Universiteit Leiden
The Netherlands

Opleiding Informatica

Detecting Facial Features in Infrared Imagery

Joos van Zweeden

Supervisor:
Mart Janssen

2nd Supervisor:
Walter Kosters

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

January 28, 2020

Abstract

Recognizing facial features in (visual light) imagery is a well-studied topic, however when the imagery is in infrared the number of available algorithms is lacking severely. This is why in this study we propose a method to use modern facial landmarking algorithms with infrared imagery. We recorded our own infrared imagery and labelled this ourselves to use for this thesis. Infrared imagery, however, is saved as temperature values while visual light imagery is normally saved in the RGB color-system. Therefore we present a transformation that transforms our infrared imagery (temperatures) into visual light imagery (RGB). We try to find an optimal translation by testing multiple different linear transformations. The results show that this method has potential as it landmarks around half the images with some of our found transformations and with variable accuracy. In 50% of cases the landmarks found are quite accurate, meaning that they look like the result of hasty manual labeling. Accuracy is likely to increase when more data is used for training the model. Also, we try to improve the results of the mouth landmark by increasing the contrast in this area as this landmark was labelled worse than most others. This method, unfortunately, greatly deteriorates our results and is thus not recommended.

Contents

Chapter 1

Introduction

This thesis describes the work that was performed for the extraction of facial features from infrared videos using image processing algorithms.

1.1 Background

This thesis was made in collaboration with Sanquin [San]. Sanquin is an institution in the Netherlands which is tasked with providing blood and blood-products for healthcare. To this end they use blood donations. Sanquin wants this to go as smooth as possible for the comfort of the donors. However, donating may have negative side effects like dizziness and even fainting. Records of these events by the bloodbank show that one in a thousand donations results in fainting of the donor [AVLD12].

Bio-feedback is a well-known and effective method to influence psycho-physiological responses [Lab95]. The hypothesis is that bio-feedback can therefore be used to prevent vasovagal responses in donors. This, however, requires measurement of the physiological responses. It will be investigated whether infrared imagery is a viable method for this purpose. The advantage of imaging is that it does not require any physical contact with the donor.

The project is in a start-up phase, and for this thesis we have tried to develop an algorithm that is able to extract facial features from infrared images. These should enable identification of various facial areas and the mean, minimum, and maximum temperatures herein, as well as changes of these temperatures over time. This thesis, however, will *only* focus on finding facial features in infrared imagery. We hope that the findings of this thesis can be of help to the bigger project. The measurement of temperatures of facial features, after finding them, should be fairly easy, especially since all the imagery is in infrared, which corresponds to the temperature of the recorded objects.

1.2 Overview

First, in Chapter 2, Earlier Research, we look at some background information and earlier research about facial recognition algorithms (Section 2.1). And in Section 2.2 we show some of the advantages and disadvantages of working with infrared imagery.

In Chapter 3, Methods, we talk about the choices we made for our algorithm and why we made these. In Section 3.1 we discuss how we collected our data and what this data is. In Section 3.2 we give some information about the algorithm we used (*One millisecond face alignment with an ensemble of regression trees* [KJ14]) and its implementation in *dlib*. We also talk about the formula we used to transform our data so we can use this in the previously mentioned algorithm. We need this transformation as the algorithm uses RGBA imagery and our data exists of temperature values. Then we discuss a method that might improve our results by "boosting the mouth region" in Section 3.2.3. This means that we try to increase the contrast in the mouth region as to give the algorithm a better chance of correctly marking the mouth, which has been proven to be rather difficult. Lastly, in Section 3.2.4 we show that this problem has a rather big solution space (in the billions). This is especially a problem considering that it takes at least a few minutes to train one model (on a i7-3770 CPU @ 3.40GHz) and thus even longer to find a solution.

In Chapter 4, Results, we show the results of our algorithm. Firstly, in Section 4.1 we define a way to normalize our results. Next, we show the results of our experiments. In Section 4.2 we show the general results of our algorithm while in Section 4.3 we show the results of each individual facial feature. It turns out that finding a good translation can have a huge impact on the performance of the algorithm. This is true for the precision (how far is the marked point away from its actual location) as well as the reliability (did the algorithm find a face at all). The best translation we found is able to find a face in about half the images but with a decent average precision. And lastly, in Section 4.4 we found that "boosting the mouth region" only has a negative effect on the results of the algorithm, as it decreased the number of faces found and lowered the accuracy.

Finally in Chapter 5, the Discussion and Conclusion, we conclude that our results show that we can use an algorithm for facial landmarking in the visual light spectrum and apply it to infrared images using small transformations on the imagery. This transformation is not optimal but finding an optimal one is hard as the solution space is very large. However, we propose this could be done for future research. We also discuss the use of larger databases and different algorithms for facial landmarking to use with our algorithm.

This thesis was done at LIACS, the computer science department at Leiden University, in cooperation with Sanquin. and was supervized by Mart Janssen and Walter Kosters.

Chapter 2

Earlier Research

Specific algorithms for facial landmarking in infrared images are described in the paper *Physiological parameter response to variation of mental workload* [MSR+18]. This study used noninvasive methods, including infrared imagery, to monitor the condition of study participants while they performed various tasks. Unfortunately we were unable to find the algorithms and code that were used in this research. We instead therefore explain the usability of some algorithms that are used for normal (non-IR) imagery.

2.1 Facial Landmarking Algorithms

Facial landmarking algorithms can be divided into a number of different types [WJ19]:

1. The holistic methods.
2. The constrained local model (CLM).
3. The regression-based methods.

A holistic method uses the global shape of the faces to find the landmarks. It uses its training images to generate a model that functions as a general face mask. It then tries to fit this mask onto a face. After this the facial landmarks can be determined as the landmarks are marked on this mask. The most well-known method is the Active Appearance Model (AAM) [CET01].

CLM also uses information of the landmarks itself to detect these and combines this with global information. It searches each landmark locally using region features [GWWJ16].

The regression-based method trains regression models and uses these to estimate the position of the landmarks through local appearance features [GWWJ16].

Table 2.1 shows information about a collection of facial landmarking algorithms. It contains the types as we discussed above, the number of points they utilize on the face and the average framerate (the speed). This

methods	type	# points	fps
TCDCN	DR	5	58
Hyperface	DR	21	5
Consensus of Examplars	C	29	1
3DDFA	DR	68	13
CFAN	DR	68	40
CFSS	R	68	25
SDM	R	68	30
3D Regression	R	68	111
Explicit Shape Regression	R	87	345
RCPR	R	194	6
One millisecond face allignment	R	194	1000
Face allignment 3000 fps	R	194	200/1500

Table 2.1: Various facial landmarking algorithm with their type (DR = deep learning (this is a regression based method combined with deep learning), C = constrained local methods and R = regression based method), the number of points it detects and their speed (in frames per second) from *Facial Landmark Detection: A Literature Survey* [WJ19, Table 5, p.21].

shows that *One millisecond face alignment* [KJ14] is a relatively fast algorithm. It has already been shown that AAMs are a viable method to perform facial landmarking even in infrared, the drawback however being that this is not a very fast method [KAM16].

There are many different ways to annotate the face and facial features and there are many databases with different notations [JdC18]. In this thesis we use the notation from the CMU Multi-PIE database [GMC+10]. This scheme is used by one of the bigger and more popular databases, the *300W faces in-the-wild challenge* [IBUGi], and can thus be seen as one of the unofficial "standards". We will go into more detail about this notation in Section 3.2.1.

2.2 Infrared Imagery

There are some advantages and disadvantages related to working in infrared as opposed to the visible spectrum. Firstly infrared can be measured irrespective of the level of illumination. It is also easier to separate the background from the actual face. Another big advantage of infrared over the visible spectrum is that the amount of noise is significantly less due to the skin on the face having an even temperature. A disadvantage, however, is that there has been less focus on the infrared facial feature extraction as opposed to the various implemented algorithms for facial landmarking in the visible spectrum [KHA+05, Ery15].

Some specific problems occur when working with infrared imagery, for instance, the variance in ambient temperature that affects facial temperatures and temperature gradients. So we have to be aware that faces can have a bigger variance in temperatures in practice than that we might find in our test data. Also prior physical activity, hair growth and patterns and the presence of spectacles (glass blocks all infrared waves) will affect the capability of adequate facial landmarking.

Chapter 3

Methods

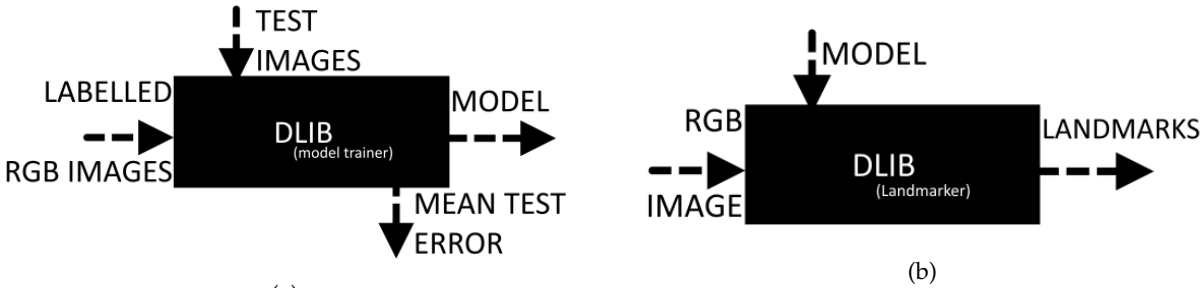
The aim of the project is to find an algorithm suitable for the identification of facial features in infrared images. To achieve this we have taken an existing algorithm and made a program that converts our infrared data into a format that can be used by a standard facial landmarking algorithm.

3.1 Data Collection

We found some databases containing labelled thermal face images [GABM14], but most of these are not (easily) accessible (anymore), are very small or of very low resolution. So we decided to create our own data-set. We recorded 31 thermal videos of different persons and took frames from these videos to create our own data-set of images. These frames exist of 348 by 464 pixels with their recorded temperature values.
Video's were recorded among canteen visitors (students and employees of the data science department, and others) of the Snellius building on May 9th, 2018. Data were recorded using Sanquin's FLIR E95 thermal camera. All participants provided informed consent for the use of video footage for this research. However, considering the privacy of our participants, we cannot share these data in accordance to the written consent of the participants.

3.2 Recognizing Facial Features using dlib

The algorithm we used for detecting the facial features is described in *One millisecond face alignment with an ensemble of regression trees* [KJ14]. It relies on learning from a small set of examples to achieve real-time and accurate predictions of the position of facial landmarks. This algorithm has also already been implemented in the C++ library *dlib*. There are various examples of applying the *dlib* library for facial landmarking on the internet [Ros]. This algorithm, however, requires the input of an RGBA encoded image, so an image with four different channels (red, green, blue and alpha). Our data however only has one channel, namely the recorded



(a) (b)

Figure 3.1: Input and output of the dlib facial landmarking functions. Which are divided into the model trainer (a) and landmarking (b) functions.

temperatures. And thus a small program is needed to convert our measurements into the correct input for the facial landmarking algorithm. Still we chose this algorithm because it was already implemented and easily obtainable, it did not require big data-sets and it performed relatively well. The basic input and output of the dlib functions are shown in Figure 3.1.

3.2.1 Labeling Faces

The recorded images were labelled according to *dlib*'s specifications. The face was labelled in an XML-file with 68 points on the face on predetermined locations, which are the jawline (17 points), eyebrows (5 points each), lips (20 points), nose (9 points) and around the eyes (6 points each). These points are displayed in Figure 3.2. Here the 68 locations have been marked on a face and are displayed as red circles. There is also a box around the head indicating the part of the picture that contains the face. We wrote a small tool in JavaScript that enabled manual marking of the 68 locations of each image. The labels were saved in a separate file so we could still change the colour transformation of the individual images. This would however only work as long as the resolution stays the same, since the location of the facial markers should not change. The training-set and test-set must be defined in two separate XML-files to be used by *dlib*.

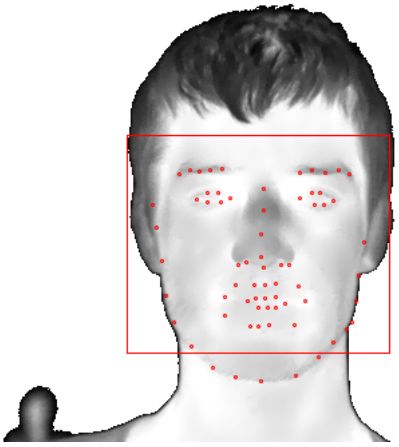


Figure 3.2: The set 68 locations have been marked on a face and are displayed as red circles

The training-set and test-set files are created automatically, using the individual XML-files, and parameters for the size of the training-test and test-sets. This way we can create a new training-set and test-set of a variable size and with either random or predetermined faces during run-time. This facilitates cross-validation evaluation.

3.2.2 Finding an Optimal Transformation

In this section we will look at how we transform our data so we can use it in the dlib facial landmarking algorithm. Our data consists of temperature values stored in CSV-files. We can read these into memory but *dlib* requires RGB data from an image format (PNG and JPG). So first we load these values into memory and we use a transformation to translate these values into RGB values. We can turn these RGB values into images. In our case we used the PNG format because this format is lossless. The formula we used was a linear function from a starting temperature to an end temperature over 255 steps.

In order to calculate these values, we used Equation 3.1:

$$\text{colour\_value} = \begin{cases} 0, & \text{if temperature} < \text{starting} \\ 255, & \text{if temperature} > \text{end} \\ \lfloor (\text{temperature} - \text{starting}) * \text{step\_size} \rfloor, & \text{otherwise} \end{cases} \tag{3.1}$$

where

$$\text{step\_size} = 255/(\text{end} - \text{starting}) \tag{3.2}$$

Figure 3.3: An image generated from temperature values using our linear transformation using the values red: 25–35, green: 15–40. blue: 25–30 and alpha 25.

Since each colour channel in PNG only has 1 byte of information, it can only have 255 different values. So we make sure that the values we calculated fall between 0 and 255.

The starting temperature is the bottom of the temperature limit so anything colder than the starting temperature will have a value of 0. The ending temperature is the ceiling so anything hotter than the ending temperature will have a value of 255. The step_size variable is the amount of colour intensity increase per degree Celsius. This is shown in Equation 3.2. But since there is a starting temperature we first subtract that from the temperature since the colour intensity should not increase for that amount.

Using Equations 3.1 and 3.2 we can start creating input for the facial landmarking algorithm by applying the transformations to every pixel of our recorded data three times, namely for the red, blue and green channel. PNG does have one more channel, The "alpha channel". This channel specifies the opaqueness of the pixel. We use this channel to partly filter out the background. When a temperature is lower than a preset temperature, this channel is set to 0 so that it is fully transparent. A large part of the background surrounding the recorded subject turns out to have a lower temperature than the person. This differs from location to location. But this is an easy implementation which might improve the results somewhat. Otherwise it is set to 1, making the pixel fully opaque. Early testing showed that shifting this value partly did not have big impacts on the results.

(a) Non-boosted face (b) Boosted face has more contrast in the mouth region.

Figure 3.4: Non-boosted contrast (a) and boosted contrast (b) side by side.

Using the C++ library *lodepng* [Van] we created a PNG file from these RGBA values. Figure 3.3 shows an image that was generated using the method specified above. The background has been mostly filtered out by the alpha limit. Everything under 25°C has been made fully transparent. Parts of the face are still recognizable (even to humans) because of the differences in temperature.

3.2.3 Boosting the Accuracy of the Mouth Region

Since in our early testing it became apparent that the algorithm was having trouble properly detecting the mouth, we decided to try and boost the accuracy of the mouth region. To do this we increased the contrast in the area where the mouth is. To find the right area we first ran the algorithm once to get an estimation of where the mouth is. Next we marked an area where the mouth is located with certainty. We did this by marking 4 positions. The first position is in between the bottom of the nose and the top of the presumed mouth. The second position is between the bottom of the chin and the bottom of the mouth. The third position is located between the right side of the presumed mouth and the right side of the face. The fourth position is located between the left side of the presumed mouth and the left side of the face. With these four positions we can mark the region of interest. This is now defined as a rectangle that is confined by these four positions. Next, we increased the contrast within the region of interest. To do this, we measure the highest and lowest temperature in this specific area. The highest temperature is used as the end temperature, the lowest as the starting temperature in Equation 3.1.

The result is that all colours within this region will remain in the same colour region, but with a far more accurate temperature discrimination. This results in a higher contrast without adding or losing any data. This method however has some limitations. Firstly, if the face is tilted too much the selected area might include an area outside of the face and thus lowering the minimum temperature drastically which results in a smaller contrast than expected. Also, this procedure substantially increases the evaluation time as creating a new image takes much time. Figure 3.4 shows the difference between a non-boosted (Figure 3.4a) and a boosted face (Figure 3.4b). The boosted face shows more contrast in the mouth area. A small problem we can run into is that sometimes a face might not be recognized; in this case we will remove the face from the training data, since if the mouth is not found it can also not be boosted.

3.2.4 Solution Space

To keep the solution space as small as possible we are only considering linear transformations. This, however, still leaves us with a fairly big solution space as all three colour channels have a start and an end temperature and the alpha channel has a starting temperature too, so at least seven values need to be set. If we limit these values between 15 and 50 degrees and we say that every step is one degree, we get $\sum_{x=0}^{34}(\sum_{y=1}^{35-x} y) = 7770$ solutions per channel. This gives us $7770 * 7770 * 7770 * 35 \approx 1.6 * 10^{12}$ solutions in total. If one combines this with the run time of a training session given in Table 3.1 we can conclude that it is not viable to do an exhaustive search.

The algorithm is substantially faster when we do not use the mouth boost. So we might want to find a good transformation without the mouth boost. And later on we can test if the mouth boost makes an improvement on our already good intervals. This can save us significant time.

(a) Using the boosted mouth region

Real	3m53.273s
User	7m48.160s
Sys	0m10.060s

(b) Not using the boosted mouth region

Real	0m38.602s
User	2m48.508s
Sys	0m10.688s

Table 3.1: Results from time while training a model in dlib (i7-3770 CPU @ 3.40GHz). *Real* refers to the real time used, *User* is the total time each thread spend in user-mode(not kernel), and *Sys* is the time spend in the kernel [(us]
.

3.3 Overview of the Evaluation Process

In Figure 3.5 we show a schematic overview of a single test run of the algorithm. For the input there is a file for each image with its landmarks, the transformation parameters for this run, the temperature values (recorded IR imagery) and a parameter defining which files are going to be used as test and training data. First all the temperature values are changed into RGB images using the parameters, and one XML file is created for training and one for testing. First the training images are used to train a model, and this model is used together with the test images to obtain a set of new landmarks for those images. These landmarks are then tested against the ones we manually labelled to get an error value.

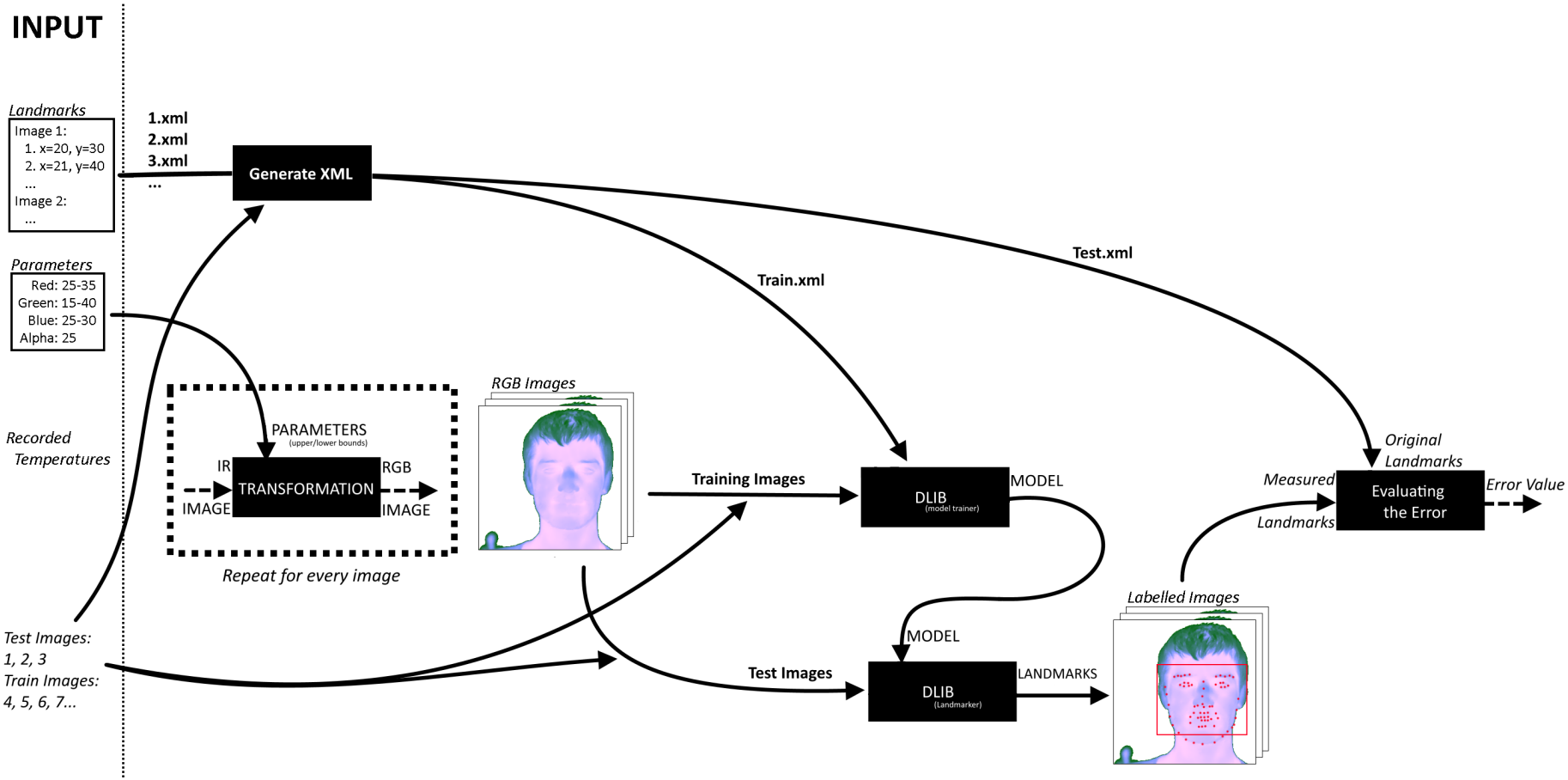


Figure 3.5: Schematic overview of the setup of the method as described in this chapter. This overview is explained in Section 3.3.

3.3.1 Source Code

The source code of this project can be found on Github at:
`https://github.com/juicenator/Facial-Landmarking-Infrared`.

Chapter 4

Results

Here we will show some results from experiments performed for testing the algorithm defined in the previous chapter. The results obtained using the *dlib* algorithm varied greatly. The results depended strongly on the transformation that was applied. This is why we first tried to find a transformation that would lead to the highest precision. After finding the best transformation, we tried to improve our results by boosting the mouth region.

4.1 Evaluating Results

To evaluate the results we used the distance between the points which the model estimated and the points labelled by hand. We normalize these results by dividing the distance between the two inner points of the eyes (side of the eye closest to the nose). This normalization is required to lessen the effect caused by the distance between the person and the camera. Most papers use the distance between the centers of the eyes [DGES+15], the *inter-ocular distance*, but since we already labelled the side of the eyes and not the middle we will use these points.

There is a chance that the algorithm detects no face at all. In this case we will not use this image in the error distance, but we will instead count the number of instances where no faces were recognized and consider this when evaluating the results separately.

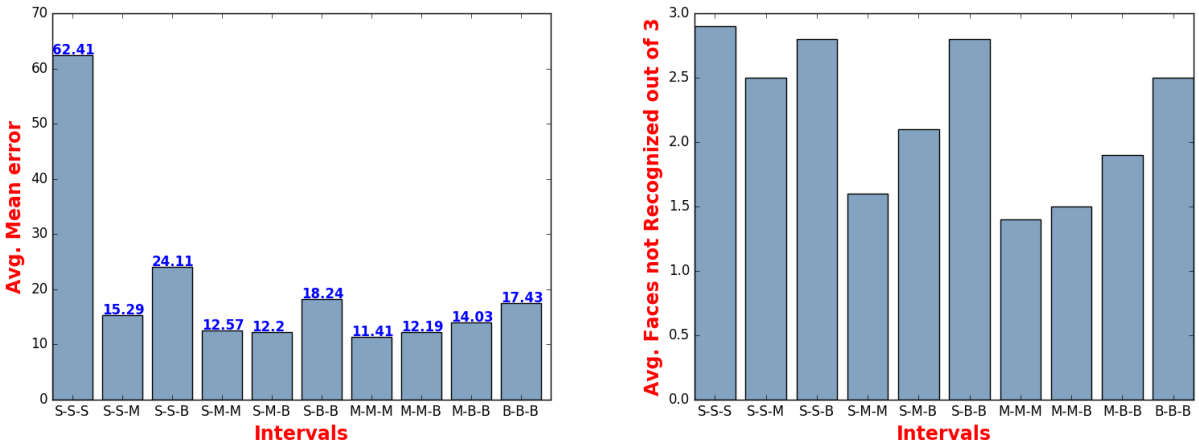
4.2 Temperature to RGB Translation

To find the right transformation we aim to explore the full solution space. But as said before, this solution space is very large (especially considering the time it takes to find a solution). Thus it was unfeasible to try and explore this space fully. Therefore we tried a few different permutations we generated randomly (and

later pseudo-randomly and handpicked) and analysed these. Below we describe a few tests we did that had interesting results.

A test we did was assigning a big interval (20–50°C) and a small interval (35–40°C) to each channel. We did this for every possible combination of big and small interval and colour channel. This showed that the intervals of the red, green and blue channel are interchangeable. As long as one uses the same values in different channels the results will be the same. This data also showed that having too many small intervals will result in poor results. This may be due to a loss of detail.

To test what intervals are the best we used a combination of three different intervals as seen in Table 4.1. We then used our training-data of 31 images to create 10 different tests. We separated 3 images from the training-data and used these images as the test-data. We did this in such a way that each image was in the test-set once (except for one). The results of these tests are in Figure 4.1. We will define the precision in our results as the average number of pixels the landmarks found by the algorithm are removed from their true position according to the labelled data. This data is normalized using the technique from Section 4.1. Note that when no face was found, the image was not used in calculating the *Average Mean error*. These results are shown in Figure 4.1a. Figure 4.1b shows the average of faces the algorithm did *not* recognize over all the tests, and we will define this as the reliability of the algorithm.



(a) Precision

(b) Reliability

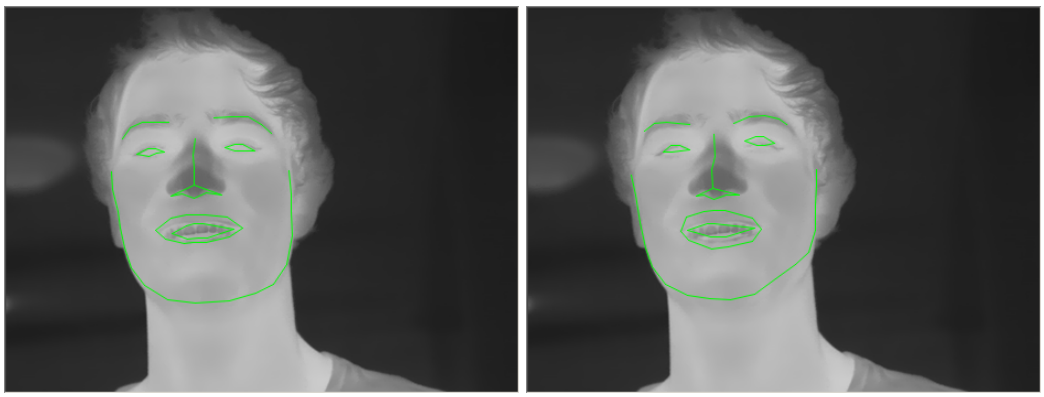
Figure 4.1: Precision and Reliability for various temperature intervals (lower is better).

Table 4.1: Letters associated with the Temperature Intervals in °C

Name	Letter	Temperature-Intervals
Small	S	35–40
Medium	M	20–40
Big	B	0–50

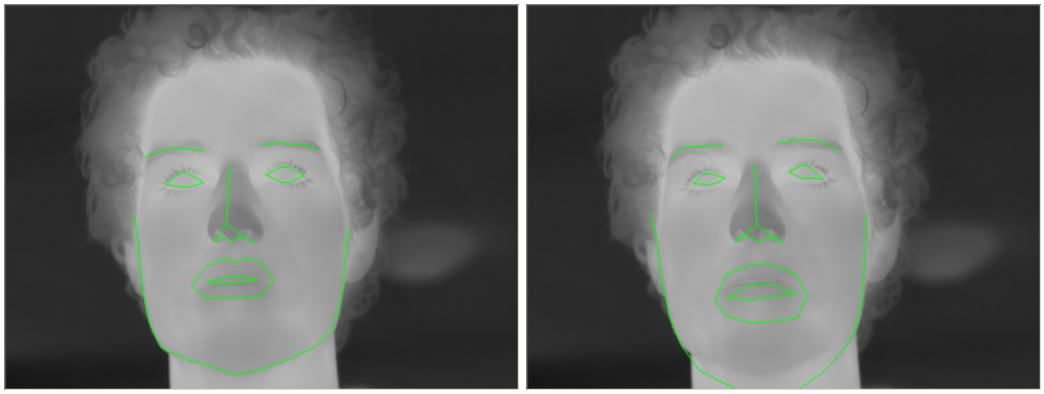
We want to minimize the values of the test error and of the number of faces that are not found. From Figure 4.1 it seems clear that having *medium* intervals has a positive effect on the results. It also seems that having all channels on a *medium* interval generates the pictures with the best results.

In Figures 4.2, 4.3 and 4.4 we can see what three different results look like with their mean error.



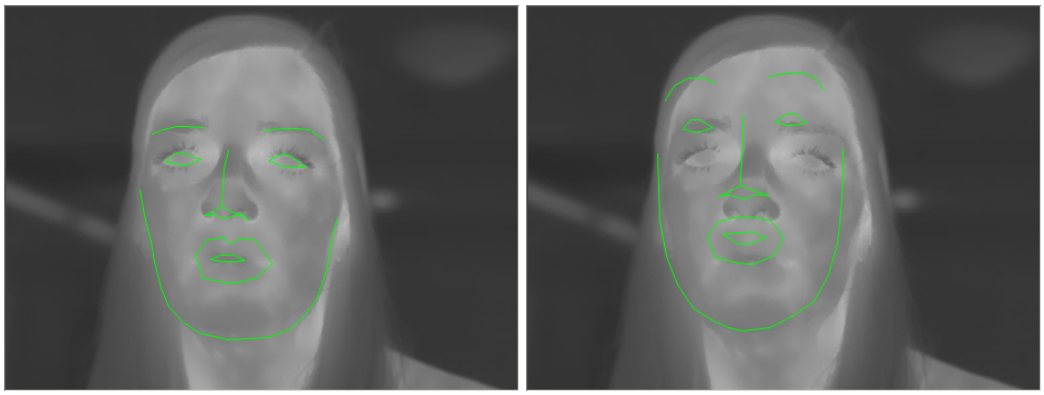
(a) Manually set landmarks. (b) Landmarks as found by the algorithm.

Figure 4.2: Comparison between the actual landmarks and the actual landmarks. The test error of this image is 7.42. This image is generated with all intervals set to *medium*.



(a) Manually set landmarks. (b) Landmarks as found by the algorithm.

Figure 4.3: Comparison between the actual landmarks and the actual landmarks. The test error of this image is 11.64. This image is generated with all intervals set to *medium*.

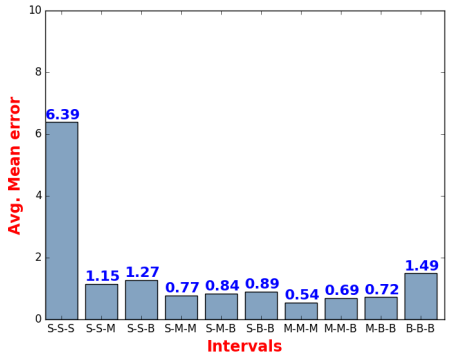


(a) Manually set landmarks. (b) Landmarks as found by the algorithm.

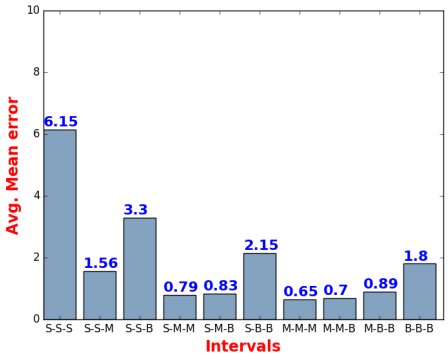
Figure 4.4: Comparison between the actual landmarks and the actual landmarks. The test error of this image is 30.43. This image is generated with all intervals set to *medium*.

4.3 Precision of Individual Facial Features

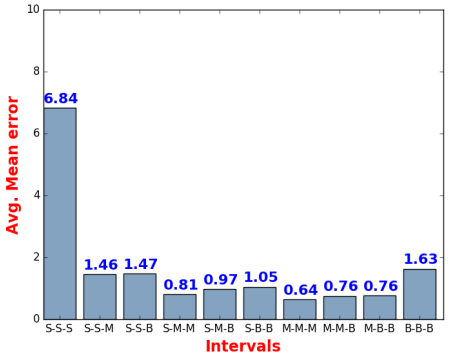
In this section we analyze the difference in error for different facial features left and right eye and brow, under the nose, the nose bridge, the inner lips, the outer lips and the jawline. Figure 4.5 shows the average error of the different facial features. From these data we can extract that most of these facial features have a similar relation between the average error of the intervals and the total average error as shown in Figure 4.1a. This seems to hold true for all the features except for the jawline (Figure 4.6). These values seem to lie closer together in relation to each other. This can be explained by the fact that when we evaluate the jawline we are observing many more specific points instead of looking at the general shape. So as every point is less likely to lie on the exact position of our pre-set points as the jawline covers a larger surface, the jaw is likely to be evaluated worse than the actual result. So both good and less good (not horrible like S-S-S) results get a higher mean error. In Figure 4.5 we can also see that both the right eye (Figure 4.5b) and the right brow (Figure 4.5d) have higher errors than their left counterparts (Figure 4.5a and 4.5c). This could be explained by several reasons. Firstly the right side landmarks of each side may be labelled worse than the right side. But as seen in Figures 4.2b and 4.3b it seems that the right side landmarks are less accurately placed on the test-data. Secondly, the algorithms used might have a tendency to put more emphasis on the left sided facial features before doing the right ones.



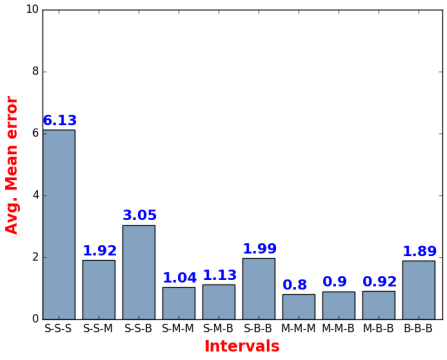
(a) Left Eye



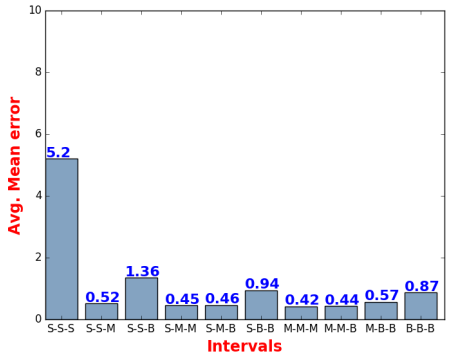
(b) Right Eye



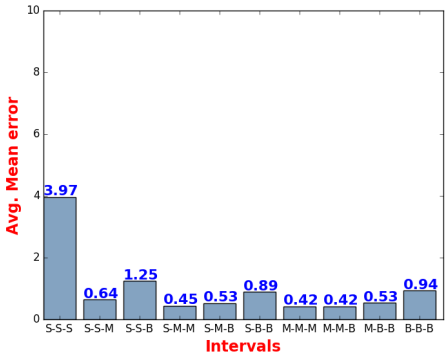
(c) Left Brow



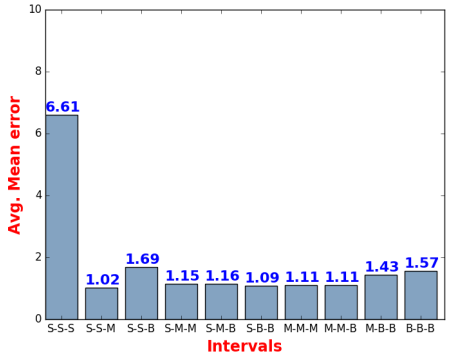
(d) Right Brow



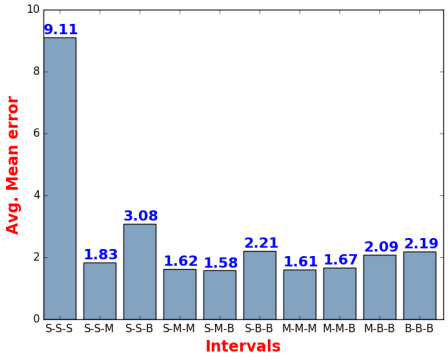
(e) Under the Nose



(f) Nose Bridge



(g) Inner Lips



(h) Outer Lips

Figure 4.5: Precision of the algorithm on different facial features. Left and right as seen from the observer.

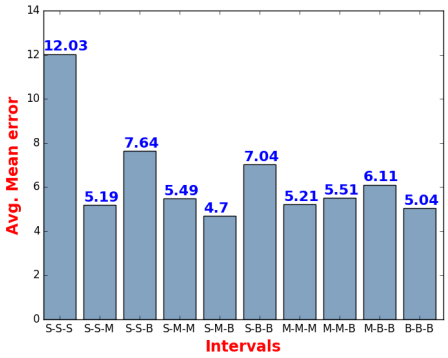
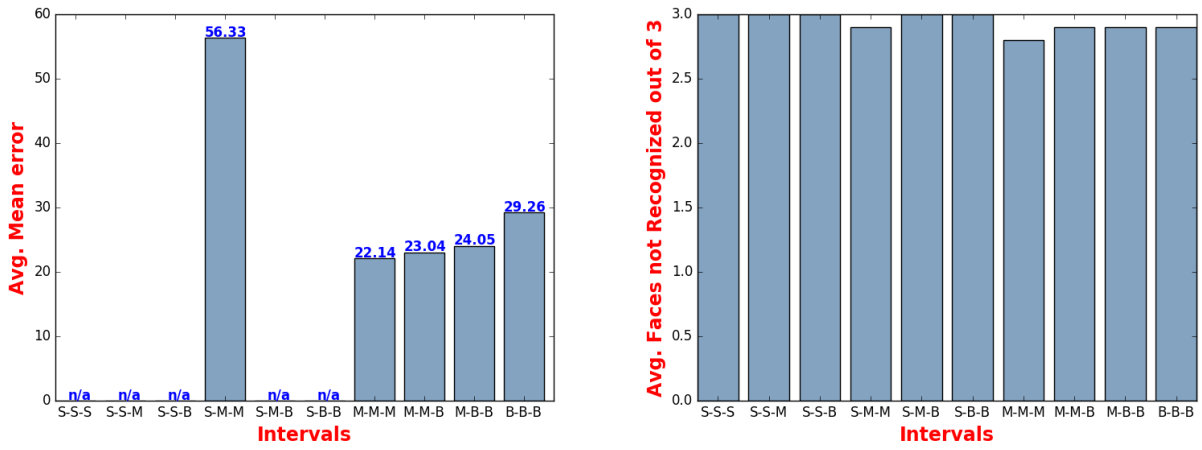


Figure 4.6: Precision of the landmarking of the jawline.

4.4 Mouth Region

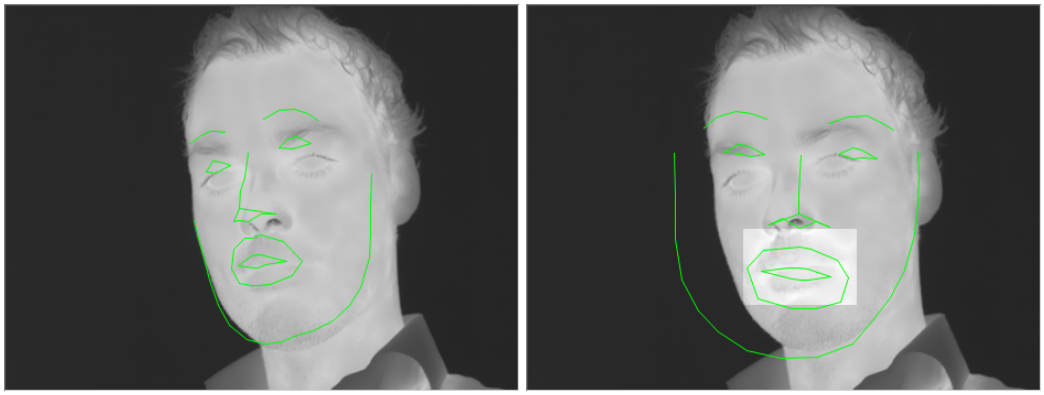
We will now look if the boosted mouth region method (see Section 3.2.3) improves our results. These results are displayed in Figure 4.7. Out of these results we can make up that the boosting method does not have a positive effect on the algorithm. Fewer faces are recognized and when they are, the algorithm is less accurate. An example can be found in Figure 4.8. Not only is the mouth region affected by this change as one might expect, but all the other landmarks are as well. This can be explained by the fact that the algorithm uses the relative positions of each landmark to fit the other landmarks. So if the mouth gets labelled wrongly, the jawline might also be placed wrongly and then the other features might also be placed worse. A reason for the decrease in reliability is that, since the algorithm uses intensity of the pixels to train the model, a large difference in intensity from the rest of the face might throw the algorithm off. Also the boosted region might not always be placed correctly. When this happens during training, the model will be trained wrongly, and when it happens during testing the mouth might not be found at all. So this method adds more steps that increase the complexity and decrease the reliability of the algorithm.



(a) Precision

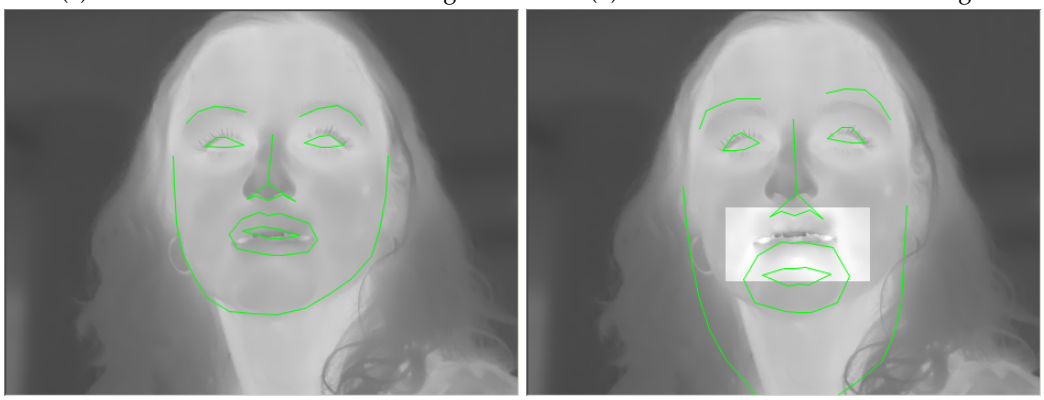
(b) Reliability

Figure 4.7: Precision and Reliability for various temperature intervals when using the mouth boost method.



(a) Found landmarks before boosting.

(b) Landmarks found after boosting.



(c) Found landmarks before boosting.

(d) Landmarks found after boosting.

Figure 4.8: Found landmarks on a non-boosted face and the landmarks found on a boosted face.

Chapter 5

Discussion and Conclusion

This study found that it is possible to successfully apply an algorithm that was developed for facial landmark detection in the visible light spectrum for infrared images. This is achieved using a simple translation from the temperature values to RGB-values. The drawback however is that this method does not always manage to succeed, in about 50% of cases it does not manage to find a face and thus place landmarks at all. Furthermore, trying to increase the precision of the mouth region, by changing the contrast of the mouth region, even worse results were obtained.

The study had some interesting findings. Firstly it is clear that the translation function for the temperature to RGB has a big impact on the precision of the facial landmarking algorithm. Managing to find a good translation with our limited search is also remarkable. So a good translation is important, but finding a somewhat decent one is not that hard, as there is a big area in the solution space that results in good translation. Secondly it is surprising that according to our results the left side of the face (right for observer) seems to be labelled less precisely than the left side, for the same landmarks. This is probably more a result of the algorithm than of the translation function.

We did encounter some problems. The limited data-set left us wondering if the results could be improved by including more examples of faces, since the data was recorded by hand (shaky) and not with a static background. The data could have been of better quality. Another problem was that when testing the jawline landmark, the jawline was compared to loose points on the line that the jaw forms instead of seeing if the jawline is correct. This had an unexpected impact on our results initially.

For further research it would be interesting to see if an even better temperature to RGB translation could be found, as we did not perform an extensive search for an optimal solution. It would also be interesting to see if an algorithm can be found that does not rely on learning algorithms but on knowledge about the temperature of different spots on the face, as we found out, for example, that the highest temperature on the face is often located between the eye and the nose. It would also be interesting to see how much the results could improve by using a bigger database, as ours was quite small. There are bigger databases available [KKM18], but these databases however did not work with our algorithm as an indication of the location of the head was

lacking. Exploring whether our translation method could work with another learning algorithm would also be interesting, as there are many different facial landmarking algorithms available.

In conclusion, the application of existing algorithms for facial landmarking developed for the visible light spectrum seems possible and a promising way forward.

Bibliography

[AVLD12] Karin Amrein, Angelika Valentin, Gerhard Lanzer, and Camilla Drexler. Adverse events and safety issues in blood donationa comprehensive review. *Blood reviews*, 26(1):33–42, 2012.

[CET01] Timothy F Cootes, Gareth J Edwards, and Christopher J Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):681–685, 2001.

[DGES+15] Abhishek Dutta, Manuel Günther, Laurent El Shafey, Sébastien Marcel, Raymond Veldhuis, and Luuk Spreeuwers. Impact of eye detection error on face recognition performance. *IET biometrics*, 4(3):137–150, 2015.

[Ery15] Melis Eryilmaz. Face segmentation in thermal images. Master's thesis, Middle East Technical University, Ankara, 2015.

[GABM14] Reza Shoja Ghiass, Ognjen Arandjelović, Abdelhakim Bendada, and Xavier Maldague. Infrared face recognition: A comprehensive review of methodologies and databases. *Pattern Recognition*, 47(9):2807–2824, 2014.

[GMC+10] Ralph Gross, Iain Matthews, Jeffrey Cohn, Takeo Kanade, and Simon Baker. Multi-pie. *Image and Vision Computing*, 28(5):807 – 813, 2010.

[GWWJ16] Chao Gou, Yue Wu, Fei-Yue Wang, and Qiang Ji. Shape augmented regression for 3D face alignment. In *European Conference on Computer Vision*, pages 604–615. Springer, 2016.

[IBUGi] Imperial College London Intelligent Behaviour Understanding Group (iBUG), Department of Computing. 300 faces in-the-wild challenge. `https://ibug.doc.ic.ac.uk/resources/300-W/`. Accessed: September 27, 2019.

[JdC18] Benjamin Johnston and Philip de Chazal. A review of image-based automatic facial landmark identification techniques. *EURASIP Journal on Image and Video Processing*, 2018(1):86, 2018.

[KAM16] Marcin Kopaczka, Kemal Acar, and Dorit Merhof. Robust facial landmark detection and face tracking in thermal infrared images using active appearance models. In *VISIGRAPP (4: VISAPP)*, pages 150–158, 2016.

[KHA+05] Seong G. Kong, Jingu Heo, Besma R. Abidi, Joonki Paik, and Mongi A. Abidi. Recent advances in visual and infrared face recognitiona review. *Computer Vision and Image Understanding*, 97(1):103 – 135, 2005.

[Kin] Davis King. dlib c++ library. `http://dlib.net/`. Accessed: July 4, 2018.

[KJ14] Vahid Kazemi and Sullivan Josephine. One millisecond face alignment with an ensemble of regression trees. In *27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014*, pages 1867–1874. IEEE Computer Society, 2014.

[KKM18] Marcin Kopaczka, Raphael Kolk, and Dorit Merhof. A fully annotated thermal face database and its application for thermal facial expression recognition. In *2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pages 1–6. IEEE, 2018.

[Lab95] Elise E Labbé. Treatment of childhood migraine with autogenic training and skin temperature biofeedback: a component analysis. *Headache: The Journal of Head and Face Pain*, 35(1):10–13, 1995.

[MSR+18] Adrian Cornelius Marinescu, Sarah Sharples, Alastair Campbell Ritchie, Tomas Sánchez López, Michael McDowell, and Hervé P Morvan. Physiological parameter response to variation of mental workload. *Human Factors*, 60(1):31–56, 2018.

[Ros] Adrian Rosebrock. Facial landmarks with Dlib, Opencv, and Python. `https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/`. Accessed: September 28, 2018.

[San] Sanquin. `https://www.sanquin.nl/`. Accessed: September 27, 2019.

[(us] ConcernedOfTunbridgeWells (username). What do 'real', 'user' and 'sys' mean in the output of time(1)? `https://stackoverflow.com/a/556411`. Accessed: October 15, 2018.

[Van] Lode Vandevenne. Lodepng. `https://lodev.org/lodepng/`. Accessed: July 4, 2018.

[WJ19] Yue Wu and Qiang Ji. Facial landmark detection: A literature survey. *International Journal of Computer Vision*, 127(2):115–142, 2019.