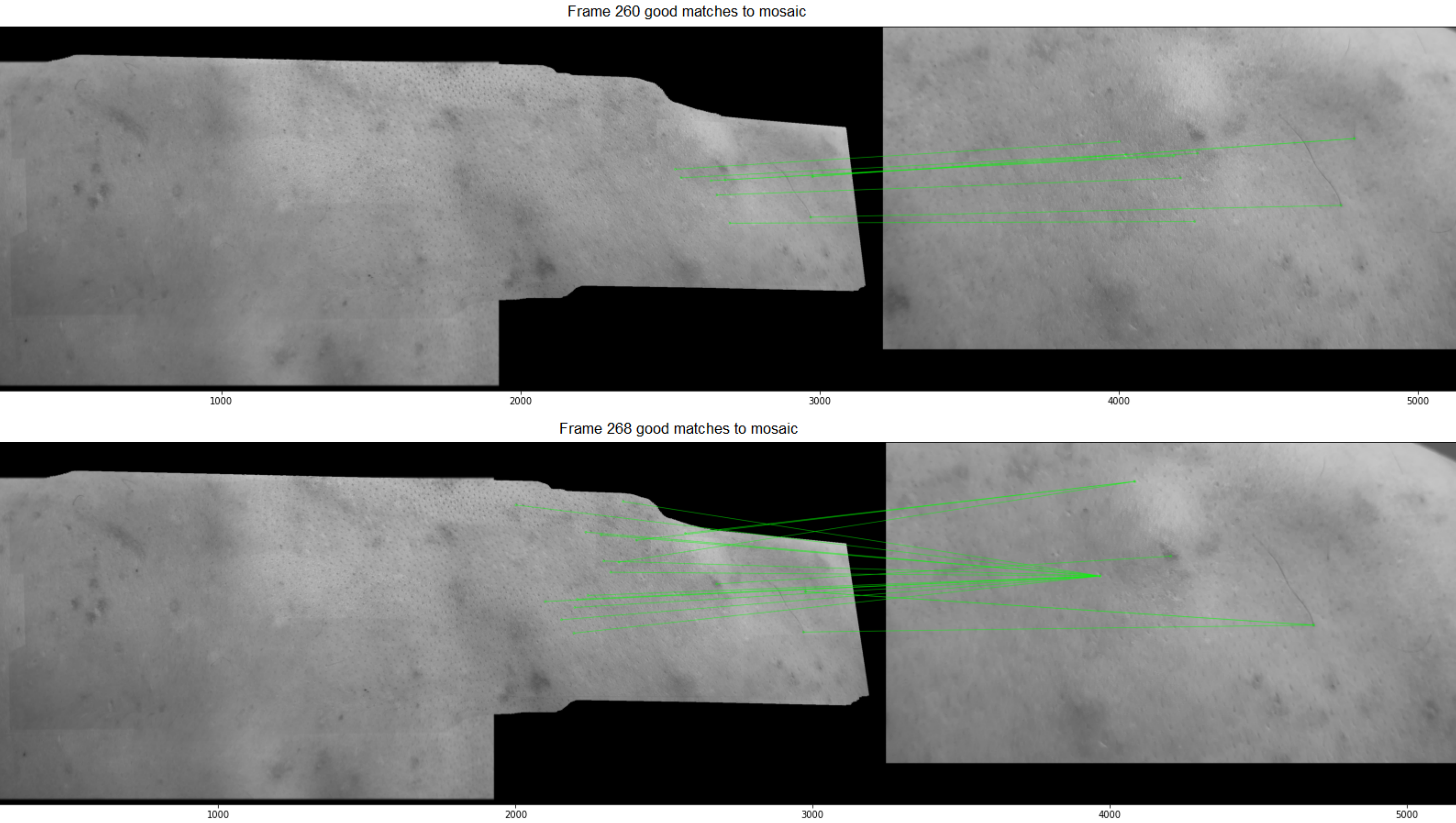

Frame 260 good matches to mosaic
1000
2000
3000
4000
5000
Frame 268 good matches to mosaic
1000
2000
3000
4000
5000

Figure 4.2: *2 examples of the good matches for dropped frames during the mosaicing process.*

frame 267 are slightly out of focus. The frames miss sharpness and this results in fewer features found.

In figure 4.2 the matched features between the dropped frames and the mosaic are shown. You can see that for frame 260, from the 309 features found only a hand full can be matched to the existing result. This small amount of features is apparently sensitive to errors and leads to a miscalculated homography. But this miscalculation is detected by the quality checks and the frame is dropped. The other example (frame 268) does match all features to a feature in the result. You see one feature being matched to multiple features in the result. The fact that Lowe's ratio does not prevent these mismatches can be explained by the lack of correct matches. If the result of the nearest neighbor matching is 2 wrong matches, the difference between the two matches will not be significant, so the matches will not be dropped. The fact that no correct matches can be found is also explained by the lack of camera focus. The difference in features found and feature descriptions is to big between the in focus mosaic and out of focus new frames.

Another thing to notice is that the images where arms or the rounding at the side of the back are in view are distorted. This is caused by the fact that at those places the skin area cannot be approximated as a plane. The homography projections projects a planar surface to another planar surface. So the calculation of the projection works for surfaces or when there is a large distance between the camera and the objects in view (as is the case for astronomy or panorama camera's). In our use case the camera is very close to the surface. So if there is curvature (like the side of the back) or differences in height (like the arms on the side) the homography projection will distort. If the majority of the features is on the back and all features on the arm can be considered outliers it's just the arm that will be projected wrongly. But if some features are on the arm/curvature and some on the back the calculation will be based on a plane drawn up along the matched features and all other points will be (slightly) distorted. The feature detection mechanism, of course, detects edges. So often the multitude of features will be found along the side of the back or arm. And so the chances of this error occurring are significant.

4.6 Projection quality

The results presented in section 3.5 showed no or a negative correlation between the normalized mutual information and the amount of matches between frames. This warrants further inspection. We have seen in figure 3.11 not only a low but also a high fit quality might correspond to failed projections. This might be caused by an empty area of overlap. The NMI is calculated over the ares of the two images that overlap. If the new frame is wrongly projected, it might be projected to a place in the reference frame were no existing image information is. In that case two empty collections will be compared and in the NMI is 1. This would mean that when the fit quality is 1 the new frame should be rejected too.

Further more the graph for result 1 shows a declining NMI value, while we also see that new frames are smaller with each successive frame. As I have introduced in section 2.3 the NMI is normalized by division with the product of the entropies of both collections under comparison. This entropy increases with increasing collection size. This means that the NMI of a smaller area of overlap will be lower then the NMI of a larger area of overlap for projections that are similar in quality. At the same time I have shown that the frames for result 1 became more and more out of focus. So it is likely that the declining NMI curve is a combination of both effects.

All and all we could use the NMI values for a safeguard of projection quality by rejecting all projections where $NMI < 0.2$ and $NMI = 1$. But we cannot state for two different NMI values the higher one corresponds to a better projection.

4.7 Further improvements

All improvements incorporated thus far have not let to a successful mosaicing process. It seems like the challenges posted by the method of data acquisition might be to big to tackle with the current mosaicing methods. There are different possibilities to further explore though.

- Sticking to the current path more effort could be invested into homography matrices checks. By trial and error more sensibility checks as discussed in 3.3 could be selected. So errors in the projection can be prevented.
  Further more, for frames were SIFT does not detect enough features, FAST could be implemented as an additional detection mechanism, with a SURF as a feature descriptor for features detected by FAST.
  Another area that needs improvement is the fact that the aproximation as a surface breaks down for curvature or differences in heigth. Here the direct method discussed in the introduction 1.4.2 could be used as a refinement technique. In our analysis the direct method had to many variables to optimize to work in our use case. But the method could be used as fine tuning after the initial projection with SIFT feature detection. The direct method implementation in Elastix can be configured to take 'non rigid' projections into account, with this mechanism not only afine projections are calculated, but also local deformations can be found and corrected. So adding this method as a finetuning step could prevent the distortions in the mosaics. Finally more flexible blending teechnique could be implemented. For instance the technique Wu et al. [24] introduced for image blending with GANs.

- Even so I think a more promising route would be to develop an app for the recording. In this app the user performing the recording could get direct feedback. If the camera would move to fast or would go out of focus the user could be warned to go back and redo a certain part of the body. A similar mechanism is already available in the panorama functions on mobile phones. This app might even use the mobile phones gyroscope to read the phones movements between frames and use these as an estimation for the expected homography.

- And finally a completely different approach that will be very interesting to explore is proposed in a paper by Nguyen et al. [30]. They train a deep convolutional neural network to calculate the homography matrix based on the two images using the pixel intensity difference as the loss function and have spend great effort to be able to do gradient descent optimization to find the best homography.

Chapter 5

Conclusions and future work

5.1 Conclusion

The research objective for this thesis as stated in the introduction 1.1 is:

> We aim to combine multiple pictures from a movie scanning a patients back or arm into one common reference frame.

This process of combing pictures is called mosaicing. I started with a literature review in the introduction (chapter 1) were I explored the steps needed to build a successful mosaic. These steps are:

1. feature detection
2. feature matching
3. calculating homography
4. warping images
5. image blending

In the chapter Methods (chapter 2) I explained the theory behind the techniques and designed experiments to test whether different techniques are suitable for the use case at hand. In the chapter Results (chapter 3) I reported the results of the experiments and the further analysis. The results showed that our aim would be difficult to achieve. I showed that all feature detection algorithms found zero features in a significant percentage of frames. From the analysis and from literature I selected SIFT as the best performing algorithm to use for further analysis. I showed that even the best performing algorithm with the telephone with the highest quality of recording (the iPhone) could not match 6% of frames.

Given the circumstances I have tried various mosaicing methods to come to an as good as possible result. These results are shown in section 3.3. The first is an intermediate result upon which I have made further improvements.

Still the best result is not yet a combination of all frames in the movie. The difficulty of creating one image out of a linear series (a movie) is that there is no tolerance for holes. If the connection between frames is lost, the result will be two (or more) images.

In this thesis I have shown that with the current method of data acquisition it is impossible to combine the different frames into one reference frame. The challenges of a very free form of data acquisition (a mobile phone camera), a very uniform surface and a surface that cannot be approximated as a plane give handicaps that are to large for a process in which a near 100% accuracy is necessary. While not all sequential frames need to be matched, some can be skipped, enough do need to be linked. And this link, in our practice, is quickly broken when the camera moves fast or is very close to the skin.

5.2 Future work

The aim of the STARPeople project is to make early detection of skin cancer possible for people at home. To realise this aim there is more work to be done. In the section 4.7 I have made suggestions for continuing this research with the current methods. I see a lot of potential options for improving the algorithm I have developed so far. By combining newer techniques and adding more checks and fallback options, it is possible to make the algorithm more robust and flexible.

Even so, I think the major issues will prolong. In some places the camera loses focus, is to close to the skin and/or moves to fast. This will always be problematic. There for I would to suggest to develop a mobile phone app that gives user feedback while recording. This way the patient can go back to the parts of the skin were the camera lost focus or moved to much and record these area's again.

Acknowledgements

This research was done in collaboration with the LUMC. I would like to thank Elena Sellentin for giving me the opportunity to participate in this project. And Jelle Mes for his dedication and unending encouragement, which helped me reach the finish line of this long running task.

Bibliography

[1] G. P. Guy Jr, C. C. Thomas, T. Thompson, M. Watson, G. M. Massetti, and L. C. Richardson, *Vital signs: melanoma incidence and mortality trends and projectionsâUnited States, 1982-2030,* Morbidity and mortality weekly report **64**, 591 (2015).

[2] M. Arnold et al., *Trends in incidence and predictions of cutaneous melanoma across Europe up to 2015,* Journal of the European Academy of Dermatology and Venereology **28**, 1170 (2014).

[3] M.-A. El Sharouni, A. J. Witkamp, V. Sigurdsson, P. J. van Diest, M. W. Louwman, and N. A. Kukutsch, *Sex matters: men with melanoma have a worse prognosis than women,* Journal of the European Academy of Dermatology and Venereology **33**, 2062 (2019).

[4] J. I. Van Der Rhee, W. Bergman, and N. A. Kukutsch, *The impact of dermoscopy on the management of pigmented lesions in everyday clinical practice of general dermatologists: a prospective study,* British Journal of Dermatology **162**, 563 (2010).

[5] P. Mirunalini, A. Chandrabose, V. Gokul, and S. Jaisakthi, *Deep learning for skin lesion classification,* arXiv preprint arXiv:1703.04364 (2017).

[6] A. N. MacLellan et al., *The use of noninvasive imaging techniques in the diagnosis of melanoma: a prospective diagnostic accuracy study,* Journal of the American Academy of Dermatology **85**, 353 (2021).

[7] H. A. Haenssle et al., *Man against machine reloaded: performance of a market-approved convolutional neural network in classifying a broad spectrum of skin lesions in comparison with 96 dermatologists working under less artificial conditions,* Annals of oncology **31**, 137 (2020).

[8] T. M. de Carvalho, E. Noels, M. Wakkee, A. Udrea, and T. Nijsten, *Development of smartphone apps for skin cancer risk assessment: progress and promise*, JMIR Dermatology **2**, e13376 (2019).

[9] J. Deeks, J. Dinnes, and H. Williams, *Sensitivity and specificity of SkinVision are likely to have been overestimated*, J Eur Acad Dermatol Venereol **34**, e582 (2020).

[10] L. University, *Observatory*.

[11] LUMC, *Dermatology*.

[12] H. Tsao et al., *Early detection of melanoma: reviewing the ABCDEs*, Journal of the American Academy of Dermatology **72**, 717 (2015).

[13] E. Bertin and S. Arnouts, *SExtractor: Software for source extraction*, Astronomy and astrophysics supplement series **117**, 393 (1996).

[14] A. Pandey and U. C. Pati, *Image mosaicing: A deeper insight*, Image and Vision Computing **89**, 236 (2019).

[15] R. J. Avila, W. Hack, and S. A. Team, *AstroDrizzle: Aligning Images From Multiple Instruments*, in *American Astronomical Society Meeting Abstracts# 220*, volume 220, pages 135–13, 2012.

[16] K. Kose et al., *Automated video-mosaicking approach for confocal microscopic imaging in vivo: an approach to address challenges in imaging living tissue and extend field of view*, Scientific reports **7**, 10759 (2017).

[17] R. Szeliski et al., *Image alignment and stitching: A tutorial*, Foundations and Trends® in Computer Graphics and Vision **2**, 1 (2007).

[18] S. Klein, M. Staring, K. Murphy, M. A. Viergever, and J. P. Pluim, *Elastix: a toolbox for intensity-based medical image registration*, IEEE transactions on medical imaging **29**, 196 (2009).

[19] D. G. Lowe, *Distinctive image features from scale-invariant keypoints*, International journal of computer vision **60**, 91 (2004).

[20] E. Rosten, R. Porter, and T. Drummond, *Faster and better: A machine learning approach to corner detection*, IEEE transactions on pattern analysis and machine intelligence **32**, 105 (2008).

[21] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, *ORB: An efficient alternative to SIFT or SURF*, in *2011 International conference on computer vision*, pages 2564–2571, Ieee, 2011.

[22] C. Harris et al., *A combined corner and edge detector*, in *Alvey vision conference*, volume 15, pages 10–5244, Citeseer, 1988.

[23] E. Karami, S. Prasad, and M. Shehata, *Image matching using SIFT, SURF, BRIEF and ORB: performance comparison for distorted images*, arXiv preprint arXiv:1710.02726 (2017).

[24] H. Wu, S. Zheng, J. Zhang, and K. Huang, *Gp-gan: Towards realistic high-resolution image blending*, in *Proceedings of the 27th ACM international conference on multimedia*, pages 2487–2495, 2019.

[25] S. A. K. Tareen and Z. Saleem, *A comparative analysis of sift, surf, kaze, akaze, orb, and brisk*, in *2018 International conference on computing, mathematics and engineering technologies (iCoMET)*, pages 1–10, IEEE, 2018.

[26] K. G. Derpanis, *Overview of the RANSAC Algorithm*, Image Rochester NY **4**, 2 (2010).

[27] J. P. Pluim, J. A. Maintz, and M. A. Viergever, *Image registration by maximization of combined mutual information and gradient information*, in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2000: Third International Conference, Pittsburgh, PA, USA, October 11-14, 2000. Proceedings 3*, pages 452–461, Springer, 2000.

[28] T. Veninga, H. Huisman, R. W. van der Maazen, and H. Huizenga, *Clinical validation of the normalized mutual information method for registration of CT and MR images in radiotherapy of brain tumors*, Journal of Applied Clinical Medical Physics **5**, 66 (2004).

[29] M. Muja and D. Lowe, *Flann-fast library for approximate nearest neighbors user manual*, Computer Science Department, University of British Columbia, Vancouver, BC, Canada **5** (2009).

[30] T. Nguyen, S. W. Chen, S. S. Shivakumar, C. J. Taylor, and V. Kumar, *Unsupervised deep homography: A fast and robust homography estimation model*, IEEE Robotics and Automation Letters **3**, 2346 (2018).

Appendix 1

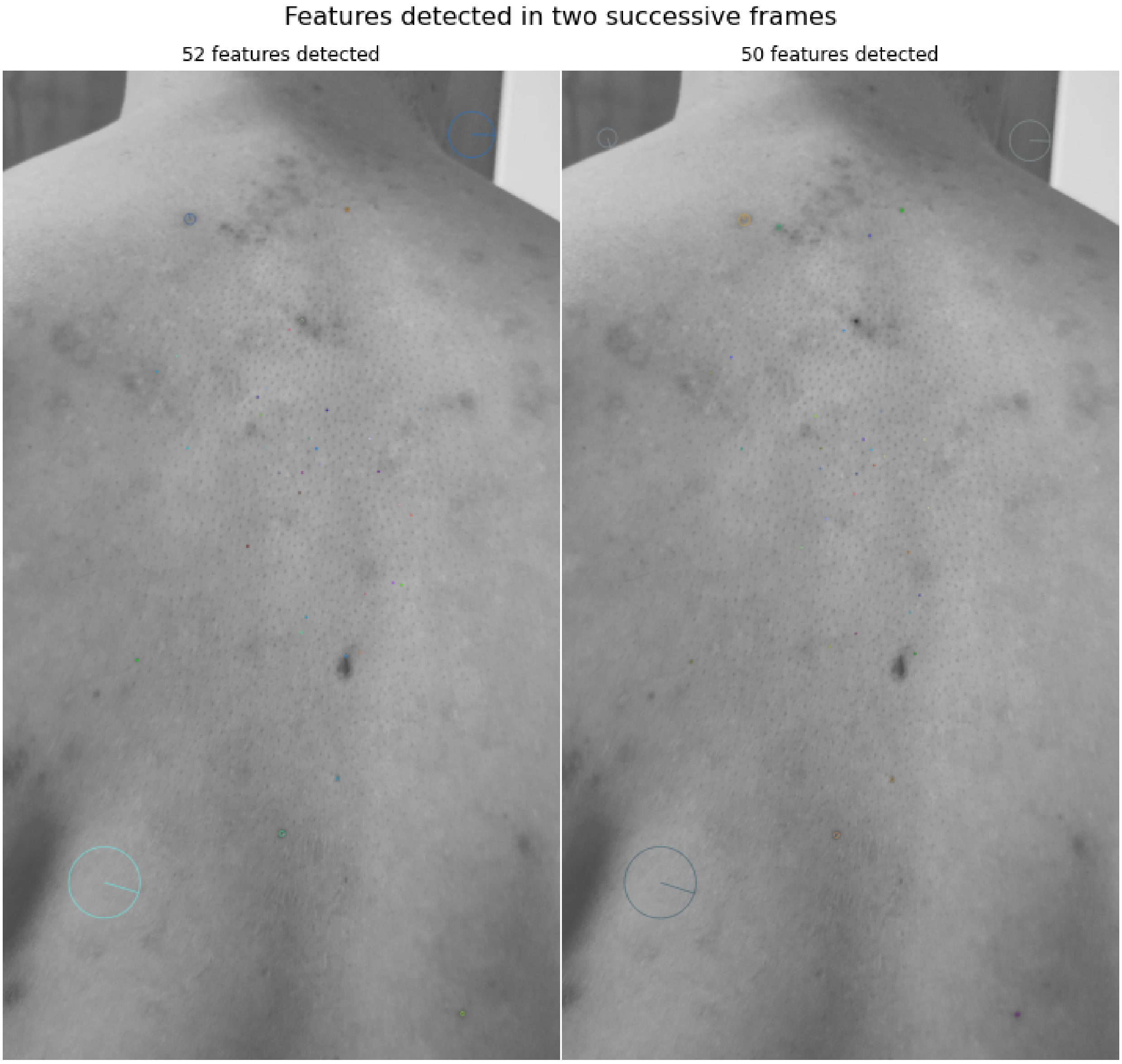


Figure 1: *Example of feature detection on skin with SIFT.*

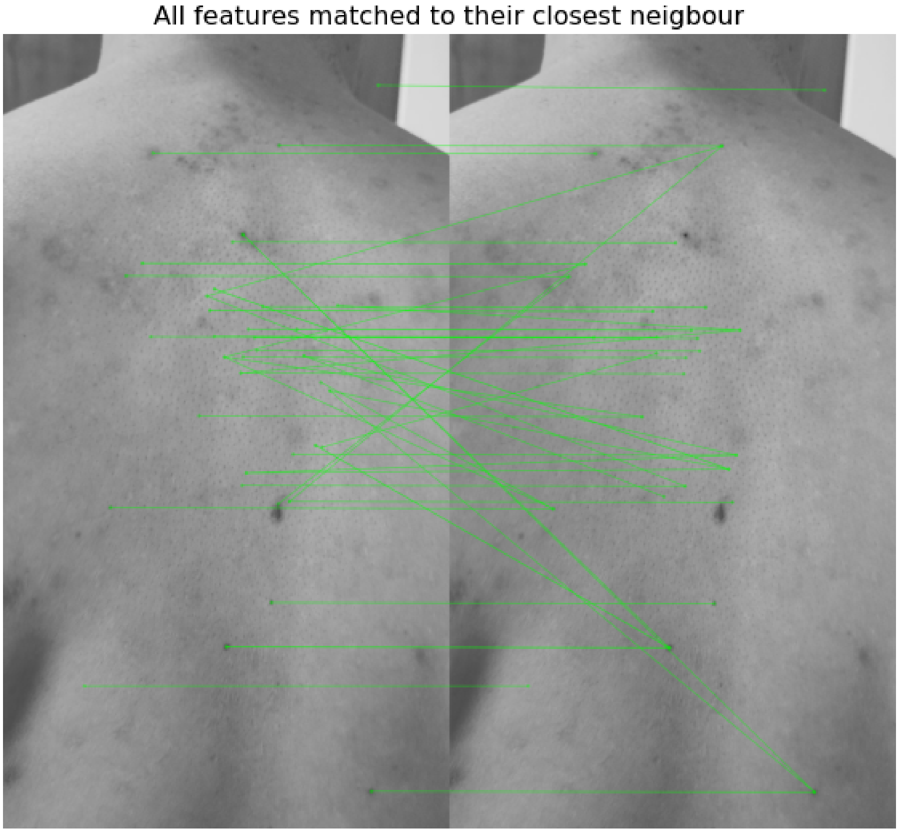


Figure 2: *Detected features matched with nearest neighbour matching.*

Appendix 2

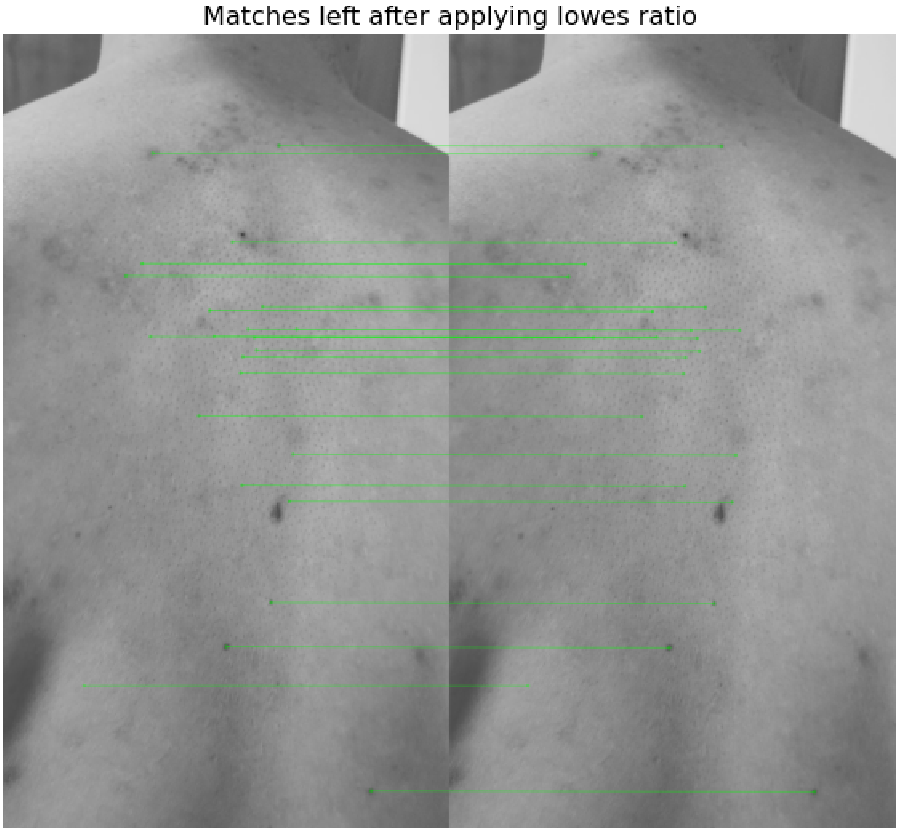


Figure 3: *Application of Lowe's ratio to filter out good matches.*

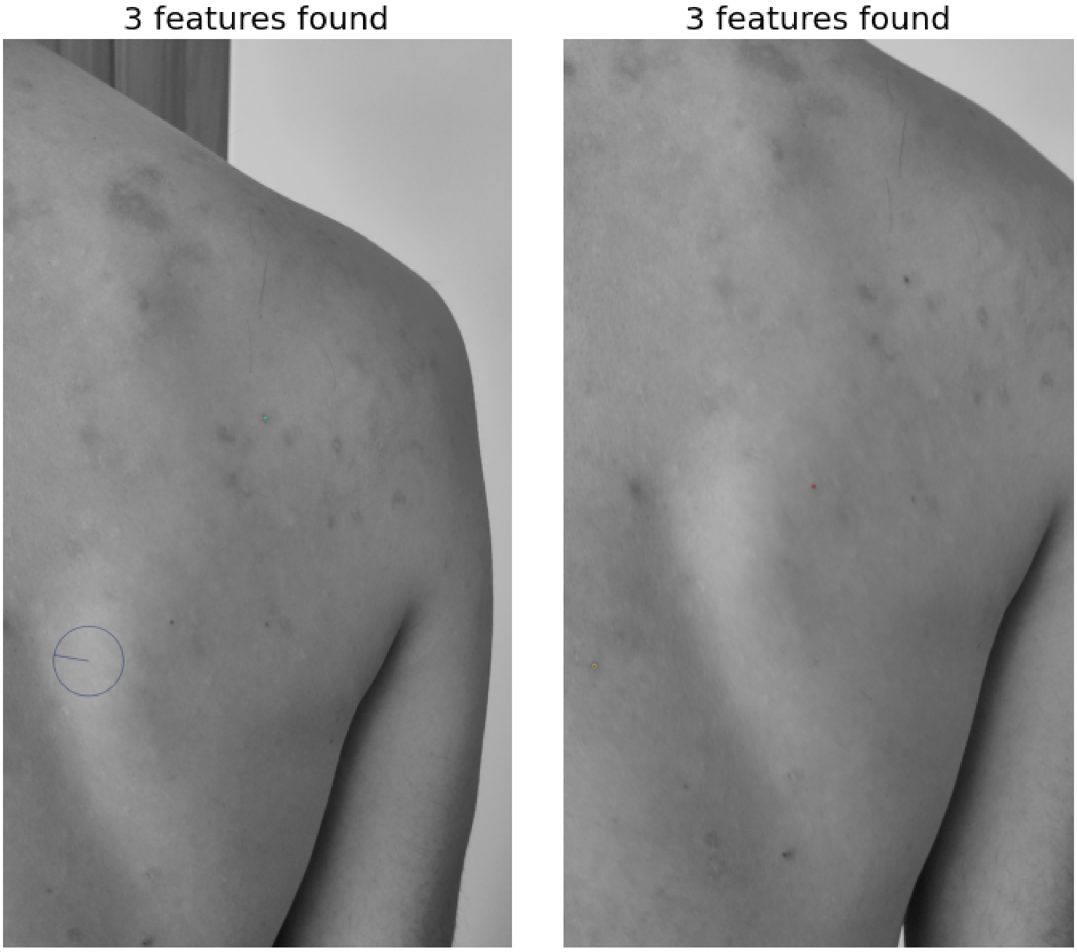


Figure 4: *Two frames where SIFT did not find enough features to calculate the projections to the reference frame of other frames.*