Point Based Emotion Classification Using SVM

Wout Swinkels

Institute of VLSI Design

Zhejiang University

Thesis advisor: Prof. Shen Haibin

A thesis for the partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering at Hasselt University

2015-2016

Acknowledgements

When I arrived in China I knew that it would be an experience that I will carry with me for the rest of my life. After 4 months I can confirm this, during my thesis I had the opportunity to work in an international research team in one of the best universities in China and the entire world. Before I came to China I did not know anything about the area of computer vision but thanks to the researchers at the Zhejiang University I gained a better understanding of this area. It took me a lot of effort to get used to the algorithms, concepts and terminology of this research area. But in the end, I can look back to it as the best period of my study with a development as researcher and human.

I would like to thank some people in special for supporting me during my thesis and for making it possible that I had this wonderful experience. First I would like to thank professor Haibin Shen for giving me the opportunity to work on my thesis at the Zhejiang University. Without his experience and knowledge it would not be possible for me to study at the Zhejiang University in the first place.

I also would like to thank the people in the research team of prof. Haibin Shen, especially Xiao Feng and Gary for making it possible for me to do my research in their lab. Without their experience and knowledge I would not have achieved the same results. Their support and advice regarding my thesis was very helpful just like the discussions about the direction of my research. But they did not help me only in the lab, they also introduced me to the Chinese culture what I really appreciated.

I want to thank professor Luc Claesen for his encouragement to study at the Zhejiang University, for the support and advice during my journey of living in China and writing my thesis.

Of course I also want to thank my family and friends for their support and interest in my research during my stay in China. In special I would like to thank my parents who supported me during my study and encouraged me to go abroad.

Wout Swinkels June 2016

Table of Contents

List of tables

List of figures

Abstract

The detection of emotions is a hot topic in the area of computer vision. Emotions are based on subtle changes in the face that are intuitively detected and interpreted by humans. Detecting these subtle changes, based on mathematical models, is a great challenge in the area of computer vision.

In this thesis a new method is proposed to achieve state-of-the-art emotion detection performance. This method is based on facial feature points to monitor subtle changes in the face. Therefore the change in distance between the feature points are extracted. This data is used by a cascade of a multi-class support vector machine, trained to detect all the emotions, and a support vector machine, trained on specific emotions, for binary classification.

This method is implemented in a real-time emotion detection application, with a processing time of less than 30 ms, to show the ability of detecting emotions in real-time.

The Extended Cohn-Kanade (CK+) dataset is used to evaluate the proposed method. The evaluation on this dataset shows that the proposed method has an average accuracy of 90% for detecting 7 basic emotions with outliers for the emotions “contempt” and “surprise”. Comparing these results to state-of-the-art algorithms indicates that the proposed method has state-of-the-art accuracies and even outperforms some state-of-the-art algorithms regarding the detection and classification speed of emotions.

Chapter 1

Introduction

In this chapter a short introduction in the area of computer vision will be given. Then some of the applications of computer vision are discussed. One of these applications is emotion detection which will be the focus of this thesis. Then the shortcomings of the current emotion detection algorithms are discussed. The objectives of this thesis are then formulated based on the problem statement and to the end of this chapter the materials and methods to achieve these objectives are introduced.

1.1 Computer vision

Computer vision is described by Reinhard Klette (2014, p. vii) in his book Concise Computer Vision as: “Computer Vision aims at using cameras for analysing or understanding scenes in the real world.” So computer vision is the discipline of analysing images to derive information about the scenes displayed on the images.

Given the broad description above it is not a surprise that computer vision has a wide area of applications. Some of these applications are:

- 3D animation (e.g. avatar of human face)
- Facial motion (e.g. person identification)
- Driver-assistance in cars (e.g. interpreting road signs)
- Fruit industry (e.g. detecting bad fruit)
- …

In this thesis the focus will be on the facial motion, this area can include person identification, emotion recognition,… But what these applications have in common is that they exist out of three different steps: face detection, feature point extraction and interpreting of the extracted information.

The detection of the face in an image is the first step in the area of facial recognition. So far there is already done a lot of research to achieve some robust algorithm to detect faces in an image. One of these algorithms is the Histogram of Oriented Gradients (HOG) algorithm introduced in 2005 by Navneet Dalal and Bill Triggs [1]. This algorithm will also be used in this thesis to extract the facial region from the images.

The second step is the extraction of feature points from the face area detected by the face detection algorithm. At the moment there is already done some research to extract the feature points from human faces [2, 3, 4, 5, 6]. In this thesis the approach described in [6] is used to extract the feature points.

The final step is to use the information provided by the feature points to check for example the identity of a person, to determine which emotion the person shows, … For this step the displacement of the feature points is calculated and used as the input to support vector machines. This approach is based on the method described in [7].

The area of emotion detection has a wide range of applications, it can, for example, be used for robots to detect the emotions of humans. This can be very useful for robots in the area of health care who can detect if a person is scared or hurt and inform a nurse or doctor. On the other hand emotion detection can also be used as an additional layer of security in visual security systems. It can, for example, be added to person identification by making the user smile while trying to get access. These are only a few applications of emotion detection.

1.2 Objectives

In this thesis the focus lies on facial motion more specific on detecting facial feature point movement, so measuring the distance that the feature points cover through the time and using this information to classify the emotion from a person. To achieve this goal it is necessary to accomplish the following individual objectives.

1. The way of finding facial feature points in an image can be done manual, semi-automatic or automatic. In this thesis the goal is to achieve this automatic. So using an image as input and extract the feature points automatically without additional human intervention. The feature points that will be tried to extract are depicted in Figure 1.

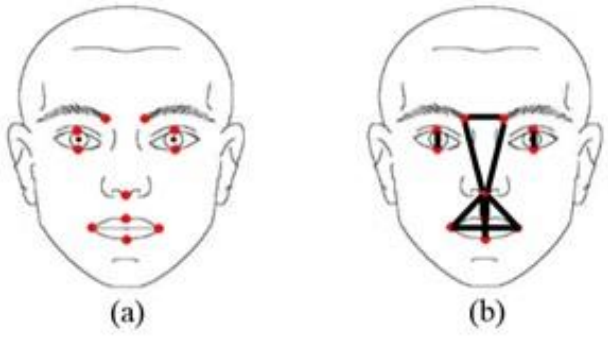

Figure 1: Eleven facial feature points of interest [7].

2. After the eleven feature points are extracted from the image the next step is to obtain the x and y coordinates of the feature points. When these coordinates are known it is also possible to calculate the distance between the feature points.
3. Once the distance between the feature points are calculated the feature points movement between two successive frames of the same person can be obtained.
4. The last objective is to achieve this procedure in real-time, so measuring the facial feature points movement in real-time. When the word real-time is used it means that the distance computation needs to be achieved in less than 33 ms. So every frame needs to be processed in less than 33 ms not only the first frame.

1.3 Material and methods

To achieve the objectives discussed in the previous section there are several libraries/toolboxes available. Some of them are:

- Computer Vision System Toolbox (Matlab)
- Open Source Computer Vision (OpenCV)
- Caffe (UC Berkeley)
- Dlib

In this thesis the OpenCV library combined with Microsoft Visual Studio 2013 will be used to implement the solution of the above mentioned objectives together with the Dlib library [8]. The Dlib library is an open-source cross platform library. This library is used for the specific tasks of face detection and the implementation of the algorithm described in [6] to obtain real-time performance. The program itself will be written in the language C++ supported by Microsoft Visual Studio.

The reason why the OpenCV library is chosen instead of the other libraries is that the research group of professor Shen has more experience with this library.

The hardware used for the development and implementation of this solution is a laptop with the following specifications:

- Operating system: Windows 8.1
- Processor: Intel Core i7-5500U CPU (2.40 GHz)
- RAM: 8,00 GB

1.4 Outline

Chapter 1: Introduction

The first chapter gives a short introduction about the research area and applications of computer vision. Then the focus will be on the application of emotion detection. The different step in emotion detection will be discussed briefly and the proposed methods for every step will be introduced.

Chapter 2: Theory

In this chapter the basic knowledge to understand the implementation of the emotion detection application will be explained. First of all the Extended Cohn-Kanade (CK+) dataset will be introduced and the corresponding evaluation method, the confusion matrix. Then the theory behind face detection algorithms will be introduced. Two specific algorithms are discussed: the Histogram of Oriented Gradients algorithm and the Viola-Jones algorithm. In the next section the theory behind the Ensemble of Regression Trees, to extract the facial feature points, will be introduced briefly. To the end of this chapter the basic theory behind Support Vector Machines will be introduced.

Chapter 3: Implementation

After introducing the theoretical background it is time to look at the implementation details of every algorithm. These details include the operational speed and parameter settings . At the end of this chapter the implementation of these algorithms are depicted in a flow chart to explain the work routine in more detail.

Chapter 4: Results

For evaluating the proposed method there are two aspects that will be discussed. The first one is the accuracy of the proposed method. The accuracy of different implementations are compared and based on these results a final implementation, with the highest average accuracy, will be chosen. This final implementation will then be compared to two state-of-the-art algorithms for face detection both on accuracy and speed.

Chapter 5: Conclusion

In this chapter an overview of the implementation and results are discussed. To the end of this chapter an overview of further improvements is proposed.

Chapter 2

Theory

In this chapter the basic knowledge to understand the implementation of the emotion detection method will be explained. First there is a short introduction about the datasets that can be used for the evaluation of emotion classifying algorithms followed by a description of the Extended Cohn-Kanade Dataset (CK+) [9, 10] used for the evaluation of the proposed method in this thesis. Then it is time to take a closer look at the algorithms used for face detection, feature point extraction and emotion classification.

2.1 Dataset

A dataset is a collection of data where the data can be represented as numbers, images, words, … For the evaluation of the proposed emotion detection method the dataset needs to contain images of different kinds of emotions where every image needs to be labelled with a specific emotion. At the moment there are different public databases available that can be used for this, some of them are:

- Extended Cohn-Kanade Dataset (CK+), University of Pittsburgh [9, 10]
- Jaffe, Kyushu University [11]
- MMI Facial Expression Database, Delft University of Technology [12]
- …

A dataset is necessary to measure the accuracy of the emotion detection method proposed in this thesis, but also to be able to compare the results obtained with other research in the area of emotion detection. So a database creates a uniform tool that can be used by researchers to compare their results. In this thesis the CK+ database is used for evaluation. The reason is that it is one of the most publically available databases used in the research area of emotion classification and thus allows us to compare the emotion classifying algorithm in this thesis with the already published algorithms.

2.1.1 Extended Cohn-Kanade Dataset (CK+)

The CK+ dataset is an extension of the Cohn-Kanade Dataset (CK) which was introduced in 2000 [10]. The CK+ dataset has 22% more sequences and also the number of subjects is increased by 27% compared to the CK dataset. This brings the total number of sequences to 593 in which every sequence can vary from 10 to 60 frames. The sequences are recorded among 123 different subjects and every sequence starts at the onset, also known as the neutral frame and ends with the frame where the

facial expressions reaches its maximum value also called the peak frame. This is depicted in Figure 2

Figure 2: Neutral frame and frame with peak value from the CK+ dataset (©Jeffrey Cohn).

The CK+ dataset is labelled with 7 different emotions (ie. anger, contempt, disgust, fear, happy, sadness and surprise). Some example images from the CK+ dataset are shown Figure 3

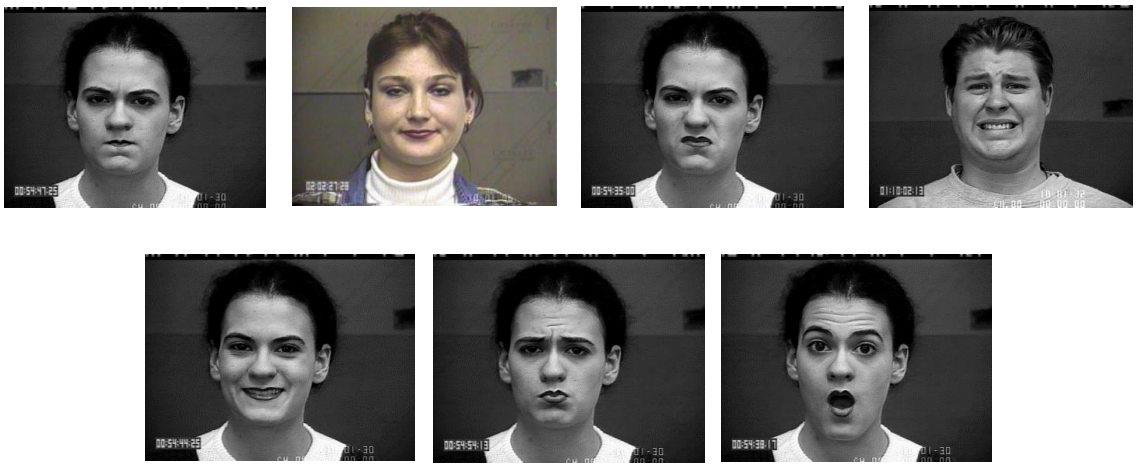

Figure 3: Example images from the CK+ dataset expressing the 7 basic emotions (©Jeffrey Cohn).

But only 327 sequences of the 593 sequences can be classified as one of the emotions mentioned above. From these 327 sequences is then only the peak frame labelled with an emotion. So this means that there are 327 frames that can be used to detect the 7 different emotions. Table 1 gives an overview of the partition of the 327 sequences among the 7 different emotions.

Table 1: Partitioning of the 7 different emotions among the 327 sequences.

Emotion	Number of sequences
Anger	45
Contempt	18
Disgust	59
Fear	25
Happy	69
Sadness	28
Surprise	83

However using the same database does not automatically mean that it is possible to compare the obtained results in this thesis with the results from other research. Therefore it is also necessary to use the same evaluation method.

2.1.2 Evaluation methods

In this thesis a confusion matrix is used to evaluate the classification of emotions. For the concept of the confusion matrix an binary classification is used but this can easily be extended to the case of multi classification.

2.1.2.1 Evaluation methods

In the case of a binary classification there are two classes that you want to separate from each other. One of the classes is labelled as positives while the other is labelled as negatives. To evaluate the predictions of the binary classification a confusion matrix can be used which is a 2 by 2 matrix. Table 2 is an example of such a confusion matrix.

Table 2: Confusion matrix for binary classification.

		Predicted condition	
Real condition		tp	fn
		fp	tn

The four cells of the confusion matrix contain different performance metrics. These metrics are defined as:

- TP/FP = true positives/false positives, these are the number of predicted positives that were classified correct/incorrect [13, 14].
- TN/FN = true negatives/false negatives, these are the number of predicted negatives that were classified correct/incorrect [13, 14].

In the confusion matrix it is possible to use absolute values, so using the number of data that is classified as TP, TN, FN or FP. But it is also possible to use relative values so dividing the TP and FP by the total number of positives and the TN and FN by the total number of negatives. In this thesis the last approach is used.

2.1.2.2 Multi classification

In this thesis the confusion matrix that will be used is a 7 by 7 matrix because there are 7 different emotions of the CK+ dataset that will be classified. Table 3 is an example of the confusion matrix that can be used for the CK+ dataset.

Table 3: Confusion matrix for the CK+ dataset.

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	ta	fc	fd	ff	fh	fsa	fsu
Contempt	fa	tc	fd	ff	fh	fsa	fsu
Disgust	fa	fc	td	ff	fh	fsa	fsu
Fear	fa	fc	fd	tf	fh	fsa	fsu
Happy	fa	fc	fd	ff	th	fsa	fsu
Sadness	fa	fc	fd	ff	fh	tsa	fsu
Surprise	fa	fc	fd	ff	fh	fsa	tsu

Besides calculating the confusion matrix is it also useful to look at the average accuracy of the emotion detection method. This can be done by adding the accuracies on the major diagonal and divide the result by the total number of emotions.

$$\text{Average accuracy} = \frac{\text{ta} + \text{tc} + \text{td} + \text{tf} + \text{th} + \text{tsa} + \text{tsu}}{\text{total number of emotions}}$$

2.2 Face detection

The first step in extracting human emotions from images is detecting the face on the image. At the moment there are several algorithms that can be used to extract human faces from images. Some of these algorithms are:

- Principal Component Analysis [15]
- Linear Discriminant Analysis [16]
- Local Binary Pattern [17, 18]
- Scale Invariant Feature Transform [19]
- Histogram of Oriented Gradients [1]
- Viola-Jones [20]
- …

In this thesis the focus lays on two of these algorithms, the Histogram of Oriented Gradients and the Viola-Jones algorithm. The reason for choosing these two algorithms is because of their real-time performance, which is a requirement because of the objective defined in the Objectives . After selecting the two algorithms, they were implemented in the real-time application for comparison regarding their processing time. The results of the comparison will be discussed in Chapter 4 but first the theoretical background of both algorithms will be highlighted to gain a better understanding of the algorithms.

2.2.1 Histogram of Oriented Gradients

Histogram of Oriented Gradients is an algorithm introduced in 2005 by Navneet Dalal and Bill Triggs and can be used for detecting human faces in images. This algorithm consists out of 6 different steps as illustrated in Figure 4. Each of these steps will be discussed in detail in the following sections.

Figure 4: Different steps in the Histogram of Oriented Gradients algorithm [1].

2.2.1.1 Normalize gamma and color

The first step is the normalization of the colour space with power law (gamma) equalization.

$$g(x, y) = f(x, y)^{\gamma}$$

- Gamma < 1 means that the values are mapped towards higher values.
- Gamma > 1 means that the values are mapped towards lower values.

This normalization, tested on grayscale, RBG and LAB colour spaces, has only a modest effect on performance. One of the reasons for this is that similar results are achieved by the subsequent descriptor normalization. Thus leaving out this step in the HOG algorithm implementation does not effects the overall performance.

2.2.1.2 Compute gradients

For the computation of the gradients the following 1-D point centred masks are used

$$g_x = [-1 \quad 0 \quad 1]$$

$$g_y = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

So suppose the gradients of the following 3x3 pixel matrix need to be calculated:

255	100	10
120	90	35
70	0	254

Then the gradients g_x and g_y of every pixels are calculated by moving the masks in the horizontal and vertical direction over every pixel. With the mask centred on the pixel whose gradient you want to calculate. In this example the border of the pixel matrix is padded with zeros for the calculation of the gradients. The results of the g_x matrix are:

100	-245	-100
90	85	-90
0	184	0

The resulting matrix is a 3x3 matrix where every cell contains the signed gradient in the x-direction of the pixel of the original 3x3 pixel matrix. If the unsigned gradients are used then the absolute values of the gradients of the previous matrix are used. In case of colour images the gradients are calculated for each colour channel. The resulting gradient vectors for the pixels are then those with the largest norm.

2.2.1.3 Weighted vote into spatial and orientation cells

Now it is time to construct the cell histograms, these cells whose histograms are calculated are local spatial regions. The orientation-based histogram bins of each cell are constructed by the weighted votes of every pixel within the cell. The histogram bins are equally divided over 0 to 360 degrees (signed gradients) or 0 to 180 degrees (unsigned gradients). In the implementation of [1] the best results were obtained by using 9 bins divided over 0 to 180 degrees, so using unsigned gradients.

The vote weight of the pixel is computed as the magnitude of the gradient. The magnitude of the gradient of pixel (x, y) can be expressed as:

$$\mathrm{mag}\left(\begin{bmatrix} g_x \\ g_y \end{bmatrix}\right) = \left[g_x^2 + g_y^2\right]^{\frac{1}{2}}$$

After calculating the vote weight of the pixel, it is also necessary to calculate the orientation of the gradient because the weights are assigned to bins based on the orientation of the gradient of that pixel. The orientation of pixel (x, y) can be expressed as:

$$\alpha(x, y) = \tan^{-1}\left[\frac{g_x}{g_y}\right]$$

In case the orientation of the gradient lies between two bins, for example the gradient orientation is 65° then the vote weight of that pixel is assigned to the bins 50° and 70° by using bilinear interpolation. In this example the distance to the bin centres is 15° and 5°. This means that the ratios are 15/20 and 5/20, so 3/4 of the vote weight of the pixel with gradient orientation 65° goes to the orientation bin of 50° and the other 1/4 goes to the orientation bin of 70°. The vote weight of every pixel in the cell is calculated and accumulated to the right orientation bin. An example of a cell histogram is shown in Figure 5.

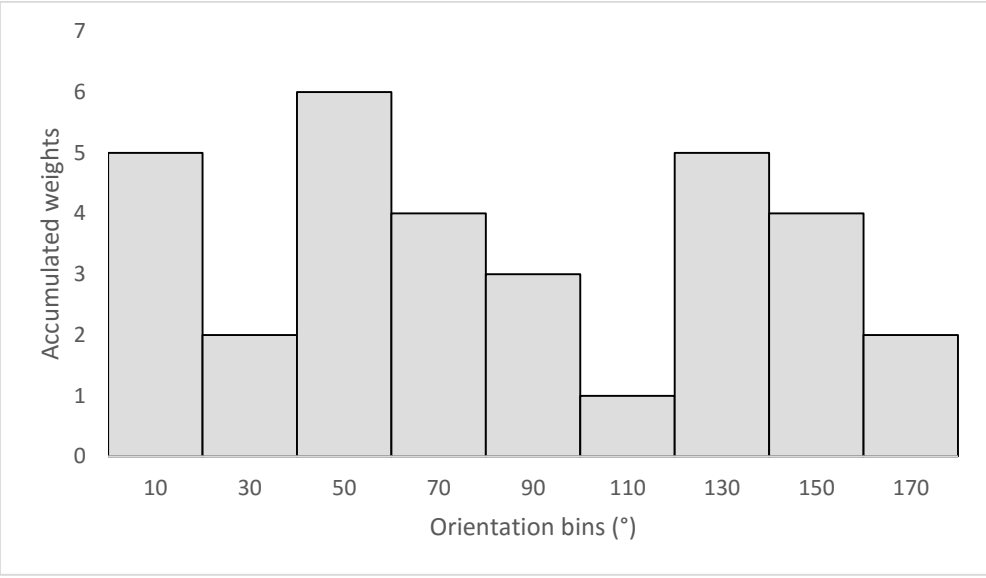

Figure 5: Example of a cell histogram.

2.2.1.4 Contrast normalize on overlapping spatial blocks

After creating the cell histograms, the gradient strengths must be normalized locally. This is because they can vary due to changes in contrast and illumination. The local normalization can be executed by grouping different cells into larger spatial blocks. After contrast normalizing each block the final descriptor can be obtained by concatenating all components of the normalized cell histograms from all block regions into a single vector.

While creating the spatial blocks each cell is used to create multiple blocks, so there is an overlap of the blocks. By doing this each cell histogram contributes several components to the final descriptor vector and this results in better performance of the algorithm.

The blocks created by grouping cells are divided into two categories based on their geometries. The first one is the R-HOG (rectangular HOG) and the second one is the C-HOG (circular HOG). Both of the block categories will be discussed briefly.

Although R-HOG blocks can be appear similar to SIFT descriptors [21] they are computed in dense grids at a single scale without orientation alignment, while SIFT's are computed at sparse, scale-invariant key points, rotated to align orientations.

The geometry of R-HOG blocks is most of the time square shaped and defined by 3 parameters ς, η and β. Where ς defines the size of the grid (ς x ς), η the size of the pixel cells (η x η) and β the number of orientation bins. In the implementation of Navneet Dalal and Bill Triggs a 16x16 block is used composed out of four 8x8 cell blocks. Also is in their implementation a Gaussian spatial window applied, so that the pixels around the edge of the block are weighted less, to increase the performance with $\sigma = 0.5 * \text{block\_width} = 8$.

For the C-HOG there are two variants which differ in their geometry. The first one has a single circular central cell Figure 6, while the second one has a central cell that is divided in angular sectors Figure 7. Just like the R-HOG is the C-HOG also defined by a few parameters, these parameters are the number of angular and radial bins, the radius of the centre bin and the expansion factor for subsequent radii. To obtain good performance it is necessary that the C-HOG has a minimum of four angular bins and two radial bins. In combination with a radius of 4 pixels for the central bin and an expansion factor of 2 results this in the best performance.

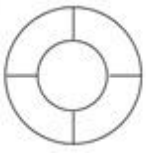

Figure 6: C-HOG with single circular central cell [1].

Figure 7: C-HOG with central cell divided in angular sectors [1].

For the block normalization 4 different schemes can be used. In these block normalization schemes v is defined as the non-normalized descriptor vector with $\|v\|_k$ the k-norm (with k = 1,2) of the vector and ϵ a small constant.

$$L2\text{-}norm: v \rightarrow \frac{v}{\sqrt{\|v\|_2^2+\epsilon^2}}$$

$L2\text{-}Hys$, $L2\text{-}\ norm$ followed by clipping (limiting the maximum values of v to 0.2) and renormalizing as in [21] [1]

$$L1\text{-}norm: v \rightarrow \frac{v}{\|v\|_1+\epsilon}$$

$$L1\text{-}sqrt: v \rightarrow \sqrt{\frac{v}{\|v\|_1+\epsilon}}$$

The performance of the $L2\text{-}norm$, $L2\text{-}Hys$ and $L1\text{-}sqrt$ block normalization schemes are almost the same while the performance of $L1\text{-}norm$ is less but still better than do not normalize the blocks at all.

2.2.1.5 Collect HOG's over detection window

The previous steps where performed on a part of the image, called the detection window. Now it is time to construct the combined feature vector and this vector is obtained by concatenating the HOG descriptors in the detection window. The size of the detection window depends on the shape of the objects that needs to be detected. In [1] humans were detected and this resulted in a window of 64x128.

2.2.1.6 Linear SVM

The last step of the Histogram of Oriented Gradients is classifying the feature vector obtained from the detection window. To do this a soft, C = 0.01, linear SVM is used.

2.2.2 Viola-Jones

Another algorithm that can be used for face detection is the Viola-Jones algorithm introduced in 2001 by Paul Viola and Michael Jones. This algorithm exist out of four different stages which are:

- Selecting haar features
- Creation of an integral image
- Training by using Adaboost
- Detection using a cascade of classifiers

In the next sections each of these stages will be described in more detail.

2.2.2.1 Selecting haar features

The classification of images is based on the value of features and this value is obtained by subtracting the sum of pixels under the white rectangle from the sum of pixels under the black rectangle. In the implementation of the Viola-Jones algorithm there are three kinds of features that are used. These three kind of features are:

- Two-rectangle features (Figure 8 A and B)
- Three-rectangle features (Figure 8 C)
- Four-rectangle features (Figure 8 D)

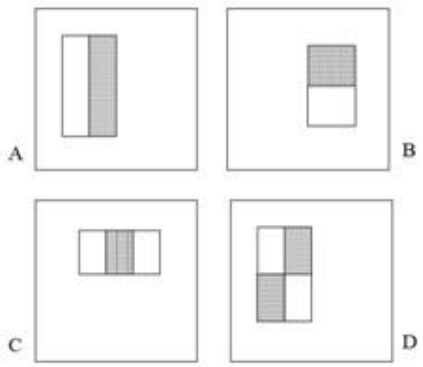

Figure 8: Examples of the three kinds of features [20].

The only orientation available for the features that are used are horizontal and vertical. Because of that the features are sensitive to simple image structures like bars, edges, … but their limitation in flexibility is allowed because of the huge gain in computational efficiency. But still there are a lot of features that need to be calculated because all possible sizes and locations of each feature are computed. This means that for a window of 24-by-24 160000 features are calculated. To speed up this calculation integral images are introduced.

2.2.2.2 Integral image

To enhance the calculation of the rectangle features an intermediate representation for the image can be used. This intermediate representation is called the integral image. Take for example point (x, y) in Figure 9 then the integral image at that point is the sum of the pixels of the grey region.

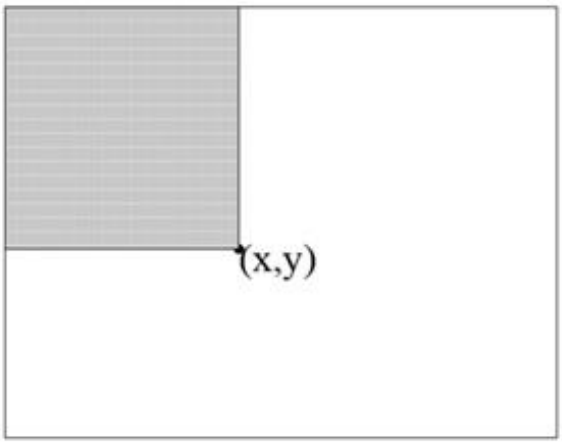

Figure 9: The sum of the pixels in the grey region is the value of the integral image at point (x, y) [20].

Another example is given in Figure 10. Suppose the sum of pixels of area D needs to be calculated. To do this 4 array references can be used annotated with the numbers 1,2,3 and 4. Then the value at location 1 is the sum of pixels of area A, at location 2 the value is A+B, at location 3 it is A+C and finally the sum of pixels at location 4 is A+B+C+D. The sum of pixels of area D can then be calculated as

$$D = 4 + 1 - (2 + 3)$$

$$D = A + B + C + D + A - (A + B + A + C)$$

$$D = D$$

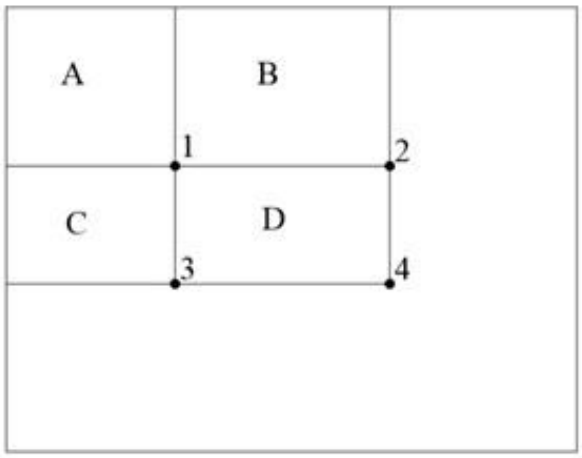

Figure 10: Example array references [20].

By using the integral image only six array references are needed to calculate two-rectangle features because of the adjacent rectangular sums. For the three-rectangle features there are eight array references needed, and nine array references for the four-rectangle features.

2.2.2.3 Learning Classification Functions

Although the computation of the individual features is very efficient, the computation of the complete set of features is not. To overcome this problem a variant of Adaboost [22] will be used. The normal Adaboost algorithm is a machine learning algorithm that combines a collection of re-weighted weak classification functions to make a stronger classifier and by doing that it will boost the classification performance of a simple learning algorithm. Re-weighting the classification functions makes sure that the images which were classified incorrect, by the weak classifier, will be highlighted.

The adapted implementation of Adaboost in the Viola-Jones algorithm will restrict the weak learner to a set of classification functions. In this set each classification function will depend on a single feature. To achieve this the weak learning algorithm will be constructed in such a way that the single rectangle features which separates the positive and negative examples the best will be selected, the other rectangle features are discarded. Then the weak learner will determine the optimal threshold classification function for each rectangular feature so that the minimum number of examples are misclassified. A weak classifier can be expressed as

$$h_j(x) = \begin{cases} 1 \ if \ p_j f_j(x) < p_j \theta_j \\ 0 \ otherwise \end{cases}$$

With

- f_j = a feature
- θ_j = a threshold
- p_j = a parity

An example of the first and second features selected for face detection is illustrated in Figure 11. It is clear that the first feature is focussing on the fact that the eye region is darker than the region across the upper cheeks. While the second feature emphasizes the difference in intensity between the bridge of the nose and the eye region.

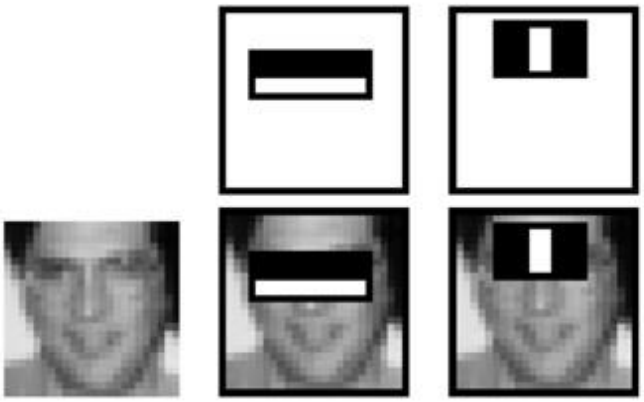

Figure 11: “The first and second features selected by Adaboost” [20].

2.2.2.4 Cascading classifiers

Every image contains a lot of negative sub-windows, this means small parts of the image that do not include faces. Therefore it is preferable to reject as much negative sub-windows in an early stage with a simple classifier instead of passing all sub-windows to a more complex classifier with a higher computation time because every sub-window has the same computation time.

Therefore the Viola-Jones algorithm is using a cascade of classifiers where the first classifier a two-feature classifier is that detects all faces with a false positive rate of 40%. The advantage of this classifier is that it can reject a lot of sub-windows with very few operations. The sub-windows that passes the first classifier of the cascade are going to the next classifier which faces more difficult sub-windows than the first classifier and therefore is more complex than the first classifier. The sub-windows that passes the second classifier are going to the next one and so on. Figure 12 depicts a schematic representation of a cascade of classifiers.

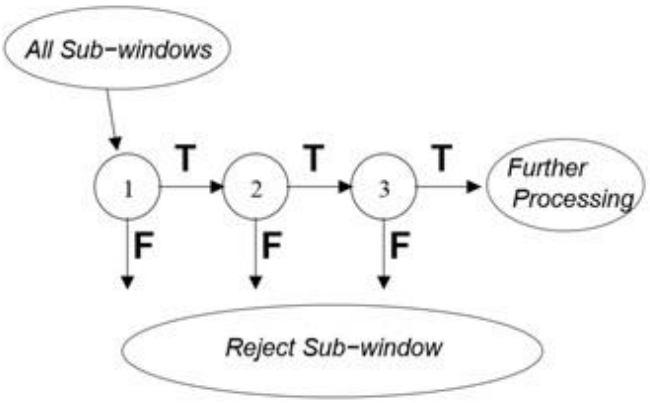

Figure 12: Schematic representation of a cascade of classifiers [20].

The cascade of classifiers used in the Viola-Jones algorithm has 38 layers with a total of 6061 features from which the first five layers of the cascade of classifiers have 1, 10, 25, 25 and 50 features [23].

To implement the cascade of classifiers the minimum acceptable false positive rate (f_i) and detection rate (d_i) of the ith classifier is selected. Then the adapted Adaboost algorithm, described in the previous section, is used to train each layer of the cascade. The number of features that are used by the classifier is increased until the target detection and false positive rates are reached for this level. Another layer is added to the cascade when the target false positive rate is still not met.

Finally, to detect faces the cascaded detector scans, after training, images on different locations with different scales. Thus for example a sub-window of 24-by-24 scans the image to search for faces. Then in every sub-window the cascaded classifier will be used to detect faces. After this is done the detector will be scaled and starts searching again for faces but only the detector is scaled not the image itself [23].

2.3 Feature point selection

After a successful facial detection it is time to extract feature points from the face region. The reason for using point-based features instead of other features is because of their low computational cost while passing these features as an input vector to the trained SVMs. These feature points are extracted by using an ensemble of regression trees as described in [6]. But before this algorithm is described there will be a short introduction about feature points.

2.3.1 Feature points

To find similarity between two different images feature points are used. In the images there will be searched for the same areas and they will be indicated with a feature point. By using feature points it is possible to compare images taken from different perspectives and see if there is a match between the two images as shown in Figure 13.

Figure 13: Matching two different images [24].

But it is also possible to work in the other way. So starting with 2 images which are taken at the same location but at different times to observe if there have been changes through the time. Applying this principle can make it possible to extract facial muscle movement by observing if the feature points added on the face of a person have changed through the time. This principle is depicted in Figure 14 and Figure 15.

Figure 14: Facial feature points in neutral image. (©Jeffrey Cohn).

Figure 15: Facial feature points in emotion image. (©Jeffrey Cohn).

Now that the use of feature points has been explained it is time to analyse the ensemble of regression trees algorithm.

2.3.2 Ensemble of Regression Trees

This algorithm uses a cascade of regressors, where each regressor is built as a regression tree, to estimate the position of facial feature points also known as facial landmarks. This cascade of regressors can be expressed as

$$\widehat{\boldsymbol{S}}^{(t+1)} = \widehat{\boldsymbol{S}}^{(t)} + r_t(I, \widehat{\boldsymbol{S}}^{(t)})$$

Where

- $\widehat{\boldsymbol{S}}^{(t)}$ = the current estimate of vector $\boldsymbol{S}$ which contains the coordinates of all the facial feature points in the image I.
- I = the image on which the facial feature points needs to be detected.
- $r_t(.,.)$ = the regressor which predicts an update vector of the image I and $\widehat{\boldsymbol{S}}^{(t)}$.

The ensemble of regression trees expands the range of outputs and ensures that this range will lay in a linear subspace of training data but only if the initial estimate $\widehat{\boldsymbol{S}}^{(0)}$ belongs to this space. To obtain an initial shape estimate the mean shape of the training data centered and scaled with respect to the bounding box output of a face detector can be used.

To train each regressor an algorithm called gradient tree boosting [25] with a sum of square error loss is used. For training the regressor the training data, that exists out face images I and corresponding shape vectors $\boldsymbol{S}$, will be used to create data triplets. These data triplets contain a face image, an initial shape estimate and the target update which can be expressed as

$$(I_{\pi i}, \widehat{\boldsymbol{S}}_i^{(0)}, \Delta \boldsymbol{S}_{\boldsymbol{i}}^{(0)})$$

Where

$$\pi_i \in \{1, \dots, n\}$$

$$\widehat{\boldsymbol{S}}_i^{(0)} \in \{\boldsymbol{S}_1, \dots, \boldsymbol{S}_n\} \backslash \boldsymbol{S}_{\pi_i}$$

$$\Delta \boldsymbol{S}_{\boldsymbol{i}}^{(0)} = \boldsymbol{S}_{\pi_i} - \widehat{\boldsymbol{S}}_i^{(0)}$$

Now that these triplets are defined it is possible to train the regression function r_0 by using the gradient tree boosting algorithm with a sum of square error loss. Once the training for the first regression function is done the triplets are updated to provide the training data for the next regressor r_1. This process is continued until a cascade of T regressors are trained which obtain, when combined, a sufficient level of accuracy.

In the regression tree, each regressor is built as a regression tree by the gradient tree boosting algorithm, decisions at each split node are made by thresholding the difference in intensities between two pixels. The 3 parameters used for this decision are

$$\theta = (\tau, u, v)$$

Where u and v are the pixels used for thresholding and where τ the threshold value is. Each node can now be defined as

$$h\left(I_{\pi_i}, \widehat{\boldsymbol{S}}_i^{(t)}, \theta\right) = \begin{cases} 1 & I_{\pi_i}(u') - I_{\pi_i}(v') > \tau \\ 0 & otherwise \end{cases}$$

These pixels, u and v, are defined in the coordinate system of the mean shape. Now it is necessary to find the pixels at the same position relative to its shape as u and v to the mean shape. Therefore the location of these points are wraped to the mean shape. This is done by using the current shape estimate before extracting the features.

2.4 Support Vector Machines

The algorithm that is used to classify the emotions, based on the feature points, is a machine learning algorithm called support vector machines [26] . In the next sections will be explained how support vector machines are searching for an optimal hyperplane and how not linearly separable data can be mapped into a higher-dimensional space to find an optimal hyperplane.

2.4.1 Optimal hyperplane selection

Support Vector Machines are trying to build a classifier by choosing a hyperplane that separates the data such that it maximises the distance between the hyperplane and the nearest data point of each class as illustrated in Figure 16. The solution of this mathematical problem, in which is assumed that the data is linear separable, is described in this section and based on [25, 27].

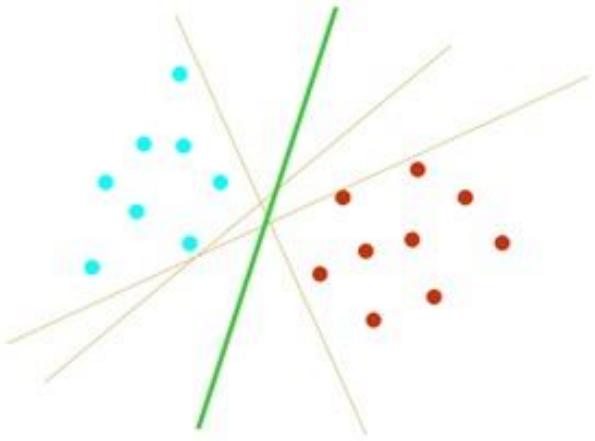

Figure 16: The possible hyperplanes for a binary classification problem where the optimal hyperplane is highlighted [28].

Suppose the dataset $\mathcal{D}$ needs to be classified and define this dataset as

$$\mathcal{D} = \{(x_i,\ y_i)|x_i\ \in\ \mathbb{R}^p, y_i\ \in \{-1,1\}\}_{i=1}^{n}$$

In this definition x_i is a p-dimensional vector containing the data that needs to be classified. In total there are n data vectors and each data vector is labelled as -1 or 1 to indicate to which class it belongs. So let's start by taking a look at the hyperplane equation that can be expressed as

$$\mathbf{w}.\mathbf{x} - \mathrm{b} = 0$$

Now 2 additional hyperplanes are selected, these hyperplanes will also separate the data and are defined as follow

$$\mathbf{w}.\mathbf{x} - \mathrm{b} = 1$$

And

$$\mathbf{w}.\mathbf{x} - \mathrm{b} = -1$$

This time no data is allowed between the two hyperplanes and every vector **x** that lays on the hyperplanes is labelled as 1 or -1. So they belong to the class 1 or -1. But not every hyperplane is selected only those who meet the following constraints.

$$\mathbf{w}.\mathbf{x_i} - \mathrm{b} \geq 1 \text{ for } \mathbf{x}_\mathrm{i} \text{ having the class } 1$$

Or

$$\mathbf{w}.\mathbf{x_i} - \mathrm{b} \leq -1 \text{ for } \mathbf{x}_\mathrm{i} \text{ having the class } -1$$

This means that every vector $\mathbf{x_i}$ that belongs to the class 1 needs to have a value equal to 1 or bigger than 1 when used in the equation $\mathbf{w}.\mathbf{x_i} - \mathrm{b} \geq 1$. If this is not the case it would mean that there is a vector $\mathbf{x_i}$ between the two hyperplanes or in other words inside the margin and this is not allowed.

Now it's time to combine the two constraints to one constraint and this can be done by multiplying both sides of the constraints with the class label y_i. The result for the first constraint is then

$$y_i(\mathbf{w}.\mathbf{x_i} - \mathrm{b}) \geq y_i(1)$$

Because $y_i = 1$ this results in

$$y_i(\mathbf{w}.\mathbf{x_i} - \mathrm{b}) \geq 1$$

For the second constraint this results in

$$y_i(\mathbf{w}.\mathbf{x_i} - \mathrm{b}) \geq y_i(-1)$$

Or

$$y_i(\mathbf{w}.\mathbf{x_i} - \mathrm{b}) \geq 1$$

Combining these two equations results in one equation that encapsulate the two constraints and can be written as

$$y_i(\mathbf{w}.\mathbf{x_i} - \mathrm{b}) \geq 1 \text{ for all } 1 \leq i \leq n$$

After defining these constraints the distance between the two hyperplanes needs to be maximized but before that is possible the distance between the two hyperplanes, also called the margin m needs, to be defined. To do this suppose that there are two hyperplanes H_0 and H_1 and a vector $\boldsymbol{x_0}$ on hyperplane H_0. To find the distance between the two hyperplanes, which are parallel, a vector with amplitude m and a direction perpendicular to hyperplane H_1 is needed. Another notation for defining a hyperplane is

$$\mathbf{w}^\mathbf{T}.\mathbf{x} = 0$$

The definition of a dot product is

$$\mathbf{x}.\mathbf{y} = \|x\|\|y\|\cos(\theta)$$

So if the dot product equals to 0 this means that $\theta = 90°$ and the two vectors are perpendicular. Because $\mathbf{w}.\mathbf{x} - \mathrm{b} = 0$ is another way of writing the hyperplane this means that the vector $\mathbf{w}$ is perpendicular with respect to $\mathbf{x}$. Thus there is already a vector with a direction perpendicular to hyperplane H_1 that can be used. Now the vector $\mathbf{w}$ can be used to create a unit vector $\mathbf{u} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$ which is perpendicular to hyperplane H_1. Multiplying $\mathbf{u}$ by m results in a new vector $\mathbf{k} = m\mathbf{u}$ with amplitude m and with a direction perpendicular to H_1. Adding $\mathbf{k}$ to point $\mathbf{x_0}$ results in a new point $\mathbf{z_0} = \mathbf{x_0} + \mathbf{k}$ located on the hyperplane H_1 as illustrated in Figure 17.

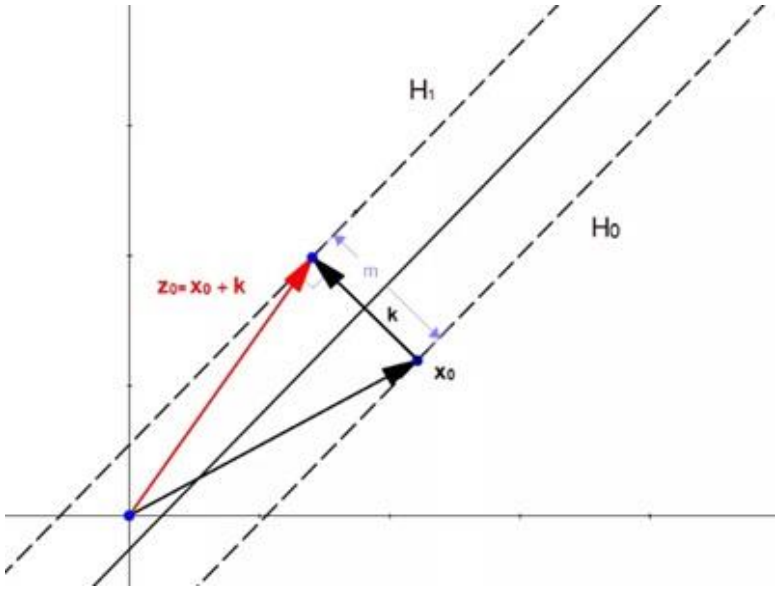

Figure 17: Hyperplanes H_0 and H_1 with the margin defined by the amplitude of vector $\boldsymbol{k}$ [27].

Hyperplane H_1 is defined as $\mathbf{w}.\mathbf{x} - \mathrm{b} = 1$ and because $\mathbf{z_0}$ is located on the hyperplane the equation can be written as

$$\mathbf{w}.\mathbf{z_0} - \mathrm{b} = 1$$

$\mathbf{z_0}$ is defined by $\mathbf{x_0} + \mathbf{k}$ so this results in

$$\mathbf{w}.(\mathbf{x_0} + \mathbf{k}) - \mathrm{b} = 1$$

In this equation $\mathbf{k}$ can be replaced by $m\frac{\mathbf{w}}{\|\mathbf{w}\|}$

$$\mathbf{w}.(\mathbf{x_0} + m\frac{\mathbf{w}}{\|\mathbf{w}\|}) - \mathrm{b} = 1$$

$$\mathbf{w}.\mathbf{x_0} + m\frac{\mathbf{w}.\mathbf{w}}{\|\mathbf{w}\|} - \mathrm{b} = 1$$

$$\mathbf{w}.\mathbf{x_0} + m\frac{\|\boldsymbol{w}\|^2}{\|\mathbf{w}\|} - \mathrm{b} = 1$$

$$\mathbf{w}.\mathbf{x_0} - \mathrm{b} = 1 - m\|\mathbf{w}\|$$

$\mathbf{x_0}$ is located on hyperplane H_0 so therefore $\mathbf{w}.\mathbf{x_0} - \mathrm{b}$ equals to -1.

$$-1 = 1 - m\|\mathbf{w}\|$$

$$m = \frac{2}{\|\mathbf{w}\|}$$

To maximize the margin m it is necessary to find the hyperplane for which $\|\mathbf{w}\|$ is as small as possible while still respecting the constraint that no points can lay inside the margin. This results in the following optimization problem which returns, after solving it, the optimal **w** and b [25].

$$\min_{\mathbf{w},\mathrm{b}} \frac{1}{2}\|\mathbf{w}\|^2 \text{ subject to } y_i(\mathbf{w}.\mathbf{x_i} - \mathrm{b}) \geq 1 \; \forall i$$

2.4.2 Kernel functions

When the input space, **x**, is not linear separable the input vector can be mapped into a higher dimensional space, $\phi(\mathbf{x})$, also called the feature space, by using a non-linear mapping function ϕ. In this space the data is linear separable and the optimal separating hyperplane can be computed [29].

However when solving the optimization problem stated above the dataset, **x**, will only be used as an inner product. This means that the mapped dataset needs to be computed in a higher maybe infinite dimension and this is can be very complex [29]. To avoid this a kernel function will be introduced

$$K(\mathbf{x_i}, \mathbf{x}) = \Phi(\mathbf{x_i})\Phi(\mathbf{x})$$

This kernel function considerably reduces the complexity of this computation because the use of a kernel makes it possible to do the computation in the dimensionality of the input space. Therefore the input data does not need to be mapped into a higher dimensional space [29]. Three commonly used kernels are [30]:

- Polynomial kernel of degree p: $K(\mathbf{x_i}, \mathbf{x}) = (\mathbf{x_i}.\mathbf{x} + 1)^{\mathrm{p}}$
- Gaussian radial basis function kernel: $K(\mathbf{x_i}, \mathbf{x}) = \mathrm{e}^{-\|\mathbf{x}_i - \mathbf{x}\|^2/2\sigma^2}$
- Sigmoid kernel: $K(\mathbf{x_i}, \mathbf{x}) = \tanh(\kappa \mathbf{x}_i.\mathbf{x} - \delta)$

Figure 18 depicts an example of a polynomial kernel with a degree of 3 used to separate a non-linear separable dataset. Thus by using the appropriate kernel an optimal separable hyperplane can be obtained in the input space.

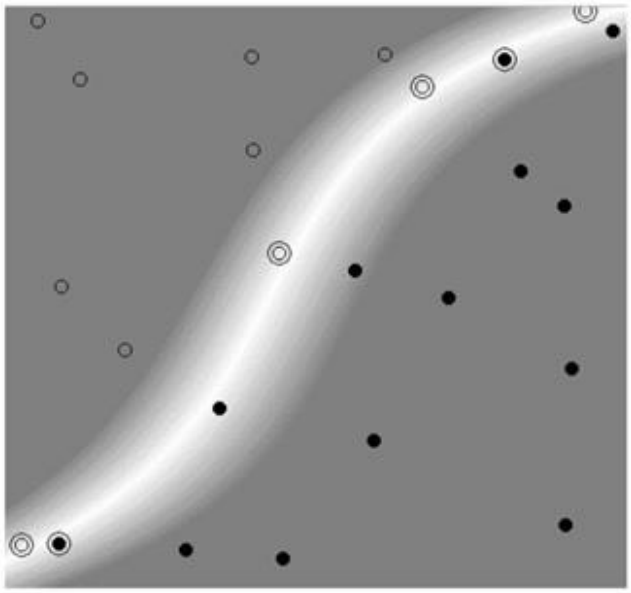

Figure 18: An example of an polynomial kernel of degree 3 [30].

In this chapter the Extended Cohn-Kanade (CK+) dataset is introduced, this dataset will be used for evaluating the accuracy of the proposed algorithm in detecting 7 basic emotions. After explaining the used dataset, the face detection algorithms that are tested and implemented in this thesis are introduced. The next section describes the algorithm that is implemented to extract the facial feature points that are used by the emotion detection implementation. To the end of this chapter a short overview of support vector machines is provided.

Chapter 3

Implementation

After explaining the different algorithms that are used in this thesis it is time to look at the implementation of the algorithms. In this chapter the workflow of the real-time emotion classification will be explained step by step. The four big steps in this program are:

- Face detection
- Facial feature points extraction
- Displacement measuring facial feature points
- SVM-based emotion classification

These different steps will be discussed in more detail in the following sections. At the end of this chapter the total program will be explained by using a flow chart to give a visual overview of the program.

3.1 Face detection

For face detection the Viola-Jones algorithm and the Histogram of Oriented Gradients algorithm were both implemented and compared based on their detection speed. The reason for comparing both algorithms on their speed is because the goal of this thesis is to develop a real-time system so the detection of the face needs to be very fast. To compare the speed of both algorithms 74 images from the CK+ dataset were used where the total detection speed was divided by the number of images that were used to obtain an average detection speed. All these images express the emotion surprise and were scaled with a factor 1, 0.5 and 0.25. As indicated in Table 4 rescaling the images speeds up the face detection. However this scaling is a trade-off between speed and range of detection because how smaller the image how smaller the detection range becomes so the rescaling is not unlimited. In this thesis the scaling factor of 0.25 is used and this factor is empirical determined. Finally Table 4 also indicates that the HOG algorithm outperforms the Viola-Jones algorithm at every scale. This is also the reason why the HOG algorithm is implemented in this thesis. To implement the HOG algorithm the Dlib library was used while the Viola-Jones algorithm was implemented by using the OpenCV library.

Table 4: Face detection speed of the HOG algorithm and Viola-Jones algorithm at different scales.

Scale	HOG algorithm	Viola-Jones algorithm
1	63,95 ms	162,35 ms
0,5	17,64 ms	44,39 ms
0,25	4,39 ms	14,88 ms

However the HOG implementation from the Dlib library is slightly different from the implementation discussed so far. In the Dlib library the HOG features, from the HOG algorithm described in the previous chapter, are replaced by the HOG features described in the Deformable Part Model algorithm [31] which results in a faster face detection.

In the Deformable Part Model the 36-dimensional HOG features are replaced by a set of 31-dimensional HOG features which contains the same information. The exact details of this implementation can be found in [31].

Besides reducing the dimensions to enhance the face detection, a pyramid of HOG is also used. In this pyramid the input image is scaled to different resolutions to see if a face is present in the image. The scaling step that is used for this is 1,2 and the image is down scaled until the size of the detection window, 64x64, is reached or the maximum pyramid level which is set on 1000. The window used for detecting the face is not scaled and has thus the same for every image in the pyramid. The scaling makes it possible to detect a face at different scales. This concept is illustrated in Figure 19.

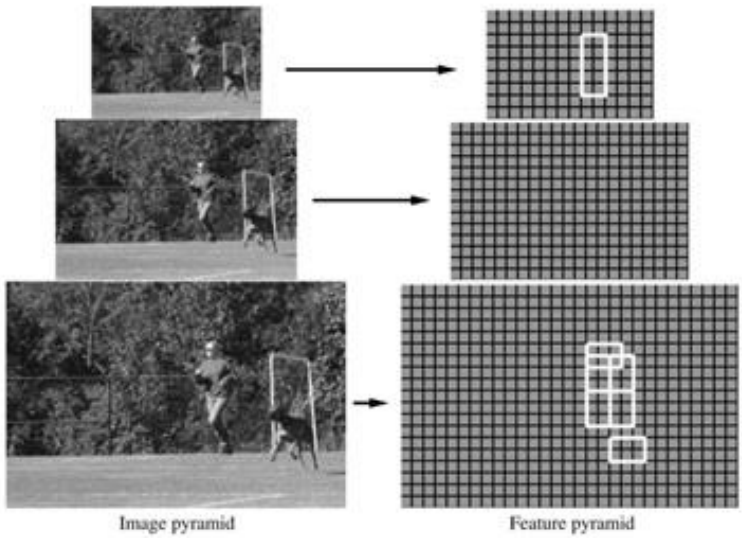

Figure 19: Example of a feature pyramid [31]

3.2 Facial feature points extraction

The extraction of the facial feature points is based on the algorithm described in [6] and uses difference in intensity between pixels to locate the correct position of the feature points. The reason for choosing this algorithm is because of it's speed. The algorithm performs feature point detection in less than 3 ms independent from the scale of the image. In total 19 different feature points are used in this thesis, these feature points are depicted in Figure 20. The reason for using these 19 different feature points will be explained in section 4.1.2 The location of the feature points and the corresponding distances that are computed are based on [7]. However in this paper only 11 feature points are used and the contraction around the mouth is also not included.

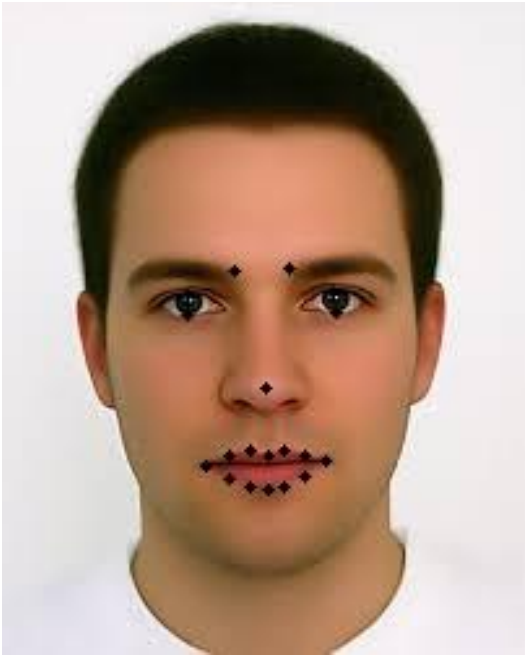

Figure 20: The 19 feature points of interest [32].

For the implementation of the Ensemble of Regression Trees algorithm the Dlib library is used. However the Dlib library returns 68 facial feature points instead of the 19 points that are preferred for this thesis. The location of these 68 facial feature points are illustrated in Figure 21. The numbering of the feature points is always in the same order. So this makes it possible to extract the 19 feature points of interest very convenient.

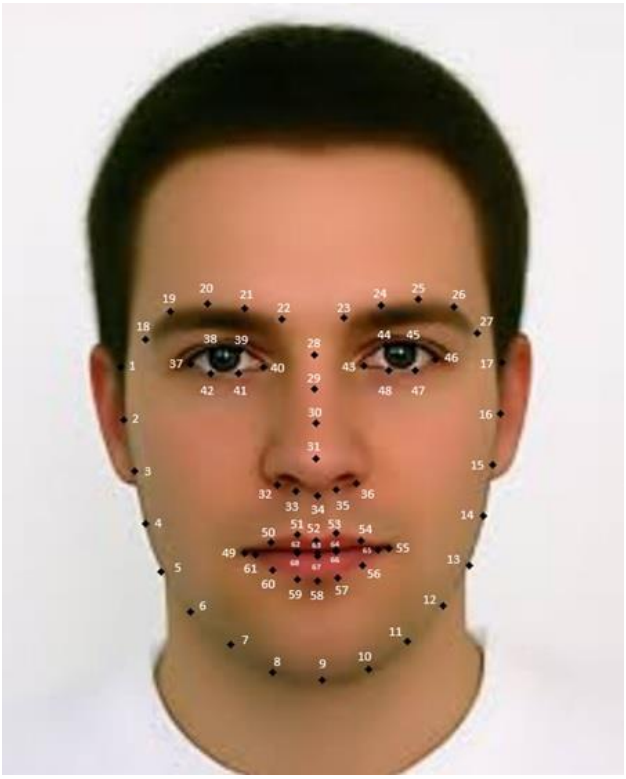

Figure 21: The order of the 68 facial feature points returned by the Dlib library [32].

The feature points around the mouth, on the nose and on the eyebrows can be extracted immediately but the feature points of the upper and lower eyelids need some additional computation. As depicted in Figure 21 the points 38 and 39 can be used for computing the right upper eyelid midpoint and the points 41 and 42 to compute the right lower eyelid. This is done by taking the midpoint between those points as x-coordinate and their maximum y-coordinate for the upper eye lid or minimum y-coordinate in the case of the lower eye lid.

To extract the feature points an Ensemble of Regression Trees [6] is trained by using the iBug-300W dataset [33, 34, 35] which contains more than 7000 images. But in this thesis another 500 images from the CK+ dataset [9] were added to this dataset to improve the facial feature extraction. For training the shape predictor based on these images the following parameters were set as follow:

- Cascade depth = 15,
- Tree depth = 4,
- Number of trees per cascade level = 500,
- Number of test splits = 20.

For the values of the other parameters used in the Ensemble of Regression Trees algorithm will be referred to [8]. The improvement is notable in the overall classification of the different emotions but this will be discussed in the next section.

3.3 Calculation displacement ratios

Once the feature points are extracted it is time to measure the displacement of the feature points in time. Therefore 11 distances from [7] are extracted by using the feature points and in addition the contraction of the mouth is also taken in consideration. These 12 distances are the following ones:

- Distance between right upper eyelid and right lower eyelid
- Distance between left upper eyelid and left lower eyelid
- Distance between inner points left and right eyebrow
- Distance between noise point and inner point left eyebrow
- Distance between noise point and inner point right eyebrow
- Distance between noise point and right mouth corner
- Distance between noise point and midpoint upper lip
- Distance between noise point and left mouth corner
- Distance between noise point and midpoint lower lip
- Distance between right and left mouth corner
- Distance between midpoint upper and lower lip
- Distance of the mouth contour

These distances are depicted in Figure 22 and Figure 23. The first image is a neutral image while the second image expresses the emotion surprise.

Figure 22: Neutral image marked with the 12 distances (©Jeffrey Cohn).

Figure 23: Image expressing surprise with the 12 distances (©Jeffrey Cohn).

To measure these distances the Euclidean distance between the feature points will be calculated by using the pixel coordinates.

$$d(p_1, p_2) = \sqrt{(p_{1_x} - p_{2_x})^2 + (p_{1_y} - p_{2_y})^2}$$

Using the pixel coordinates means that the distance is measured in pixels. These distances can be used to train the SVM however this has a drawback. If the pixel distance would be used that means that, for applying the SVMs in a real-time application, the person in front of the camera, who's emotion needs to be detected, needs to be at the same distance in front of the camera as the persons in the database that is used to train the SVM. To overcome this problem displacement ratios will be calculated. This displacement ratio will divide each distance from the neutral image by each corresponding distance from a subsequent image.

$$Displacement\ ratio = D_r = \frac{Distance\ neutral\ image}{Distance\ subsequent\ image}$$

So for example if the displacement of the inner eyebrow feature points needs to be measured then first the Euclidean distance between those two points in the neutral frame will be calculated. Then the same distance is calculated in any subsequent frame and finally the ratio of those distances will be computed. Working with ratio's also means that there are 3 different situations:

- Displacement ratio < 1: distance is bigger than in the neutral expression
- Displacement ratio > 1: distance is smaller than in the neutral expression
- Displacement ratio = 1: distance is the same as in the neutral expression

Note however that obtaining a ratio of 1 is almost impossible because this means that the facial neutral expression has to be exact the same as in the neutral image used as reference.

3.4 SVM-based emotion classification

Once the displacement ratios are computed it is time to classify the emotions. To do this two different SVMs were trained. One multi-class SVM that separates the seven emotions (anger, contempt, disgust, fear, happy, surprise and sadness) and one binary SVM that separates contempt and fear. The reason for this second SVM is that it gives a big improvement for the accuracy of fear but this will be discussed in more detail in Chapter 4.

In the previous chapter a SVM was introduced as a binary classifier and this is also the case so a multi-class SVM does not exist. If a multi-class SVM is implemented this means that it is actually a combination of multiple binary SVMs. For this thesis the "one-against-one" approach is used which means that for k number of classes $k(k-1)/2$ binary classifiers are constructed. For example suppose that the emotions anger, contempt, disgust and fear need to be classified. This results in $k = 6$ which are the following SVMs:

- SVM anger/contempt
- SVM anger/disgust
- SVM anger/fear
- SVM contempt/disgust
- SVM contempt/fear
- SVM disgust/fear

Training a SVM with training data from the ith and jth classes will result in solving the following two-class classification problem [36].

$$\min_{\boldsymbol{w}^{ij},b^{ij},\boldsymbol{\xi}^{ij}} \frac{1}{2}\left(\boldsymbol{w}^{ij}\right)^T \boldsymbol{w}^{ij} + C\sum_t \left(\xi^{ij}\right)_t$$

$$\text{subject to } \left(\boldsymbol{w}^{ij}\right)^T \phi(\boldsymbol{x}_t) + b^{ij} \geq 1 - \xi_t^{ij}, \text{ if } \boldsymbol{x}_t \text{ in the } i\text{th class},$$

$$\left(\boldsymbol{w}^{ij}\right)^T \phi(\boldsymbol{x}_t) + b^{ij} \leq -1 + \xi_t^{ij}, \text{ if } \boldsymbol{x}_t \text{ in the } j\text{th class},$$

$$\xi_t^{ij} \geq 0.$$

What this basically means is that all data points $\boldsymbol{x}$ are passed through every SVM where each binary classification is considered as a vote. At the end the data point is assigned to the class with the highest vote for that data point. However when two classes have the same number of votes the data point will be stored in the class which name appears first in the array where the class names are stored [36].

For training the SVM the OpenCV library is used which on his turn is based on the LibSVM library [36]. But before the SVM can be trained some parameters need to be set, in this thesis there are only two parameters set the others are chosen by OpenCV. These parameters are:

- SVM type = C_SVC, which is defined as:
  - "*C-Support Vector Classification. n-class classification* $(n \geq 2)$*, allows imperfect separation of classes with penalty multiplier C for outliers.*" [37]
- Kernel type = RBF, which is defined as:
  - *"Radial basis function (RBF), a good choice in most cases.* $\boldsymbol{K}(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}, \gamma > 0$" [37]

After training the SVMs the implementation details are known. These details are shown in Table 5.

Table 5: Implementation details SVMs.

Parameter	Multi-class SVM	Binary SVM
svm_type	C_SVC	C_SVC
Kernel_type	RBF	RBF
gamma	3,375e-002	5,0625 e-001
C_value	3,125e+002	2,5
term_crit	1000	1000

The definition of these parameters can be found on [37]. The choice of using a RBF-kernel instead of another kernel is based on previous research [7, 38].

3.5 Real-time emotion detection

In this section the real-time emotion detection application will be discussed, which is developed in Visual Studio C++ and makes use of the Matlab engine, the OpenCV library and the Dlib library. This application is developed to make it possible to extract emotions at real-time on a lightweight notebook by using the on board webcam. For explaining the work flow of the program a flowchart will be used. This flowchart is depicted in Figure 24.

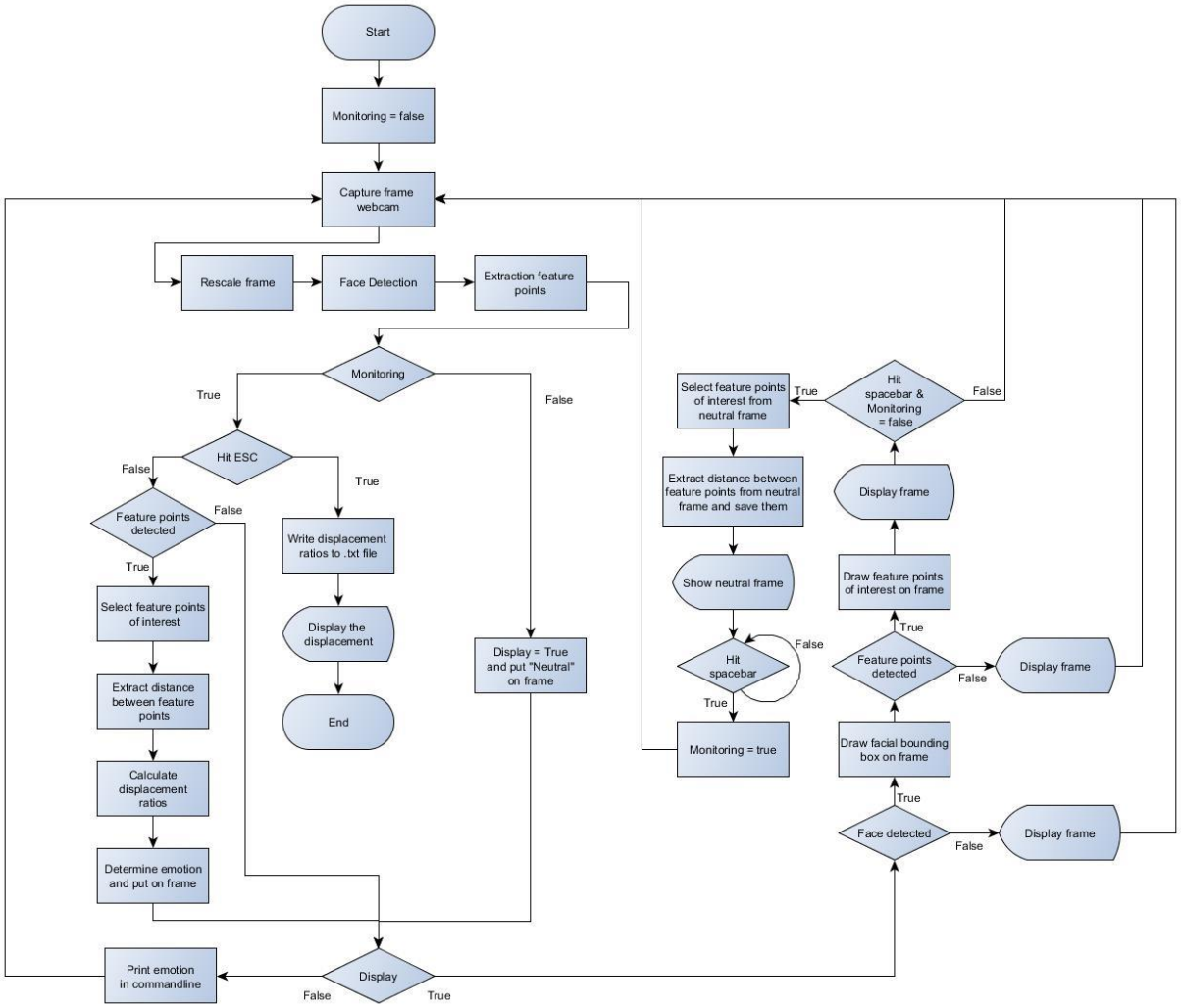

Figure 24: Flowchart of the real-time emotion detection application.

When the application is launched the parameter Monitoring will be set on false. This parameter indicates if an emotion will be extracted or if the neutral frame, which is used to calculate the displacement ratios, will be captured. If the variable Monitoring is set on false this means that the frames that are analysed will be used to pick the neutral frame, if it is set on true then the emotions are extracted from the frames.

The next step is capturing the frames that are coming from the webcam. These frames are rescaled to ¼ of the original size. The rescaled frames are then given to the adapted

HOG algorithm, implemented by using the Dlib library, to detect the face(s) on the frame. After the face(s) are detected, the face detector returns every face detected in the frame, a for-loop is used to extract the 68 facial feature points on every face that is detected. For this application that results in one face.

Now the program will check if the variable Monitoring is true or false. In the first stage of the program this variable is false so now the variable Display will be set on true. This variable indicates if the processed frames will be shown or not. For the frame that is chosen as neutral frame it is necessary that the feature points are located well on the face that is detected. Therefore it is necessary that the user can see the location of the feature points before he decides to use the current frame as a reference. At the same time the text “Neutral” will be put on the frame.

The variable Display is set true so then the program will check if there were any faces detected. When no faces are detected the current frame will be displayed and the user of the application just sees the frame from the webcam. In case there is a face detected, a rectangle bounding box will be drawn around the face and now it will be checked if the facial feature points were extracted from the frame. When they are not detected the frame will displayed with the bounding box around the face but without the facial feature points. If they are detected they will be shown on the frame and displayed to the user.

At this point in the program the user can select an appropriate frame as a neutral frame. And if the user does not hit the spacebar then this process will repeat itself. In case the user hit the spacebar and monitoring is false, which is the case, then the current frame will be used as a neutral frame. This means that from the 68 facial feature points the 19 feature points of interest are extracted and the corresponding 12 distances are calculated. These distances are then saved to use them later in the program to calculate the displacement ratios.

The frame that is chosen as a neutral frame will now be displayed to the user until he hit the spacebar again. As soon as the user hit the spacebar the variable monitoring is set to true, because now the emotion needs to be detected, and the program continues.

Again the frames are captured from the webcam and rescaled. Then the face is detected and the facial feature points are detected. But now monitoring is true so then the program will check if the user has hit the ESC-button. If this is done all the displacement ratios, which were saved in the program, are written to a .txt file. Then these displacement ratios are plot by sending the data to Matlab. After the plotting the program will stop.

When the user does not hit the ESC-button the program checks if the facial feature points were detected. If they were not detected the frames will be displayed when the variable Display is true.

Suppose that the feature points were extracted from the face then the feature points of interest are selected, thus the 19 feature points and the corresponding distances are extracted. This time the displacement ratios of the 12 distances are also calculated by using the neutral distances. The results of these calculations are given to the cascade of

SVMs which determine the emotion. This time the emotion is also put as a text on the frame. In case the variable Display equals to false then the frames are not shown but the emotion will be printed on the command line. This is done to improve the overall speed.

The total processing time of resizing the frame, detecting the face, extracting the facial feature points, calculating the displacement ratios and determining the emotions takes less than 30ms. The equipment that is used for implementing and running the application in real-time is described in section 1.3.

In the begin of this chapter the implementation of the HOG algorithm and the Viola-Jones algorithm is explained. Both algorithms are compared based on their processing speed and more details are provided regarding the HOG algorithm. The next section describes the implementation details of the ensemble of regression trees algorithm for extracting the facial feature points. After extracting the facial feature points the calculation of the displacement ratios is described, these ratios are used as input vectors for the support vector machines. The support vector machines are used for the emotion classification and their implementation details are described in more detail in this chapter. At the end these steps are combined in a real-time emotion detection algorithm which is described in the last section of this chapter.

Chapter 4

Results

In this chapter the results of this thesis will be discussed and a new technique in the area of computer vision is introduced. This is the cascading of a multi-class SVM with SVMs for binary classification to improve the accuracy of emotion recognition but first an overview of the different methods that have been used to achieve the final result will be explained. These methods are divided into two different categories based on the used shape predictor, the mathematical model of the trained ensemble of regression trees. These categories are:

- Shape predictor Dlib library
- Shape predictor Dlib library trained with an additional 500 images from the CK+ dataset

The first category is again divided into three more subcategories. These subcategories are:

- Emotion prediction based on 11 feature points
- Emotion prediction based on 19 feature points
- Emotion prediction based on 19 feature points with cascaded SVM(s)

The second category is subdivided into two categories which are the same as the last two subcategories of the category shape predictor Dlib library. After these results are discussed, the results in this paper are compared with the results from recent state-of-the art algorithms of emotion detection. In all the implementations analysed in this thesis a multi-class SVM is used to classify emotions.

4.1 Shape predictor Dlib library

This category is subdivided into three different categories. These categories are:

- Emotion prediction based on 11 feature points
- Emotion prediction based on 19 feature points
- Emotion prediction based on 19 feature points with cascaded SVM(s)

In each category the SVMs are trained and tested on the CK+ dataset. Therefore the sequence of each emotion is divided into two groups, the training data and test data. The exact division of the sequences is depicted in Table 6. The same division is used for the results where the shape predictor includes 500 images of the CK+ dataset.

Table 6: Division training and testing data for every emotion sequence.

	Training	Testing
Anger	23	22
Contempt	9	9
Disgust	30	29
Fear	13	12
Happy	35	34
Sadness	15	14
Surprise	41	41

4.1.1 11 feature points

The first implementation uses 11 feature points to calculate 11 distances, which are depicted in Figure 25. These distances are the same as the ones for the 12 feature points only these distances do not include the mouth contour.

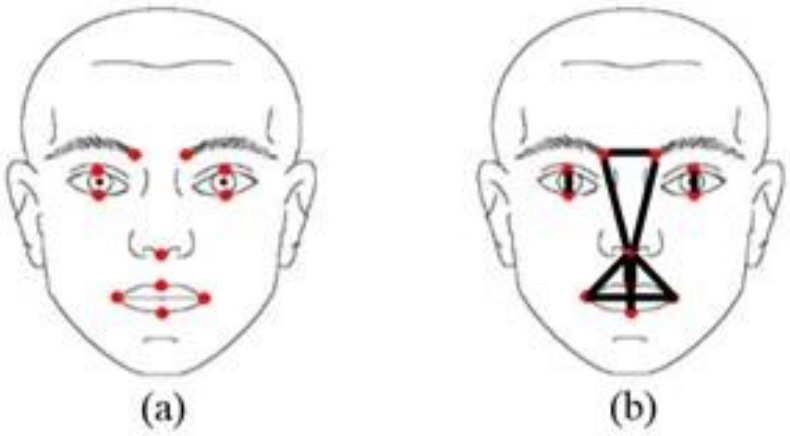

Figure 25: The 11 feature points (a) and 11 distances (b) used for extracting emotions [7].

The choice of the 11 feature points is based on [7]. In this paper these 11 feature points are suggested to identify the principle muscle actions for the recognition of 6 basic emotions (anger, disgust, fear, happiness, sad and surprise). In this thesis the first implementation uses only these 11 feature points and 11 distances. An additional advantage of using 11 distances to classify emotions is that the training of the SVM becomes much faster, because the dimensions from the input data are not so large as when for example the whole image for classification is used.

The results of extracting these 11 feature points to estimate the emotions are shown in the confusion matrix of Table 7. This matrix indicates that the accuracy of detecting fear and sadness is not satisfying. The reason for their poor results is because they are confused with other emotions. For fear there is even a chance of 41.67% that it will be detected as sadness. On the other side sadness has a chance of 21,43% to be confused with anger. However the average accuracy is still 80,23% but this is not comparable to current state-of-the art algorithms for emotion detection.

Table 7: Emotion accuracy using 11 feature points obtained by using the Dlib shape predictor.

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	81,82%	0%	13,64%	4,54%	0%	0%	0%
Contempt	11,11%	88,88%	0%	0%	0%	0%	0%
Disgust	10,34%	0%	89,66%	0%	0%	0%	0%
Fear	0%	0%	0%	50%	8,33%	41,67%	0%
Happy	0%	0%	5,88%	0%	94,12%	0%	0%
Sadness	21,43%	7,14%	7,14%	7,14%	0%	57,14%	0%
Surprise	0%	0%	0%	0%	0%	0%	100%

4.1.2 19 feature points

While expressing emotions the mouth is one of the features that changes a lot and at the moment only 4 feature points are taken in consideration for monitoring the movement of the mouth. Therefore it could be interesting to look not only at the movement of those 4 feature points but take a closer look at the contraction of the mouth. So by measuring the circumference of the mouth in the neutral frame and in the subsequent frames it is possible to determine if the mouth contracts. This means that the input vector of the SVM is now extended from 11 features to 12 features. To extract the mouth contour an additional 8 facial feature points will be add which results in a total of 19 facial feature points. These 19 facial feature points are depicted in Figure 20.

Table 8 shows the confusion matrix of using these 19 feature points for extracting the emotions. By analysing this matrix a few remarks can be made. For example the accuracy of contempt increases with 11,12% and reaches an accuracy of 100%. The accuracy of fear is increased from 50% to 58,33%. Besides that there is also a small increase in the accuracy of happy but more important, the accuracy of sadness is has now an accuracy of 71,43% which results in an average accuracy of 85,47%.

These results confirm that the contraction of the mouth is an important feature that needs to be taken in consideration for classifying emotions.

Table 8: Emotion accuracy using 19 feature points obtained by using the Dlib shape predictor.

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	81,82%	0%	9,09%	4,54%	0%	4,54%	0%
Contempt	0%	100%	0%	0%	0%	0%	0%
Disgust	10,34%	0%	89,66%	0%	0%	0%	0%
Fear	0%	0%	0%	58,33%	8,33%	33,33%	0%
Happy	0%	0%	2,94%	0%	97,06%	0%	0%
Sadness	7,14%	7,14%	0%	14,29%	0%	71,43%	0%
Surprise	0%	0%	0%	0%	0%	0%	100%

4.1.3 19 feature points cascaded SVM(s)

After analysing the confusion matrix from Table 8, separate SVMs where trained for separating the emotion which should be detected and the emotion with which they are confused. The results of those experiments where that two additional SVMs for binary classification can be used. These SVMs are:

- **SVM fear/sadness**. This SVM separates images that expressing the emotion fear from images containing the emotion sadness. As can be seen in Table 8, the emotion fear is confused with sadness but sadness is also confused with fear. By implementing this SVM in cascade with the multi-class SVM the confusion from sadness with fear is reduced from 14,29% to 7,14%. However there is no reduction in confusion from fear with sadness.
- **SVM contempt/sadness**. This SVM separates images containing the emotion contempt and sadness. In this case sadness is confused with contempt however contempt is not confused with any other emotion. Using this SVM in cascade with the multi-class SVM reduces the confusion with contempt entirely.

Using this technique of cascaded SVMs, where the performance of a multi-class SVM is improved by cascading it with a SVM for binary classification, is a new technique developed in the area of computer vision during this thesis. This technique is not reported in current publications on the domain of emotion recognition and is therefore introduced in the area of computer vision during the research of this thesis.

The results of implementing the cascade of SVMs are shown in Table 9. As mentioned before the confusion with fear has dropped to half the confusion when not using the cascade of SVMs. Also the confusion of contempt has decreased to 0%. This results in an average accuracy of 87,51%

Table 9: Emotion accuracy using 19 feature points in a cascade of SVMs obtained by using the Dlib shape predictor.

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	81,82%	0%	9,09%	4,54%	0%	4,54%	0%
Contempt	0%	100%	0%	0%	0%	0%	0%
Disgust	10,34%	0%	89,66%	0%	0%	0%	0%
Fear	0%	0%	0%	58,33%	8,33%	33,33%	0%
Happy	0%	0%	2,94%	0%	97,06%	0%	0%
Sadness	7,14%	0%	0%	7,14%	0%	85,71%	0%
Surprise	0%	0%	0%	0%	0%	0%	100%

4.2 Shape predictor Dlib library (CK+)

After analysing the location of the facial feature points on the CK+ images it became clear that the feature points located around the mouth are not always located on the correct position. To overcome this problem an additional 500 images from the CK+ dataset were added to the training set of the shape predictor and retrained on this dataset. The new shape predictor is again trained by using the ensemble of regression trees algorithm.

This category is also divided into some subcategories. These subcategories are:

- Emotion prediction based on 19 feature points
- Emotion prediction based on 19 feature points with cascaded SVM(s)

In each category the SVMs are trained and tested on the CK+ dataset. The division of each sequence is the same as in the previous section.

4.2.1 19 feature points

Again 19 feature points are used to calculate the 12 distances and to classify the 7 basic emotions. The results, of using the retrained shape predictor for the extraction of the feature points, are presented in Table 10.

Table 10: Emotion accuracy using 19 feature points obtained by using the retrained shape predictor.

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	81,82%	9,09%	4,54%	0%	0%	4,54%	0%
Contempt	0%	100%	0%	0%	0%	0%	0%
Disgust	0%	0%	96,55%	0%	0%	3,45%	0%
Fear	0%	33,33%	0%	58,33%	8,33%	0%	0%
Happy	0%	0%	2,94%	3%	94,12%	0%	0%
Sadness	14,29%	14,29%	0%	7,14%	0%	64,29%	8,33%
Surprise	0%	0%	0%	0%	0%	0%	100%

The overall accuracy achieved with this method is 85,02% what is slightly less than using the same method but with the shape predictor of the Dlib library but when the technique of cascaded SVMs is applied the new shape predictor will outperform the shape predictor from the Dlib library. This will be discussed in the next section.

4.2.2 19 feature points cascaded SVM(s)

When the separate SVMs are tested to separate the emotion that needs to be detected from the emotions to which it is confused it is found that one additional SVM for binary classification can be used. This is the SVM contempt/fear.

- **SVM contempt/fear**. This SVM separates images that expressing the emotion contempt from images containing the emotion fear. As can be seen in Table 11, the emotion fear is confused with contempt but not the other way around. By implementing this SVM in cascade with the multi-class SVM the confusion from fear with contempt is reduced from 33,33% to 0%.

The result of this new technique, where a multi-class SVM is cascaded with a SVM for binary classification, are shown in Table 11. As mentioned before the confusion of fear with contempt is reduced from 33,33% to 0%. This results in an average accuracy of 89,78%.

Table 11: Emotion accuracy using 19 feature points in a cascade of SVMs obtained by using the retrained shape predictor.

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	81,82%	9,09%	4,54%	0%	0%	4,54%	0%
Contempt	0%	100%	0%	0%	0%	0%	0%
Disgust	0%	0%	96,55%	0%	0%	3,45%	0%
Fear	0%	0%	0%	91,67%	8,33%	0%	0%
Happy	0%	0%	2,94%	3%	94,12%	0%	0%
Sadness	14,29%	14,29%	0%	7,14%	0%	64,29%	8,33%
Surprise	0%	0%	0%	0%	0%	0%	100%

The approach described in this section is also used in the real-time emotion detection application described in the previous chapter. Now that the results of the final implementation are discussed is it time to compare these results with state-of-the-art emotion classification algorithms.

4.3 Comparison algorithms

The results of the final implementation used in this paper are now compared to several state-of-the-art emotion detection algorithms. These algorithms are compared with the algorithm used in this thesis based on two different aspects:

- Accuracy of emotion detection
- Speed of emotion detection

For this comparison the focus lays on two specific papers. These are:

- Paper 1: Learning Active Facial Patches for Expression Analysis [39]
- Paper 2: Image Ratio Features for Facial Expression Recognition Application [40]

Both of the papers are using the CK+ dataset for evaluating their emotion detection algorithm however they do not give any details about how many sequences are used for training and testing. Also the number of sequences used for every emotion is not mentioned. Therefore it is impossible to use exactly the same training and testing settings as used in the papers.

4.3.1 Accuracy of emotion detection

In this section the proposed method will be compared with [39] and [40] based on the accuracy of the detected emotions. The results of this comparison are depicted in Table 12.

Table 12: Accuracy comparison.

	Proposed method	Paper 1	Paper 2
Anger	81,82%	71,39%	**92,31%**
Contempt	**100%**	/	/
Disgust	**96,55%**	95,33%	93,62%
Fear	**91,67%**	81,11%	90,57%
Happy	94,12%	**95,42%**	84,62%
Sadness	64,29%	**88,01%**	86,05%
Surprise	**100%**	98,27%	90,24%
Neutral	/	/	90,74%
Average	**89,78%**	88,26%	89,74%

Table 12 indicates that the proposed method of this thesis outperforms the two papers on the average accuracy but also for the emotions contempt, disgust, fear and surprise. Paper 1 achieves the highest accuracies for the emotions happy and sadness. The highest accuracy for anger is reached by the algorithm of paper 2. However this comparison is not totally reliable because both papers do not classify the emotion contempt.

For the neutral emotion this is the case for the proposed method of this thesis and paper 1. So to make a fair comparison both emotions contempt and neutral will be left out. The result of leaving out those two emotions are shown in Table 13.

Table 13: Accuracy comparison when leaving out contempt and neutral.

	Proposed method	Paper 1	Paper 2
Anger	81,82%	71,39%	**92,31%**
Disgust	**96,55%**	95,33%	93,62%
Fear	**91,67%**	81,11%	90,57%
Happy	94,12%	**95,42%**	84,62%
Sadness	64,29%	**88,01%**	86,05%
Surprise	**100%**	98,27%	90,24%
Average	88,08%	88,26%	**89,57%**

Now the comparison between the proposed method and the methods described in paper 1 and 2 is more reliable. The proposed method is now just outperformed by the other two methods however it still detects 3 out of the 6 basic emotions better than the two papers. The reason why the other two papers outperform the proposed method is because the accuracy of sadness is very low compared to the other emotions. This results in the fact that the average accuracy is pulled down. So sadness is left out then the results are the same as in Table 14

Table 14: Accuracy comparison when leaving out contempt, sadness and neutral.

	Proposed method	Paper 1	Paper 2
Anger	81,82%	71,39%	**92,31%**
Disgust	**96,55%**	95,33%	93,62%
Fear	**91,67%**	81,11%	90,57%
Happy	94,12%	**95,42%**	84,62%
Surprise	**100%**	98,27%	90,24%
Average	**92,83%**	88,30%	90,27%

The results in Table 14 show that, when leaving out sadness, our proposed method outperforms the current state-of-the-art algorithms. Also has the proposed method the best accuracy for 3 out of 5 emotions listed in Table 14. Now that the proposed method is compared by looking at the accuracy it is time to compare them by looking at the speed of the algorithms.

4.3.2 Speed of emotion detection

The algorithm that is proposed in this thesis has a processing time of less than 30 ms. The processing time is defined as the time it takes for resizing a frame, detecting the face, extracting the facial feature points, calculate the displacement ratios and determine the emotion by using SVMs. Therefore the algorithm used in this thesis can perform at real-time by analyzing more than 30 fps.

Comparing this real-time performance with the other papers is not possible because the papers do not mention anything about real-time performance. Therefore it is assumed that the algorithms proposed in those papers do not have real-time performance. This means that the proposed method also outperforms the other algorithms on the area of operational speed.

This chapter describes different implementation details that are tested on their accuracy. These implementations are based on the shape predictor from the Dlib library and an self-trained shape predictor including additional images from the CK+ dataset. To the end of this chapter the final implementation is compared to two state-of-the art emotion detection algorithms based on their accuracy and speed.

Chapter 5

Conclusion

In this chapter a summary of the implementation and the performance of the proposed algorithm are discussed. To the end of this chapter, an overview of further suggestions for improvements is provided.

5.1 Implementation and performance

The first step in detecting emotions is finding the location of the face in an image. One of the objectives from this thesis is to achieve emotion detection in real-time therefore the face detection needs to happen with almost no latency. Therefore the HOG algorithm is used and implemented by using the Dlib library with a latency of less than 5 ms.

The extraction of the feature points is executed fully automatic as defined in the first objective of this thesis. To achieve this an ensemble of regression trees is used and implemented by using the Dlib library. The extraction of the features results in 19 facial feature points used for calculating 12 distances between the feature points and is done in less than 3 ms.

After the 12 distances are computed the displacement ratios are derived. The neutral frame is used as a reference for calculating these displacement ratios.

The corresponding 12 displacement ratios are the input vectors for the multi-class SVM but also for the cascaded SVM for binary classification. After the input vectors are given to the SVMs an emotion is given as output.

The processing time so resizing the frame, detecting the face, extracting the facial feature points, calculating the displacement ratios and determining the corresponding emotion is less than 30 ms. The real-time performance that is requested is thus achieved. At the processing time of 30 ms an average accuracy of 89,78% is achieved which is comparable to current state-of-the-art papers.

5.2 Further improvements

In this section some suggestions are made to increase the average accuracy of the current algorithm, both on the CK+ dataset as in the real-time emotion detection application.

The accuracy of sadness is very low compared to the other emotions. To improve this accuracy the extraction of the facial feature points can be improved. At the moment there is already an improvement compared to the shape predictor of the Dlib library but the location of the feature points is still not optimal.

Another option to improve the accuracy of sadness is a more detailed research towards the working of SVMs and how to find the optimal hyperplane for the emotion of sadness. Besides looking to the possibilities of SVMs, other machine-learning classifiers can be implemented to classify the emotions based on the proposed method.

In the real-time application the emotions happy, surprise and anger can be detected well. However the other emotions are hard to detect. The reason for that is that the feature points are calculated on every frame and due to environmental noise and change in illumination the feature points are not located stable. To overcome this problem a more detailed research can be done on stabilizing these feature points.

The last suggestion for improvement is that the user of the real-time application is able to move in a 3-dimensional space. At the moment it is not possible to move in a 3-dimensional space because the neutral frame that is used as a reference for calculating the distances is not scaled. Therefore as soon as the neutral frame is picked in a 3-dimensional space the user is restricted to move in a 2-dimensional plane to make sure that the correct emotion is extracted. This problem can be investigated in further research.

In this chapter a short overview is given of the final implementation that is used for the emotion detection together with the results regarding the accuracy and the speed of the proposed algorithm. Towards the end of this chapter some suggestions are made for further improvement.

References

[1] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05),* vol. 1, pp. 886-893, 2005.

[2] S. K. Paul and M. S. Uddin, "Extraction of Facial Feature Points Using Cumulative Distribution Function by Varying Single Threshold Group," in *2012 International Conference on Informatics, Electronics & Vision (ICIEV)*, Dhaka, 2012.

[3] W. Zhao, J.-S. Park and S.-W. Lee, "Fully automatic face detection and facial feature points extraction using local Gabor filter bank and PCA," *2011 International Conference on Machine Learning and Cybernetics (ICMLC),* vol. 4, pp. 1789-1792, 2011.

[4] S. Zhu and J. Zhao, "Facial Feature Points Extraction," *Fifth International Conference on Image and Graphics, ICIG '09,* pp. 195-199, 2009.

[5] Y. Kawarazaki, G. Duan, T. Shinkawa and Y.-W. Chen, "Viewpoint-specific active appearance model for robust feature point extraction," in *6th International Conference on Computer Sciences and Convergence Information Technology (ICCIT)*, Seogwipo, 2011.

[6] J. Sullivan and V. Kazemi, "One Millisecond Face Alignment with an Ensemble of Regression Trees," in *Conference on Computer Vision and Pattern Recognition*, Columbus, Ohio, 2014.

[7] A. S. M. Sohail and P. Bhattacharya, "Classifying Facial Expressions Using Point-Based Analytic Face Model and Support Vector Machines," *2007 IEEE International Conference on Systems, Man and Cybernetics,* pp. 1008-1013, 2007.

[8] D. E. King, "Dlib-ml: A Machine Learning Toolkit," *Journal of Machine Learning Research,* no. 10, pp. 1755-1758, 2009.

[9] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar and I. Matthews, "The Extended Cohn-Kanade Dataset (CK+): A complete expression dataset for action unit and emotion-specified expression," *Proceedings of the Third International Workshop on CVPR for Human Communicative Behavior Analysis (CVPR4HB 2010),* pp. 94-101, 2010.

[10] T. Kanade, J. F. Cohn and Y. Tian, "Comprehensive database for facial expression analysis," *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition (FG'00),* pp. 46-53, 2000.

[11] M. J. Lyons, S. Akemastu, M. Kamachi and J. Gyoba, "Coding Facial Expressions with Gabor Wavelets," in *Third IEEE International Conference on Automatic Face and Gesture Recognition*, Nara, 1998.

[12] M. Pantic, M. Valstar, R. Rademaker and L. Maat, "Web-based database for facial expression analysis," in *2005 IEEE Conference on Multimedia and Expo*, 2005.

[13] D. M. W. Powers, "Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation," School of Informatics and Engineering Flinders University of South Australia, PO Box 2100, Adelaide 5001, South Australia, 2007.

[14] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters,* vol. 8, no. 27, pp. 861-874, 2006.

[15] M. Turk and A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience,* vol. 3, no. 1, pp. 71-86, 1991.

[16] K. Etemad and R. Chellappa, "Discriminant Analysis for Recognition of Human Face Images," *Journal of the Optical Society of America A,* vol. 14, no. 8, pp. 1724-1733, 1997.

[17] T. Ojala, M. Pietikhenl and D. Harwood, "Performance evaluation of texture measures with classification based on Kullback discrimination of distributions," *Proceedings of the 12th IAPR International Conference on Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision & Image Processing,* vol. 1, pp. 582-585, 1994.

[18] T. Ojala, M. Pietikäinen and D. Harwood, "A Comparative Study of Texture Measures with Classification Based on Featured Distribution," *Pattern Recognition,* vol. 29, no. 1, pp. 51-59, 1996.

[19] D. G. Lowe, "Object Recogition from Local Scale-Invariant Features," *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999,* vol. 2, pp. 1150-1157, 1999.

[20] P. Viola and M. Jones, "Robust Real-time Object Detection," in *Second International Workshop on Statistical and Computational Theories of Vision - Modeling, Learning, Computing, and Sampling*, Vancouver, Canada, 2001.

[21] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision,* vol. 60, no. 2, pp. 91-110, 2004.

[22] Y. Freund and R. E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *Journal Of Computer And System Sciences,* vol. 55, no. 1, pp. 119-139, 1997.

[23] P. Viola and J. Michael, "Rapid Object Detection using a Boosted Cascade of Simple Features," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2001,* vol. 1, pp. I-511 - I-518, 2001.

[24] A. Huamán, "Feature Matching with FLANN," [Online]. Available: http://docs.opencv.org/2.4/doc/tutorials/features2d/feature_flann_matcher/feature_flann_matcher.html#feature-flann-matcher. [Accessed 18 March 2016].

[25] T. Hastie, R. Tibshirani and J. H. Friedman, The elements of statistical learning: data mining, inference, and prediction, New York: Springer-Verlag, 2001.

[26] V. N. Vapnik, The Nature of Statistical Learning Theory, New York: Springer-Verlag, 2000.

[27] A. Kowalczyk, "SVM - Understanding the math : the optimal hyperplane," SVM Tutorial, 8 June 2015. [Online]. Available: http://www.svm-tutorial.com/2015/06/svm-understanding-math-part-3/. [Accessed 13 June 2016].

[28] S. R. Gunn, "Support Vector Machines for Classification and Regression," University of Southampton, Southampton, 1998.

[29] M. Hofmann, "Support Vector Machines - Kernels and the Kernel Trick," University of Bamberg, Bamberg, 2006.

[30] C. J. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery 2,* vol. 2, no. 2, pp. 121-167, 1998.

[31] P. F. Felzenszwalb, R. B. Girshick, D. McAllester and D. Ramanan, "Object Detection with Discriminatively Trained Part-Based Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence ,* vol. 32, no. 9, pp. 1627-1645, 2010.

[32] M. Gruendl, "Average faces," 1 July 2002. [Online]. Available: http://www.uni-regensburg.de/Fakultaeten/phil_Fak_II/Psychologie/Psy_II/beautycheck/english/durchschnittsgesichter/durchschnittsgesichter.htm. [Accessed 21 March 2016].

[33] C. Sagonas, E. Antonakos, G. Tzimiropoulos, S. Zafeiriou and M. Pantic, "300 faces in-the-wild challenge: Database and results," *Image and Vision Computing,* vol. 47, pp. 3-18, 2016.

[34] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou and Pantic Maja, "A semi-automatic methodology for facial landmark annotation," in *2013 IEEE Conference on Computer Visison and Pattern Recognition Workshops*, Oregon, USA, 2013.

[35] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou and M. Pantic, "300 Faces in-the-Wild Challenge: The first facial landmark localization Challenge," in *2013 IEEE International Conference on Computer Vision Workshops (ICCVW)*, Sydney, Australia, 2013.

[36] C.-C. a. L. C.-J. Chang, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology,* vol. 2, no. 3, pp. 27:1--27:27, 2011.

[37] O. d. team, "Support Vector Machines," OpenCV, 16 June 2016. [Online]. Available: http://docs.opencv.org/2.4/modules/ml/doc/support_vector_machines.html#cvsvm-train. [Accessed 16 June 2016].

[38] S. L. Happy and A. Routray, "Automatic Facial Expression Recognition Using Features of Salient Facial Patches," *IEEE Transactions on Affective Computing,* vol. 6, no. 1, pp. 1-12, 2015.

[39] L. Zhong, Q. Liu, P. Yang, B. Liu, J. Huang and D. N. Metaxas, "Learning Active Facial Patches for Expression Analysis," *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 2562-2569, 2012.

[40] M. Song, D. Tao, Z. Liu, X. Li and M. Zhou, "Image Ratio Features for Facial Expression Recognition Application," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics),* vol. 40, no. 3, pp. 779-788, 2010.