Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Measurement and Information Systems

Microexpression Detection Using Hybrid Expert Systems

Scientific Students' Association Report

Author:

Balázs Frey
Gábor Révy

Advisor:

dr. Gábor Hullám
Dániel Hadházi

2020

Contents

Kivonat

A mikrokifejezések olyan arckifejezések, melyek jellemzője, hogy univerzálisak, azaz minden embernél ugyanazt jelentik, másrészt tipikusan csak egy pillanatig jelennek meg. A felismerésükhöz jelenleg szakértői tudásra van szükség. E feladat automatizálása lehetőséget teremtene a mikrokifejezések széleskörű felhasználására.

Az emberek arckifejezéseit megfigyelve meghatározhatjuk a pillanatnyi érzéseiket, a reakciójukat egy őket ért hatásra, mint például egy reklámanyagra, vagy felhasználhatjuk bizonyos mentális betegségek detektálására, például depresszió vagy PTSD esetén.

A mikrokifejezések automatizált detektálásának két fő iránya a gépi tanulás és a szakértői képfeldolgozás alapú megközelítés.

A gépi tanulás során annotált minták segítségével a kialakított modellünk képes megtanulni a minták közös tulajdonságait. Ennek a megközelítésnek nagy előnye, hogy egy robosztus eljárást kapunk. Hátránya, hogy ez a robosztusság addig jellemző, amíg a felismerendő kép ugyanabból a háttéreloszlásból kerül ki, mint a tanítóminták. Ha a modellünket egy új környezetben vagy más célra kell felhasználnunk, a modellt újra kell tanítani feltéve, hogy rendelkezésünkre állnak más tanítóminták. Ezen megközelítésnek további hátrányai, hogy a tanításhoz sok minta kell, valamint a modellünk fekete dobozként működik, megértése és testreszabhatósága erősen korlátozott.

A szakértői rendszer több, gyakran egyedileg kialakított képfeldolgozó algoritmust használ. A megközelítés előnye, hogy kevés minta esetén is használható, valamint a rendszer teljesen átlátható, átalakítható, hiszen ismerjük és akár külön-külön módosíthatjuk is a komponenseit. Hátránya, hogy nagy robosztusság eléréséhez több komponensre van szükség, ami növeli a rendszer komplexitását, valamint a priori tudást is fel kell használni a komponensek építésekor. Az alacsonyabb robosztusság ellenére a rendszer további előnye, hogy testreszabhatósága miatt könnyen átalakítható különböző kontextusokhoz (pl. fényviszonyok, kép felbontása).

Ebben a dolgozatban egy hibrid megoldást mutatunk be, melynek alapját egy gépi tanulás alapú referenciapont-felismerő adja. A referenciapontok segítségével, szakértői képfeldolgozó és egyéb jelfeldolgozó algoritmusokat felhasználva képesek vagyunk különböző jellemzőket meghatározni. Az egyes részfeladatokra a szakirodalomban publikált megközelítések alapján javaslunk megoldásokat. A különböző algoritmusokat valós esetekben - videókon és képeken - alkalmazzuk és kiértékeljük az eredményeket. Továbbá a dolgozatban ismertetjük a gyakorlati alkalmazás során felmerült problémákat és tapasztalatokat is. Célunk, hogy a detektált jellemzők segítségével összetett arckifejezéseket, érzelmeket tudjunk meghatározni.

Abstract

Microexpressions are facial expressions that appear only for a moment. Their most important feature is that they are universal, that is they have mean the same mining to all people. In many scenarios, their recognition currently requires expert knowledge. Automating this task would provide an opportunity for its widespread use.

By observing people's facial expressions, we can determine their momentary feelings, their reaction to an effect, e.g. an advertisement, we can detect certain mental illnesses such as depression or PTSD.

The two main approaches for automated detection of microexpressions are machine learning and expert image processing.

Following the machine learning approach, we use annotated samples to enable our model to learn the common features of the samples implicitly. An advantage of this approach is that it results in a robust solution. On the other hand, this robustness is present as long as the input image is in the same manifold as the training samples. If we need to use our model in a new environment or for different purpose, the model needs to be retrained provided that appropriate training samples are available. Another disadvantage of this approach is that training requires a lot of samples, and our model behaves like a black box, so its explainability and customizability is severely limited.

An expert system uses several custom-designed image and signal processing algorithms. The advantage of this approach is that it can be used even with a small number of samples, and the system is completely transparent and adaptable, because we know and can modify its components individually. The disadvantage is that in order to achieve high robustness, several components are required, which increases the complexity of the system, and a priori knowledge must also be used when building these components. Despite the lower robustness, an additional advantage of this approach is that it can be directly adapted to different contexts (e.g. lighting conditions, image resolution) due to its easy customizability.

In this report, we propose a hybrid solution, utilizing a machine-learning-based landmark point recognizer. With the help of landmark points, we are able to determine different properties using expert image processing and signal processing algorithms. We use these algorithms in real cases - videos and pictures. In this thesis we analyze related tasks, overview corresponding methods of relevant publications, based on which we propose solutions to these tasks and examine their behaviour in practical applications. Our goal is to detect complex facial expressions and emotions with the help of the detected gestures.

Chapter 1

Introduction

Microexpressions are the reflection of emotions on the face for a very short time. Automating the detection of facial expressions would allow a wide range of uses, e.g. to study reactions for an advertisement or to help the diagnosis of mental illnesses. Experts usually distinguish 7 different basic emotions: anger, disgust, fear, happiness, sadness, surprise and contempt. Our final goal is to detect them. To do this, there are two main approaches: machine learning and expert image processing based systems. The machine learning based solution has two main requirements: an appropriate architecture and a lot of data. There are several machine learning based solutions with high accuracy on a given dataset, however their generalization capabilities are often limited, i.e. when a system trained on a particular dataset (e.g. studio quality images with adequate lighting) is used in another environment (e.g. low resolution images produced by phones), it will likely underperform. Furthermore, deep neural network based methods - which are typically used in this context - work as black boxes, i.e. they lack explainability making it hard to prove their proper operation. Additionally, they are usually not robust to small visual perturbations (e.g. abrasions on skin), which may cause the lack of the detection of microexpressions. Another problem can be the availability of samples for purposes other than research. In many cases, datasets and learned models may not be used for commercial applications.

In an expert system several image processing algorithms are utilized to make a prediction from an image or video. The abilities of the algorithms are known and can be fine-tuned for a given situation e.g. lighting conditions, shutter speed, angle. In our solution we would like to detect emotions in two main stages. The first step is to detect the motion of the muscles, the so-called action units on the face. Based on the moved action units the emotions can be determined. The first step utilizes a landmark detection algorithm to locate the key areas on the face. Using image processing and time series analysis the state and state change of the defined areas can be detected. Based on literature research we started to investigate and implement algorithms focusing on these particular areas of the face. The eye plays an important role in non-verbal communication, thus the gaze of the eye is examined. This includes locating the pupil and detecting blinks. The mouth, more precisely its properties carry also key information. Its shape, curvature, thickness, pose and the change of these greatly influence the predicted emotion. Furthermore, the movement of the eyebrows is also relevant in facial expressions, as it can used to discriminate between fear, anger and surprise.

Detecting elemental emotions in this way allows compound emotions to be detected as they often consist of a sequence or combination of elemental emotions. In addition, the

detection of microexpressions may also help to interpret the intentions of a communicating partner, and to detect non-verbal communication features.

A considerable advantage of the expert system based approach compared to the machine learning approach is that the result, e.g. the detection of complex features, such as a compound emotion, can be explained. In other words, a compound emotion can be traced back to its components, which may allow the fine tuning of the detection, i.e. the adaptation of such methods for various different scenarios. Another reason to use expert systems or hybrid systems that have an expert knowledge based component is, that appropriate datasets annotated with compound emotions are not publicly available yet.

In this thesis we propose a hybrid method to detect microexpressions based on videos and images. In the second chapter the theoretical background of detecting microexpressions is described, which is followed by previous solutions for this problem. In this section we introduce facial landmarks which form the basis of our solution. In the third chapter the investigated and implemented algorithms are explained. Besides the key elements of the algorithms, we demonstrate the practical aspects of these methods especially their applicability in certain scenarios. In the last chapter we test our methods on a selected set of images and discuss possibilities for further improvement.

Chapter 2

Background

2.1 Microexpressions

To recognize different microexpressions, the joint movement of different facial muscles is observed. This is dealt with by the discipline of psychology. In our work we began to deal with the recognition of the $6 + 1$ basic facial expressions: happiness, surprise, sadness, anger, fear, disgust and contempt. Contempt has been accepted by many emotion experts to be a universal emotion. These emotions are described as discrete, meaning that they are distinguishable based on a person's facial expression. Another interesting property of these expressions is that they are universal, i.e. they have the same emotion behind them for all people.

So that we know what movements need to be examined on the face, we have investigated the literature on this topic. Our two main sources were the work of Du and Martinez [8] and a comprehensive book on facial expressions edited by Paul Ekman and Erika Rosenberg-Band [10]. Although the paper of Du and Martinez addresses compound facial expressions, the basic expressions are also described in it. Articles on this topic, as well as these two papers, use the FACS system to describe each expression. The Facial Action Coding System [9] is a system, first described by Carl-Herman Hjortsjö, and later further developed by Paul Ekman and Wallace V. Friesen. In this latter work facial movements are taxonomized based on their appearance on the face. Movements on the face are grouped and named. An identifier is also assigned to each of them for easier reference. This identifier is called AU (action unit). The system also contains a scoring procedure to differentiate the strength of the movement of an action unit. This is marked from A (mild strength = "traced") to E (strong = "maximum"). An example is AU 4E. AU 4 is called the "Brow Lowerer", and E means "maximum". The description of AU 4E is: "Brow pulling together or lowering is *maximum*" (from [9]). Description of the joint movement of AUs can also be found in the book.

Action units and their descriptions were collected for each expression one-by-one based on [9]. We tried to focus on AUs that are probably recognizable by an algorithm based on a picture or video. Table 2.1 summarizes the AUs corresponding to the investigated emotions followed by the detailed description of each AU.

expression	AUs
happiness	$6; 12; 25$
surprise	$1, 2; 5; 25; 26$
sadness	$1, 4; 6; 11; 15; 17$
anger	$4; 7; 11; 27$
fear	$1, 2; 5; 20; 25$
disgust	$9; 10; 24$
contempt	$R12; R14$

Table 2.1: Action units related to the 6+1 basic emotions

AU 6 The muscles around the eye constrict and the cheek is lifted upwards. The movement may cause crow's feet lines (wrinkles next to the outer edge of the eyes).

AU 12 The corners of the lips are pulled back and upward.

AU 25 The lips part, the teeth may be exposed.

AU 1,2 Inner (AU 1) and outer (AU 2) corners of the eyebrows are raised, thus the entire eyebrow is pulled upwards. Horizontal wrinkles may appear on the forehead.

AU 5 Upper eyelid is raised, eye aperture widens.

AU 26 Jaw drops by relaxation, teeth separate.

AU 1,4 Inner corners of the eyebrows are pulled slightly closer causing slight vertical wrinkle between the eyebrows or on the forehead.

AU 11 Upper lip is pulled upward and laterally, thus the the upper middle portion of the nasolabial furrow deepens.

AU 15 The corner of the lips is pulled down. Can cause the stretching of the lower lip and wrinkles below it.

AU 17 Chin boss is pulled upward and the lower lip is pulled upward. This causes the shape of the mouth to appear $\cap$ or increase this shape if present in neutral.

AU 4 Eyebrows are lowered and pulled closer together. Vertical wrinkles are produced between the eyebrows.

AU 7 Eyelids are tightened, eye aperture is narrowed. Lower eyelid is raised covering more of the eye than usually.

AU 27 The mandible is pulled down, cheeks flatten and stretch. The mouth may shape a vertical oval form.

AU 20 Lips are pulled back laterally, become flat and stretched. This can cause wrinkles at the lip corners. The nostril's opening is elongated.

AU 9 The skin along the sides of the nose is pulled upwards causing wrinkles along the sides and across the root of the nose. Eye aperture is narrowed. Center of the upper lip is pulled upwards.

AU 10 Upper lip is raised. The nasolabial furrow is deepened. Nostril wings are widened.

AU 24 Lips are pressed together without pushing up the chin boss.

AU R12 Lip corner is raised a trace on the right side (could also be L12).

AU R14 Right lip corner is tightened with slight lateral movement. May cause wrinkling around the lip corner.

2.2 Machine learning based algorithms

2.2.1 Landmark detection

Landmark detection is a fundamental building block of many applications such as face recognition, pose recognition, emotion recognition and many more in augmented reality. In image localization tasks we use machine learning algorithms to detect objects by predicting coordinates of the object's bounding box. In case when a finer prediction is required, determining the direction of gaze or key points along the mouth is necessary, so that the shape of the mouth can be extracted and determined whether the person is smiling or frowning. Additionally, predicting points that help defining the edge of the face can be useful.

In many computer vision applications, algorithms need to recognize these essential points of interest rather than just bounding boxes on the input image. We refer to these points as landmark points. In our work we used Dlib toolkit [16] to detect essential landmark points. The detection consists of the following steps: (1) detection of the bounding box of the face, and (2) identifying landmark points on faces.

2.2.2 Detecting bounding boxes

Frontal face detector is a tool for detecting the positions of faces in an image. It is based on a histogram of oriented gradients (HOG) [6, 20] and linear SVM [2]. Dlib also provides a CNN based object detector, but it is computationally intensive and it is not suitable for real-time videos.

2.2.3 Detecting landmark points

Shape predictor is a tool that takes in an image region containing an object and outputs a set of point locations that define the pose of the object. The predictor uses the state-of-the-art method from Kazemi and Sullivan based on an ensemble of regression trees [15]. This method is using a training set of labeled facial landmarks on an image (see Figure 2.1). These images are manually labeled, specifying specific (x, y)-coordinates of regions surrounding each facial structure.

For facial landmark detection we use Dlib's [16] 68-face-landmark shape predictor that was trained on the iBUG 300-W face landmark dataset [26]. The method is using a cascade of regression trees, training them via gradient boosting with a squared error loss function. It also introduces a prior probability on the distance between pairs of input pixels which allows the boosting algorithm to efficiently explore a large number of relevant features.

2.2.4 Other machine learning-based solutions

There are many other solutions based on machine learning. They are mostly end-to-end approaches.

CNN A very common approach is to use end-to-end Convolutional Neural Networks for the classification of emotions or using CNN as a feature extractor and an another method for classification. Ouellet [22] used a 7-layer CNN trained on ImageNet [7] for feature extraction. An SVM (support-vector machine) is then trained to classify

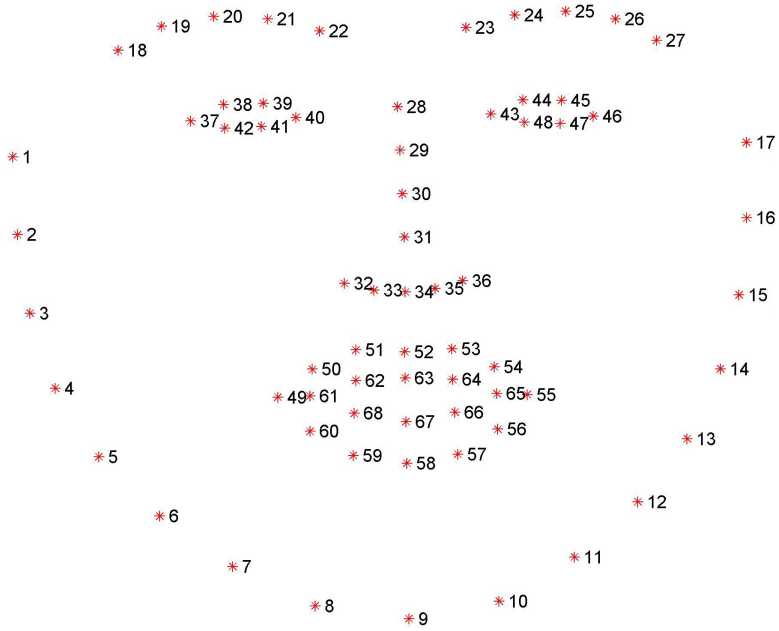


Figure 2.1: Layout of landmark points

the emotion based on the output of the CNN, which is generated using the Extended Cohn-Kanade (CK+) Dataset [19] which consists of 700 sequences from footages of 210 actors labeled with the 7 basic emotions. This system reached 94.4% accuracy.

One CNN-based study is worth highlighting: in their work [3], Breuer and Kimmel investigated the connection between the features learned by the layers of the neural networks and the action units of FACS [9]. They trained a simple CNN on the CK+[19], the FER2013 [4] and the NovaEmotions [30] dataset. The FER2013 dataset is created using the Google image search API. It contains nearly 36000 images divided into 8 classes (7 emotions class + 1 neutral class). The NovaEmotions dataset contains over 42000 images taken of 40 people during gaming. Their CNN consists of 3 blocks each containing a filter map layer with activation and a max-pooling layer. The blocks are followed by a fully connected layer containing 512 neurons and the output layer for the classes. They reached an accuracy of 98.62% on the CK+, 72.1% on the FER2013 and 81.3% on the NovaEmotions dataset. The interesting part is that they employed Zeiler et al. [38] and Springenberg's [29] methods to visualize the filters responsible for a given emotion classification task. They showed that there is a significant correlation between the learned features and action units of the FACS.

AE Autoencoders are able to learn the efficient encoding and decoding of high-dimensional data by dimensionality reduction. The solution of Zeng et al. [39] uses a deep sparse autoencoder as the last step of the classification process. First, facial landmarks are located using the active appearance model [5]. Then HOG [6] (histogram of oriented gradients), LPB [28] (local binary patterns) and gray value descriptors from patches around 51 chosen landmark points are applied. These features are then compressed using the PCA (principal component analysis) method. The output of the PCA is then fed into the AE. The CK+[19] is used for training the system which reached an accuracy of 95.79% on the 7 and 89.84% on the 8 class classification.

RNN Recurrent neural networks are able to capture temporal information thus are considered more suitable for videos. Yan et al. built an emotion recognition system [37] that fuses the information extracted from the audio and the video frames. A CNN based RNN is a part of this system. The input images are fed into a VGG-Face [24] feature extractor network. The features extracted are then utilized as input for a BRNN [27] (bidirectional RNN) to model the changes of the face. Their system was trained on AFEW 6.0 [1] and CHEAVD [17] databases. AFEW 6.0 (Acted Facial Expressions In The Wild) consists of 773 training, 383 validation and 593 testing video samples selected from hollywood movies and TV reality shows. CHEAVD (Chinese Natural Emotional Audio Visual Database) contains 1981 training and 243 validation video clips selected from Chinese movies and TV programs labeled with 8 emotion classes. The accuracy of the CNN-BRNN is 44.46% on the AFEW 6.0 (validation set result) and 51.03% on the CHEAVD dataset.

2.3 Expert system based solutions

Among previous solutions, there are some expert systems for detecting facial expressions. Pantic and Rothkrantz created a hybrid system called ISFER [23] (Integrated System for Facial Expression Recognition). It utilizes several algorithms to detect the movement of action units[9] using a still dual facial view (front and side views) image as input. Based on the detected movement of the AUs the expression is determined. The algorithms used in the system are diverse, and the features are tracked redundantly. Algorithms include a neural network to find features of the eyes, ACM snake to track the eye, fuzzy classifier [35] to classify mouth expressions and chain codes [34] to localize the eyebrow contour. The average classification accuracy of 6+1 expressions (surprise, fear, disgust, anger, happy, sad + blended notional expression) is 90.57%. The drawback of this approach is that the method is not openly available, and the data set upon which the method was fine tuned is overly specific, i.e. it is inappropriate for general video based detection.

Another expert system [12] is developed by Ghimire et al.. Using the Dlib [16] toolkit facial landmark points are extracted from the picture. The face is then divided into 29 regions, some of which are previously selected to extract geometric features from using LPB [28] (local binary pattern) and NCM [14] (normalized central moments) descriptors. These features are then fed into an SVM to classify the expression. This solution reached an accuracy of 97.25% on the CK+ [19] database.

Chapter 3

Methods for detecting microexpressions

3.1 Gaze detection

In this section several algorithms related to gaze detection are described. Gaze detection includes the detection of the eye and the iris. Blink detection is also part of this task which can be an essential feature when the aim is to identify surprise, disgust or fear.

3.1.1 Basic method

The first, naive solution was to use a simple landmark-based algorithm. The input image is converted into grayscale and fed into the landmark detector (see subsection 2.2.3. Using the eye landmark points (as shown in Figure 3.1), an eye-mask is created and the eye is cut out from the image. This eye-image is divided into left and right parts. The gaze is determined by comparing the number of bright pixels between the two sides. The main problem with this algorithm is that it only works under laboratory conditions. In case of inadequate lighting, for example in non-homogeneous lighting conditions, the image of the camera is not suitable for the "pixel counting". Another problem is when the person in front of the camera is not looking directly into the camera. In this case, the upper or lower eyelid may cover too much of the eyeball. Also, landmark detection is less accurate, thus parts outside the eyes may also be included in the mask.

3.1.2 EyeTab-based method

The main solution for the gaze detection task is a more complex algorithm based on an open source gaze estimator called EyeTab [36]. Although it contains many device and environment specific constants and algorithms, it provided a good basis. In the following the whole algorithm is described.

3.1.2.1 Eye localization

In the first step eye ROIs (region of interest) are determined. Originally, OpenCV Haar-like feature based cascade classifiers [33] were used to detect eyepairs. However, by investigating the results, we found, that the estimation of the eyes' ROI is more accurate if it is

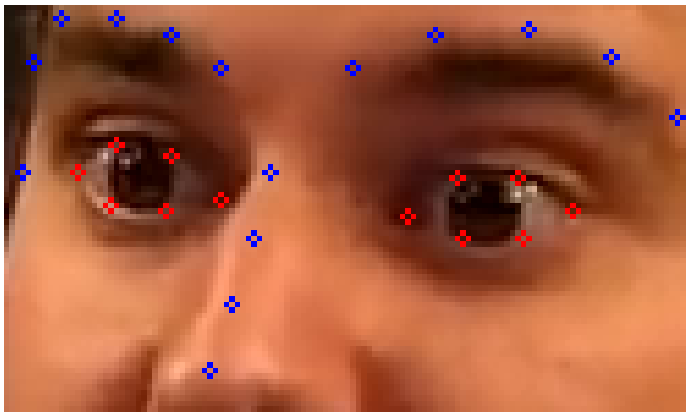


Figure 3.1: Landmarks on the eye.

defined by the landmark points than by the cascade classifiers. Thus we defined the ROIs by expanding the rectangles created by the eye landmark points. Since the edges of the eyes are recognized accurately by the landmark detector based on the investigation of the results, the ROI rectangles defined by the landmark points have only been expanded in the vertical direction by a factor of 2.

3.1.2.2 Removing specular reflection of light

The next step is the removal of glare caused by different light sources. The algorithm starts with converting the eye image to grayscale, then Gauss low-pass filtering is applied on it. Using morphological closing[1] the eyelashes are suppressed. The detection of glare is performed by thresholding the image using the 50th percentile. This method selects bright areas in the image, from which the small areas are then selected (areas that are not greater than the half of the width of the eye). These small reflection area proposals are then inpainted with a method described by Alexandru Telea [31]. Figure 3.2a shows an example for the result of this algorithm.

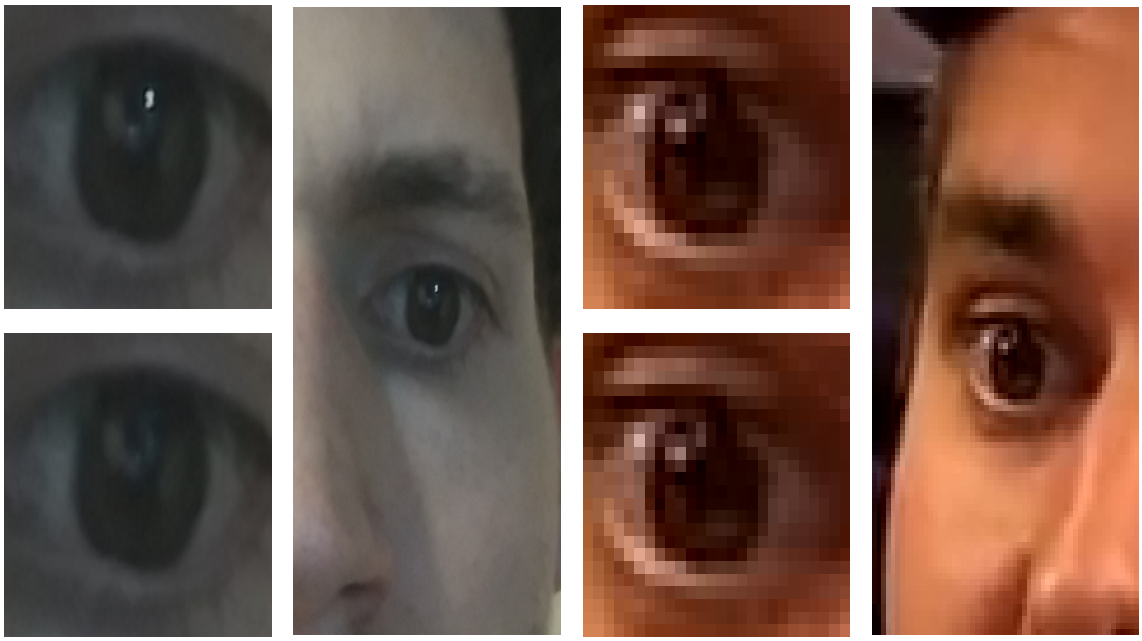
Applying this method resulted in varied results. It rarely suppresses the real flares, but only bright areas out of the eye region are inpainted (as shown in Figure 3.2b) or the picture is left unchanged. The following pupil center detector proved to be robust enough to return good results in these cases, too, which makes the necessity of the specular reflection removal step questionable.

3.1.2.3 Pupil detection

The pupil is detected by combining two algorithms: one is gradient based and the other is isophote based.

In the original implementation [36] the red channel of the RGB image (see Figure 3.3a) was used as input, however based on our experiments, applying homomorphic filtering on the full image and cutting the eye-ROIs (see Figure 3.3b) results in slightly more stable pupil estimations. Homomorphic filtering can be used to correct results of inadequate lighting, for example a shady eye socket. Only low frequency components are filtered out, therefore edges are preserved.

[1] Morphological closing on an image is defined as a dilation followed by an erosion. [25]



(a) Specular reflection is correctly removed.

(b) Bright area in the top right corner is impainted.

Figure 3.2: Two examples of the specular reflection removal algorithm.

The homomorphic filtering first maps the (min-max) normalized grayscale image to logarithmic scale. Then, on this image a high-pass filter is applied by subtracting its Gauss filtered version from it. Finally, it is mapped back to the original scale and cut under 0 and over 1. The algorithm can be formally written as follows:

$$I_{homomorphic} = T(\exp(\log(I) - \log(I) * G_\sigma))$$
$$T(x) = \min(x, 1)$$

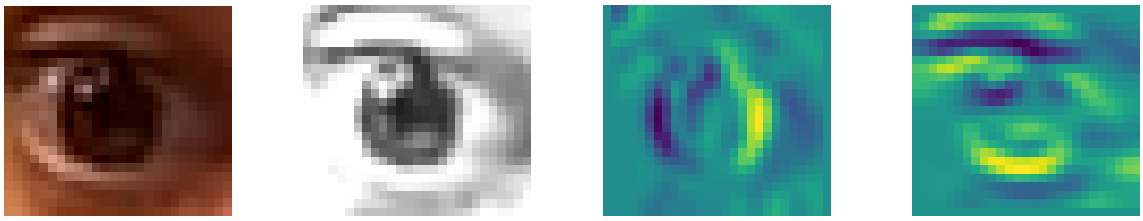
Here, G_σ denotes the convolution kernel of the Gauss blurring and $*$ means convolution. T cuts the values under 0 and over 1.

First, the gradient based pupil map is created. Directional gradients (see Figure 3.3c and 3.3d) and gradient magnitudes (see Figure 3.3g) are computed on the eye image and are thresholded (see Figure 3.3e and 3.3f). The next step exploits the fact that the vector from the pupil center pointing to the edge of the iris and the gradient vector in that point are parallel or meet at an acute angle. Every point in the image is taken as a center candidate and the squared dot products are accumulated for every other point. Thus the gradient map can be calculated as:

$$R_{grad}(x,y) = \sum_{x',y'} \left\{ \left\langle \frac{[x'\ y'] - [x\ y]}{\|[x'\ y'] - [x\ y]\|_2}, \frac{\nabla \mathbf{I}(x',y')}{\|\mathbf{I}(x',y')\|_2} \right\rangle^2 \cdot (255 - \mathbf{I}(x',y')) \cdot \mathrm{Ind}\left\{ \|\nabla \mathbf{I}(x',y')\|_2 > 0 \right\} \right\}$$

Here, Ind denotes the indicator function. I is the homomorphic luminance compensated input image and ∇ indicates the operator of the gradient. $\|\cdot\|_2$ and $\langle\cdot,\cdot\rangle$ denote the

operators of the $L2$ norm and the scalar product, respectively. Since we want to combine the two methods the resulting accumulated values are stored as a feature map as shown in Figure 3.3h.

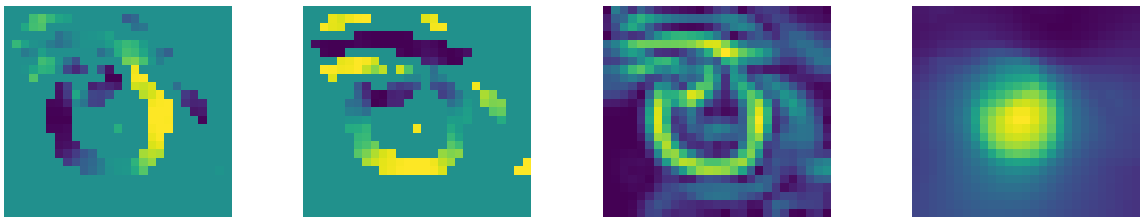


(a) RGB image of the eye.

(b) Homomorphic filtered image.

(c) Gradients along the x axis.

(d) Gradients along the y axis.



(e) Thresholded gradients along the x axis.

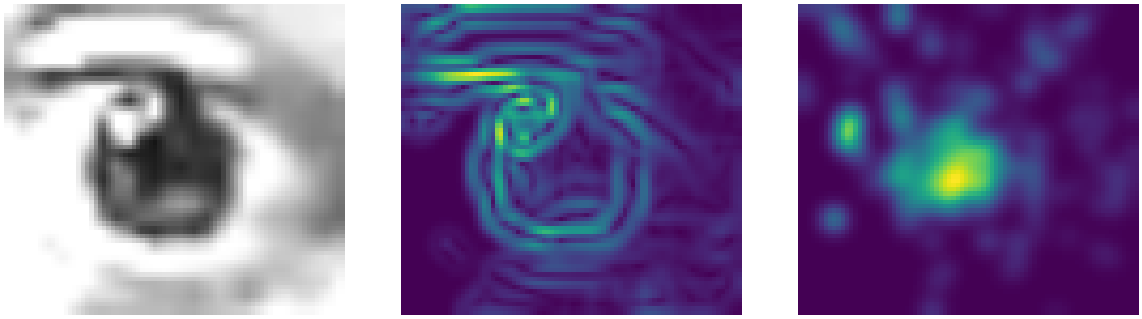
(f) Thresholded gradients along the y axis.

(g) Gradient magnitudes.

(h) The computed pupil feature map.

Figure 3.3: Steps of pupil detection using the gradient based method

In the next step the isophote-based pupil map is created. This method was described by Valenti et al. [32]. An isophote (see Figure 3.4b) is a curve connecting points with the same intensity in the image. The isophotes are independent of rotation and linear lighting changes, i.e., brightness and contrast, which makes the method more robust. Based on their curvature, their center can be calculated, thus we obtain another center map as shown in Figure 3.4c.



(a) Homomorphic filtered image.

(b) Curvature of the isophotes.

(c) Resulting feature map.

Figure 3.4: Steps of pupil detection using the isophote based method

A combined feature map is formed by weighting the center maps returned by the two methods. The point with the highest value in the map is chosen as the center of the pupil.

3.1.2.4 Eyelid detection

The next step is to locate the upper and lower eyelids of the eye. Here, the red channel of the eye image serves as the input. First, the points of the upper eyelid are determined, so the ROI image is cut in half. The eyelashes are suppressed by applying morphological closing [25]. Horizontal texture is then extracted using a derivative of Gaussian filter, formally described as:

$$\mathbf{I}_y^{'} = \nabla_y \left(\Big((\mathbf{I} \oplus \mathbf{M_k}) \ominus \mathbf{M_k} \Big) * \mathbf{G}_\sigma \right)$$

where $\oplus$ and $\ominus$ denote the operation of morphological dilation and erosion, respectively. $\mathbf{M_k}$ is the 5×5 disk-shape structuring element of the morphological operation. $\mathbf{G}_\sigma$ is the 2 dimensional, isotropic Gaussian function with σ parameter and $*$ denotes convolution. ∇_y is the gradient along the y axis. The suspected position of the iris is masked out and in each column of the image the pixels with the highest intensity are selected:

$$MT(x) = \arg\max_{y'} \{\mathbf{I}'_\mathbf{y}(x, y')\}$$

If there is another pixel with similarly high intensity in the column between the highest-intensity point and the bottom, then it replaces the highest-intensity point:

$$MT'(x) = \arg\max_{y' \in \left[MT(x)+5,\ldots,MT(x)+100\right]} \{\mathbf{I}'_\mathbf{y}(x, y')\}$$

$$LID(x) = \begin{cases} MT'(x), & \text{if } \mathbf{I}'_\mathbf{y}(x, MT'(x)) > \mathbf{I}'_\mathbf{y}(x, MT(x)) - \Delta \\ MT(x), & \text{if } \mathbf{I}'_\mathbf{y}(x, MT'(x)) \leq \mathbf{I}'_\mathbf{y}(x, MT(x)) - \Delta \end{cases}$$

Here, Δ denotes the intensity difference, which is set to 50 (the intensities of the image are scaled into the range [0,255]). The idea behind this maximum-search is, that due to the inaccuracy of the eye-detection the top of the eye-socket can be visible in the ROI image which is also a horizontal line next to the upper eyelid. This way the points of the limbus are determined. Using the RANSAC[2] method a parabola is fitted to the points.

The lower eyelid is determined using almost the same algorithm. The difference is that the image is also split into right and left side on which diagonal derivative of Gaussian filter with opposite direction is applied to extract the texture of the lower eyelids. In the last step only the maximum search is performed since there is no other expected line. The resulting points are filtered by intensity.

3.1.2.5 Limbus detection

The next task is to determine the points of the limbus. Based on the investigations, in this step the homomorphic filtered image is used as input as shown in Figure 3.5a. Median

[2]RANSAC [11] (Random sample consensus) is an iterative algorithm for fitting a model on a set of observed data.

filter is applied on this image to reduce the noise in the iris while keeping its contours clear (see Figure 3.5b). A polar coordinated representation of the image is then created (see Figure 3.5c). This can be formulated as:

$$\mathbf{I}^{pol}_{(x_0,y_0)}(\rho,\theta) = \text{Median}\left\{\,\text{Hom}\{\mathbf{I}\}, \text{Box}_{5,5}\,\right\}(x_0 + \rho\cdot\cos\theta, y_0 + \rho\cdot\sin\theta)$$

Here, $(x_0.y_0)$ denotes the center of the image in cartesian coordinates. Hom and Median denote the homomorphic and median filtering, respectively, and $\text{Box}_{5,5}$ is the 5×5 kernel window of the median filter. This image is blurred using a low-pass filter and the contour search space is reduced by cutting the top and bottom of the image. The horizontal lines are extracted using a derivative of Gaussian filter along the rows as shown in Figure 3.5d. This is computed by:

$$\mathbf{I}^{pol}_{(x_0,y_0)}{}' = \mathbf{I}^{pol}_{(x_0,y_0)}(\rho,\theta) * \nabla_\rho \mathbf{G}_\sigma,$$

where $\mathbf{G}_\sigma$ is the isotropic Gaussian function with σ parameter, ∇_ρ is the operator of the gradient along the ρ axis and $*$ denotes the operation of convolution. The angles belonging to the top and bottom of the limbus are masked out (as shown in Figure 3.5e) since they are usually covered by the eyelid. On each column maximum search is performed to determine the border between the iris and the sclera (the white of the eye), formally:

$$Limb(\theta) = \begin{cases} \text{not defined}, & \text{if } \theta \notin \Theta \\ \arg\max\limits_{\rho} \{\mathbf{I}^{pol}_{(x_0,y_0)}{}'(\rho,\theta)\}, & \text{if } \theta \in \Theta \end{cases}$$

Here, θ denotes the set of angles retained. These points are then transformed back into the cartesian representation.

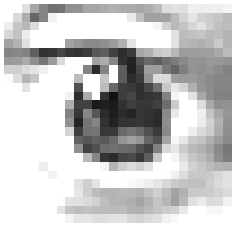
3.1.2.6 Blink detection

The pupil detection algorithm proposes a region of interest also in frames, when the eyes of the examined person is closed or is blinking. Since blinking is also a facial microexpression, detecting these frames is also an important task. In order to do that, local Hessian based image analysis is applied to distinguish true and false region of interests proposed by the pupil detection method.

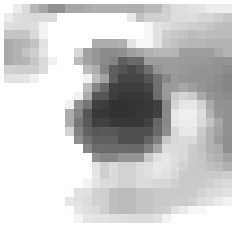
A typical pupil in a grayscale intensity image (also after utilizing homomorphic luminance compensation method) is a round region, which is darker compared to its neighboring pixels. These regions can be highlighted by local Hessian based filtering method, which is defined by:

$$\mathbf{H}_\sigma = \alpha(\sigma)\cdot\nabla^2(\mathbf{I} * \mathbf{G}_\sigma)$$

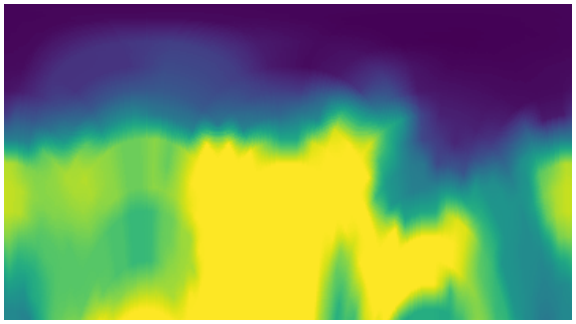
where ∇ is the gradient operator, $\mathbf{G}_\sigma$ is the 2 dimensional, isotropic Gaussian function with σ parameter and $*$ denotes the operator of the convolution. $\alpha(\sigma)$ is a normalization scalar which compensates for the multiplicative dependency of the norm of the matrix on the σ parameter. Based on the scale space theory [18] $\alpha(\sigma) = \sigma^2$. The value of σ depends



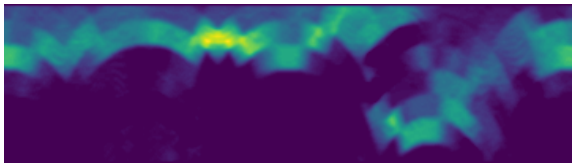
(a) Homomorphic filtered image.



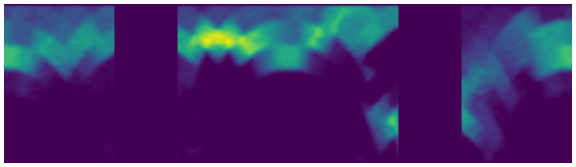
(b) Median filtered input.



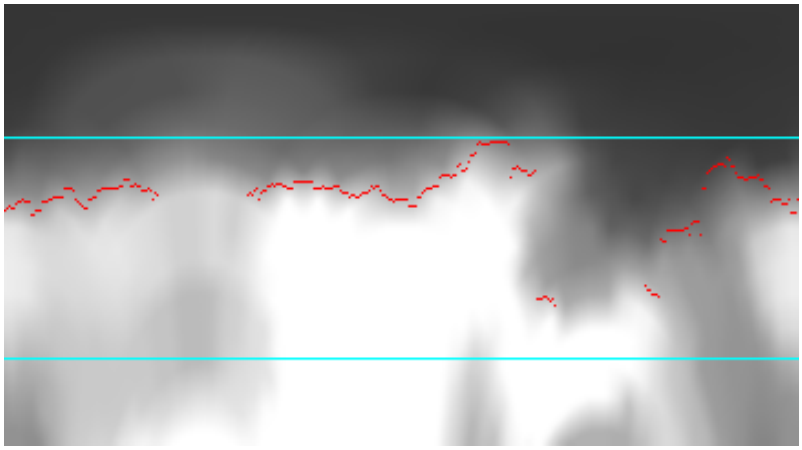
(c) Polar transformed image.



(d) Polar image filtered with derivative of Gaussian.



(e) Angles belonging to top and bottom masked out.



(f) Resulting limbus points in polar coordinates.

(g) Resulting limbus points in cartesian coordinates.

Figure 3.5: Steps of limbus detection

on the size of the blob which is proposed as a pupil candidate, which is determined by the solution of the optimization problem:

$$\sigma(r) = \arg\max_{\sigma} \Big\{ (\mathbf{D}_r * \alpha(\sigma) \cdot \mathbf{G}_\sigma)(0,0) \Big\} = r/\sqrt{2}$$

where $\mathbf{D}_r$ denotes the binary image of the $(0,0)$ centered homogeneous sphere with r radius:

$$\mathbf{D}_r(x,y) = \begin{cases} 1, & \text{if } \Big\| [x\ y] \Big\|^2 \leqslant r \\ 0, & \text{if } \Big\| [x\ y] \Big\|^2 > r \end{cases}$$

Please note that the local Hessian operator assigns a 2×2 matrix to each pixel of the examined image. Since the shape of the visible pupil highly depends on the direction of the gaze and the distance between the lower and the upper eyelids, its radius can not be precisely proposed by the pupil proposal algorithm, therefore a set R of possible r-s is defined. An ellipse is fitted to the limbus points using the RANSAC [11] method, the axes of which are used as radius proposals. Furthermore we found, that the eye landmark points are stable in the two corners of the eye. Based on this fixed ratios $(\frac{1}{2}, \frac{5}{12}, \frac{1}{3})$ of the half distance between the corners of the eye are added to the proposals.

The best fitting $r \in R$ is defined by its corresponding scale, in which the largest amplitude curvature of the intensity image is the minimal (which corresponds to our observation, that the pupil can be approximated by a dark blob):

$$r^* = \arg\max_{r} \Big\{ \lambda_{\max}\{\mathbf{H}_{\sigma(r)}\} \Big\}$$

where λ_{max} denotes the maximal amplitude eigenvalue of the Hessian matrix, and (x_0, y_0) denotes the proposed center of the pupil. Eigenvalues of the Hessian are calculated by:

$$\begin{aligned} \lambda_1\{\mathbf{H}\} &= \frac{\operatorname{trace}\{\mathbf{H}\} + \sqrt{\operatorname{trace}\{\mathbf{H}\}^2 - 4\det\{\mathbf{H}\}}}{2} \\ \lambda_2\{\mathbf{H}\} &= \operatorname{trace}\{\mathbf{H}\} - \lambda_1\{\mathbf{H}\} \end{aligned}$$

The first equality can be derived based on the following observations:

$$\begin{aligned} \operatorname{trace}\{\mathbf{H}\} &= \operatorname{trace}\{\mathbf{Q}^T\mathbf{\Lambda}\mathbf{Q}\} = \operatorname{trace}\{\mathbf{\Lambda}\mathbf{Q}\mathbf{Q}^T\} = \operatorname{trace}\{\mathbf{\Lambda}\} = \lambda_1 + \lambda_2 \\ \det\{\mathbf{H}\} &= \det\{\mathbf{Q}^T\mathbf{\Lambda}\mathbf{Q}\} = \det\{\mathbf{Q}^T\} \cdot \det\{\mathbf{\Lambda}\} \cdot \det\{\mathbf{Q}\} = \det\{\mathbf{\Lambda}\} = \lambda_1 \cdot \lambda_2 \end{aligned}$$

where $\mathbf{H} = \mathbf{Q}^T\mathbf{\Lambda}\mathbf{Q}$ is the eigenvalue–eigenvector decomposition of the Hessian matrix, which always exists, because $\mathbf{H}$ is symmetric. Please note that $\mathbf{Q}$ is an orthonormal matrix, therefore $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$.

We chose this way of calculation instead of the most commonly utilized power iteration [21], because this is faster in practice and similarly robust.

After we get the scale, the only task is to examine the circularity of the examined blob, which can be measured by:

$$C_{(r)}(x_0, y_0) = \lambda_{\max}\Big\{\mathbf{H}_{\sigma(r)}(x_0, y_0)\Big\} \cdot \lambda_{\min}\Big\{\mathbf{H}_{\sigma(r)}(x_0, y_0)\Big\}$$

Since the Hessian matrix is symmetric, it can be diagonalized by an orthonormal matrix, which inducts that the product of the eigenvalues is equal to the determinant of the Hessian, which can be computed faster than the value of the lower amplitude eigenvalue, therefore computed by:

$$C_{(r)}(x_0, y_0) = \det\Big\{\mathbf{H}_{\sigma(r)}(x_0, y_0)\Big\}$$

From there, the openness of the examined eye is calculated by:

$$\mathrm{Ind}\Big(C_{(r)}(x_0, y_0) > c\Big) \cdot \mathrm{Ind}\Big(\lambda_{\max}\Big\{\mathbf{H}_{\sigma(r)}(x_0, y_0)\Big\} > 0\Big)$$

Here, c is the "roundness/non-prolongation" set to 0.3. The first term is true if the portion of the eigenvalues is not too high, therefore the proposed pupil region is circular, while the second term is true, if the region is darker than its neighboring pixels.

3.1.3 Hough transform based method

In addition to the EyeTab-based method, we also examined Hough transformation based pupil detection. In this section, we describe the basics of this method.

3.1.3.1 Pupil detection

The method is using two main algorithms: edge detection and Hough transform based circle detection.

The first step is to find the bounding boxes for both eyes based on landmark points provided by Dlib. Inside the bounding box of an eye we need to find the edges of the pupil.

Edge detection The motivation behind the idea is that there is a high intensity change at the boundary line between the iris and the sclera, which appears in the form of a high-amplitude gradient on the monochromatic image. By highlighting regions where there is a high intensity change, we are more likely to find the edge of the pupil.

In order to produce the edge image we run the following steps on the input image:

- Conversion to grayscale image
- Convolution with gaussian kernel with kernel size of 5×5 and $\sigma = 2$.

- Convolution with vertical and horizontal Sobel kernels.
- Computation of $\sqrt{v^2 + h^2}$ for each pixel, where v and h are the values of the pixels in the same position on the result image of the vertical and horizontal Sobel convolution.
- Thresholding gradients to keep only pixels above the 90th percentile.
- Application of morphological erosion.

These steps can be formalized in the following way:

$$\mathrm{Gr}(\mathbf{I}, \sigma) = \mathrm{Ind}\left(\| \nabla(\mathbf{I} * \mathbf{G}_\sigma)\|_2 > \mathrm{T}_{90}\left(\| \nabla(\mathbf{I} * \mathbf{G}_\sigma)\|_2\right)\right)$$

where $\mathbf{G}_\sigma$ is the 2D Gaussian kernel, ∇ is the gradient operator, which is implemented by Sobel filters, $\mathrm{T}_{90}(\mathbf{X})$ is the 90th percentile of the intensities of $\mathbf{X}$, $\mathrm{Ind}(\cdot)$ is the indicator operator. The operators are computed element-wise.

$$\mathrm{Edges}(\mathbf{I}, \sigma) = \mathrm{Gr}(\mathbf{I}, \sigma) - \mathrm{Gr}(\mathbf{I}, \sigma) \ominus \mathbf{Box}_{3,3}$$

where $\ominus$ is the morphological erosion, while $\mathbf{Box}_{3,3}$ is the 3×3 sized box kernel.

The result of the algorithm are shown on Figure 3.6 and Figure 3.7. As it can be seen, the algorithm detects borders of the edges. This can be beneficial if the visible shape of the iris is elliptical.

Hough circle Hough transform [13] is an algorithm, that can find objects on images, that do not have a continuously tractable border, but the shape of the object can be defined by closed function of small number of parameters.

In a two-dimensional space, a circle can be described by:

$$(x - a)^2 + (y - b)^2 = r^2,$$

where (a, b) is the center of the circle, and r is the radius. A given (x, y) point votes for all (a, b, r) values that satisfy the equation above, so our Hough space will be three-dimensional. The method is based on the fact that for each point (x, y) of the original circle, whose center is at (a, b), the distance from (a, b) is r.

In practice, the circles can be detected by finding local maxima in the Hough space representation ($Acc[a, b, r]$). Initially, every element in the Acc tensor is zero. For a specific radius r we iterate through each edge point (x, y) in the original space, formulate a circle that is centered at (x, y) with radius r and for every point (x', y') of its contour increase the value of the accumulator grid cell $Acc[x', y', r]$. The elements in the accumulator matrix can be interpreted as the number of object pixels in the input space that are located at r distance from (a, b).

So that

$$\mathrm{Acc}(a, b, r) = \sum_{(x,y)} \mathbf{I}(x, y) \cdot \mathrm{Ind}\left(\sqrt{(x - a)^2 + (y - b)^2} = r\right)$$

where $\mathbf{I}$ is the input image and (x, y) are edges of the input image.

After this, we can find local maxima in the accumulator matrix that are corresponding to the circle centers in the original space. Demonstration of the Hough circle detection is shown on Figure 3.8.

The result of running the circle detection algorithm on Figure 3.6 is shown on Figure 3.9.

Selection of the optimal proposal In some cases the edge detector can result in an edge map such that the maximum value of the circle detector accumulator corresponds to a false result.

To solve this problem we evaluate three ROIs with the highest accumulator values. An (x_0, y_0, r) can be a proposed circle (ROI) in the Hough domain, if it is a local maximum. For a given proposal, the evaluation steps are the following:

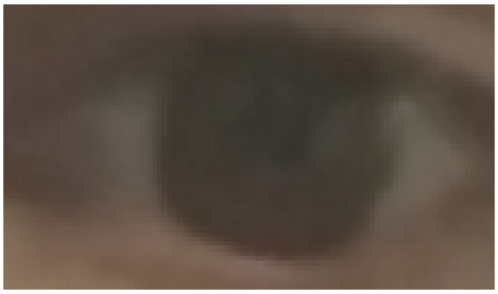
- Create a mask so that all pixels inside the contour of the circle are zeros and all other pixels are ones.
- Invert the mask and the monochromatic image of eye.
- Multiply the mask and the monochromatic image pixel-wise then sum up the values.
- Divide the result by the r^2 where r is the radius of the circle.

The method claims that we should select the circle with the highest circle value which is equal to the dark pixels per circle area. So that

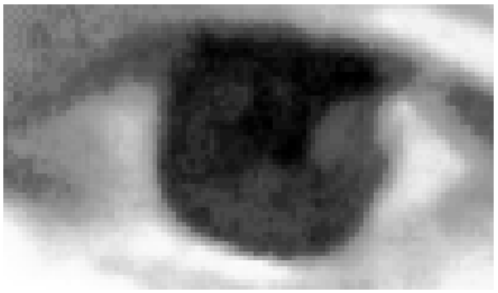
$$\mathrm{Cp}(x_0, y_0, r) = \frac{\mathrm{trace}\left((255 \cdot \mathbf{1} - \mathbf{I}_{eye})^T \cdot (\mathbf{1} - \mathbf{Mask}_{x0,y0,r})\right)}{r^2}$$

where $\mathbf{I}_{eye}$ is the patch of the monochromatic eye image, located in the position defined by the (x_0, y_0, r) proposal, $\mathbf{Mask}_{x0,y0,r}$ is the binary mask of the circle, with center in (x_0, y_0) and r radius and $\mathbf{1}$ is a matrix of ones.

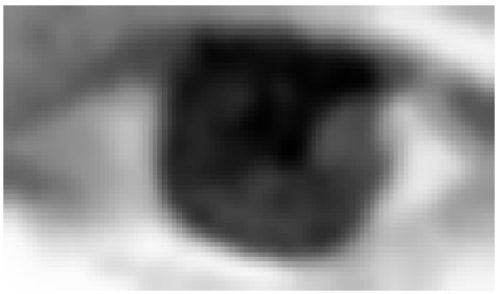
As you can see on Figure 3.10 regardless of whether the red circle has the highest value in the circle detector accumulator, the green circle has the highest circle value so it is a better result.



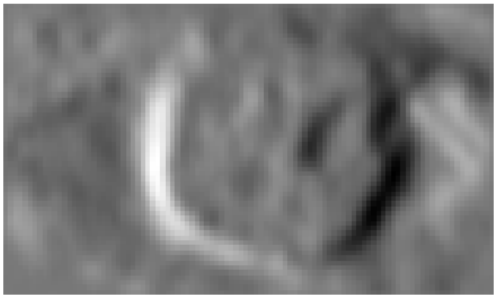
(a) Original image



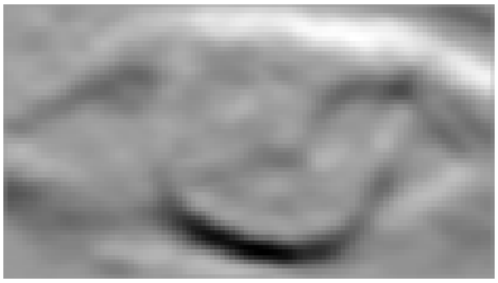
(b) Monochromatic image



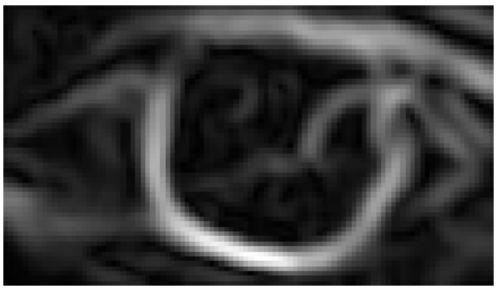
(c) Gaussian blurred image



(d) Result of vertical convolution



(e) Result of horizontal convolution



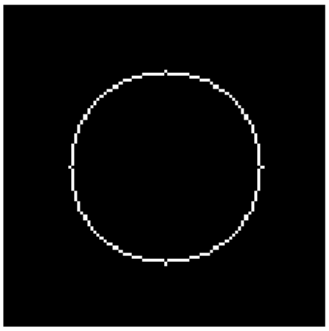
(f) Amplitude of gradients

(g) Result of thresholding

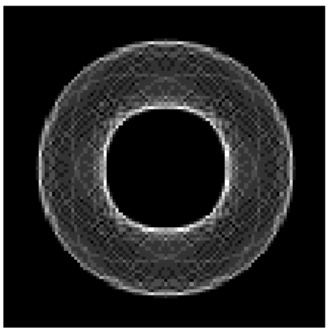
(h) Edges of the eye

Figure 3.6: Find edges on eye images.

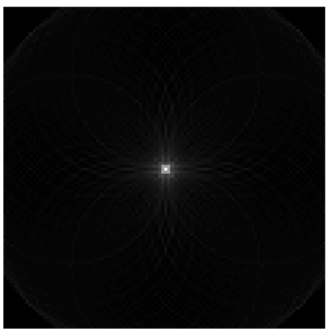
Figure 3.7: Edge detection on eye.



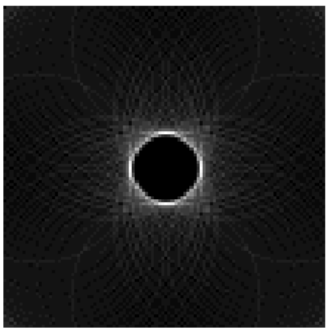
(a) Original image. $a = 50, b = 50, r = 30$



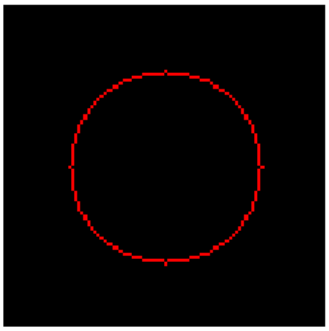
(b) Hough space for $r = 10$. Maximum value is 0.15625.



(c) Hough space for $r = 30$. Maximum value is 0.5.



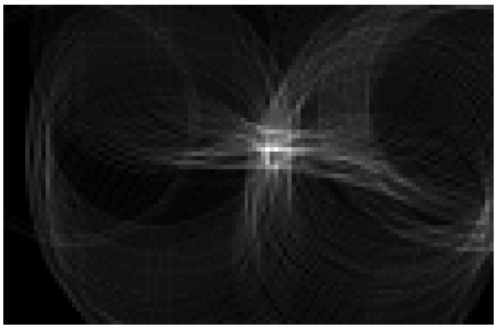
(d) Hough space for $r = 40$. Maximum value is 0.11207.



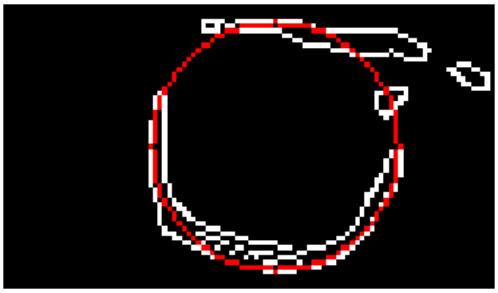
(e) Restored image. Selecting the maximum value from accumulator with $a = 50, b = 50, r = 30$.

Figure 3.8: Hough transformation of a circle.

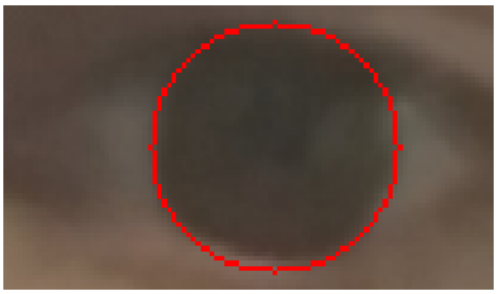
(a) Edge map



(b) Hough space of r=28

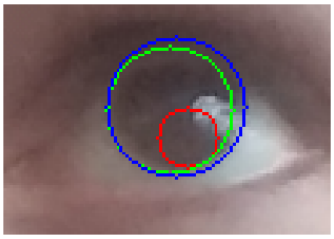


(c) Found circle with parameters of $a = 61, b = 33, r = 21$.



(d) Found circle is fully covering the pupil.

Figure 3.9: Finding circle of pupil.



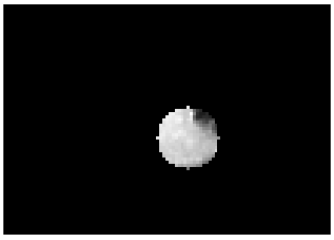
(a) Original image and the three ROI circles, ranked by the accumulator matrix in descending order: *red*, *green*, *blue*.

.

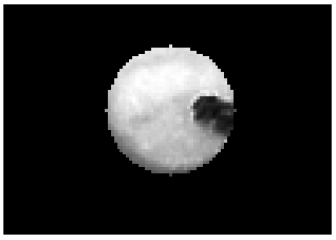
(b) Mask with parameters of $60, 43, 10$.

(c) Mask with parameters of $54, 34, 21$.

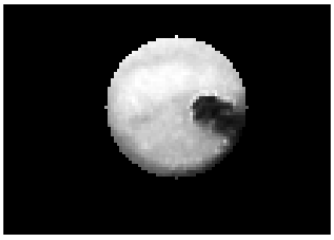
(d) Mask with parameters of $56, 33, 23$.



(e) Cp = 133951.5



(f) Cp = 147585.4



(g) Cp = 143990.6

Figure 3.10: Evaluation of circles.

3.2 Eyebrow-raising and contraction detection

In this section our algorithms for detecting eyebrow-raising and contracting are described. Videos are required as input for these methods as a change in eyebrow position is detected. In our work, these algorithms were applied on 30 fps videos. During investigating the accuracy of the landmark points on videos, and distances between several landmark points, we found, that the landmark points of the eye are accurate enough to calculate distance from. Thus first, the video is processed frame-by-frame to extract landmark points. This is the most time-consuming part of the task because from here only the time series of the position of the landmark points is processed. The next step is to extract the distances of particular points.

Most of the parameters described below are set empirically, some of them may need to be readjusted when using a different frame rate.

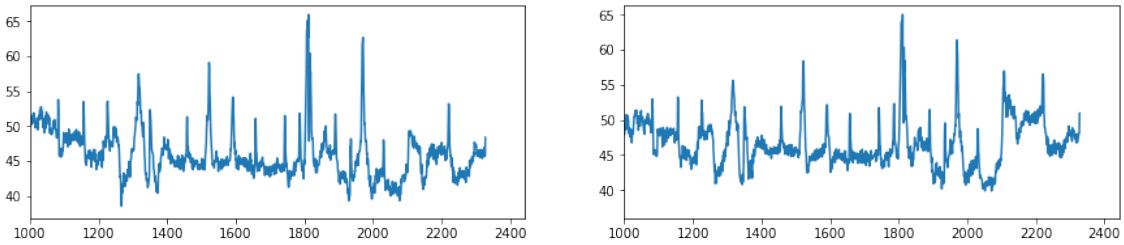
3.2.1 Median filter-based solution

This method was the first approach that we investigated. The algorithm uses the eyebrow-eye distances (x_α) as its input, more precisely the distance between the center point of the eyebrow (based on indices 20 and 25 in Figure 2.1) and the mean of the eye landmark points (based on indices of 37-40 and 43-36 for the left and the right eye respectively) as shown in Figure 3.11a. The idea is very simple: median filtering is applied using two kernels of different size. One is narrow (box_{11}) and the other one is wide (box_{61}) (see results in Figures 3.11b and 3.11c). The narrow kernel is used to smooth the noise. The wide kernel is used to smooth the eyebrow raising, and thus form a baseline to compare to. The difference of the two time series (created with the two kernels) is taken as Figure 3.11d shows. Overall it can be calculated as follows:

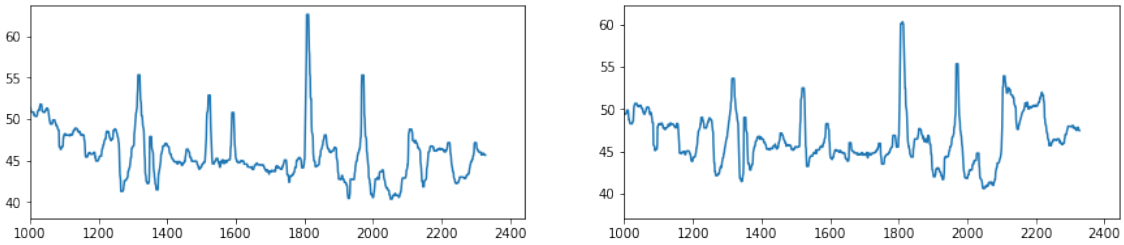
$$dx_\alpha = \text{med}\{x_\alpha, \text{box}_{11}\} - \text{med}\{x_\alpha, \text{box}_{61}\},$$

where (box_n) is the n width box kernel, x_α is the input time series, and $\text{med}\{\cdot,\cdot\}$ is the median filtering. The state of the eyebrow of one eye (see Figure 3.11e) is considered raised at a given timestamp if the difference in that timestamp is greater than an ε and the ratio of the difference time series (dx_α) to the difference time series for the other eye (dx_β) is greater than $\frac{1}{2}$.

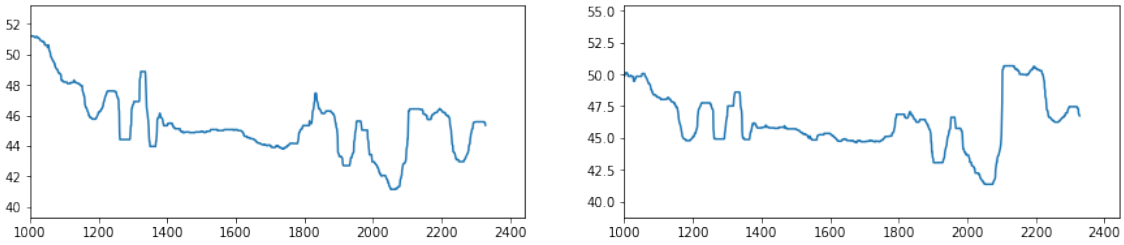
$$\left(dx_\alpha > \varepsilon\right) \wedge \left(\frac{dx_\alpha}{dx_\beta} > \frac{1}{2}\right)$$



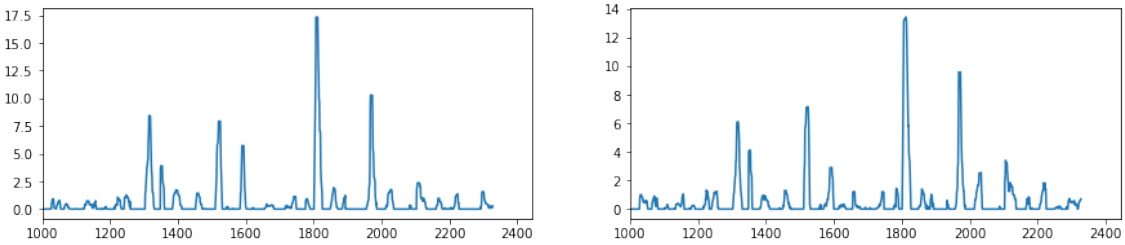
(a) Eyebrow-eye distance.



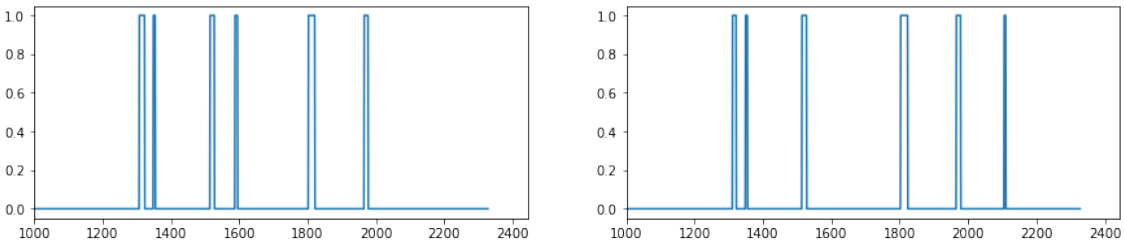
(b) Median filtering applied to the time series using the narrow kernel.



(c) Median filtering applied to the time series using the wide kernel.



(d) Difference of the two median filtered time series.



(e) State of the eyebrows (raised - 1, unraised - 0).

Figure 3.11: Steps of eyebrow raising detection with the median filter-based approach

3.2.2 Probability density-based solution

3.2.2.1 Eyebrow raising

For this method eyebrow-nose bottom and eyebrow-eye distances are investigated. Another input for this method is the output of the blink detection. In a basic scenario the distance between the mean of the eye landmark points (with indices of 37-40 and 43-36 in Figure 2.1) and the center point of the eyebrow (20 and 25) is calculated. However, we found that during a blink, all of the eye landmark points move a little bit, causing a jump in the distance. Several solutions have been investigated for this phenomenon. For example the coordinates were interpolated during a blink. However, raising eyebrows is often accompanied by blinking, and in this case that moment would be missed. Our final solution is that during a blink the distance between the tip of the nose (34) and the center point of the eyebrow (20 and 25) is taken into account.

First, the eyebrow-eye distance is processed (see Figure 3.12a). Using a 20 wide sliding window mean and variance is calculated for each of the 20 consecutive records. In the next step the relative likelihood is calculated for the next 3 points one by one using the probability density function of the normal distribution defined by the observed distances in the window. By multiplying the 3 points joint probability is calculated. Our null hypothesis was that if the records come from the same distribution, then they are independent given the window; formally:

$$p(x_{win+3}, x_{win+2}, x_{win+1} \mid x_{win}) = \prod_{i=1}^{3} p(x_{win+i} \mid x_{win})$$

The joint probability in our case can be computed as:

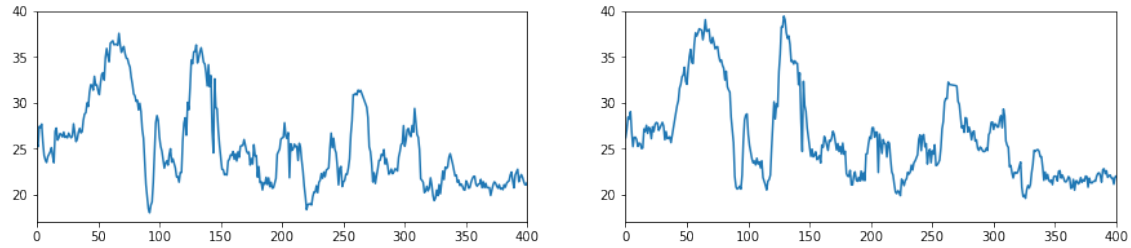
$$\prod_{i=1}^{3} p(d_{win+i} \mid win) = \prod_{i=1}^{3} \frac{1}{\sigma_{win}\sqrt{2\pi}} \cdot e^{-\frac{(d_{win+i}-\mu_{win})^2}{2\sigma_{win}^2}},$$

where d_{win+i} denotes the distance after the window with i frames; σ_{win} and μ_{win} denote the standard deviation and the mean of the window respectively. The joint probability is calculated after each window in the time series. For ease of use, we work with the logarithm of the values (see Figure 3.12b. In the next step eyebrow movements are detected. First, a binary threshold is applied. In our case values under -15 are the candidates. Using a maximum filter with a width of 5, movements close to each other are merged into one "sequence". In these sequences, local minimum search is performed to find the timestamp belonging to the most salient movement. These are points, which belong with low probability to the distribution defined by the window, thus are outliers. These points are filtered further to eliminate points belonging to the lowering and contracting of the eye. This is performed by comparing the window mean and distance value belonging to that timestamp. Until this point, time series belonging to each eye are treated separately. However, while investigating the landmark points on videos, we found, that moving the eyebrow on only one side (e.g. left) affects the landmark points on the other side (i.e. right), thus "generating movement". This movement is much smaller on the unraised side (i.e. right). Thus pairing the movements of the two sides is necessary. Detections on the two sides with a distance in time less than 0.1 s are considered the same movement. If the

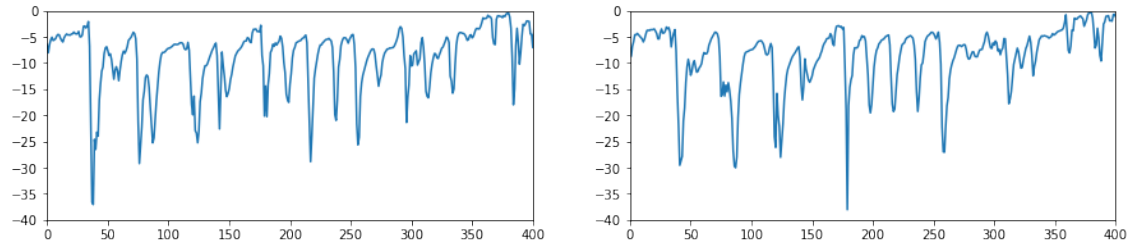
distance on one side belonging to such a movement is less than half than the other side, then it is a "generated movement", else it is a bilateral eyebrow-raising.

The majority of this process is also performed on the eyebrow-nose tip distance (see Figure 3.12d) time series. Joint probabilities are computed and are filtered using a harder threshold (-20). Filtering for only raising movement is also applied, Figure 3.14b shows the resulting detections.

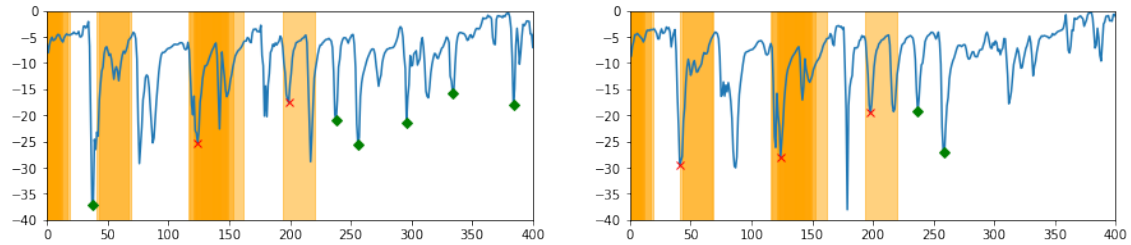
Blink timestamps resulting from the gaze detection part are used to mark possibly fake detections in a window (with a width of 0.43 s) as Figure 3.12c shows. If such a movement can also be found in the eyebrow-nose tip distance time series (with a maximal difference in time of 0.17 s), then it is considered as a real movement, else as a fake movement (see Figure 3.12f). Examples are shown in Figure 3.13.



(a) Eyebrow-eye distance.



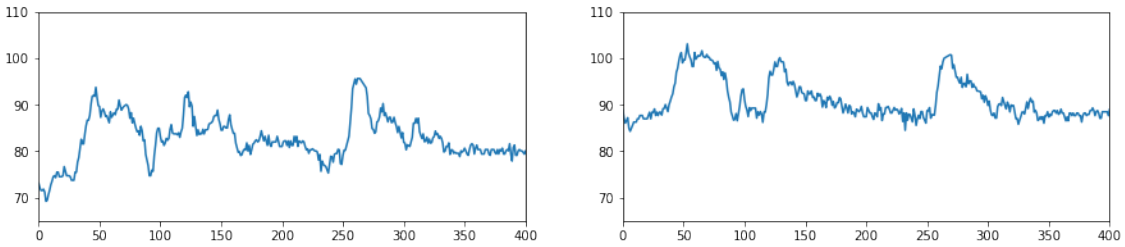
(b) Log of joint probabilities calculated based on the eyebrow-eye distance.



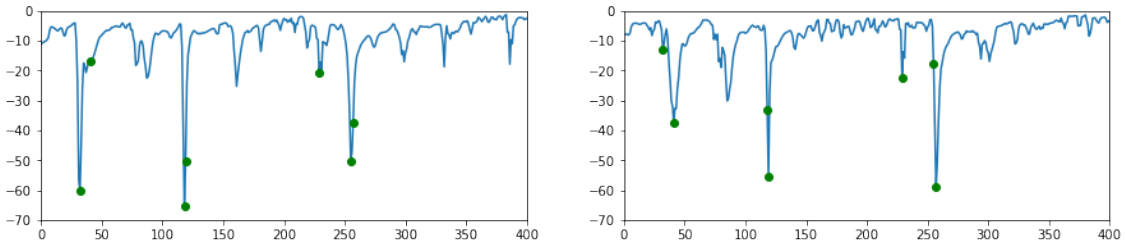
(c) Filtered detections calculated based on the eyebrow-eye distance. Orange zones denote the blinking timestamps with a window around them. Red x's mark the detections falling in the zone of blinking and green diamonds denote the rest of the detections.

3.2.2.2 Eyebrow contraction

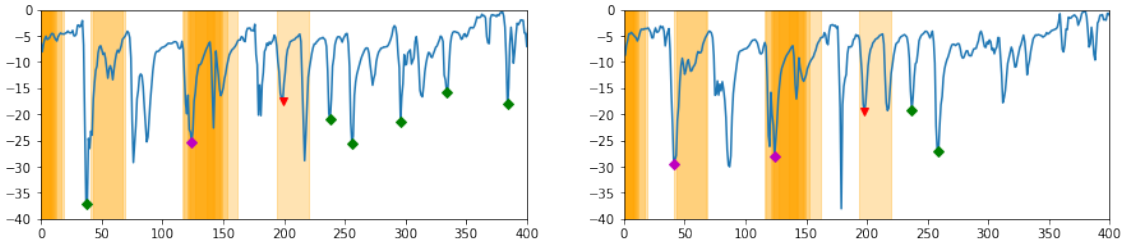
A similar method is used to detect eyebrow contraction. The input for this method is the rate of two distances as shown in Figure 3.14a. The first distance is between the inner edges of the eyebrows (with indices of 22 and 23 in Figure 2.1). The second distance is between the inner corner of the eyes (with indices of 40 and 43). We found, that these points are stably recognized by the landmark detector. The idea behind the rate is that during the contraction the two eyebrows approach each other. Joint probabilities are calculated the same way as described previously (see Figure 3.14b). As the next step, the



(d) Eyebrow-nose distance.



(e) Filtered detections calculated based on the eyebrow-nose distance.



(f) Final filtered detections calculated based on the eyebrow-eye distance. Orange zones denote the blinking timestamps with a window around them. Green diamonds mark the detections out of the blink-windows. Magenta diamonds are the detections that fall into the orange zone but can be detected based on the eye-nose distance, thus are retained. Red triangles denote the detections that fall into the orange zone and can not be detected based on the eye-nose distance thus are permanently removed.

Figure 3.12: Steps of eyebrow raising detection with the probability density-based approach

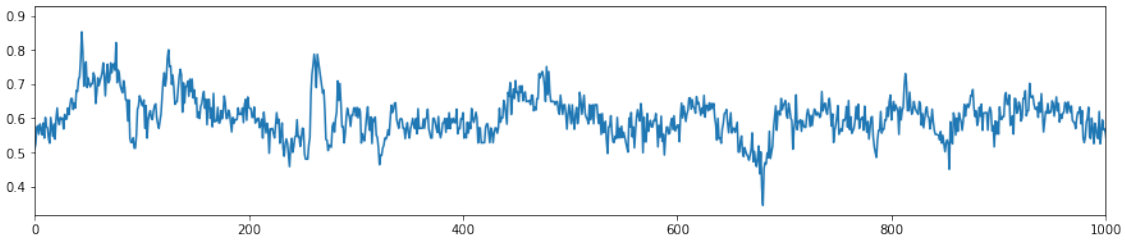
local minimas are detected and filtered using a threshold (-8). The resulting detections are further filtered to belong only to contractions. This is achieved by comparing the distance rate at that timestamp to the mean of the window before it. If its value is smaller than the mean, then it is considered as a contraction else as a raising (see Figure 3.14c).

(a) Example belonging to the first green diamond (at around 38) in Figure 3.12f. There is no blink during the eyebrow-raising.

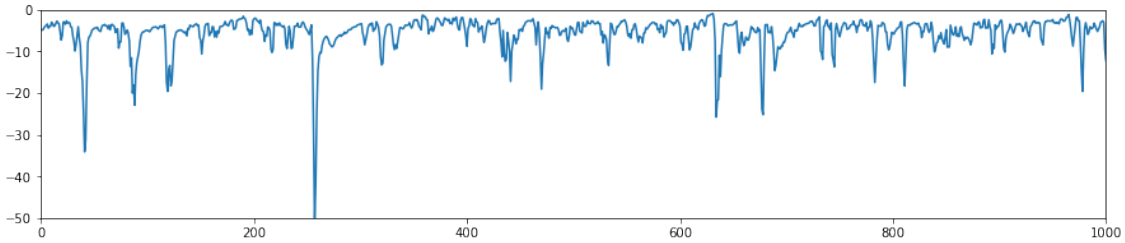
(b) Example belonging to the first purple diamond (at around 124) in Figure 3.12f. There is a blink during the eyebrow-raising, but it is also detected based on the eyebrow-nose distance.

(c) Example belonging to the first red triangle (at around 199) in Figure 3.12f. There is a blink during the eyebrow-raising, but it is not detected based on the eyebrow-nose distance thus removed from the detections.

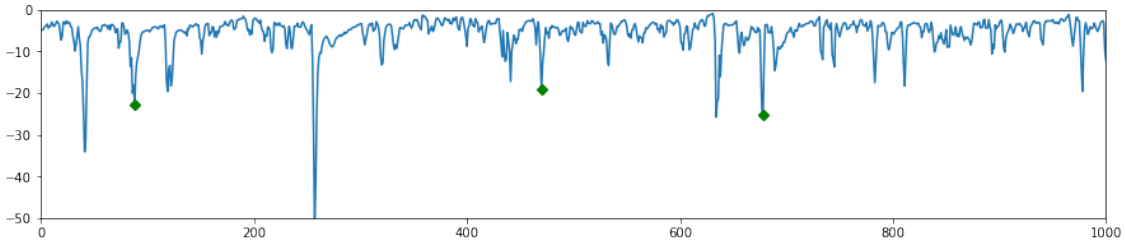
Figure 3.13: Examples belonging to the detections shown in Figure 3.12f.



(a) Rate of distances between the inner edges of the eyebrows and the inner corners of the eyes.



(b) Log of joint probabilities calculated based on the distance rate.



(c) Final filtered detections calculated based on the distance rate.

Figure 3.14: Steps of eyebrow contraction detection with the probability density-based approach

Figure 3.15: Example belonging to the first green diamond (at around 88) in Figure 3.14c.

3.3 Mouth detection

Examination of the eyes and their surroundings alone does not provide sufficient information about the mood of the subject, therefore we examined the lips and their surroundings.

In this section, we introduce procedures in order to extract information about the position and size of the mouth.

3.3.1 Openness detection

One of the basic parameters we determined was whether the subject's mouth was open. Detecting mouth openness is crucial because opened mouth can mean surprise, fear etc. Dlib provides several landmark points around the mouth, based on which we can determine this parameter.

(a) Closed mouth.

(b) Opened mouth.

Figure 3.16: Determining mouth openness.

First, we calculate the average height of the upper and lower lips. It is the mean of the length of the blue and the green lines showed on Figure 3.16. Then we compare the minimum of these two values with the average distance between lips that is the mean of the length of the red lines. Mouth is open if

$$\frac{\| \vec{r_1}\|+\| \vec{r_2}\|+\| \vec{r_3}\|}{3} > min\left(\frac{\| \vec{b_1}\|+\| \vec{b_2}\|+\| \vec{b_3}\|}{3}, \frac{\| \vec{g_1}\|+\| \vec{g_2}\|+\| \vec{g_3}\|}{3}\right) * 0.7.$$

where $\vec{r_i}$ vectors are marked by the red, $\vec{g_i}$ by the green, $\vec{b_i}$ by the blue lines on Figure 3.16.

3.3.2 Visible lip size

The next parameter we examined is the visible size of lips. For this, we fit one polygon to the upper and another one to the lower lip points defined by Dlib, and calculate their area.

Visible lip size can be used to determine whether the subject was biting his or her lower lip, or the degree of rotation along an axis perpendicular to the sagittal plane. As it can be seen in Figure 3.18 biting the lower lip decreased its visible size.

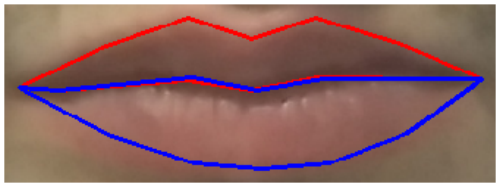


Figure 3.17: Region of upper and lower lips.

(a) $LS_{low} = 3416$ **(b)** $LS_{low} = 2431$ **(c)** $LS_{low} = 5132$

Figure 3.18: Analyzing lower lip size (LS_{low}) on different mouth states. ((a) Neutral, (b) Biting lip, (c) Heavy smiling).

3.3.3 Detecting gap between lips

One of the biggest challenges in the analysis of the mouth was to determine the shape and the curvature of the gap. We can extract a lot of information from these parameters of a closed mouth, such as whether the person is smiling.

3.3.3.1 Fitting line

As a first approach, we fitted a line to the mouth. Here, we had to decide two important things: the image preprocessing steps and the fitting algorithm. For preprocessing, two methods were investigated: edge detection and blob detection.

Edge detection, although worked in the case of the eyes, was unfortunately not effective in the case of mouth. Based on our observations, the mouth gap can be represented by a thick, dark, horizontal line even when it is fully closed, so the edge detection algorithm resulted in two, well-separable edges and a lot of false edges.

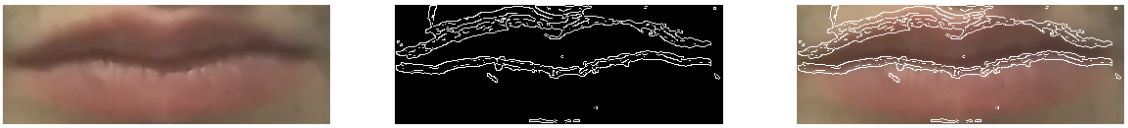


Figure 3.19: Edge detection on mouth.

The other approach is blob detection. We assume, that dark, linear blobs can approximate the gap between the lips. Because of the non-ideality of light conditions, parts of the lips are usually in shadow, while in other scenarios, it can reflect the backlight. Therefore, the gap cannot be detected continuously in most of the times, however parts of the gap can be detected as separate dark blobs. Because blobs are usually detected based on the curvature of the local intensities in image processing tasks, therefore we can utilize local Hessian based filtering (described in subsection 3.1.2.6) for this purpose too.

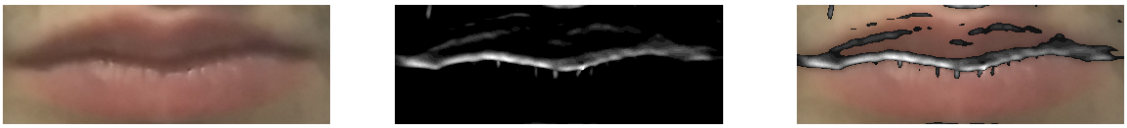
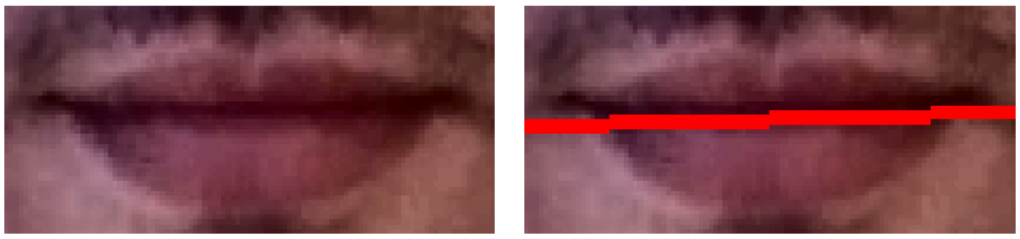


Figure 3.20: Blob detection on mouth.

By thinning the dark region found by the algorithm, we can determine the center of the cavity. We have to fit a line to this, for which we have chosen Hough transformation that was applied to the eyes as well.

Hough line The basic procedure was extended with several constraints. One of these is the angle of the line and X axis. Those lines were kept where this angle was in the $[-\frac{\pi}{8}, +\frac{\pi}{8}]$ interval. The other constraint is the distance of the middle of the line from the vertical center of the mouth bounding box. If the equation of the line is $y = m \cdot x + b$ then the middle of it is the value of $m \cdot x_{half} + b$ where x_{half} is the half of the width of the mouth bounding box. Lines were kept if this distance was no more than 10% of the height of the bounding box.



(a) On a neutral mouth the algorithm can find the main line of the mouth.

(b) Mouth with heavy curvature cannot be modelled with a straight line.

Figure 3.21: Fitting line on mouth gap.

As it can be seen in Figure 3.21 the method can determine the position of the gap on simple images, but it is not sufficient to follow its curvature.

3.3.3.2 Fitting parabola

Previous results indicated that we need a more accurate modeling of the mouth gap than a straight line. After several experiments, a quadratic curve proved to be the most efficient, ensuring the most accurate fit at a given computational cost.

The advantage of a parabola over a straight line is that it can model not only the angle with the X-axis, but also the curvature of the mouth gap. Parabolic alignment was also

performed with the aforementioned Hough-transformation, and for preprocessing we also used blob detection.

Hough parabola In two-dimensional space parabolas can be described by:

$$y = c \cdot (x - a)^2 + b.$$

For a given (x, y) point we look for all (a, b, c) values that satisfy the equation above. The basic idea of the method is that for every (x', y') point that lies on a parabola whose equation is $y = c \cdot (x - a)^2 + b$ we can define another parabola whose equation is $y = -c \cdot (x - x')^2 + y'$ so that (a, b) will lie on it.

Our three-dimensional accumulator can be computed in the following way:

$$\mathrm{Acc}(a, b, c) = \sum_{(x,y)} \mathbf{I}(x, y) \cdot \mathrm{Ind}\left(y - c \cdot (x - a)^2 = b\right)$$

where $\mathbf{I}$ is the input image and (x, y) are edges of the input image.

After the original implementation, we faced three main problems by applying the method on the binarized blob image:

- The algorithm could not handle small curvature changes near zero curvature, i.e. neutral mouth expression.
- The output of blob detection is not always robust, so in a video recording, the fitted parabola made large jumps between consecutive frames.
- The algorithm can only handle those cases where the directrix is parallel to the X-axis.

Handling small curvature changes. During the parabola search, several curvature values must be examined. Initially, a uniform scale was used in the interval (-0.04,0.04) with a step size of 0.004. By setting the step size, we face a trade-off between computation cost and the accuracy of fitting.

After several studies, we observed that fine step size was only required in the near-neutral mouth state, but not in edge cases such as heavy smiling.

As a solution, we started using a logarithmic scale, which significantly improved the accuracy of the parabolic fit.

(a) Fitting parabola with linear scale. $\alpha = 0$.

(b) Fitting parabola with logarithmic scale $\alpha = 001587$.

Figure 3.22: With logarithmic scale we can distinguish small curvature changes.

Handling noise. Due to the ever-changing head position and light conditions in the example video, the blob detection output had to be preprocessed for a more accurate parabolic fit. The following steps were performed.

- Thresholding: The values of the blob detection output above the 85th percentile according to the output values are preserved. Based on this criteria we can create a binary mask.
- Filtering mask region: The size of the mask is narrowed to the middle area of the mouth, discarding false values at the edge of the image (e.g., pixels of mustache).
- Finding dominant components: In each column of the mask, we find the dominant component. Dominant components are the longest related values of 1 in a given column.
- Finding middlepoints: In the next step, the middlepoint is determined for each column, which is the weighted average of the coordinates of the pixels of the dominant component according to the intensities of the original image.
- Filtering middlepoints: There are cases where some outliers appear in the resulting output. These are eliminated using a median filter.
- Drawing polygon: We closed the curve of the middlepoints by piecewise linear interpolation, on which we can run our parabolic fitting algorithm.

These steps can be formalized in the following way:

$$\mathbf{BE} = \mathrm{Ind}\left(\mathbf{E}_{blob} > \mathrm{T}_{85}(\mathbf{E}_{blob}) \cdot \mathbf{Mask}_{corners}\right)$$

$$\begin{aligned}
&\mathbf{DC}(x) = [a, b] \Leftrightarrow \\
&\Leftrightarrow \Big(\forall d \in [a, b] : \mathbf{BE}(x, d) = 1\Big) \wedge \\
&\wedge \neg\exists(a', b') : \Big(\Big((b' - a') > (b - a)\Big) \wedge \Big(\forall d \in [a', b'] : \mathbf{BE}(x, d) = 1\Big)\Big)
\end{aligned}$$

where $\forall$ is the universal, $\exists$ is the existential quantifier, $\neg$ is the operator of logical negation, while $\wedge$ is the logical and.

$$\mathbf{mp}(x) = \text{median}\left\{ \frac{\sum_{y \in DC(x)} y \cdot \mathbf{BE}(x,y) \cdot \mathbf{E}_{blob}(x,y)}{\sum_{y \in DC(x)} \mathbf{BE}(x,y) \cdot \mathbf{E}_{blob}(x,y)}, \mathbf{box_n} \right\}$$

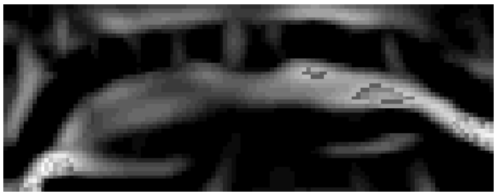
where median$(\cdot)$ is the median filtering and $\mathbf{box_n}$ is the n width box kernel.

The result of these steps are shown on Figure 3.23

Handling not horizontal directrix. As we described before, we chose a simpler algorithm to find parabolas in order to reduce computational cost. We only look for parabolas whose directrixes are parallel with X-axis (see Figure 3.24).

To solve this problem we apply a line fitting algorithm to find the main line of the mouth. We apply the line fitting algorithm after the preprocessing steps mentioned in the previous point. After we found the main line, we rotated the image until the line became parallel with X-axis. We used zero padding and zero order (nearest neighbour) interpolation for the rotation. To this rotated image we applied our parabola fitting algorithm then we rotated back the image (see Figure 3.25).

(a) Original image



(b) Output of blob detection

(c) Thresholding

(d) Filtering mask region

(e) Finding dominant components

(f) Finding middlepoints

(g) Filtering middlepoints

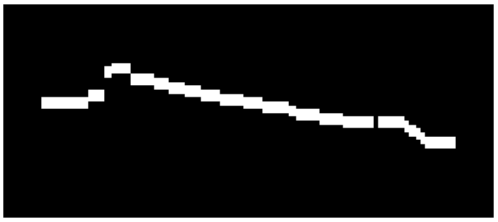
(h) Finding parabola

(i) Result with preprocessing

(j) Result without preprocessing

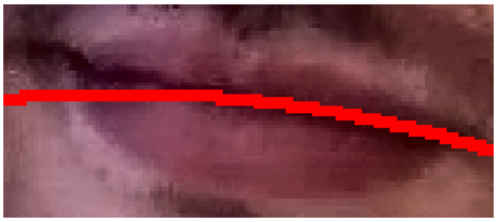
Figure 3.23: Preprocessing steps on mouth images.

(a) Original image



(b) Preprocessed image

(c) Fitted parabola

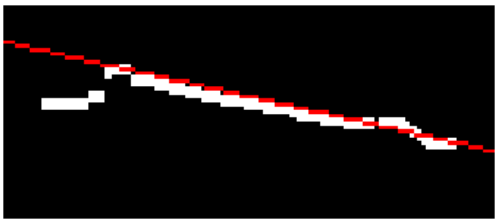


(d) Result image

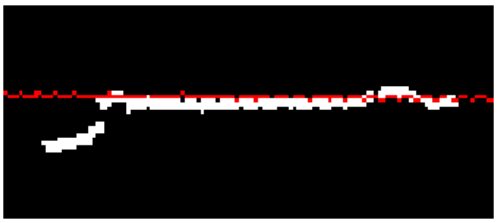
Figure 3.24: Fitting parabola without rotation.

(a) Original image

(b) Preprocessed image

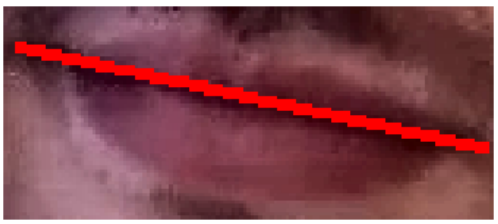


(c) Main line of preprocessed image



(d) Rotated preprocessed image

(e) Fitted parabola (with 0 curvature



(f) Result image

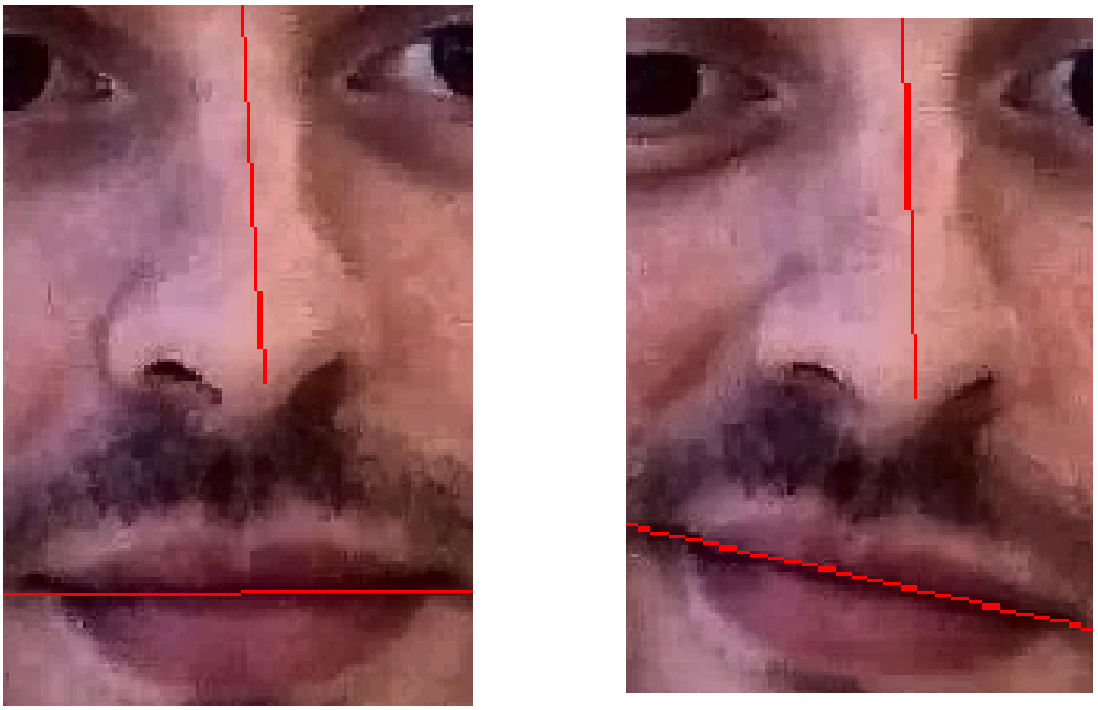
Figure 3.25: Fitting parabola with rotation.

3.3.4 Mouth angle

Half-sided mouth is a typical sign of contempt, the detection of which can be useful.

Unfortunately, we cannot compare the angle of the mouth with the X-axis because different head positions may exist in a video. It should be examined relative to the face, so we decided to compare it with the central line of the nose, because their relative positions are rotation invariant. To find the main line of the mouth we reused the line described in the *Handling not horizontal directrix* paragraph. The nose line is defined by linear interpolation of the landmarks of the nose then applying a line fitting algorithm on it.

After these steps we simply compare the angle between the mouth and the nose and if that angle is less than 80 degree we report a half-sided mouth parameter, see Figure 3.26.



(a) Neutral face $\alpha = 87.8°$

(b) Half-sided mouth $\alpha = 76.5°$

Figure 3.26: Determining angle between nose and mouth line.

Chapter 4

Results and discussion

In this report we examined several different algorithms that allowed to extract information about facial expressions from images. The algorithms were analyzed in the following test images and compared with neural network-based solutions.

Figure	Eyes	Eyebrows	Mouth openness	Mouth curve	Mouth line degree	Facs Aus	Facs result
a)	Open	Not raised	Closed	0.0037 (Heavy downward)	80.189	15	sadness
b)	Open	Not raised	Closed	-0.0007 (Slightly upward)	87.606	-	neutral
c)	Open	Left raised	Closed	0 (Not curved)	76.497(*)	R14	contempt
d)	Open	Not raised	Closed	-0.0031 (Heavy upward)	86.629	12	happiness
e)	Open	Not raised	Open	-0.0021 (Heavy upward)	82.052	12,25	happiness
f)	Open	Raised	Open	0 (Not curved)	89.153	1,2;26	surprise

Table 4.1: Evaluation of different kind of faces shown on Figure 4.1. (*) denotes a half-sided mouth

As Figure 4.1 shows, algorithms are able to detect basic features on the sample images. The parabola fitted to the mouth shows that the detection of curvature and angle of the mouth are quite accurate. Also, the boundary of the iris is predicted to its real position in most of the cases. Based on the predictions, the action units can be detected thus the microexpressions can be determined as Figure 4.1 shows.

We also examined the test images with a popular, publicly available neural network-based solution[1]. The results are shown in Figure 4.2. According to the documentation, the neural network was trained to distinguish angry, disgusted, fearful, happy, neutral, sad, surprised classes. As can be seen, this neural network is capable of capturing some expressions from the images. However, due to the inaccuracy of the landmark detector, key features on the face are sometimes missed, e.g. the shape of the mouth is not accurate in Figure 4.2a and the curvature of the eyebrow in Figure 4.2c.

Figures 4.1a and 4.2a display a "sad" expression characterized by a closed mouth with a heavy downward curve. The expert system based approach correctly identified the downward curve and the closed mouth. The neural network based method classified this image mainly as a "neutral" expression with a moderately high probability of 0.75, and as a "sad" expression with a lower probability of 0.23. This is possibly due to the fact that the landmark points of the mouth, particularly the edges, were not properly aligned.

Figures 4.1b and 4.2b display a "neutral" expression with no particular features. Both methods identified this expression correctly.

[1] `https://github.com/justadudewhohacks/face-api.js/`

In Figures 4.1c and 4.2c a "contempt" expression can be seen. The characteristic feature of this expression is asymmetry, which is manifested in a closed, half-side upward elevated mouth line and the eyebrows raised on one side. The raised eyebrow (on the left side) is detected by the expert system, besides the half-side elevated mouth line indicated by a decreased mouth angle (74.497°) compared to the ideal 90°) between the mouth line and the nose line. The neural network based solution classified this expression as "happy", which is acceptable as "contempt" expression was not among the trained classes.

Figures 4.1d and 4.2d show a smiling face, open eyes and unraised eyebrows. The upward curve of the mouth is followed accurately by the fitted parabola. The "happy" emotion is correctly predicted by both approaches.

Figures 4.1e and 4.2e show an expression with slightly open, upward-curving mouth and unraised eyebrows. This means "happiness" based on the action units. These features were found by our expert system. Also it was predicted correctly by the neural network.

On Figure 4.1f an emotion with open mouth and raised eyebrows is visible. This can be interpreted as "surprise", which follows from the features detected by our methods. However, the prediction 4.2f of the neural network based solution is "neutral" with a probability of 0.98. Based on the demos shown on the page of the API, this may be the result of the training on images showing exaggerated emotions.

The results show, that classic image processing algorithms and expert systems can compete with neural network-based solutions, because they can achieve high accuracy without the requirement of large training dataset. Furthermore, their output can be explained, because they do not behave as a black box model. Please also note, that there are many tunable parameters in a neural network based solution (e.g. the architecture of the network, the training dataset, the utilized augmentation and regularization methods, etc.), therefore, there may exist a more accurate neural network based solution for this task, but the lack of interpretation still degrades the usability.

4.1 Future work

As can be seen from the investigated methods described in this paper, the examination of the action units is not yet exhaustive. There are even more key features on the face worth detecting since they can carry important information about the emotions of a person thus making the prediction more accurate. Based on the action units related to the basic emotions, wrinkles play an important role: crow's feet lines next to the eye, horizontal and vertical wrinkles on the forehead, the nasolabial furrow, wrinkles between the eyebrows, at the root of the nose and next to the mouth are key indicators of particular emotions. A further development aspect could be the redundant detection of some motions, e.g. the motion of eyebrows, facial muscles near the mouth, etc. This would allow a more robust detection. Currently, only the contraction and raising of the eyebrow are detected using multiple frames. However, this could be used for several other features, e.g. by tracking the changes of features across multiple frames. In such a case when these methods would be used for the detection of microexpressions based on recordings, made by a non-stationary device, e.g. a handheld phone, further steps are necessary to allow adequate detection, such as image stabilization. Furthermore, other machine learning based algorithms could be integrated. The landmark detector played a fundamental role in our solution, as it provided relevant reference points to build upon. If this method would be enhanced with additional features or other similar methods would be developed, their integration would be a viable choice.

(a) Mouth curving downward.

(b) Neutral face.

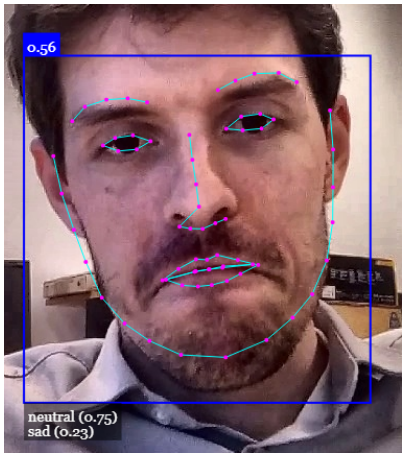
(c) Half-sided mouth.

(d) Smiling.

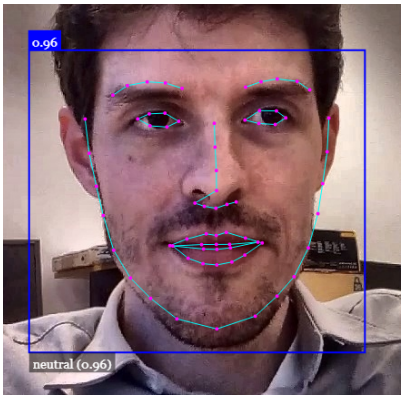
(e) Smile with open mouth.

(f) Open mouth with the eyebrows raised.

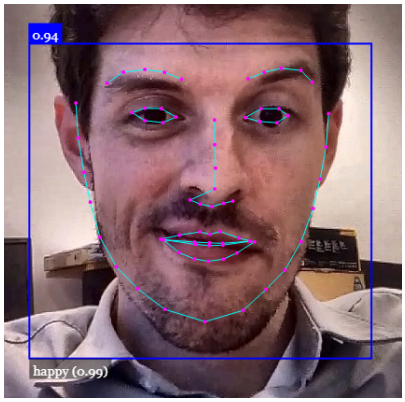
Figure 4.1: Results on different kind of faces using our solution.



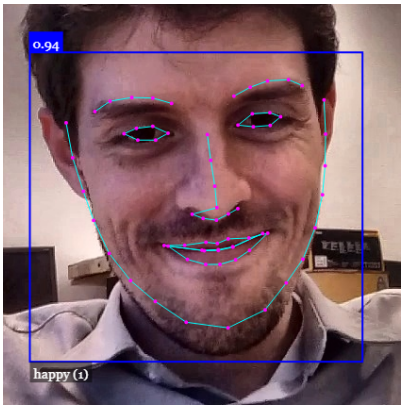
(a) neutral(0.75), sad(0.23)



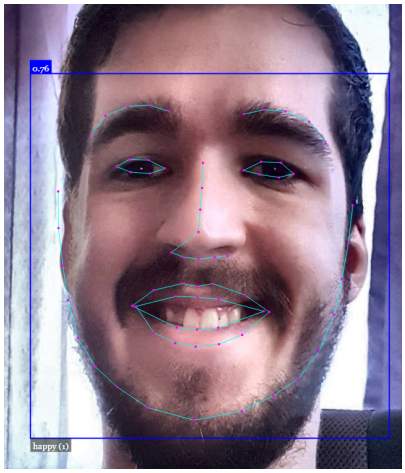
(b) neutral(0.96)



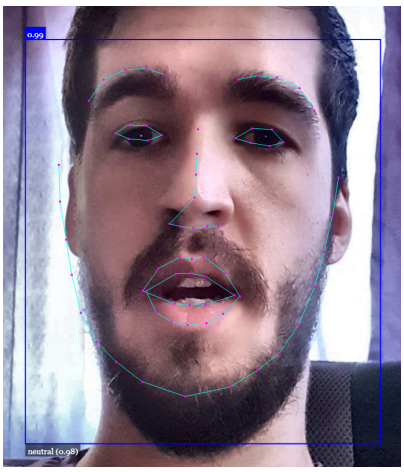
(c) happy(0.99)



(d) happy(1)



(e) happy(1)



(f) neutral(0.98)

Figure 4.2: Results on different kind of faces using the neural network based solution.

Acknowledgements

We would like to thank the support of Gábor Hullám and Dániel Hadházi.

The research of Gábor Hullám was supported by the ÚNKP-20-5 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund, and the János Bolyai research scholarship.

This research has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.2-16-2017-00013).

Bibliography

[1] Sarah Adel Bargal, Emad Barsoum, Cristian Canton Ferrer, and Cha Zhang. Emotion recognition in the wild from videos using images. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pages 433–436, 2016.

[2] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[3] Ran Breuer and Ron Kimmel. A deep learning perspective on the origin of facial expressions. *arXiv*, pages arXiv–1705, 2017.

[4] Pierre-Luc Carrier, Aaron Courville, Ian J Goodfellow, Medhi Mirza, and Yoshua Bengio. Fer-2013 face database. *Universit de Montral*, 2013.

[5] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active appearance models. *IEEE Transactions on pattern analysis and machine intelligence*, 23(6):681–685, 2001.

[6] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[8] S Du and AM Martinez. Compound facial expressions of emotion: from basic research to clinical applications. *Dialogues in clinical neuroscience*, 17(4):443, 2015.

[9] Paul Ekman, Joseph C. Hager, and Wallace V. Friesen. *Facial action coding system: the manual*. Research Nexus, 2002.

[10] Rosenberg Ekman. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, USA, 1997.

[11] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[12] Deepak Ghimire, Sunghwan Jeong, Joonwhoan Lee, and San Hyun Park. Facial expression recognition based on local region specific features and support vector machines. *Multimedia Tools and Applications*, 76(6):7803–7821, 2017.

[13] Peter E Hart and RO Duda. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[14] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2):179–187, 1962.

[15] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 06 2014. DOI: 10.13140/2.1.1212.2243.

[16] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[17] Ya Li, Jianhua Tao, Linlin Chao, Wei Bao, and Yazhu Liu. Cheavd: a chinese natural emotional audio–visual database. *Journal of Ambient Intelligence and Humanized Computing*, 8(6):913–924, 2017.

[18] Tony Lindeberg. Scale-space theory in computer vision, 1993.

[19] Patrick Lucey, Jeffrey F Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar, and Iain Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *2010 ieee computer society conference on computer vision and pattern recognition-workshops*, pages 94–101. IEEE, 2010.

[20] Robert K McConnell. Method of and apparatus for pattern recognition, January 28 1986. US Patent 4,567,610.

[21] RV Mises and Hilda Pollaczek-Geiringer. Praktische verfahren der gleichungsauflösung. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 9(1):58–77, 1929.

[22] Sébastien Ouellet. Real-time emotion recognition for gaming using deep convolutional network features. *arXiv*, pages arXiv–1408, 2014.

[23] Maja Pantic and Léon JM Rothkrantz. An expert system for multiple emotional classification of facial expressions. In *Proceedings 11th International Conference on Tools with Artificial Intelligence*, pages 113–120. IEEE, 1999.

[24] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *Proceedings of the British Machine Vision Conference (BMVC)*. British Machine Vision Association, 2015.

[25] William K. Pratt and James E. Adams Jr. *Digital Image Processing, 4th Edition. J. Electronic Imaging*, 16(2):029901, 2007. DOI: 10.1117/1.2744044. URL https://doi.org/10.1117/1.2744044.

[26] Christos Sagonas, Epameinondas Antonakos, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 300 faces in-the-wild challenge: Database and results. *Image and vision computing*, 47:3–18, 2016.

[27] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[28] Caifeng Shan, Shaogang Gong, and Peter W McOwan. Facial expression recognition based on local binary patterns: A comprehensive study. *Image and vision Computing*, 27(6):803–816, 2009.

[29] J Springenberg, Alexey Dosovitskiy, Thomas Brox, and M Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*, 2015.

[30] Gonçalo Tavares, André Mourão, and João Magalhães. Crowdsourcing facial expressions for affective-interaction. *Computer Vision and Image Understanding*, 147(C): 102–113, 2016.

[31] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of graphics tools*, 9(1):23–34, 2004.

[32] Roberto Valenti and Theo Gevers. Accurate eye center location and tracking using isophote curvature. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[33] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.

[34] Joseph N Wilson and Gerhard X Ritter. *Handbook of computer vision algorithms in image algebra*. CRC press, 2000.

[35] JC Wojdeł and LJM Rothkrantz. Mixed fuzzy-system and artificial neural network approach to the automated recognition of mouth expressions. In *International Conference on Artificial Neural Networks*, pages 833–838. Springer, 1998.

[36] Erroll Wood and Andreas Bulling. Eyetab: Model-based gaze estimation on unmodified tablet computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 207–210, 2014.

[37] Jingwei Yan, Wenming Zheng, Zhen Cui, Chuangao Tang, Tong Zhang, and Yuan Zong. Multi-cue fusion for emotion recognition in the wild. *Neurocomputing*, 309: 27–35, 2018.

[38] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[39] Nianyin Zeng, Hong Zhang, Baoye Song, Weibo Liu, Yurong Li, and Abdullah M Dobaie. Facial expression recognition via learning deep sparse autoencoders. *Neurocomputing*, 273:643–649, 2018.