Facial Expression Recognition using Local Binary Patterns

with classification based on Support Vector Machines

AALBORG UNIVERSITY
Department of Electronic Systems
Vision, Graphics and Interactive Systems
9th Semester project

Autumn 2012

Maxime Coupez
Kim-Adeline Miguel
Julia Alexandra Vigo

Department of Electronic Systems
Fredrik Bajers Vej 7
DK-9220 Aalborg Ø
http://es.aau.dk

Title:
Project Title

Theme:
Interactive Systems

Project Period:
Fall Semester 2012

Project Group:
12gr942

Participant(s):
Maxime Coupez
Kim-Adeline Miguel
Julia Alexandra Vigo

Supervisor(s):
Zheng-Hua Tan

Copies: 1

Page Numbers: 61

Date of Completion:
November 29, 2012

Abstract:

Since the last decade, a lot of researches have been carried out about emotion recognition. The number of projects conducted in this field demonstrates the interest and the importance of systems which can recognize human mood.
In this project, an emotion recognition system is developed, using a Microsoft Kinect. This recognition is achieved in 3 steps: Face detection, extraction and classification of facial features, this structure being the usual modus operandi in emotion recognition research.
Face detection is performed using Viola-Jones' algorithm, then Local Binary Patterns (LBP) are used to extract facial features. Finally, Support Vector Machines (SVM) classify these features into six predefined emotions.
The system is implemented to run on a computer using a Kinect and works for one person in front of it. The classifier is trained with the Karolinska Directed Emotional Faces database, which includes enough different faces to obtain a satisfying result.

Preface

This report documents the semester project entitled *Facial expression recognition using Local Binary Patterns*. The project was carried out during the 9th semester of specialization *Vision, Graphics, and Interactive Systems* under the Department of Electronic Systems at Aalborg University in Autumn 2012.

The report is divided into four parts plus appendices: *Introduction*, *Feature Detection*, *Feature Classification*, *Implementation* and *Evaluation*. The first part review the general structure of a facial expression recognition system and its main issues, and concludes with a state of the art of existing systems. Analysis of possible solutions and design of our system are contained in the following two parts, and the fourth part describes our implementation. The last part evaluates the performance and accuracy of our system and concludes on the project as a whole.

References to secondary literature sources are made using the syntax [number]. The number refers to the alphabetically sorted bibliography found at the end of the report, just before the appendices.

We would like to thank our supervisor at Aalborg University Zheng-Hua Tan for supporting us in this challenging project.

A CD is attached to this report which includes:

- Source code of the developed program.
- PDF file of this report.

Aalborg University, November 29, 2012

Maxime Coupez
<mcoupe12@es.aau.dk>

Kim-Adeline Miguel
<kmigue12@es.aau.dk>

Julia Alexandra Vigo
<jvigo12@es.aau.dk>

Contents

Part I

Introduction

Contents

The main motive of this project is to understand facial expression recognition systems and their applications. A review of the architecture of such systems will be done, along with a state of the art of already existing algorithms. After this study, issues coming along with this kind of recognition system will be studied. In the last part, the requirements of this project will be formulated.

Chapter 1

Motivations

A facial expression is a "visible manifestation of the effective state, cognitive activity, intent, personality, and psychopathology of a person" [8]; facial expressions represent a huge part in dialogue and interaction with other humans. Indeed, facial expressions carry more informations than speech, informations on which humans can relay for interaction. Facial expressions have a considerable effect on a listening interlocutor; in a conversation, it represents 55 percent of information received by a listener, while 38 percent are conveyed by voice intonation and the remaining 7 percent by the spoken words [21].

Since Antiquity, researchers have been interested in emotion and more particularly in emotion recognition. One of the most important studies on facial expression analysis impacting on modern day science of automatic facial expression recognition is the work carried out by Charles Darwin [4]. In 1872, Darwin wrote a book that established general expression principles, expression means and expression description for both humans and animals [6]. He also classified various kinds of expressions. This can be considered as the beginning of facial expression recognition.

Nowadays, with the emergence of new technologies and computers, research is now focused on computer-based automatic facial expression recognition. Because facial expressions are major factors in human interaction, this research field will improve the domain of Human-Machine Interaction. Indeed, emotion recognition will enable computers to be more responsive to users' emotions, and allow interactions to become more and more realistic.

Another domain where facial expression recognition is an important issue is robotics. With the advances made in robotics, robots tend to mimic human emotion and react as as human-like as possible, especially for humanoid robots. Indeed, since robots are being more and more present in our daily lives, they need to understand and recognize human emotions.

A lot of applications in the robotics field have already emerged. For example, Bartlett et al. have successfully used their face expression recognition system to develop an animated character mirroring the expressions of the user (called CU Animate) [3]. They have also been successful in deploying the recognition system on Sony's Aibo Robot and ATR's RoboVie [3]. Another interesting application has been demonstrated by Anderson and McOwen, called "EmotiChat" [2]. It is a regular chatroom, except the fact that their facial expression recognition system is connected to the chat and convert the users' facial expressions into emoticons. Because facial expression recognition systems' robustness and reliability are constantly increasing, lots of innovative applications will appear.

There are also various other domains where emotion recognition can be used: Telecommunications, behavioural science, video games, animations, psychiatry, automobile safety, affect-sensitive music jukeboxes and televisions, educational software, etc [4].

This project focuses on facial expression recognition from a video stream. Indeed, facial expression recognition can be performed *statically* on input images, or *dynamically* on video sequences. Systems can also be *obtrusive*, or *non-obtrusive*, the former based on a device mounted on the user's head or body, therefore following each of his movements and perform facial expression recognition without much losses, while the latter can encounter difficulties if the user is not properly situated. However, non-obtrusive systems allow more natural user interactions. We chose our system to be non-obtrusive, and will detail its setup further in the next section.

1.1 Environment Setup

Our system will use the camera embedded into a Microsoft Kinect to record the user's video input. We will consider a casual use of the camera, the user sitting in front of the computer, the camera being next to it, as seen in **Insert picture of the setting & ref to figure**. This camera provides a 640×480 pixels frame resolution, while recording at 30 FPS.

For development and training purposes we will use some pre-existing emotion datasets, in order to validate the efficiency of the system before testing it in real conditions.

1.2 Facial Expression Datasets

Databases are very important for facial expression recognition system.

Using the same databases as in previous studies allows performance and accuracy comparisons between new implementations and previously obtained results. Since most studies draw their results on the same databases, it is then relatively easy to compare them and choose a database suitable for our system.

Databases are however difficult to build. Indeed, it has to be obtained following a meticulous procedure while being exhaustive so it can be considered as representative. The majority of actual databases use posed expressions rather than spontaneous ones, this choice having a major influence on facial expression recognition systems. This explains why some databases are updated, and now integrate spontaneous expressions. Even with this transition from posed expressions to spontaneous expressions, there are other requirements that should be met to have a standardized database.Its content should be of different resolutions and scales, and should also contain exposition under different conditions, i.e changes in lightning, occlusions or different head angles [4].

Constructing a database is then a tedious task because of all these requirements to meet. Consequently, most studies are based on already existing datasets. The 3 datasets described afterwards are popular and freely available facial expression datasets which have been used a numerous amount of times in the past few years. Our system will then be trained and tested with one or several of these databases.

1.2.1 Japanese Female Facial Expression Database (JAFFE)

This database contains 213 images of 7 facial expressions (6 basic facial expressions: happy, angry, afraid, disgusted, sad, surprised, and 1 neutral facial expression). Each expression has been photographed three or four times. Each image has been rated on 6 emotion adjectives by 60 Japanese subjects. All images come from 10 Japanese female models. The database was planned and assembled by Miyuki Kamachi, Michael Lyons, and Jiro Gyoba [17].

This database contains only posed expressions. The photos have been taken under strict and controlled conditions: similar lighting and hair tied so there is no facial occlusion [4].

An example of images contained in the database is given by Figure 1.1. In this figure, this is a female subject displaying 7 different emotional expressions (neutral, happy, angry, afraid, disgusted, sad, surprised).

1.2.2 Karolinska Directed Emotional Faces Database (KDEF)

The Karolinska Directed Emotional Faces (KDEF) contains 4900 pictures of human facial expressions. The material was developed in 1998 by Daniel Lundqvist, Anders Flykt and Professor Arne Ohman at Karolinska Institutet, Department of Clinical

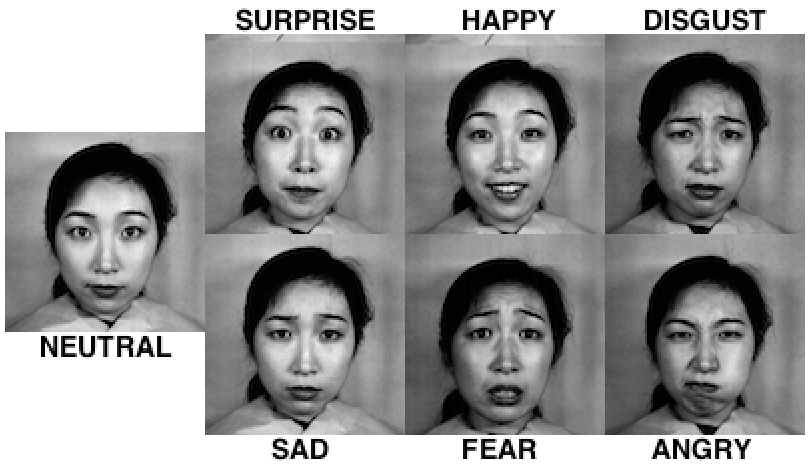


Figure 1.1: Example of images from JAFFE database

Neuroscience, Section of Psychology, Stockholm, Sweden [14].

The database was first developed for psychological and medical research purposes. It was created so it could used in perception, attention, emotion, memory and backward masking experiments. This database has also been built under controlled conditions. Indeed, researchers tried to maintain constant and soft lighting for each subject. Furthermore, they made their subjects wear the same grey T-shirts, and used a grid to center the participants faces while they were shot. This grid was also used to place eyes and mouths at the same position in fixed image coordinates during scanning [14].

The database contains 70 individuals (35 males and 35 females), from 20 to 30 years, each one displaying 7 different emotional expressions (neutral, happy, angry, afraid, disgusted, sad, surprised). Each expression has been photographed (twice) from 5 different angles (-90, -45, 0, +45, +90 degrees: i.e. full left profile, half left profile, straight, half right profile, full right profile) [14].

An example of images contained in the database is given in Figure 1.2which represents a female subject photographed from a straight angle and displaying 7 different emotional expressions (neutral, happy, angry, afraid, disgusted, sad, surprised).

1.2.3 Montreal Set of Facial Displays of Emotion Database (MSFDE)

This database contains facial expressions of European, Asian, and African subjects, from both genders. Each expression was created by directly asking the subject to

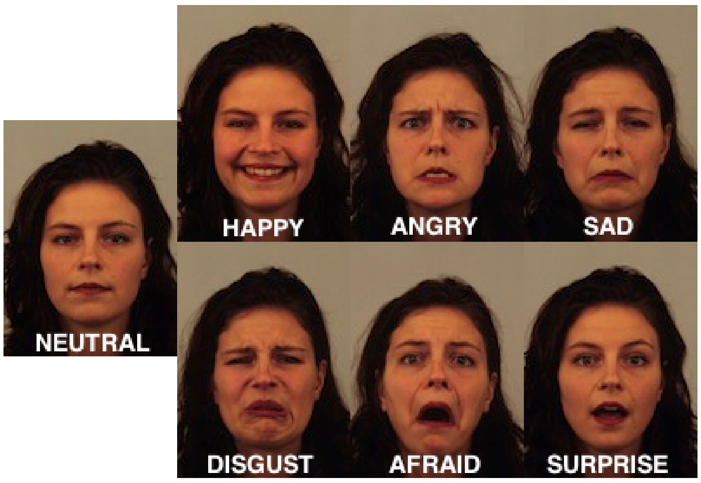


Figure 1.2: Example of images from KDEF database

express this emotion [27].

The database contains expressions of happiness, sadness, anger, fear, disgust, and embarrassment, along with a neutral facial expression. All expressions have been photographed at 5 different levels of intensity [27].

An example of images contained in the database is given in Figure 1.3, where an African female subject displays 7 different emotional expressions (neutral, happy, angry, afraid, disgusted, sad, ashamed).

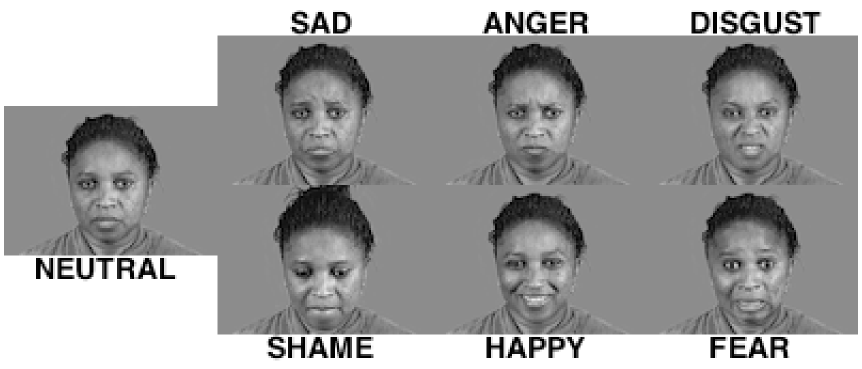


Figure 1.3: Example of images from MSDFE database

Chapter 2

Facial expression recognition

After having stated the conditions and motivations of this project, we will now describe the general structure a a facial expression recognition system. It can indeed be roughly summed up as classification applied to a pre-processed image.An overview of pre-processing steps will be done in this chapter, while feature extraction and classification will be more detailed respectively in Parts 5 and 7. Following sections will be about issues raised by facial expression recognition systems, and key requirements these systems have to meet in order to be considered acceptable.

2.1 General structure

Facial expression recognition is a system enabling an automatic recognition of emotions displayed by a human face. Facial expression recognition can be image or video-based; it can also be computed in real-time if needed. Researchers usually try to recognize emotions out of static images. It can also be achieved real-time on video streams : While the person displays his/her emotions, the facial expression recognition system analyses the video, and detect the displayed emotion.

In both cases, facial expression recognition process is structured as in Figure 2.1

2.1.1 Image Acquisition

The first step is "Image Acquisition". Images used for facial expression recognition can be static images or image sequences, with the latter giving more informations about the displayed expression, i.e steps in muscles movement. About static images, facial expression recognition systems usually take 2D greyscale images as inputs. We can however expect future systems to use color images; first because of the increasing affordability of technologies and devices capable of capturing images or image sequences; then because colors can give more information on emotions, for example blushing [5].

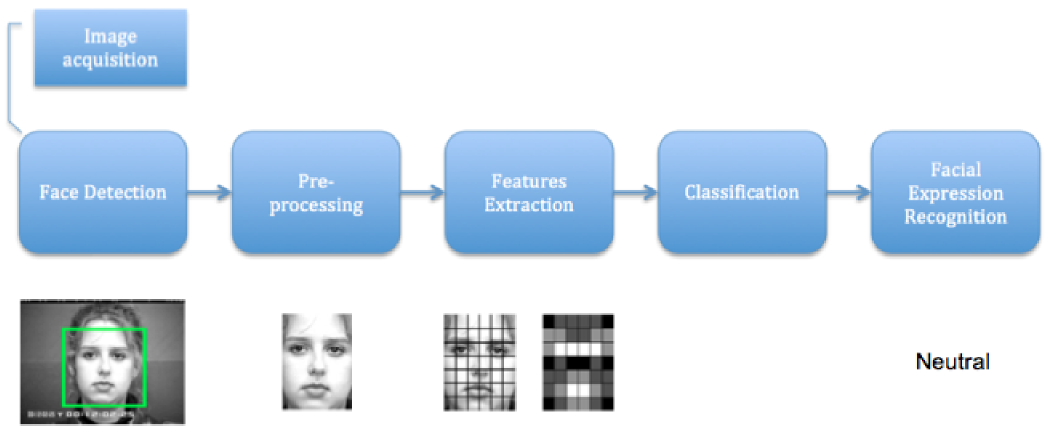


Figure 2.1: Facial Expression Recognition process

2.1.2 Face Detection

Second step is "Face Detection". Indeed, in a static image and even more in an images sequence, this is an obvious need. Once the face has been detected, all other non-relevant information can be deleted. This step could hence be included in the next step, which is "Pre-processing", but because of its importance it can be considered as a step in itself. If working with image sequences or video streams, the face has to be detected and tracked. One of the most used and famous detection and tracking algorithm is the Viola-Jones algorithm, which will be explained further in Chapter 4. This algorithm can be trained to detect all kind of objects, but is mostly used for face detection.

2.1.3 Pre-processing

Third step is "Pre-processing", where image processing algorithms are applied to the image in order to prepare it for the next step. Pre-processing is usually about noise removal, normalization against the variation of pixel position or brightness, segmentation, location or tracking of parts of the face. Transformation, scaling and rotation of the head in the image or image sequence have an effect on emotion recognition. In order to solve this problem, the image can be geometrically standardized, with the eyes generally used as reference points [5].

2.1.4 Feature Extraction

Once the image has gone through the "Pre-processing" step, the next one is "Feature Extraction". In this step, data is converted "into a higher representation of shape, motion, color, texture, and spatial configuration of the face or its components" [5]. One of the main goals of this step is to reduce the dimensionality of the input data. The reduction procedure should retain "essential information possessing high discrimination power and high stability" [5]. There are a lot of features extraction methods. The most famous are : Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Problem Based Learning (PBL), Hidden Markov Models (HMM), Eigenfaces, Gabor Wavelets. This step will be detailed further in Part 5. The extracted data is then used in the "Classification" step.

2.1.5 Classification

The classification step is the culminating point of the facial expression recognition process. There are many kinds of classification algorithms, some of them can even be used in the feature extraction part, as it will be detailed in Part 5. This step takes into input a model previously trained with pre-processed data, and test data made of feature vectors extracted from the image we want to label. Feature vectors from pre-processed data and test data have to be obtained using the same feature extraction algorithm. The chosen classifier then outputs a value corresponding to the label of the class the picture belongs to.

2.2 Issues

2.2.1 Datasets

Databases can be a source of issues. As said previously, databases should meet a number of requirements in order to be as exhaustive and efficient as possible.

In order to build a facial expression recognition system efficient in live conditions,, it should be able to recognize spontaneous expressions rather than posed expressions. Indeed, spontaneous expressions are closer to reality than posed expressions, the latter being exaggerated to facilitate their labelling and recognition. While creating a database of spontaneous expressions, Sebe and al [24] made some observations of the major problems they encountered [4]:

- The same emotions can be expressed at different intensities by different subjects;

- As soon as the subject is aware of being photographed and studied, the authenticity of the emotion is lost;
- Because of the laboratory conditions, even if the subject is not aware of being photographed or recorded, the subject is not encouraged to display spontaneous expressions.

In order to overcome these problems, they came up with a method. Their solution was to record facial expressions with a camera hidden in a video kiosk displaying emotion inducing videos. Subjects were notified of the recording after it was done, and were asked a permission to use recorded sequences for research studies. The subjects then explained which emotions they felt and expressed, their replies being documented even if it did not match the recorded expressions [24].

The researches found that a wide range of expressions are hard to induce, particularly fear and sadness. They also found that spontaneous expressions could be misleading: some subjects express one emotion while feeling another one (for example, one subject was showing sadness while being happy) [24].

In a nutshell, databases bring some issues that can affect the authenticity of the recognition system. It depends of the type of expressions: spontaneous or posed expressions. If the system aims to recognize facial expressions of people unaware of it, spontaneous expressions databases will be used but, as seen previously, it can lead to authenticity issues. If the system aims to recognize facial expressions of people asked to express certain emotion, posed expressions databases will be used, but the result will not be close to reality.

2.2.2 Real-time

The primary goal of the facial expression recognition system described in this paper is to perform real-time facial expression recognition. As a real-time application, the processing time should be taken into account. Indeed, if it is too long, the system will not be responsive enough to be labelled as real-time.

This is one of the challenges of this kind of system, because processing is usually really heavy no matter which algorithm is used. Most facial expression recognition applications should be working in real-time conditions, for example in robotics or in surveillance. A solution could be to find new algorithms for Facial Expression Recognition or to improve and lighten already existing algorithms.

2.2.3 Conditions

Another one of the challenges of this kind of system is to be independent of recording conditions. It means that the recognition should not be disturbed for example by occlusions, or difference in the lighting, or even by the angle between the face and the camera. These examples cover almost all conditions that can change during the recording, and have an influence on the recognition system.

Occlusion

"Occlusion" defines all elements covering the face, partly or entirely. For example, a beard, a scarf masking the bottom of the face, glasses or bangs. By hiding a part of the face, these occlusions can affect the recognition. Indeed, facial expression recognition systems are based on comparison of features, and if all the features cannot be compared because something is covering a part of the face, the recognition accuracy is affected. In order to compensate for this problem, some databases includes data with occluded faces, for example with beards, glasses or scarves. This is the case for the AR Face database. Some examples of images contained in this database are given by Figures 2.2 and 2.3 [18].

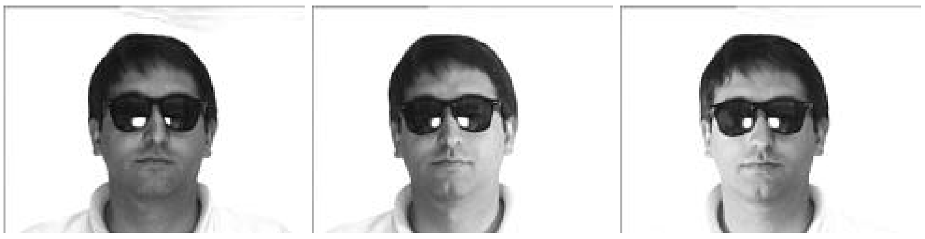


Figure 2.2: Example eye occlusion in the AR Face database

Lighting

As for occlusion, lighting is an element that can affect recognition effectiveness. With different lighting conditions from those in the database, recognition will be less efficient. All conditions different from those recorded in the database on which the recognition process is based will impact the system. If images are brighter or darker than reference data, some details can disappear, or some features will not be recognized as well as if the conditions were identical. In order to compensate for

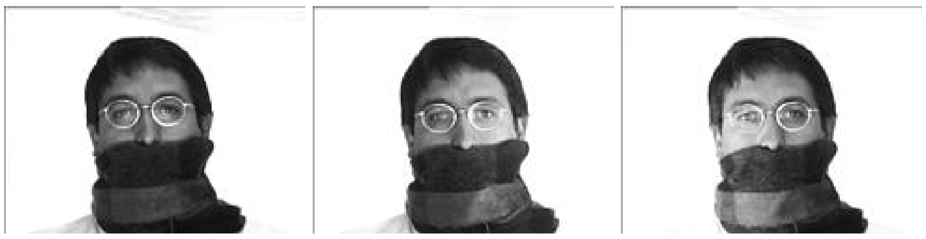


Figure 2.3: Example of face occlusion in the AR Face database

this problem, databases can include data with different lighting conditions. This is the case for the AR Face database and an example of the images contained in this database is given by Figure 2.4 [18]:

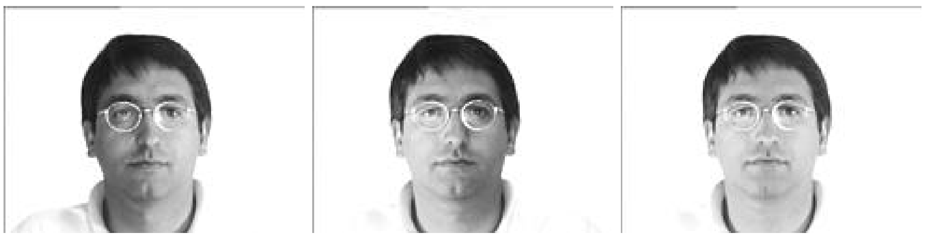


Figure 2.4: Example of different lighting conditions (from left to right: dark to bright) in the AR Face database

Angle

Head angle is one of the most impacting conditions during recognition. Indeed, some features can be occluded or disappear if the head does not properly face the camera. This issue also rises if the system is not tuned to work with head angles other than straight profile. For example, with a profile angle, one eye, half of the nose and of the mouth disappear. If the database does not contain samples with profile faces, the recognition will fail. However, if the database contains images from straight angle as well as from different profile angles, recognition will be possible. An example of different images taken from different angles for one emotion, "Fear", from the KDEF database is given in Figure 2.5:

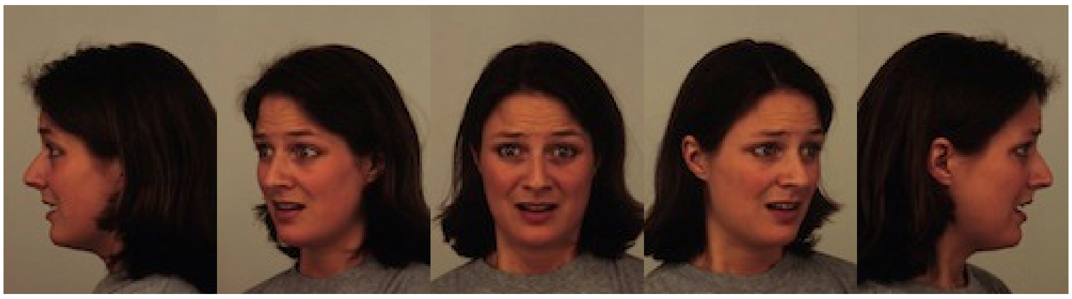


Figure 2.5: Example of different angles for the "Fear" emotion (from left to right: full left profile, half left profile, straight, half right profile, full right profile) in the KDEF database

2.3 Requirements

Based on everything said previously, our facial expression recognition system can be defined by some requirements prior to its implementation. Additional requirements may be defined further. Here are the requirements already defined :

- Able to recognize basic emotions : As explained before, facial expression recognition systems are able to recognize 6 basic emotions and the neutral state. This system should be able to do the same: recognize the 6 basic emotion that are "Happiness", "Fear", "Surprise", "Disgust", "Sadness", "Anger" and the neutral state.

- Able to work in real-time : This system should be able to recognize facial expressions in real-time. It means that it can recognize expressions based on a video sequences. It also means that the algorithm for the feature extraction has to be carefully optimized so it is able to compute facial features in a decent amount of time.

- Recognition from straight angle of the face : This system should be able to recognize facial expression from a straight angle of the face. It means that the system should be able to detect faces in front of the camera lens, and recognize expressions in these faces. It might not be able to recognize emotions on a face on a profile or half-profile angle.

- Recognition with no occlusion : This system should be able to recognize emotions with no occlusion on the subject's face. It means that the face should not be covered in any way: no glasses, no beard or no scarf. The face should also be complete, not cut and not masked.

- Recognition with no changes in lighting : This system should be able to recognize emotions under constant lighting conditions during the recognition process. Moreover, the intensity level of the light should be as close as possible to the one of the database. This way the lighting would not have any influence on the recognition process.

Part II

Feature detection

Contents

Before getting to the main part of this project which is Feature extraction, there is a mandatory step that is Feature detection. In order to avoid as much computation and processing as possible, only parts containing regions of interest for a Facial Expression Recognition system have to be processed. It consequently means that detection of interesting features has to be performed beforehand, which is the goal of face detection. This part will explain how face detection works in general. We will then introduce the Viola-Jones algorithm, a performant and cost-effective method for face detection and tracking.

Chapter 3

Face detection

Face detection is the first step after image acquisition. It represents a requirement for a Facial Expression Recognition system. All the background is not taken into account. It allows to focus only on what is interesting in the input image: the face, which helps reducing the processing during the next step that is feature extraction.

3.1 Detection

Finding out if the input image or video sequence represents or contains a particular object is what is called Detection. Usually after the detection step comes the recognition step. For this system, the recognition step consists in facial expression recognition. But depending on the recognition, there can be another step that is the tracking step. Tracking consists in following a moving target on the images of a video sequence [7].

To represent an object detector, the term "black box" can be used. The box gets an image as its input and at a high level the output can be considered as an image with annotations saying where the object of interest appears, if it appears [7]. For example, the output can look like in figure 3.1 [7].

But at a low level, the output is not anymore an annotated image. The object detector has a basic component that is something required to say if an instance of the object of interest is contained in a certain region or sub-region of the original image or not. This is what a binary classifier does [7]. For example, what a binary classifier does can look like in figure 3.2 [7].

3.2 Classifiers

Classification aim to solve the problem of identifying in a set of categories or sub-populations to which a new observation belongs. It is based on a training set of data that contains instances whose category affiliations are known. A classifier is an al-

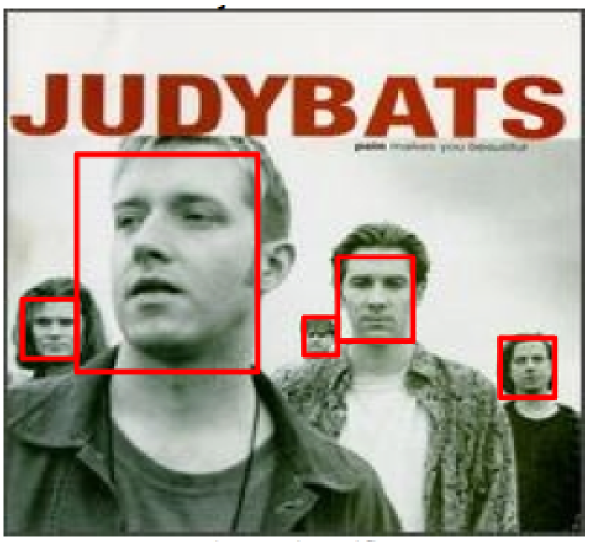


Figure 3.1: Example of an output of face detection

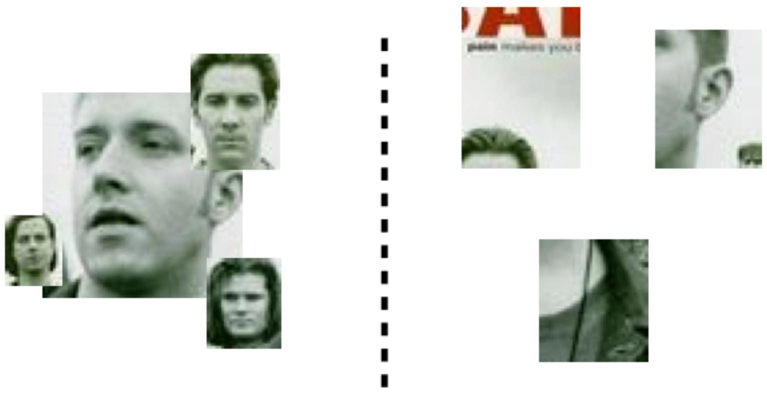


Figure 3.2: Example of what does a binary classifier for face detection

gorithm that implements classification. Classifiers groups data into categories. This can be done based on some measures of inherent similarity; for example, vectors represent the distance between instances, ad this in a multidimensional vector space [29].

Chapter 4

Viola-Jones

Viola-Jones algorithm is "a visual object detection framework that is capable of processing images extremely rapidly while achieving high detection rates". 3 main points characterize this algorithm. The first one is the use of what is called an "Integral image". It is a new representation of the image and it allows the features used to be computed very quickly. The second one is the use of a learning algorithm based on AdaBoost. It results in giving extremely efficient classifiers. The third and last one is the use of a method to combine classifiers. This method combine classifiers in "cascade". It allows to focus on the promising object-like regions by discarding the background in a very quick way [28].

4.1 Overview

The Viola-Jones algorithm works as following [7]:

- "The Viola-Jones detector is a strong, binary classifier build of several weak detectors"
- "Each weak detector is a simple binary classifier"
- During the learning part, a cascade of weak classifiers is used and trained in order to attained the desired hit/miss rate using the learning algorithm based on AdaBoost
- The input image is divided in several rectangular sub-regions in order to detect objects. The cascade computes each of these sub-regions
- To classify a sub-region as "positive", it has to go through all of the stages of the cascade
- All the process involving the sub-regions is repeated at different scales

4.2 Haar features

The features used by Viola and Jones are called Haar features and are based on Haar wavelets. Haar wavelets are single wavelength square waves. It is composed of one high interval and one low interval. In two dimensions, a square wave is represented by a pair of adjacent rectangles: one rectangle that is light and one rectangle that is dark. The true Haar wavelets are not the one used for this rectangle combinations that is used for visual object detection. They use instead rectangle combinations that are better suited to recognition tasks. That is because of this difference that these features are called Haar features rather than Haar wavelets (they can also be called Haarlike features) [11].

For example, the figure 4.1 shows the first two Haar features in the original Viola-Jones cascade [11]. These are all features from the original set of features. In figure 4.2, there is an example of features from the extended set of features [7]. In the figure 4.3 , it is an example of a early stage in the Haar cascade. Each black and white rectangle represents a feature. And that is what the algorithm hunts for in the image [10].

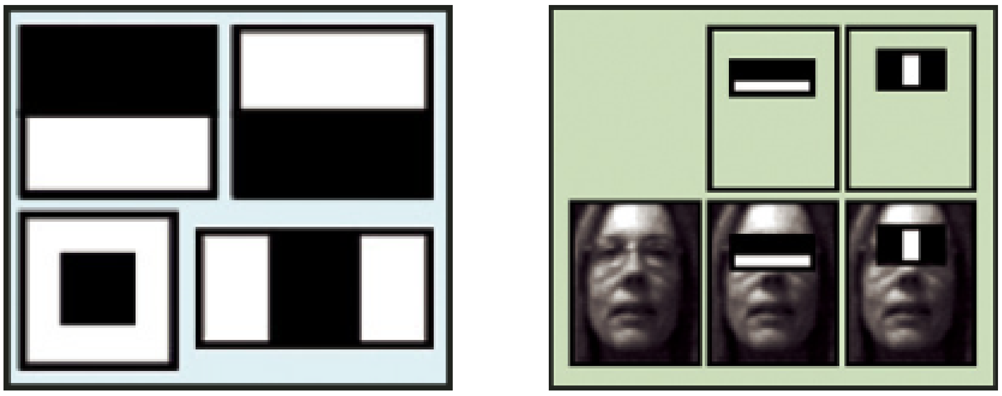


Figure 4.1: Example of the first two Haar features

To detect if a Haar feature is present or not, basic subtraction is used. The subtraction consists in subtracting the average pixel value of the dark-region from the average pixel value of the light-region. Then it is a simple comparison. The result of the subtraction is compared with a threshold. If the result is above the threshold, then the feature is considered as present and it can go to the next stage [11]. There is about 20 to 30 different stages to detect the presence of Haar features. The first stage is a very coarse scan of the image. The second stage is more detailed, the third

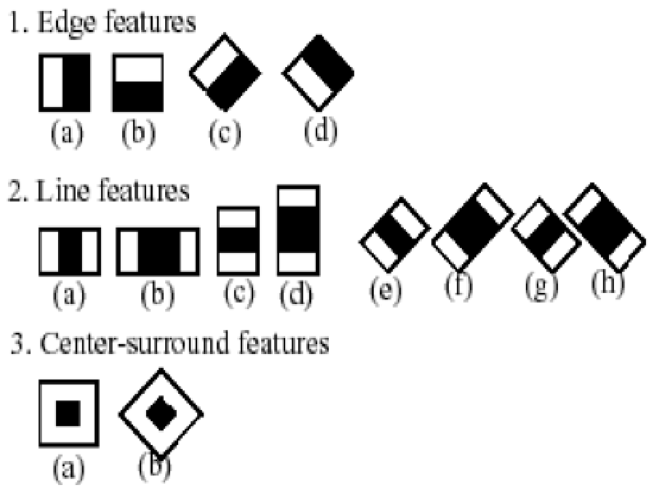


Figure 4.2: Extended set of features

stage is once again more detailed and harder to pass, the fourth stage is even harder and it goes on and on till the end. The more it moves froward into the cascade, the more it becomes harder. The features become increasingly complex and larger. All this processing takes indeed more time to compute [10].

For example, the figure 4.4 shows the later stage in the Haar cascade. There are many more patterns of black and white rectangles that need to match the input image [10].

It exists three kinds of feature that Viola and Jones use: a two-rectangle feature, a three-rectangle feature and a four-rectangle feature. To find the value of the two-rectangle feature, it consists in the difference between the sum of the pixels that are in the two rectangular regions. The regions (or rectangles) are the same: they have the same size and the same shape. And they are horizontally or vertically adjacent. The three-rectangle feature are calculated by the sum of the pixels of the two outside rectangles subtracted from the sum of the pixels in the center rectangle. The last kind of feature is the four-rectangle feature consists in the difference between the diagonal pairs of rectangles [28].

For example, the figure 4.5 shows the different kinds of rectangle features used by the Viola-Jones algorithm. The images (A) and (B) show the two-rectangle features. The image (C) shows the three-rectangle feature, and the image (D) shows the four-rectangle feature [28].

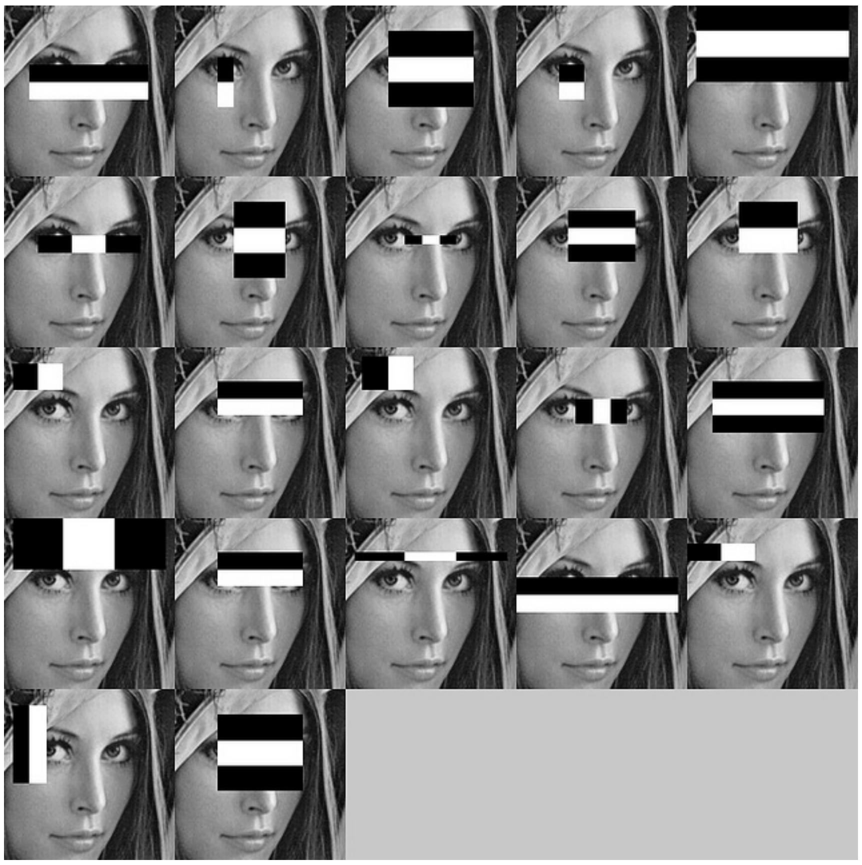


Figure 4.3: Example of an early stage in the Haar cascade

Viola and Jones admit that rectangle features can be considered as primitive features. In contrast with other features, rectangle features are quite coarse (even though they are sensitive to the presence of edges, bars and other simple image structure). It appears that however, a rich image representation is provided by this set of rectangle features and furthermore this representation supports effective learning. In comparison with the extreme computational efficiency provided by rectangle features, their limited flexibility is not much of a problem [28].

4.3 Integral image

Viola and Jones used an intermediate representation of an image that they called "integral image". This integral image allows to compute very quickly the rectangle

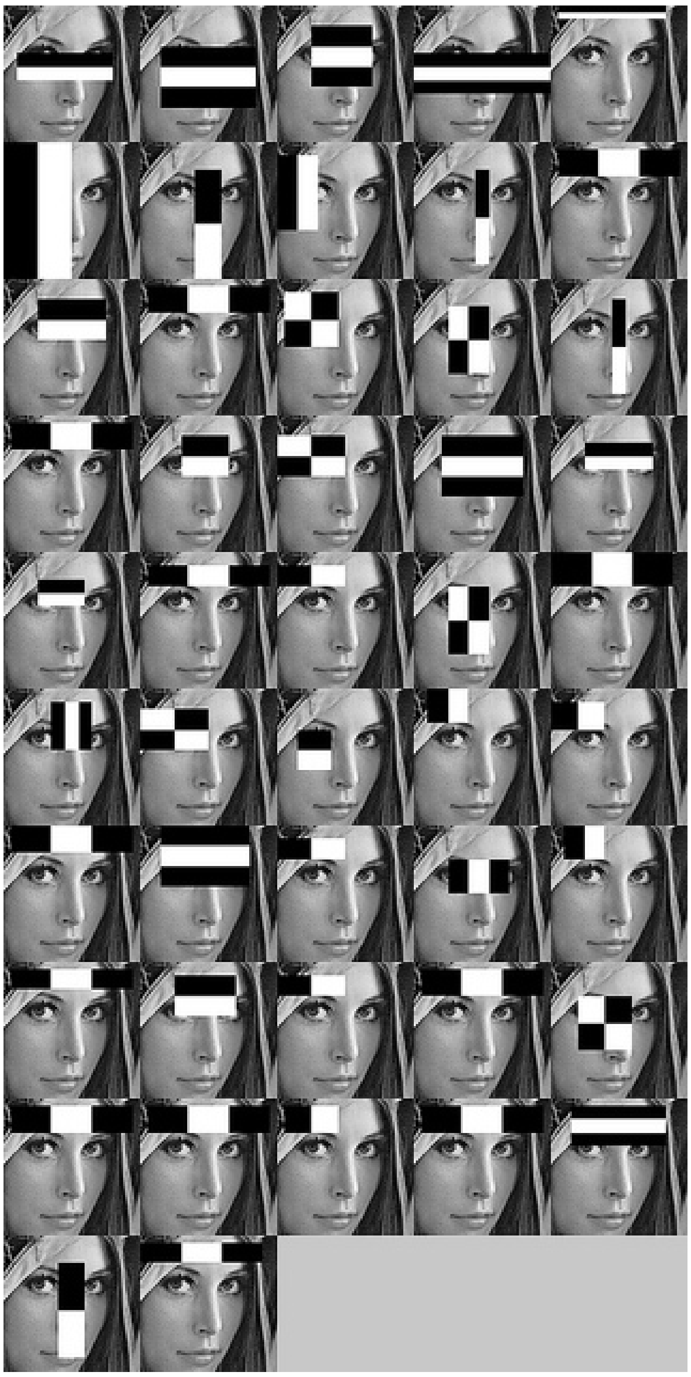


Figure 4.4: Example of the later stage in the Haar cascade

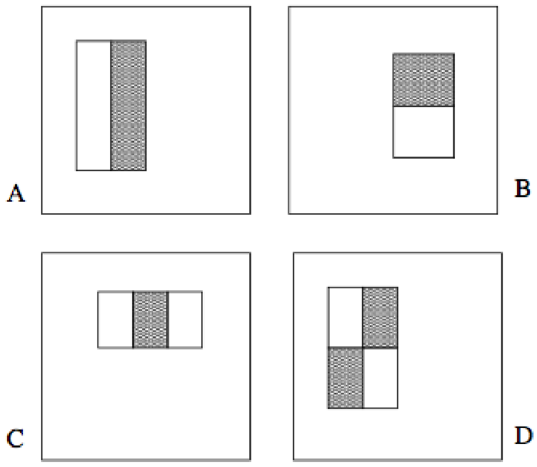


Figure 4.5: Example of the different kinds of rectangle features

features [28]. This technique allows to determine the presence or absence of hundreds of Haar features. It can be determined for every image location and for several scales efficiently. In general, adding small units together is what is called "integrating"; here, the small units are pixel values. The sum of all the pixels above and to the left of each pixel is the the integral value. And this value can be found for each pixel. That is why this technique is efficient: the entire image can be integrated only with a few calculations per pixel. This integration starts at the top left corner and go through all the image to the right and down [11].

It means that the integral image, at location x, y contains the sum of the pixels above and on the left of x, y, x, y included:

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

where $ii(x,y)$ is the integral image and $i(x,y)$ is the regional image. In the figure 4.6, the value of the integral image at point (x,y) is the sum of all the pixels above and on the left [28].

Using the following pair of recurrences:

$$s(x,y) = s(x,y-1) + i(x,y) \tag{4.1}$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \tag{4.2}$$

(where $s(x,y)$ is the cumulative row sum, $s(x,-1) = 0$, and $ii(-1,y) = 0$) the integral image can be computed in one pass over the original image.

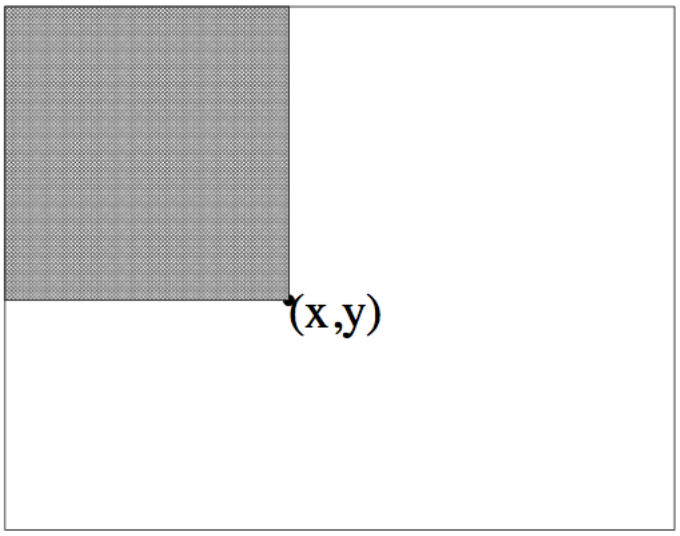


Figure 4.6: Integral image

Thanks to the integral image, any rectangular sum can be calculated in four array references. In the figure 4.7, the sum of the pixels in rectangle D can be calculated with four array references. The value of the integral image at the point 1 is the sum of the pixels in rectangle A. The value at the point 2 is $A + B$, at the point 3 is $A + C$, and at the point 4 is $A + B + C + D$. The sum in D can then be computed as $4 + 1 - (2 + 3)$ [28].

4.4 Weak classifiers and AdaBoost

Features are extracted from a sub-region of an input image. This sub-region has a size that is usually of 24 by 24 pixels. Each of all the features types are moved and scaled across the entire input image (In a 24 pixel by 24 pixel sub-region, it means that there are about 160,000 possible combinations to process) [25].

AdaBoost is a machine-learning method used by Viola and Jones in order to select the specific Haar features to use. It is also used to set the threshold levels. This method is based on the combination of many weak classifiers to form a strong one. It is called a weak classifier because this kind of classifiers obtains the right answer only a little more often than a random guess would which is not particularly good. The purpose of using so many weak classifiers is to get a right answer with a higher rate of success. All of this is based on the verified hypothesis that if each of the

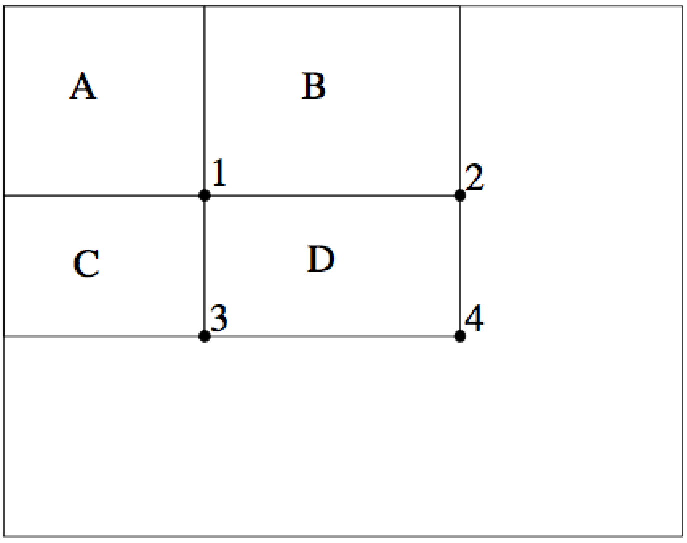


Figure 4.7: Integral image with four array references

weak classifiers pushes the final answer a little bit in the right direction each time, it means that at the end, the correct answer is obtained. This combination of several weak classifiers represents a strong one [11].

AdaBoost works the following way: it chooses a set of weak classifiers that are going to be combined and assigns to each of these classifiers a weight (see figure 4.8). The result of this weighted combination is a strong classifier [11]. One of the difficulties and challenges for this learning algorithm is to associate a large weight for each good classifier and a smaller weight for each poor classifier. In order to succeed in selecting a small group of good classifiers but with still significant variety, AdaBoos is quite an aggressive algorithm [28].

Experiments have been tested with a classifier constructed from 200 features and using AdaBoost. The result would give reasonable results. The detection rate of the classifier was of 95% and it obtain only 1 false positive in 14084 on a testing dataset (see figure 4.9)[28].

With this experiment, it is an initial evidence that the 200-feature classifier is an efficient technique for object detection. It means that a boosted classifier constructed from rectangle features is also an efficient technique for object detection. The results of this experience are convincing in terms of detection. But they may not be sufficient for real-world tasks. This boosted classifier requires 0.7 seconds to scan an 384×288 pixel image. So regarding the computation time, it is probably faster than

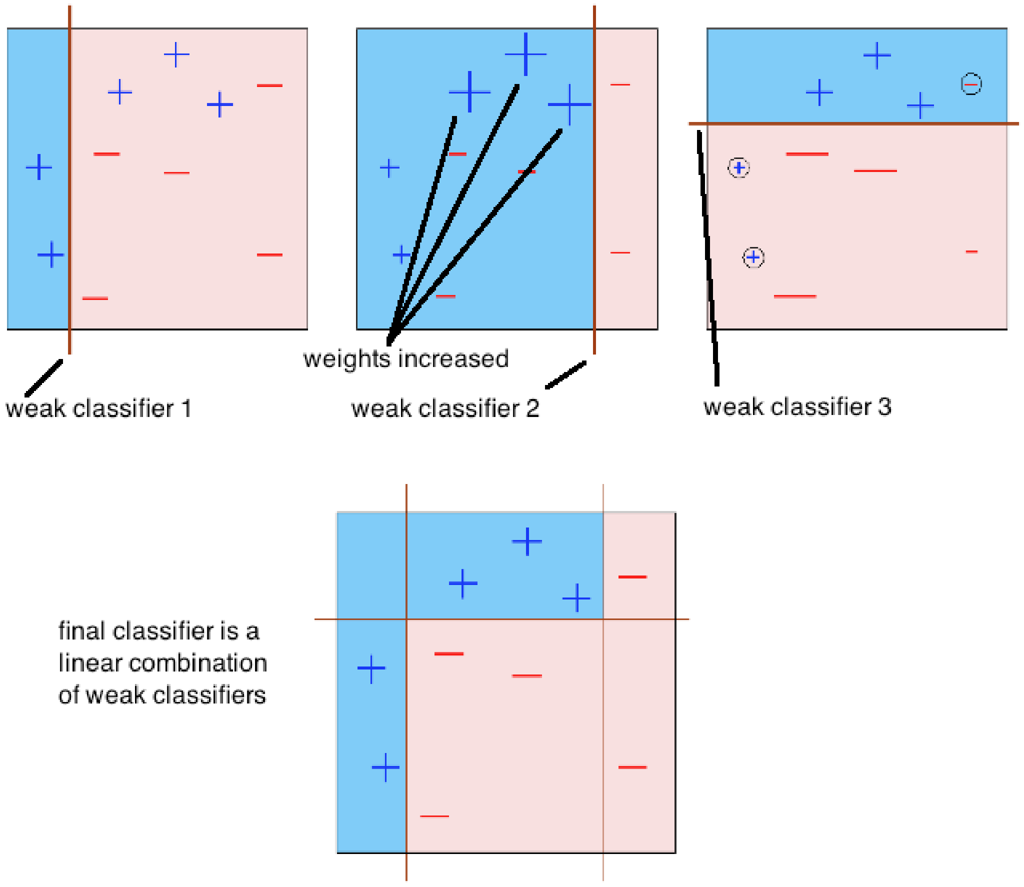


Figure 4.8: AdaBoost method

any other system already published. In order to improve the system so that it will suit to real-world tasks, the detection performance must be improved. But the most straightforward method to do that is to add features to the classifier and doing that will immediately decrease the speed of this system; it will increase computation time [28].

4.5 Classifiers cascade

What Viola and Jones did to classify image regions and sub-regions in an efficient way is to combine AdaBoost classifiers as a filter chain. It is constructed as a cascade and that is why Viola and Jones named it "Classifiers cascade" comes from. This

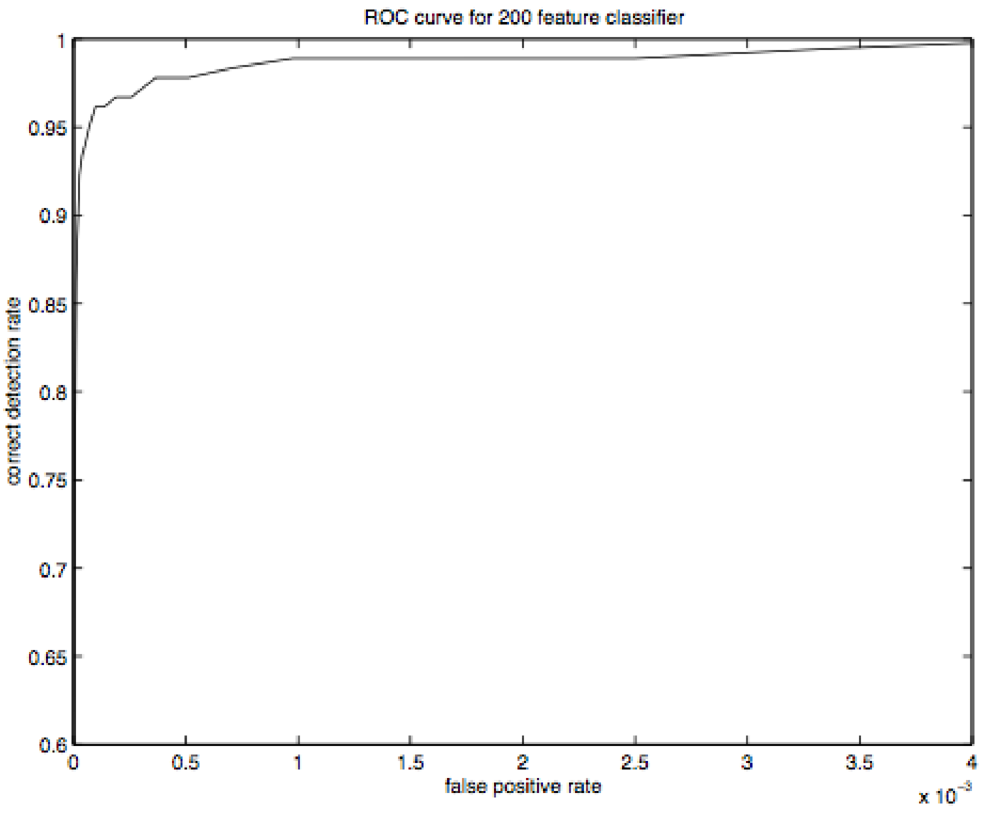


Figure 4.9: Receiver operating characteristic (ROC) curve for the 200 feature classifier

chain is composed for each filter of a separate AdaBoost classifier that has a fairly small number of weak classifiers. As in figure 4.10, the classifier cascade represents a chain of filters. If an image sub-regions makes it through the whole cascade, it is classified as "Face". If not it is classified as "Not Face" [11]. Using this algorithm with the classifiers cascade method allows to reduce significantly the computation time and to increase significantly the detection performance [28].

Tests were made to see if the cascade method was feasible. To do that, two simple detectors were trained. One of them was a 200-feature classifier and the other one was a cascade of 10 20-feature classifiers. Figure 4.11 gives the ROC curves that compare the performance of the two classifiers. Between the two classifiers, regarding their accuracy, the difference is little and not significant. On the other hand, regarding their speed, the difference is large and significant. The classifier cascade is about 10 times faster. This is because as soon as the first stage, most of the non-faces are discarding so in all the next stage, the classifier never evaluate them again [28].

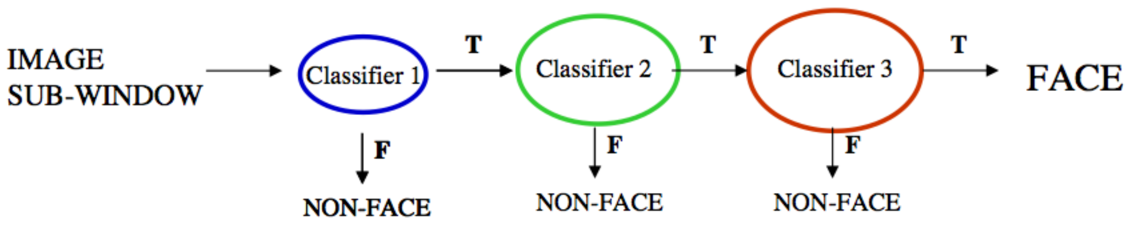


Figure 4.10: Cascade of boosted classifiers

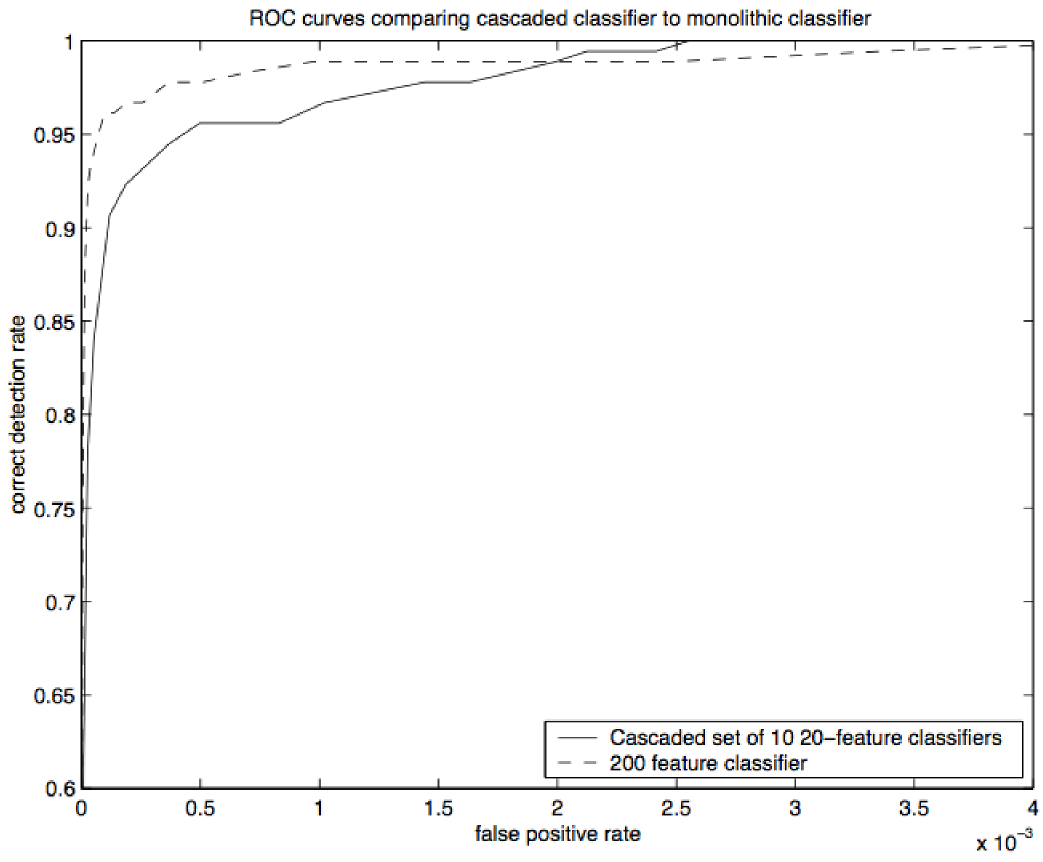


Figure 4.11: ROC curves of a 200-feature classifier and of a classifier cascade that contains 10 20-feature classifiers

The cascade method has been made in a way that there must not be false negative. That means that a face must not be classified as "Not Face". To do that, the assistance threshold has been set low for each level. This way, in the training set, it is low enough to pass all or almost all face examples. All the training images that passed previous stages are classified by filters trained to do it for each level. A region is immediately classified as "Not Face" if even one of these filters did not succeed to pass this region. If one of these filters succeed to pass a region, then it is up to the next filter in the chain. If a region succeed to pass through all the filters that are present in the chain, then this regions is classified as "Face" [11].

The key of this method is to construct smaller boosted classifier (yet more efficient). This way, they will detect nearly all the positive instances while rejecting a lot of the negative sub-regions. Before to use complex classifiers to achieve low false positive rates, simple classifiers are called to reject most of the sub-regions [28].

The order of the filters in the cascade is not random. The importance weighting that AdaBoost assigns is on what is based the order of the filters. The filters that are the more heavily weighted are called early. This way, they eliminate the sub-regions that are not face as soon as possible. In figure 4.12, it shows the first 2 features from the Viola-Jones cascade with a face behind. The first feature used is the one with the eye region being darker than the cheek region. The second feature used is the one with the eyes region being darker than the bridge of the nose [11].

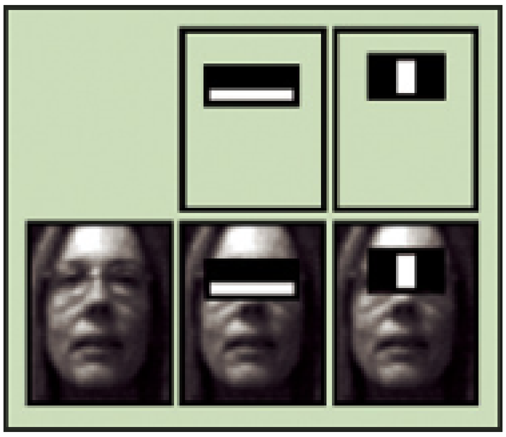


Figure 4.12: The first two Haar features in the original Viola-Jones cascade

The structure in itself of the cascade means that within any single image, there is a majority of sub-regions that are negative. This way, since the earliest stage, the cascade tries to reject as many negatives as possible. Because, on the contrary, when a positive instance occurs, it will trigger the evaluation of all the classifiers of the cascade. And this is an really rare event [28].

Following are the different numbers about cascade classifiers (see figure 4.13) [19]:

- 100% detection rate and 50% false positive rate is achieved by a 1 feature classifier
- 100% detection rate and 40% false positive rate is achieved by a 5 feature classifier
- 100% detection rate and 10% false positive rate is achieved by a 20 feature classifier

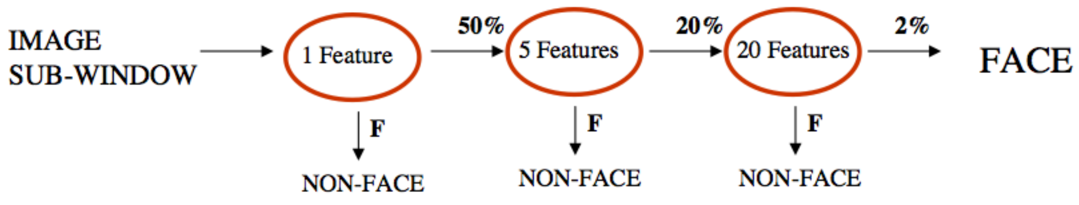


Figure 4.13: Cascade of boosted classifiers rate

4.6 Test set and training

The face training set is composed of about 5000 faces (hand labeled). All the faces are scaled and have the same resolution (24 × 24 pixels). All the faces come from face images on the internet chosen randomly. Some typical face examples are shown in figure 4.14 [28].

To resume, the training set is composed of [19]:

- about 5,000 faces
  - All frontal
- 300 million non faces sub-regions
  - from 9,400 non-face images
- Face are normalized

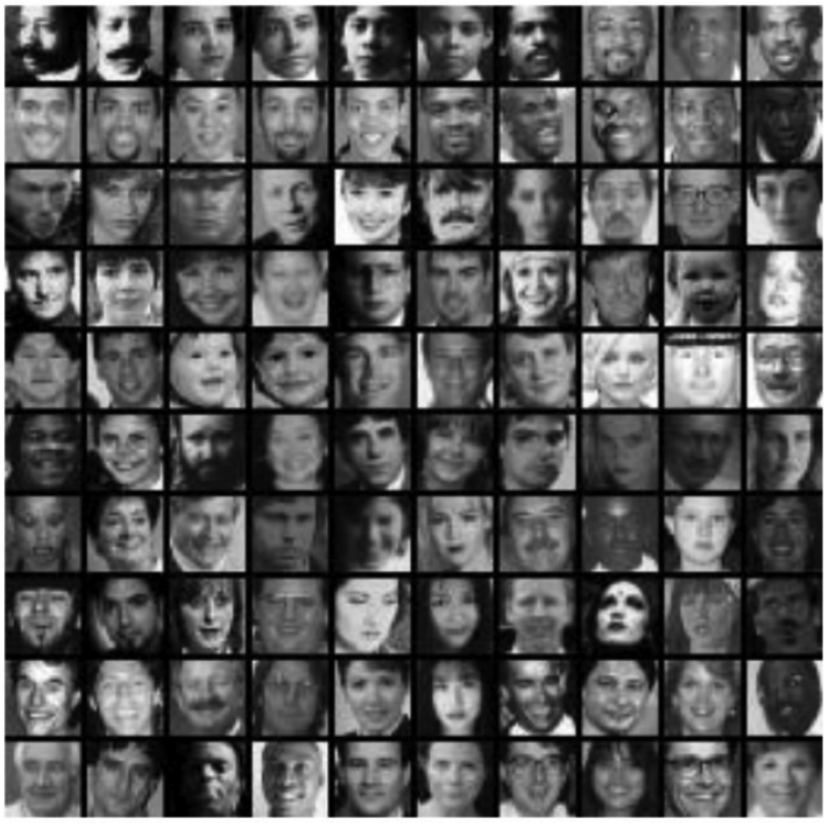


Figure 4.14: Example of frontal upright face images used for training

  - Scale, translation

- Many variations
  - Between individuals
  - Lightning
  - Pose (rotation of the head)

Usually, there is two parts into a test set: the first part is the training set and the second part is the validation set. Typically, the training set is composed by about 5,000 positives samples (faces) and 10,000 negative samples (non faces: usually it is non face sub-regions chosen from non-face images) [7]. For this kind of training, with a 32 layer classifier the total time is usually of several weeks [28].

Viola-Jones training stage proceeds with the following step [7]:

- "Given the number K of possible features (about 160,000 on a 24×24 gray-level image)
- Fix the number L of desired stages in the cascade
- Iterate until L weak classifiers have been selected:
  - Given reweighed data from the previous stage
  - Train all K weak classifiers (find the best threshold to classify the training set)
  - Select the best classifier at this stage
  - Reweight the data"

As said previously, depending on how good a weak classifier is, a weight is associated to it. The weak classifiers are associated to weights that depends on their classification error. The weak classifiers are combined linearly in function of those weights which represents a huge computational cost [7].

Part III

Feature extraction and classification

Contents

This part will focus on the steps following face detection, which are feature extraction and classification. Chapter 5 will present the main issues on feature extraction, and usual feature extraction methods, while the following chapter will focus on the Local Binary Patterns algorithm. Classification will then be introduced in Chapter 7, with a general overview of the classification problem, along with a presentation of some classification algorithms. The last chapter will describe Support Vector Machine classification.

Chapter 5

Feature extraction

After having detected the face, for example using Viola-Jones algorithm, it is necessary to perform feature extraction in order to process data before classification. This step takes an image as input, and extracts vectors characterizing its main features. In the case of a facial expression recognition system, resulting vectors contain informations about spacial configuration of facial features, but can also encode informations about shape, texture or movement of the image's content [5].

There are however many kinds of algorithms outputting features vectors, and the choice of an effective one depends on many criteria. These feature extration methods can generally be ranked among 2 categories: they can either be appearance-based or geometry-based, depending on the way they extract feature vectors. The aim of this chapter is to explain the differences between these 2 types of feature extraction methods, and provide some examples from each category.

Before developing a facial expression recognition project, it is important to know what already exists; the state of the art of facial expression recognition system. In this chapter, an overview will be given of the existing systems before to decide on a feature extraction system for the project.

Two main categories of feature extraction algorithms can be distinguished : *appearance-based* or *geometry-based.* The first ones are algorithms that try to find basic vectors characterizing the whole picture, usually by a dimensionality reduction method. These algorithms lead to a simplification of the dataset, while retaining the main characteristics of the picture. However, these methods have to be carefully parametrized, so they do not encounter the "curse of dimensionality", which is about processing high-dimensional data.

Examples of appearance-based methods : Principal Component Analysis, Linear Discriminant Analysis, Hidden Markov Models, Eigenfaces.

The second type of feature extraction algorithms are geometry-based algorithms. These methods tend to locate important features, and build the feature vectors depending on those regions of interest. The key point of these methods is that the face is not a global structure anymore. Indeed, it has been summarized in a set of features regions, which are themselves translated into feature vectors.

Examples of geometry-based methods : Gabor Wavelets, Local Binary Patterns.

5.1 Principal Component Analysis (PCA)

This is a statistical method; one of the most used in linear algebra. PCA is mainly used to reduce high dimensionality of data and to obtain the most important information out of it. PCA computes a covariance matrix and a set of values called the eigenvalues and eigenvectors from the original data [9]. Its output is a new coordinate system with lower dimensions, obtained from transformed high dimensionality of data, while preserving the most important information. Since it is a statistical method, it can also be used in the classification step.

5.2 Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis is also a statistical method, used to classify a set of objects into groups. It is done by looking at a specific set of features describing the objects. LDA as PCA are used to establish a linear relationship between the dimensions of the data. LDA uses this relationship to model the differences into classes, while PCA does not take any differences into account in the linear relationship. The idea behind LDA is to perform a linear transformation on the data to obtain a lower dimensional set of features [9]. Like PCA, LDA can also be used as a classification algorithm.

5.3 Local Binary Patterns (LBP)

This is an geometry-based method. Its first application was to describe texture and shape of an image by extracting informations from the neighbourhood of a central pixel. These informations are the output of the thresholding of intensity values from the neighbourhood pixels with the intensity value of the central pixel [9]. This method will be detailed in Chapter 6, and will be used in our facial expression recognition system.

5.4 Hidden Markov Models (HMM)

These models are a set of statistical models used to characterize the statistical properties of a signal [23]. It can be used as a classification algorithm, and can also be developed to recognize expressions based on the "maximum likelihood decision criterion" [15].

5.5 Eigenfaces

Eigenfaces are a set of eigenvectors. These eigenvectors are derived from the covariance matrix of a set of images; and this in a high-dimensional vector space. The eigenvectors are ordered and each one represents the different amount of the variation among images. Characterization of the variation between face images is then possible [26].

5.6 Gabor Filters

Gabor filters are applied in order to extract a set of Gabor wavelet coefficients. Filter responses are obtained when Gabor filters are convolved with face image. These representations of face image display desirable locality and orientation performance [12]. However, the main limitation of the Gabor feature extraction is processing time.This algoritm is very time-consuming, and dimensions of resulting vectors are prohibitively large [22].

Conclusion paragraph that opens on LBP

Chapter 6

Local Binary Patterns

Some feature extractions are widely used and studied as the Principal Component Analysis (PCA) and the Linear Discriminant Analysis (LDA) to characterize and describe the face. They in fact describe the whole face. But these method are not efficient when the lightning changes or when the pose of the head changes. This is quite a challenge for these method. That is why some researchers turned to local descriptors. These local descriptors describe the face by characterize the parts of the face in function of their importance. The Local Binary Pattern (LBP) feature extraction is a local descriptor and it is widely used [1].

The LBP operator is known to be an effective texture descriptor. It has been introduced in 1996 by Ojala et al. [20]. It is also one of the most widely used. This operator has a lot of advantages. One of its main advantages is that this operator is highly discriminative. The other advantages are its invariance to gray-level changes and its computation efficiency. Its computation efficiency is suitable for image analysis but it may not be efficient enough for real-time analysis [1].

LBP is known to be a great operator for texture description but why is it used for face description? Because faces can be seen as a composition of micro-patterns. And describing micro-patterns is what the LBP operator does [1].

6.1 Overview

Globally, a gray-scale image of a face is divided into small regions. LBP histograms are extracted from each of those small regions. Then these histogram are concatenated into a single feature vector [13].

The LBP operator works with one central pixel and its eight neighbor pixels. It is basically a comparison with a threshold between the central pixel and the neighbor pixels. The intensity value of the central pixel sets the threshold. Then for each neighbor pixels, the same method is applied. If a neighbor pixel has a value superior to the one of the central pixel or equal to, a 1 is assigned to this pixel. If the pixel value of the neighbor pixel is inferior to the one of the central pixel, then a 0 is assigned to this pixel. Then, when all the neighbor pixels are done, a LBP code for the central pixel can be obtained by concatenating all the values of the eight neighbor pixels into a binary code. And for a better comprehension for the human, this code can also be transformed from a binary code to a decimal one. The figure 6.1 shows an example of the LBP process for a central pixel and its eight neighbor pixels [13].

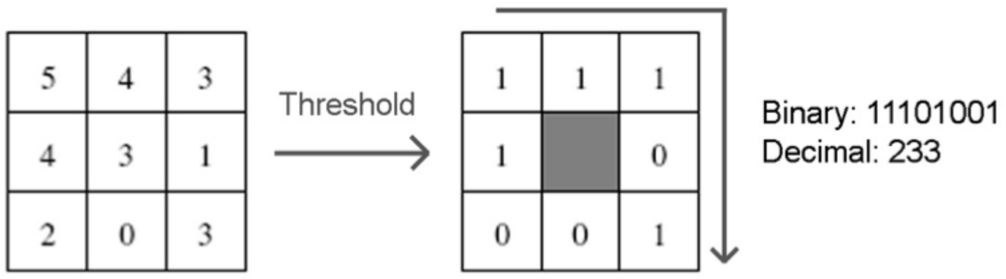


Figure 6.1: The LBP operator

The LBP operator works with one central pixel and its eight neighbor pixels. It is basically a comparison with a threshold between the central pixel and the neighbor pixels. The intensity value of the central pixel sets the threshold. Then for each neighbor pixels, the same method is applied. If a neighbor pixel has a value superior to the one of the central pixel or equal to, a 1 is assigned to this pixel. If the pixel value of the neighbor pixel is inferior to the one of the central pixel, then a 0 is assigned to this pixel. Then, when all the neighbor pixels are done, a LBP code for the central pixel can be obtained by concatenating all the values of the eight neighbor pixels into a binary code. And for a better comprehension for the human, this code can also be transformed from a binary code to a decimal one. The figure 6.1 shows an example of the LBP process for a central pixel and its eight neighbor pixels [13]. The figure 6.2 shows an example of the LBP operator applied on a facial image from the JAFFE database [16].

6.2 Improvements

6.2.1 Circular LBP

In order to be able to use the LBP operator at different scales, a new form of the operator was used. This new form is the "Circular LBP Operator". This way, the operator can be extended to other neighborhoods than only eight pixels; it can be extended to neighborhood of various sizes [9].

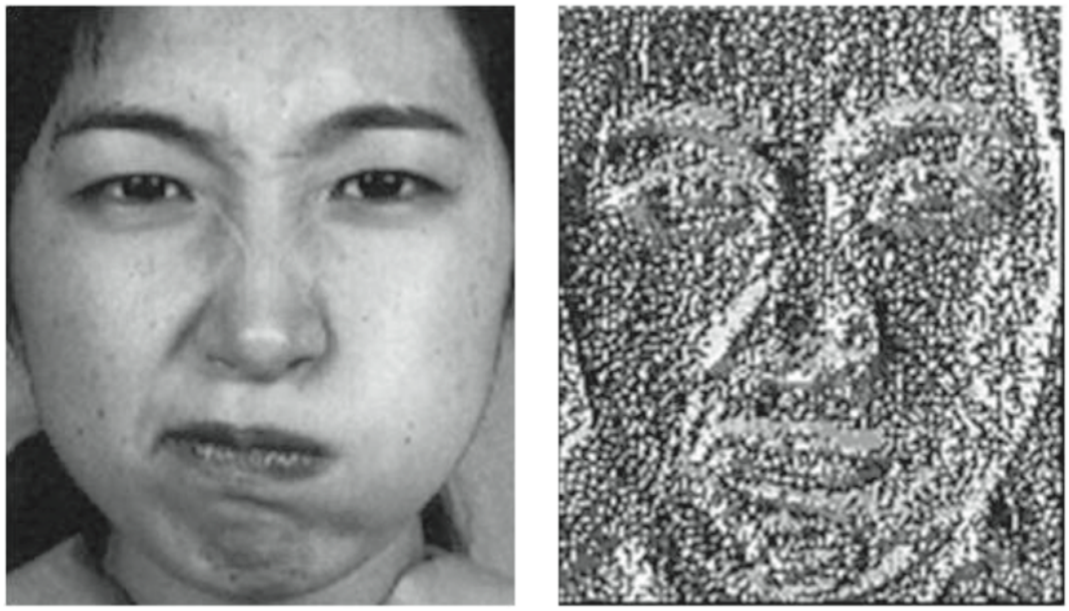


Figure 6.2: Example of the LBP operator applied on a facial image from the JAFFE database

This operator still compares the intensity value of the central pixel with the one of its neighbors but the neighbors pixels are now calculating by a circle as follows [9]:

- P represents the number of sampling points (neighbor pixels)
- C represents the central pixel and has the coordinates (x_c, y_c)
- R represents the distance between each of the neighbor pixels and the central pixel

To find the coordinates of each P neighbor pixel, the formula are the following [13]:

$$x = x_c + R\cos(2\pi n/P) \tag{6.1}$$

$$y = y_c + R\sin(2\pi n/P) \tag{6.2}$$

The figure 6.3 shows the circular LBP operator with a different number of neighbor pixels P and with various sizes [13]. For the circular LBP operator, the following notation is used: (P, R) and for a pixel p, the following notation is used: $LBP_{P,R}(p)$ [9].

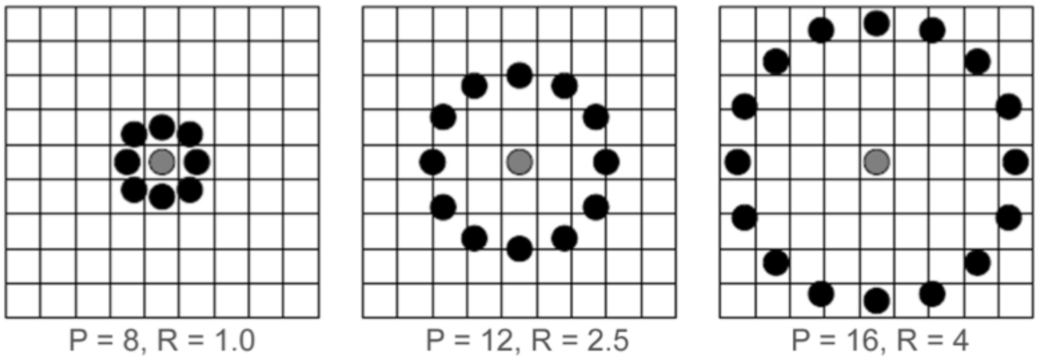


Figure 6.3: The circular LBP operator with different radius R and different number of neighbor pixels P

Most of the time, when a circular LBP operator is used, the coordinates of the neighbor pixels (calculated with the formula given above) may not fall exactly on a pixel. In this case, using a bilinear interpolations of the neighbor pixel intensity values is recommended for the comparison. For example, the circular operator with $P = 8$ and $R = 1.0$ looks like the basic LBP operator; the only difference is that for the calculation, in case of the neighbor pixels do not fall right into single pixels, these pixel has to be interpolated first [9].

What is resulting of the LBP operation is the texture. The formula of the texture is the following [9]:

$$T = t(I_C, I_0, I_1, ..., I_{P-1}) \tag{6.3}$$

With I_C, the intensity value of the central pixel, I_p for $p = 0, 1, ..., P$, the intensity values of the P neighbor pixels, with T, the texture in the local neighborhood of C [9].

But this texture can also be calculated in an other way. It consists in subtracting the central pixel intensity value from the pixel values of the computed neighborhood values. This way, the texture T is the combination of the differences between the intensity values of the neighbor pixels and the central pixel, and of the intensity value of the central pixel I_C. The formula resulting is the following [9]:

$$T = t(I_C, I_0 - I_C, I_1 - I_C, ..., I_{P-1} - I_C) \tag{6.4}$$

Based on the work of Ojala et al. [20], $t(I_C)$ describes the overall luminance of an image. The overall luminance is not related to the local texture of the image. And by consequent, relevant information for texture analysis are not provided. Even after dropping this element from the formula above, the texture description still contained the most of the texture information with the following formula [9]:

$$T = t(I_0 - I_C, I_1 - I_C, ..., I_{P-1} - I_C) \tag{6.5}$$

The above formula makes the texture description invariant against shifts in the intensity values. But against scaling of the intensity values, the texture description is not invariant. To obtain a texture description invariant to these scalings, only the signs of the differences are taken into account (and not the difference in itself). The new formula is as follows [9]:

$$T = t(s(I_0 - I_C), s(I_1 - I_C), ..., s(I_{P-1} - I_C)) \tag{6.6}$$

where,

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{6.7}$$

For the pixels with an intensity value superior or equal to the one of the central pixel, the sign assigned would be 1. For the pixels with an intensity value inferior to the one of the central pixel, the sign assigned would be 0 [9].

The last step is to assign a binomial weight to each sign, for the LBP operation.These weights are summed and it produces for the central pixel C of coordinates (x_C, y_C), the LBP code. The formula is as follows [9]:

$$LBP_{P,R}(x_C, y_C) = \sum_{p=0}^{P-1} s(I_p - I_C)2^p \tag{6.8}$$

The use of the formula above to calculate LBP produce a large number of patterns and it does not help to reduce the size of data. That is why Uniform Local Binary Pattern are used; this way, LBP are used in an efficient way for image processing [9]. This kind of LBP will be explained in the subsection below.

6.2.2 Uniform LBP

A Local Binary Pattern can be called uniform only if it contains 2 or less bitwise transitions. A bitwise transitions can be a transition from 0 to 1 or the opposite: from 1 to 0 [9].

In fact, there can be only 2 or 0 bitwise transitions for a uniform LBP. Indeed, the LBP is circular so there cannot be only 1 bitwise transition. If at a moment on the circle, one pixel is at 1 and then after two pixel for example, there is a pixel at 0, this is one bitwise transition. But then, going on on the circle, there will be inevitably another bitwise transition from 0 to 1 to go back at the pixel with the value of 1 [9].

If a pattern contains 0 bitwise transition, it means that it is composed only of 0s or of 1s [9].

For a uniform LBP with a bit-length of 8 bits, containing 2 bitwise transitions, with P numbers of neighbor pixels, there are $P(P-1)$ possible combinations. So there are $8(8-1)+2=58$ possible patterns ("+2" is for the 2 patterns with 0 bitwise transition; the patterns with all 1s or all 0s). The uniform LBP has the following notation:

$$LBP^{u^2}_{P,R}$$

with P numbers of neighbor pixels and R, the value of the radius [9].

There are 2 main advantages coming along with the uniform LBP. The first one is the reduction in range of possible pattern that the uniform LBP allows. With the uniform LBP operator, there are 58 possible combinations as seen above. With the basic LBP operator, for the same bit-length, there are $2^8 = 256$ possible combinations. More than four times more possible combinations [9].

The second advantages is that even though there is a reduction of dimensionality thanks to the uniform LBP, the patterns are still discriminative between various structural features. These structural features still detected by the uniform LBP are the spot, the spot/flat, the line end, the edge and the corner. These features are shown in the figure 6.4 [9].

6.3 Facial Expression Recognition based on LBP

The LBP operator can be used to extract features for a Facial Expression Recognition system. The main idea is to obtain a set of features vectors with the LBP operator. The process is first to train models to recognize the 6 basic emotions. The models

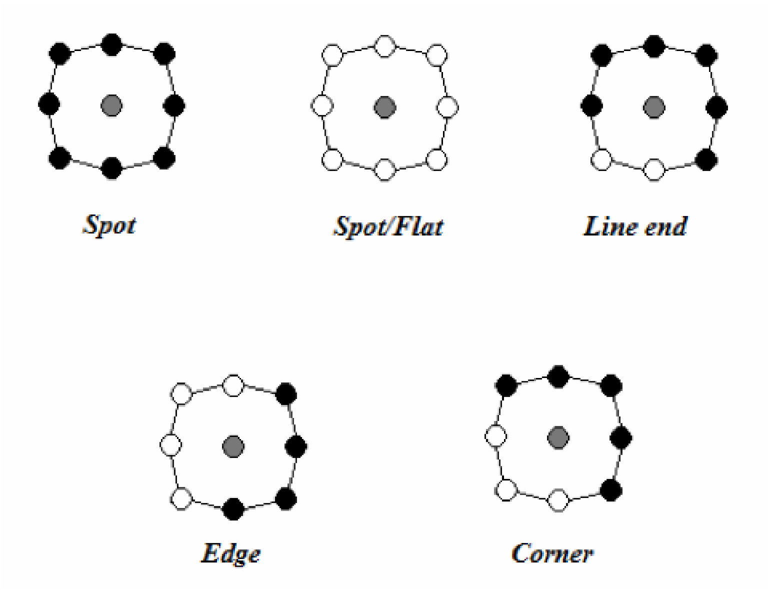


Figure 6.4: Patterns that can be detected with uniform LBP

are trained with the features vectors extracted. Then to recognize an emotion on a sample face image, features vectors are extracted from this sample face image with the LBP operator. And these vectors extracted are then test against the models built in the training phase [9].

6.4 Histogram computing

To obtain these features vectors mentioned above, the code obtained thanks to the LBP operator are concatenated into an histogram. And this is this histogram that gives the features vectors.

To compute this histogram, first the face is partitioned into regions. Most of the time, the face is partitioned as following: 7 rows and 6 column so in total 42 regions. This way of partitioning is the most used for Facial Expression Recognition systems. It suits well the parts of the face that are important features for emotion recognition. The dimension of the face for all the images tested should be kept constant. Then for all the pixels of the face, the LBP code, $LBP^{u2}_{8,2}$, is computed. Then, the codes resulting are grouped region-wise into different bins. Here, because the LBP code is the one with $P = 8$ and $R = 1.0$, the number of bins in the histogram is of 59 (56 bins for the uniform LBP with 2 bitwise transitions, 2 bins for the uniform LBP with 0 bitwise transition and 1 bin for the non-uniform LBP). All these bins are concatenated column-wise; as it can be seen in the figure 6.5. The final vector obtained with this histogram is a vector of $42 \times 59 = 2478$ rows and 1 column (the histogram is computed for each region partitioned on the sample image) [9].

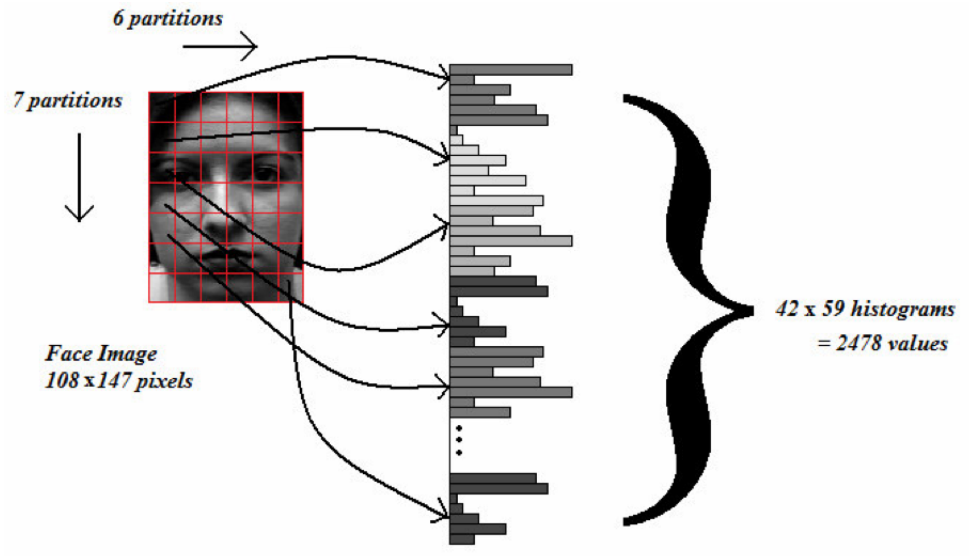
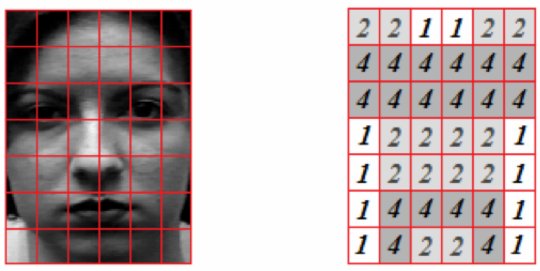


Figure 6.5: Example of a vector based on histogram computing

The regions obtained when the sample image is partitioned have not the same contribution to the emotion expressed. For example, the eyes bring more information about an emotion than the cheeks. So for each region, a weight is assigned. Thus, the histogram of each region is multiply with a corresponding weight. The weights are assigned as in the figure 6.6 [9].



2	2	1	1	2	2
4	4	4	4	4	4
4	4	4	4	4	4
1	2	2	2	2	1
1	2	2	2	2	1
1	4	4	4	4	1
1	4	2	2	4	1

Figure 6.6: Weight assignment for each region

Chapter 7

Feature classification

bla

7.1 Supervised and unsupervised learning

bla

7.1.1 Supervised learning

bla

7.1.2 Unsupervised learning

bla

Chapter 8

Support Vector Machine

bla

8.1 Overview

bla

8.1.1 Margin maximization

bla

8.1.2 SVM kernels

bla

8.2 Combining LBP and SVM

bla

Part IV

Implementation

Contents

Bla bla bla

Part V

Evaluation

Contents

Content

Chapter 9

Results

bla

9.1 Section name

bla

Chapter 10

Issues

10.1 Feature extraction

bla

10.2 Real-time

bla

Conclusion

In case you have questions, comments, suggestions or have found a bug, please do not hesitate to contact me. You can find my contact details below.

Jesper Kjær Nielsen
jkn@es.aau.dk
http://kom.aau.dk/~jkn
Niels Jernes Vej 12, A6-302
9220 Aalborg Ø

Bibliography

[1] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28(12):2037–2041, 2006.

[2] Keith Anderson and Peter W. McOwan. A real-time automated system for recognition of human facial expressions. *IEEE Trans. Systems, Man, and Cybernetics Part B*, 36(1):96–105, 2006.

[3] Marian Stewart Bartlett, Gwen Littlewort, Ian Fasel, and Javier R. Movellan. Real time face detection and facial expression recognition: Development and application to human computer interaction. *Proc. CVPR Workshop on Computer Vision and Pattern Recognition for Human-Computer Interaction*, 5:53, 2003.

[4] Vinay Bettadapura. Face expression recognition and analysis: The state of the art. *Tech Report*, 2012.

[5] Claude C. Chibelushi and Fabrice Bourel. Facial expression recognition: A brief tutorial overview. `http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/CHIBELUSHI1/CCC_FB_FacExprRecCVonline.pdf`, 2003.

[6] Charles Darwin. *The Expression of the Emotions in Man and Animals.* John Murray, 2. ed. edition, 1904.

[7] Fabrizio Dini. An application of viola-jones algorithm: face detection and tracking. `http://www.micc.unifi.it/dini/download/dbmm2008-Dini.pdf`, 2008.

[8] Gianluca Donato, Marian Stewart Bartlett, Joseh C. Hager, Paul Ekman, and Terrence J. Sejnowski. Classifying facial actions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21(10):974–989, 1999.

[9] Abhiram Ganesh. Evaluation of appearance based methods for facial expression recognition, 2008.

[10] Adam Harvey. Adam harvey explains viola-jones face detection. `http://www.cognotics.com/opencv/servo_2007_series/part_2/sidebar.html`, 2012.

[11] Robin Hewitt. How face detection works. `http://www.cognotics.com/opencv/servo_2007_series/part_2/sidebar.html`, 2007.

[12] Yousra Ben Jemaa and Sana Khanfir. Automatic local gabor features extraction for face recognition. *International Journal of Computer Science and Information Security*, 3(1), 2009.

[13] Bram K. Julsing. Face recognition with local binary patterns, 2007.

[14] Emotion Lab. Karolinska directed emotional faces (kdef). http://www.emotionlab.se/resources/kdef.

[15] Jenn-Jier James Lien. Automatic recognition of facial expressions using hidden markov models and estimation of expression intensity, 1998.

[16] Weifeng Liu, Yanjiang Wang, and Shujuan Li. Lbp feature extraction for facial expression recognition. *Journal of Information and Computational Science*, 8(3):412–421, 2011.

[17] Michael Lyons. The japanese female facial expression (jaffe) database. http://www.kasrl.org/jaffe.html.

[18] Aleix Martinez and Robert Benavente. The ar face database. http://www-sipl.technion.ac.il/new/DataBases/Aleix%20Face%20Database.htm.

[19] University of British Columbia. The viola/jones face detector, 2001.

[20] Timo Ojala, Matti Pietikainen, and David Harwood. A comparative study of texture measures with classification based on feature distributions. *Pattern Recognition*, 29:51–59, 1996.

[21] Maja Pantic and Leon J.M. Rothkrantzi. Automatic analysis of facial expressions: the state of the art. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(12):1424–1445, 2000.

[22] Lekshmi V Praseeda and M Sasikumar. Facial expression recognition from global and a combination of local features. *IETE Tech Rev*, 26(1):41–46, 2009.

[23] Lawrence R. Rabiner and Biing-Hwang Juang. Fundamentals of speech recognition, 1993.

[24] N Sebe, M S Lew, Y Sun, I Cohen, T Gevers, and T S Huang. Authentic facial expression analysis. *Image and Vision Computing*, 25:1856–1863, 2007.

[25] Padhraic Smyth. Face detection using the viola-jones method, 2007.

[26] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[27] UQUAM. The msfde. http://www.er.uqam.ca/nobel/r24700/Labo/Labo/MSEFE.html.

[28] Paul Viola and Michael Jones. Robust real-time object detection, 2001.

[29] Wikipedia. Statistical classification. http://en.wikipedia.org/wiki/Classifier_(mathematics).

Appendix A

Appendix A name

Here is the first appendix