Aalborg University

Master's Thesis

Computer Vision, Graphics and Interactive Systems

Emotion recognition in blurred images with local features and machine learning

Author:
Rasmus L. Kristensen

Supervisors:
Zheng-Hua Tan
Zhanyu Ma
Jun Gou

June 3, 2014

AALBORG UNIVERSITY
DENMARK

School of Information and Communication Technology
Computer Vision, Graphics and Interactive Systems
Niels Jernes Vej 12
9220 Aalborg Ø
http://www.create.aau.dk/vgis/

Title:
Emotion recognition in blurred images with local features and machine learning

Project type:
Master's Thesis

Project period:
September, 2013 - May, 2014

Participant(s):
Rasmus Lyngby Kristensen

Supervisor:
Zheng-Hua Tan (*Aalborg Univeristy*)

Supporting Supervisors:
Zhanyu Ma (*Beijing University of Post and Telecommunication*)
Jun Gou (*Beijing University of Post and Telecommunication*)

Circulation: 4
Number of pages: 130
Appendix: 6
Finished June 4th, 2014

Synopsis:

Facial Expression Recognition (FER) has important extensions to the development of the next generation of *Human Machine Interface*s (HMI). In this project, it was proposed to solve the problem of FER in blurred images by using the blur invariant local feature descriptor *Local Frequency Descriptor* (LFD). Further, it was proposed to use *Spectral Clustering* (SC) to reduce the dimensionality of local features for FER.
The recognition accuracy of LFD was compared against that of the blur invariant local feature descriptor *Local Phase Quantization* (LPQ) and the popular local feature descriptor *Local Binary Patterns*. The recognition accuracy of SC was compared against that of *Principal Component Analysis* (PCA). All recognition accuracies was measured using *Support Vector Machine*s (SVM).
It was found, that LPQ seems to actually provide better recognition accuracy than LFD. Further, it was found that SC in general provides similar or slightly better results than PCA. However, the SC method is more computational expensive.

Preface

You are about to read the documentation of a project on facial expression recognition. It can be tricky to select the line of attack and in which part of the field one wants to contribute. In 2011, a challenge on facial expression recognition was held during the IEEE conference on Face and Gesture Recognition. In the meta-analysis of the competing systems published after the challenge was held, the organizers state that the problem of recognizing six prototypic facial expressions like anger or fear can be considered largely solved. They also state that most proposed systems up until that point could only recognize prototypic expressions from facial expression databases containing largely exaggerated expressions. Therefore, subsequent research must try to expand into the unknown realms and try to incorporate a touch of realism into their research.

In this project, I try to introduce this touch of realism by considering the recognition of facial expressions in blurred images. Often, images sampled in the real world are of poor quality compared to those sampled in a laboratory. One common form of degradation is blurring. I apply a new type of local feature to the problem as well as a new type of dimensionality reduction method. The proposed system is compared against some of the most promising existing systems.

One of these existing systems is based on the *Local Directional Pattern variance* feature descriptor. I implemented the feature extractor myself based on descriptions found in the paper where it was first proposed. Unfortunately, the implementation contained some form of bug, which reduced the recognition rate significantly below what was expected. Despite having debugged the code numerous times and double checked the values of all calculated variables, it was not possible to enhance the recognition rate. I tried to establish contact to all of the three researchers who proposed the feature descriptor, but with no luck. Personally, I find the idea of the feature descriptor quite appealing. Therefore, I have kept the description of it in this project. However, the reader should be aware before hand, that the descriptor is not a part of the final experiments.

During the project period, I spent five months at *Beijing Univeristy of Posts and Telecommunication* (BUPT). There, I had the opportunity to be involved in professional discussions both regarding the content of this project but also the content of other projects. While I was there, I was attached to the PhD group at the *Pattern Recognition and Intelligent Systems* (PRIS) lab. It gave me the opportunity to work together with Doc. Zhanyu Ma and Prof. Jun Gou. The stay provided me with a lot of both personal and professional experiences. Therefore, I would like to use this opportunity to acknowledge and extend my gratitude to Doc. Zhanyu Ma and Prof. Jun Gou for their great contribution to my work. At the same time, I would also like to acknowledge Ass. Prof. Zheng-Hua Tan who both supervised this project and who established the connection to the PRIS lab at BUPT. Acknowledgement should also be given to Julie Damms Studiefond, S. C. Van Fonden, Henry og Mary Skovs Fond, and Knud Højgaards Fond for granting me

the funds needed to support my stay at BUPT.

I have tried to make this report as condensed as possible in order to reduce the number of *Too-Long-Didn't-Read*s (TLDR) suffered by the reader. However, I have also tried to make the report as self sustained and transparent as possible. Therefore, the technical methods and the experiments are explained in such details, that a non-specialist should be able to verify the results.

This report opens with a motivation chapter which also serves as an introduction. Then, the previous research in the field is outlined and a choice of focus is made based thereon. After the outline of previous work, the methods used in the project is described. The report ends with an evaluation chapter and a conclusion. Note that there are six appendixes after the conclusion.

Rasmus Lyngby Kristensen

Reading guide
Through the report, references to sources will be given and they will be collected in a bibliography at the end of the report. The Harvard method is used as reference style. This means that a source is referenced as [surname, publishing year]. Resources located on the project CD is referenced by /cdrom/ followed by the file path to the resource. Figures, tables and equations are numbered according to the chapter they occur in. When using technical abbreviations, the full name of the abbreviation is given the first time it appears. A complete list of the used abbreviations can be found immediatly after the preface.

Abbreviations

AU Action Unit
AAU Aalborg University
DBN Deep Belief Network
EAI Emotion Avatar Image
FACS Facial Action Coding System
FER Facial Expression Recognition
FERA 2011 Facial Expression Recognition and Analysis challenge 2011
HCI Human Computer Interaction
HMI Human Machine Interface
ICA Independent Component Analysis
JAFFE Japanese Female Facial Expression
KDEF Karolinska Directed Emotional Faces
LBP Local Binary Patterns
LBP-TOP Local Binary Patterns from Three Orthogonal Plains
LDA Linear Discriminant Analysis
LDP Local Directional Pattern
LDPv Local Directional Pattern variance
LFD Local Frequency Descriptor
LMD Local Magnitude Descriptor
LPD Local Phase Descriptor
LPQ Local Phase Quantization
LPQ-TOP Local Phase Quantization from Three Orthogonal Planes
MRF Markov Random Field
OAA One-Against-All
OAO One-Against-One
PCA Principal Component Analysis
PIE Pose, Illumination, Expression
RBF Radial Basis Function
RGB Red-Green-Blue
ROI Region Of Interest
SC Spectral Clustering
SIFT Scale Invariant Feature Transform
STFT Short-Term Fourier Transform
SVM Support Vector Machine
TFD Toronto Face Database

Table of Contents

Motivation and Problem statement

1

Facial Expression Recognition (FER) sees multiple uses in many different fields. FER can be used to extract the underlying emotions of a given person when he or she is exposed to different stimuli. Therefore, it is commonly used in psychology, behavioral science and clinical practice [Caleanu, 2013]. According to the instructions of the first FER challenge, FER has important applications to the design of intelligent user interfaces to computers and machines [Valstar et al., 2011]. Especially when constructing interfaces that adapts to the user. One can think of the imaginary HAL9000 computer from the movie "2001: A Space Odyssey" as an example. In one of the scenes, HAL is aware of main character David Bowmans emotions and can make the following observation:

> "*Look Dave, I can see you're really upset about this. I honestly think you ought to sit down calmly, take a stress pill, and think things over.*"
>
> - HAL9000, "2001: A Space Odyssey"

HAL is sensing the emotions of David and can give a meaningful response. It is assessed that 55% of the information exchanged in a conversation between two humans is carried by facial expressions [Mehrabian, 1968]. As a consequence, FER is a very important part of multi-modal systems with modern *Human Computer Interaction* (HCI) [Khan, 2013]. This extends directly into robots which are intended for close interactions with humans. Such a robot must be able to do FER for the interaction to go smoothly [Bettadapura, 2012]. As the HAL example nicely illustrates, FER systems can also benefit elderly, mentally ill or disabled persons who are not able to fully express their needs through spoken or written communication. A system could e.g. detect fear in an elderly person and use that information to locate and take actions against the source.

According to Bettadapura [2012], the first FER system was proposed by Suwa et al. [1978]. The subject did not achieve much further attention before 1990 as indicated in the survey by Samal and Iyengar [1992]. This statement can be supported by doing a search for "*facial expression recognition*" on `http://ieeexplore.ieee.org`. The earliest article returned was written in 1991. Over the course of the following 23 years, the interest for FER has increased tremendously and many well-functioning recognition systems has been proposed [Bettadapura, 2012]. However, according to Khan [2013] there is a need for developing systems which are robust against changes in illumination and light intensity. Further, Zhen and Zilu [2012] states that the recognition rate achieved by existing methods are too low for actual, real-life, practical implementations. Lastly, Tian [2004] and Dhall [2013]

argues that most methods so far considers only high quality, clear face photos obtained in a laboratory environment and that new methods are needed which considers distorted photos.

Aalborg University is currently doing a research project with the title "*Durable Interaction with Socially Intelligent Robots (iSocioBot)*" [Tan et al., 2013]. The key contribution of that project is to design and construct a socially intelligent robot which mimics and interprets human social interaction through multi-modal inputs such as sound and vision. The primal use of the robot is aimed at the service-sector. Based on the above discussion, it is assessed that the robot must be able to do FER to be properly social. The report you are currently holding in your hands documents a project which addresses the FER problem. Specifically, it seeks to uncover a method which could be used by the iSocioBot to recognize facial expressions. At the same time, it also seeks to address some of the standing FER problems outlined above.

1.1 Problem Statement

It is a problem to recognize facial expressions in blurred images with a high recognition rate. So far, the blur invariant *Local Phase Quantization* (LPQ) descriptor has been used to solve the problem. The *Local Frequency Descriptor* (LFD) is a refinement of LPQ, which has not yet been applied to *facial expression recognition* (FER). It is a promising descriptor because it has been showed to outperform LPQ for face recognition in blurred images. This project seeks to answer the following questions: Will LFD provide a higher recognition rate than LPQ for FER in blurred images? Furthermore, will LFD provide better results than the two popular feature extractors *Local Binary Patterns* (LBP) and *Local Directional Pattern variance* (LDPv), which has provided promising results for recognizing facial expressions in sharp images?

The choice of local features for solving the FER problem in blurred images gives rise to a significantly high feature dimensionality. This is in particular true when the image grid division method is used. A high feature dimensionality can hinder the subsequent classification process. It is hypothesized, that local feature descriptors extracted from images containing similar facial expressions will cluster together. Likewise, it is hypothesized that local feature descriptors extracted from images containing different facial expressions will spread far apart. *Spectral Clustering* (SC) is a clustering method which can transform the extracted feature descriptors into a lower dimensional space where samples of the same class forms tight clusters. Therefore, this project also seeks to answer the following question: Will SC reduce the dimensionality of the local feature descriptors to a reduced feature space which better discriminates the clusters of different classes than the so far popular *Principal Component Analysis* (PCA) reduction method?

Previous Work 2

This chapter describes the work which has previously been done on the FER task. As noted by Valstar et al. [2011], researching on FER has been popular over the last two decades and many scientific articles concerning the topic has been published. It would be an exhausting job to read through all the literature, but luckily summarizing surveys have been published regularly. The following summation of previous work takes the three surveys written by Khan [2013], Bettadapura [2012] and Caleanu [2013] as its starting point. The surveys describe 10, 18 and 10 scientific papers, respectively. A retell of the details of their findings would be pointless as the interested reader could simply pick up any of the surveys and get the information directly from the source. Instead, this chapter tries to give an overview of the taxonomy used in the field of FER as well as an overview of recent FER methods. In addition to the surveys, specific scientific articles and other surveys are included to further elucidate the matter where needed.

The three surveys referenced above jointly describes the terms and procedures which over the decades has become the de facto standards within the field. This chapter follows a logical progression starting at how facial expressions are formed and defined in the real world, then explains how facial expressions can be described mathematically, and ends with a description of some of the most used facial expression databases. To enforce this progression, the chapter is divided into sections which individually describes and analyses important subparts of the FER problem. Each section contains relevant references to previous scientific articles containing information covered by that section. Thus, this chapter does not present a description of previous work in a time-line continuous fashion. Instead, it follows a problem-oriented flow that divides the previous work into groups based on similarity.

2.1 The origins of facial expressions

This section provides an overview of the creation and recognition of facial expressions seen from a humane perspective. Facial expressions seem to be one of the most basic channels through which humans can communicate emotions. They are indeed so fundamental to the human race, that the underlying emotion of a specific expression is perceived identically across all human cultures. To prove this fact, a team of researchers traveled to New Guinea. There they found an isolated civilization which had not had their expression interpretation contaminated by movies or pictures from other cultures. Through a set of stories and images of facial expressions they concluded that the civilization did indeed perceive the same emotions from the stories as the rest of the human population [Ekman

and Friesen, 1971]. The findings of the study was supported by Izard [1994]. However, it is often so in science that nothing is simply black or white. Jack et al. [2009] has used an eye-tracker to investigate which parts of the face is emphasized by people from western and eastern cultures. It turns out that people from western cultures look on the eyes, nose and mouth when they are decoding the expression. Conversely, people from eastern cultures looks only on the eyes and nose. In their article they argue, that the method used by eastern people is inadequate to reliably distinguish some facial expressions made by western people. However, they also argue that most expressions are perceived equally across both cultures. This implies, that the following description of facial expressions and their underlying emotions applies more or less to all humans, regardless of ethnic origin.

Emotions originates from various sources. Fasel and Luettin [2003] identifies four sources that inflicts the emotional state of a human. These are:

Mental State
: Felt emotions, Convictions, Cogitations

Non-Verbal Communication
: Unfelt emotions, Emblems, Social winks

Psychological Activities
: Manipulators, Pain, Tiredness

Verbal Communication
: Illustrators, Listener responses, Regulators

What can be deduced from the above list is that quite a lot of factors can trigger an emotion and thus a facial expression. This knowledge will become relevant later in this chapter. For now, please remember that human emotions are more complex than one might expect at first glance.

2.2 How facial expressions are formed and described

This section describes how humans form facial expressions and how the various subparts of facial expressions are termed. A facial expression is formed continually by the face muscles. According to Ekman and Rosenberg [1997], an expression is composed of three temporal segments, namely: the onset, the apex and the offset. The onset is the beginning of the expression from the point where the expression starts to form until just before its peak. The apex is the exact moment when the expression is full. The offset is the decline of the expression until the face is again at neutral. Due to what is known as micro expressions, it can be relevant to consider all three segments when recognizing expressions. A micro expression is an expression which never reaches the peak. This can happen if a person tries to hide their true emotion or if the emotion suddenly changes during the onset. If a FER system is required to recognize micro expressions, both the onset and offset must be considered as well as the apex.

An expression can be spontaneous or posed. A spontaneous expression is based on a true, underlying emotion. Conversely, a posed expression is artificial and not based on a true emotion. They arise when a person is asked to act a facial expression. Due to human nature, it is very difficult to obtain true spontaneous expressions. As a consequence, most

FER researchers work with posed expressions. Unfortunately, spontaneous and posed expressions are not identical [Ekman and Rosenberg, 1997]. Thus, most proposed FER systems might not work if they were to be used in the real world. Due to the difficulties involved in spontaneous expressions, little research has gone into this problem.

During the eons of evolution, humans have developed a quite good method of recognizing expressions. Through the eye-tracker experiment mentioned earlier, a set of gaze heat maps were created. They show that humans put emphasis on the regions surrounding the eyes, nose and mouth when they decipher facial expressions [Jack et al., 2009]. A similar experiment yielding similar results was done by Khan et al. [2012]. This indicates that those regions convey more information regarding the expression than other regions. Kotsia et al. [2008] constructed an FER system and tested it against partly occluded facial expression images. They proved that occluding different parts of the face inflicts the recognition rate of different types of expressions. It is therefore evident, that different parts of the face contains information about different expressions and that the expression information is concentrated around the eyes, nose and mouth.

Evolution is probably also the reason behind the complex structure of the human face. It contains roughly 40 muscles which work together to form and shape the face. Thus, a human can produce a lot of different facial expressions, though most of them does not necessarily convey an emotion. The *Facial Action Coding System* (FACS) was proposed by Ekman and Friesen [1977] as a way to scientifically describe human facial movements. FACS is a comprehensive system that describes the activation of face muscles. It was originally developed as a tool of psychology to record facial expressions. Some face muscles can be activated alone and others can only be activated in groups. Activation of a single muscle or a single muscle group is the lowest possible level of motion the face can do. FACS term these low level motions *Action Unit*s (AU). Every AU has a unique value. As an example, AU1 is a raised inner brow and AU26 is a dropped jar. In total, FACS defines 9 AUs in the upper face, 18 AUs in the lower face and 5 AUs which can not be classified as belonging to either the lower nor the upper face. A given facial expression can be fully described as a combination of AUs. Other similar coding schemes exists, but FACS is the most widely used [Bettadapura, 2012].

While AUs are a way of describing facial expressions by their lowest level of facial motion, there also exists a global approach. In the global approach, the face is considered as one entity and the expression is judged from the entire face in its entirety, thus "global". Ekman and Friesen [1971] defines the following six emotions used in their cross-culture research: happiness, sadness, anger, surprise, disgust and fear. The six facial expressions associated with these six emotions has become known as the basic expressions. Several authors add a neutral facial expression to the list for a total of seven basic facial expressions. These expressions have been heavily used in the FER literature and continues to be used [Bettadapura, 2012; Khan, 2013]. However, another set of expressions was used for the first *Facial Expression Recognition and Analysis challenge 2011* (FERA 2011). They operate with just five basic emotions, which are: anger, fear, joy, relief, and sadness [Valstar et al., 2011]. Though the seven basic emotions has been used extensively, it has been argued that they are not sufficient to properly describe the human emotions. This is supported by the complexity of the underlying emotion creating system, which was briefly mentioned in the

previous section. Parrott [2001] argues that the basic emotions can have sub-emotions. As an example, a person could be happy with pride or happy because they are relieved. In total, Parrott defines 136 emotions, each composed of a primary emotion, a secondary emotion and a tertiary emotion. Du et al. [2014] coins the term *compound emotions* to these kinds of joint emotions. They define 22 basic emotions which they argue should replace the previous seven. These are as follows: neutral, happy, sad, fearful, angry, surprised, disgusted, happily surprised, happily disgusted, sadly fearful, sadly angry, sadly surprised, sadly disgusted, fearfully angry, fearfully surprised, fearfully disgusted, angrily surprised, angrily disgusted, disgustedly surprised, appalled, hatred, and awed.

A problem when constructing a FER system is the changes of facial features. Tian et al. [2001] defines the two terms: permanent facial features and transient facial features. Permanent features are elements which are constant in the face. This could be the mouth, nose and eyes. The transient features are volatile, meaning that they occur and disappears as the face moves. An example could be furrows and wrinkles which changes as the skin stretches. Another way facial features can change is through occlusion by hair, clothes, glasses and the like. One has to be aware of these changes when selecting the facial expression model and the type of input data.

2.3 Types of input data

This section provides an overview of the possible forms of input data which can be used to recognize facial expressions. Obviously, it is implicit that the term "data" refers to some form of image-like data in the terminology used here.

Facial expressions can be recognized in either the time continuous or time static domain. Put another way, an FER system will either process video or images. Consider the method proposed by Cohen et al. [2002] as an example of a system which processes video. It is based on a wireframe model of a standard face and a Tree-Augmented-Naive Bayes classifier. In each frame of the video sequence, they define a set of 12 salient points which they use to fit the standard wireframe model. The points are also used to track the face from frame to frame. By considering how the wireframe is deformed across the frames, 12 facial motion measurements can be calculated for each neighboring pair of frames. These are fed into the classifier which then determines the presence of AUs as defined by the FACS. Basic facial expressions are assigned to the frames by considering which AUs are active. The authors emphasizes that accurate tracking is very important across the frames.

The importance of accurate tracking is also noted by Zhao and Pietikainen [2007]. They developed a feature descriptor based on *Local Binary Patterns* (LBP) which they call *Local Binary Patterns from Three Orthogonal Plains* (LBP-TOP). LBP-TOP belongs to a range of features which consider the video feed as a three dimensional space. The first two dimensions of the space are formed by the two orthogonal spatial dimensions which spans the image plane. The third dimension is formed by the time shift in the video sequence. LBP is a local feature, meaning that it considers the local regions surrounding every pixel in the image. The LBP-TOP feature expands this locality to consider the local region in all three dimensions of the video. Face tracking is vital for LBP-TOP because

the faces must be aligned precisely on top of each other from frame to frame in order to extract the features properly. Dhall et al. [2011] defined *Local Phase Quantization from Three Orthogonal Planes* (LPQ-TOP) by changing the LBP descriptor in the work by Zhao and Pietikainen [2007] with the *Local Phase Quantization* descriptor. By doing so, they obtained a higher recognition rate but the importance of precisely aligned faces prevailed. Indeed, face tracking turns out to be a vital part of most FER systems which processes video.

An example of a system which processes still images could be the system proposed by Shan et al. [2005]. Their system is quite similar to the baseline system from FERA 2011. The system is based on the standard LBP feature extractor. In every image, the LBP feature descriptors are extracted and presented to a classifier made of *Support Vector Machine*s (SVM) combined in a *One-Against-One* (OAO) classification scheme. The output of the classifier is an integer from 1 to 6, which codes for one of the six basic expressions (no neutral). Their system does not identify AUs. Instead they use a global recognition approach.

All three examples of FER systems presented above use *two-dimensional* (2D) spatial data. Even though a video can be seen as *three-dimensional* (3D), every single image in the sequence is a 2D image. Another possibility is to use spacious 3D distance data as input and recognize facial expressions based on the depth information. Both recognition of AUs and the global approach can be done in 3D. As with its 2D counterpart, several 3D based systems has been proposed [Fang et al., 2011]. However, most authors continues to focus on 2D images [Caleanu, 2013].

New types of input data has begun to emerge in recent years. For instance, He et al. [2013] propose a system which uses a Deep Neural Network based on Boltzmann Machines to recognize facial expressions from thermal infrared images. However, FER applications based on thermal images requires a special transducer and as a consequence their availability to a practical system could be limited.

2.4 General feature types

In order to recognize objects contained in an image, it is often necessary to extract some form of image features. Feature extraction is equivalent to reducing the highly complex input image to some simpler form, which is easier to work with. In general, two types of image features exist: geometric features and appearance based features. This section describes the differences between these two feature types and provide some examples of proposed systems which uses these features. In addition, a description of the difference between engineered features and learned features is provided at the end of the section.

Geometric features describe some form of physical parameters of objects found in the image. They can be shapes, positions, lengths, widths, angles, distances, areas and the like. Usually, a set of easily distinguishable salient points are defined on the face and their positions and mutual distances are used to describe the face [Caleanu, 2013]. As an example of a modern system which recognizes the seven basic facial expressions based

on geometric features, consider the work done by Saeed et al. [2012]. They base their system on the 68 fiducial points defined by Lucey et al. [2010]. However, they optimize the processing speed by only considering eight of the points. First, they detect the face using a Haar-cascade detector[Viola and Jones, 2001]. Then they find the fiducial points by the method developed by Belhumeur et al. [2011]. The distances between the eight points are calculated and used as the features. The distances are fed into an *One-Against-All* (OAA) SVM classifier implemented by LIBSVM [Chang and Lin, 2011]. The facial expression is indicated by an integer number, following the same scheme as described previously.

Appearance based features are usually formed by putting the image or a part of the image through an image filter or a bank of image filters. This often implies, that the image or image part is convolved with some filter kernels which reduces the dimensionality of the image. Gabor wavelets are an example of a filter method often used for FER [Bettadapura, 2012]. Often when using filters, the filter responses from a number of fiducial points are used instead of the response from the entire image. An example system following this approach with Gabor wavelets was developed by Vukadinovic and Pantic [2005]. Other forms of appearance based features includes *Principal Component Analysis* (PCA), *Independent Component Analysis* (ICA), local features such as LBP and the like. The filter approaches are often referred to as component based approaches and the approaches based on PCA/ICA/LBP etc. are often referred to as holistic methods.

It is noted by Caleanu [2013] that the highest recognition rate seems to occur when geometric features are used together with appearance based features. As an example, Hsu et al. [2013] proposed a system which achieves the highest recognition rate when the feature types are combined. Caleanu [2013] note, that neither the geometric based features nor the appearance based features seems to perform better than the other. In contradiction to this statement, it is stated in the meta analysis of FERA 2011, that appearance features seem to clearly outperform geometric features [Valstar et al., 2012]. However, they also state that the best performance seems to occur when the features are combined.

Another way to distinguish between feature types is by engineered features or learned features. The features described so far are all engineered features in the sense that they have been explicitly designed by humans to solve a specific task. In contrast, learned features are automatically learned from the input data by a learning algorithm without intervention from humans. A system using learned features were proposed by Ranzato et al. [2011]. They propose to use a *Markov Random Field* (MRF) as the bottom layer of a *Deep Belief Network* (DBN). The MRF can model an image very well on pixel level. Combined with the learning ability of the DBN, they create a system which is fairly robust against occlusion. To make their system achieve recognition rates comparable to systems using engineered features, they need a very large database. Therefore, they need to use the *Toronto Face Database* (TFD) for training [Susskind et al., 2010]. It is reported that the TFD is the largest database of faces to date [Ranzato et al., 2011]. Note however, that it is not explicitly a database of facial expressions. The TFD is used to let their model learn a set of very descriptive face features. The trained model is then used to extract features from a facial expression database. They report an accuracy which outperforms systems using *Scale Invariant Feature Transform* (SIFT) [Lowe, 1999] features and Gabor wavelets for occluded FER. They also state, that they expect learned features to outperform engineered

features even more when larger datasets become available.

2.5 General schemes for recognizing facial expressions

This section describes the general "angles-of-attack" which can be followed when designing an FER system. In addition, it also describes some of the most likely state-of-the-art candidate systems.

By considering the recent FER surveys, it becomes evident that there is two general approaches. Either the system recognizes AUs and use those to determine the expressions, or it uses the global approach where the expressions are recognized directly from the images. The following text uses the baseline system proposed for FERA 2011 to describe these two approaches. Then some examples are given on systems using both approaches, which could very well be the current state-of-the-art for FER. Be advised, however, that the state-of-the-art is constantly changing and that there might be current systems which are unknown to the author of this report.

For FERA 2011, two baseline systems are proposed: one for the AU recognition approach and one for the global expression recognition approach. Since the face detection and feature extraction processes are identical for the two systems, lets start by describing those. The Viola-Jones face detector [Viola and Jones, 2001] is used to detect the face and a similar Haar-cascade eye detector is used to find the eyes within the face. The location of the eyes are used to normalize the size and rotation of the faces. The *Region Of Interest* (ROI) containing the face is then divided into a number of cells which forms a grid. From each cell in the grid, LBPs are extracted and histograms are formed based on their number of occurrences. The histograms resulting from each cell are then concatenated together to form one, large face descriptor. The dimensionality of the descriptor is reduced by PCA and SVMs are used to do the classification.

With the basis sorted out, lets move to the specific baseline system which uses the AU approach. In the baseline system for AU recognition, a separate binary SVM is trained for each AU independently. Thus, the SVMs are trained to only recognize a single AU. The facial expression on a given face is then determined by investigating which AUs were active in that image.

Now, lets consider the other baseline system, namely the one which uses the global approach, In the baseline system for the global approach, a set of OAA SVM classifiers are trained. Each SVM is trained to recognize one of the five basic expressions. Thus, the facial expression can be determined directly from the output of the classifier system.

In the meta analysis of FERA 2011, the organizers states that the most popular form of classifier is SVMs. They were used by 83% of the competitors [Valstar et al., 2012]. As mentioned, the baseline system also uses SVMs together with LBP. This combination was first proposed by Shan et al. [2005]. They used the combination due to previous publications which reported good results when combining SVMs with appearance features.

A system implementing many of the same ideas as the baseline system following the AU

approach was proposed by Velusamy et al. [2013]. They enhance every step of the process, except the classifier. By doing so, they achieve a recognition accuracy which seems to be the state-of-the-art in the AU approach.

An example of a state-of-the-art system which uses the global approach could be the one proposed by Kabir et al. [2010]. They use a local feature inspired by LBP called *Local Directional Pattern variance* (LDPv). Whereas LBP considers the pixel intensities directly, LDPv extracts the edge response of the image texture before constructing patterns. By doing so, it becomes more robust against noise than LBP. Besides changing the feature extractor, their method is more or less similar to the baseline system of the FER challenge. Another system which could also be the state-of-the-art was proposed by Zhen and Zilu [2012]. They base their system on the LPQ feature descriptor [Ahonen et al., 2008]. The novelty of the LPQ feature is that it computes the local Fourier Transform around every pixel. The phase information of four frequencies are extracted from the local Fourier response and that information is used to construct patterns similar to those seen in LBP. The LPQ feature is more robust against blurred degradation of the image than LBP and LDPv. Zhen and Zilu [2012] proves that LPQ outperforms LBP, but they do not compare their findings against LDPv. It can be noted that Yang and Bhanu [2011] and Yuan et al. [2012] achieved very promising results by combining LBP and LPQ.

As noted previously, some parts of the face conveys more information about the facial expressions than other parts. Ahonen et al. [2004b] used this information to construct a face recognition system based on LBP. The system uses Chi-square statistics to compare training and test histograms extracted using the image grid division approach. The similarity measure of each histogram is weighted depending on the position of its cell in the image. Shan et al. [2009] use a similar approach for FER.

As noted in the introduction to FERA 2011 [Valstar et al., 2011], it is very difficult to compare different FER methods due to the lack of a standard facial expression test-set. The lack of a standard was also pointed out by Khan [2013]. As a consequence, it is difficult to actually determine what the current state-of-the-art is. Further, it is also difficult to determine which of the AU approach or the global approach is the better. By considering the list of methods compared by Caleanu [2013], it seems that both approaches achieve equally good results. The outcome of FERA 2011 was presented in a meta analysis done by Valstar et al. [2012]. Based on the submissions for the challenge, they conclude that the global approach is more popular than the AU recognition approach, but they do not state which is best. They do however state, that existing systems already provide good results on the global approach, but that far more research is needed on the AU approach.

2.6 Databases of facial expressions

A quite substantial number of facial expression databases exists. This section will define a set of parameters based on the existing databases, by which the characteristics of a given database can be described.

As mentioned above, no database has gained the status of being the "standard". Different

researchers uses different databases or different subsets of the same database. As mentioned previously, this is a problem when comparing different FER methods. A result published using one specific database can not be held directly against another result obtained using a different database. The reason is due to the different parameters of the different databases. As an example, it is not possible to compare one study which report a high recognition accuracy using a database with a larger number of samples against another study publishing a lower accuracy using a database with a low number of samples. The higher accuracy of the first study may arise simply because the study used more training data which led to a better fitting of their model. Another problem could be differences in how explicit the facial expressions are in different databases. One database could contain images of persons who overplayed the expressions a lot, making them easier to recognize. Another database might have put emphasis on realistic, subtle expressions which would make them harder to recognize. By doing this simple example based reasoning it can clearly be seen, that it is urgent to develop a standard test set. FERA 2011 tried to do just that [Valstar et al., 2011]. Though their relatively small dataset has been applied by some authors [Dhall et al., 2011; Dahmane and Meunier, 2014], it has been noted that a larger standard test-set is still needed [Valstar et al., 2012].

As noted by Bettadapura [2012], it is immensely difficult to obtain good, lab-environment grade images of facial expressions which are based on actual emotions. He refers to a study which tried to obtain true facial expressions by putting up a video kiosk in a pedestrian street. The kiosk would show videos which were intended to wake certain feelings in the viewers. The viewers were secretly video filmed while watching the videos. After the videos had played, the viewers would be told that they had been recorded and asked if the researchers could use the video for scientific purposes. The researchers found that it was difficult to induce facial expressions on viewers simply by videos and some conflicting examples were obtained, e.g. a person looking sad when they actually felt happy. Despite these difficulties, the study resulted in the database which is now know as the *Spontaneous Expression Database* [Sebe et al., 2007]. Due to the difficulties in obtaining true emotion-based facial expressions, most facial expression databases use actors which are instructed to do facial expressions by an instructor. The acted expressions might be quite different from the true expressions. Therefore it has been argued that many of the systems proposed so far would not work nearly as well in the real world as they do on paper [Fasel and Luettin, 2003]. Furthermore, none of the known databases uses expressions obtained while the subjects were speaking. This is a problem, because as Fasel and Luettin [2003] noted, most facial expressions seems to occur during speech and social interaction.

Based on the databases described by Khan [2013], Bettadapura [2012] and Caleanu [2013], the following database describing parameters are defined:

Parameter	Value	Description
Nr. of subjects	$\mathbb{N}$	The number of subjects contained in the database.
Data type	Video or Image, and 2D or 3D	The type of input data.
Nr. of samples	$\mathbb{N}$	The number of images or videos in the database.
Expressions	List of expressions	The different expressions contained in the database.
Expression type	Spontaneous or Posed	How the subjects made the expressions.
Obstructions	Yes or No	If the images contain obstructions such as glasses or scarfs.
Labels	AU coded, discrete labels or none	How the samples are labeled.
Gender	% female	The amount of female subjects in the database.
Age span	$\mathbb{N}$ to $\mathbb{N}$	The age span of the subjects in the database.
Ethnicity	% of different ethnicities	The ethnic combination of subjects in the database.
Background	Simple or Complex, and Constant or Varying	The background type used in the images.
Lighting	Constant or Varying	How the lighting changes between different samples.
Color	Yes or No	If the images are colored or gray scale.
Poses	List of angles	Azimuth and Elevation angles used to photograph subjects in the database.
Price	$\mathbb{R}$	The price of the database.

Table 2.1. Definitions of descriptive parameters concerning facial expression databases.

The parameters defined in Table 2.1 can be used to select a proper database based on the requirements of the study. Furthermore, they can also be used to highlight the differences between different databases. Therefore, they can be used to compare two databases with each other in a sensible way.

Trough a literature survey, the following databases has been found:

1. CMU-Pittsburg Database (also known as the Cohn-Kanade database) [Kanade et al., 2000]
2. The Extended Cohn-Kanade database [Lucey et al., 2010]
3. MMI Facial Expression Database [Pantic et al., 2005]
4. Spontaneous Expressions Database [Sebe et al., 2007]
5. The AR Face Database [Martinez and Benavente, 1998]
6. CMU Pose, Illumination, Expression (PIE) Database [Sim et al., 2002]
7. The Japanese Female Facial Expression (JAFFE) [Lyons et al., 1998]

8. The Karolinska Directed Emotional Faces (KDEF) Database [Lundqvist et al., 1998]
9. FER Challenge Dataset [Valstar et al., 2011]

A detailed description of the above databases is not provided here as this chapter is not meant as a detailed lexicon of FER methods. As stated in the beginning, this chapter is meant to provide the reader with an overview of current FER methods and customs. Following that line of thought, a small note about the most used database is provided here [Caleanu, 2013; Khan, 2013], namely the Cohn-Kanade database. It is used in many studies, but it suffers from some fundamental problems. First of all, the database was created to be used in the study of AUs. Therefore, the images are not labeled with basic expression labels. Instead, each image has been assigned AUs. Today however, it is also extensively used for FER studies using the global approach. The authors behind the Cohn-Kanade database has published a dictionary which links AUs to emotional expressions. As an example, if AU 6 and 12 or 12C or 12D is active, then the subject shows a happy emotion. However, some expressions might be less clear. Surprise, for instance, requires AU 1, 2, 5B, and 26 or 27 to be active. Further, it has some varieties where some of the AUs are missing. As a result, it can be difficult for a non trained person to decipher the actual facial expression. Of course one can always judge the expressions by looking directly on the images, but different persons might judge the same expression differently. Thus, different FER authors using the Cohn-Kanade database might use different expression labels for the same images. Some authors might also select a subset of the database due to difficulties in labeling all images with sufficient confidence. As a consequence, authors might actually use slightly different databases, even though they report using the Cohn-Kanade database. As authors does usually not state precisely how they divided and labeled the database, it can be quite difficult to compare different studies which uses the Cohn-Kanade database.

Most of the databases in the above list are free. Some databases are free for research but not for commercial use. An example of such a database is the KDEF database. It has previously been used for FER studies at *Aalborg University* (AAU) and therefore the university already has access to the database. Even though the database is rarely used for FER systems in general, it would make sense to use it here to allow for easy comparison with previous studies done at AAU.

Possible solutions 3

This chapter clarifies the choices which has to be made in order to construct a FER system, based on the previous work outlined in Chapter 2. Following the clarification, an argumentation for making these choices are presented. The chapter culminate in a delimitation which clearly defines the areas of focus selected for this project. The delimitation is supported by the problem statement presented in Section 1.1.

In general, the following choices has to be made:

1. Type of input data
    - Video or still images. 2D RGB, 3D depth, thermal or other.
2. Type of recognition
    - AU approach or global expression approach
3. Type of feature
    - Geometric or appearance features. Engineered or learned.

Prior to constructing the system, a choice has to be made for all of the above. Of course, a choice could also be to replace the "or's" in the above list with "and's". Under all circumstances, it is important to take the requirements of the finished system as the starting point in order to make these choices wisely.

No comparison of still image approaches against video approaches was encountered during the literature survey documented in Chapter 2. Therefore, it is actually unknown if one is better than the other. However, the video approach contains more information than the still image approach, because it considers the spatial distribution of the facial features in time. Therefore, it would be expected that the video approaches will provide higher recognition rates than the still image approaches. It would also be expected that the video approaches are more computationally heavy than the still image approaches, simply because they have to process more information.

As with the comparison of still images and video, it seems that no studies have been conducted on the advantages of different image spaces. Therefore, it can not be stated which of 2D or 3D is best.

The choice to use either the AU recognition approach or the global recognition approach should be based on the system application. If the system is required to explain the details of the face motions, the AU approach should be used. If instead it is enough to explain only the emotions of the facial expressions, the global approach could be used just as well.

When choosing the type of feature to use, the complete set of tasks required of the finished system must be considered. If the system is required to do other vision tasks besides FER, such as face recognition, the appearance features would be the better choice. The appearance features could be used to solve both tasks by simply changing the classifier. If the system is only required to do FER, then geometric features could be used. However, as described in Chapter 2, appearance features has been proved to outperform geometric features.

The above reasonings form the base for selecting a possible solution for this project, and thus the areas of focus. The rest of this chapter describes which methods has been chosen for investigation in this project. As written in Chapter 1, one of the motivations behind the project is the social intelligent robot currently under development at AAU. Due to this, the choices are made with the social robot in mind.

In general, a robot can be considered as a system with limited computing power. Unless it can maintain a constant connection with a server, it needs to drag all its processing resources with it. This is why the resources needs to be limited. This includes a limit on the processing capability and the amount of memory. It is chosen to use 2D still images as input because they are expected to require less computing power than video. It could be interesting to investigate 3D depth information as well, but that has not been selected for investigation in this project.

Because the robot is social, it will most likely need to do face recognition besides FER. As mentioned previously, appearance features could be used to solve both tasks. Furthermore, appearance features has been proved to outperform geometric features. Therefore, they are chosen as the feature type. Regarding the choice between the AU approach and the global approach, both seems to be an equally good choice. However, very interesting methods based on appearance features using the global approach has been published in recent years. Therefore, it is chosen to compare some of these methods against each other.

Because the human resources involved in this project are fairly limited, not all of the aspects of a complete FER system can be addressed in equal detail. Some of the aspects are selected to be researched in detail, as stated by the problem statement. The selected aspects form the focus points of this project. Consequently, the focus points form the contributions to the field made by this project.

From the literature on FER, it is apparent that LBP is a very popular type of appearance feature. It even forms the basis of the baseline system proposed in the first FER challenge. It has also been shown that it outperforms another popular appearance feature, namely the Gabor wavelets. However, in recent years, even better local features has been proposed by several authors. Of these, particularly the LDPv and the LPQ features has been reported to yield promising recognition rates. A new type of local feature called *Local Frequency Descriptor* (LFD) was proposed by Lei et al. [2011]. From the previous work uncovered by the literature survey documented in Chapter 2, no previous implementation using LFD for FER has been found. Besides their supposedly higher recognition rates, one of the main benefits of LPQ and LFD is their robustness against image blurring. As noted previously, FER under non-laboratory conditions is a field which needs more research. This project will seek to meet this demand by constructing a blur-invariant system.

As outlined by the problem statement in Section 1.1, this project will seek to determine which of LBP, LDPv, LPQ or LFD performs best for FER with blurred test images. Because the focus is on the feature extractor, some delimitations is made. First of all, this project will concern still images instead of video. Secondly, the seven basic expressions will be recognized by the discrete expressions. In order to compare the feature extractors, the rest of the system needs to remain constant. Therefore, only two types of dimensionality reduction methods are tried along with just a single classification method.

As mentioned previously, PCA is a popular method for dimensionality reduction of local features. Shan et al. [2009] proves that PCA provides better recognition rates than another popular dimensionality reduction method called *Linear Discriminant Analysis* (LDA). There exists a promising clustering method called *Spectral Clustering* (SC). It transforms the data to a new space in which similar data samples forms tight clusters. The new space can be of lower dimensionality than the original, and therefore be regarded as a feature space of lower dimensionality. Here, it is proposed to use SC as a competitor to PCA for dimensionality reduction.

To test the system, it is chosen to use the Cohn-Kanade database and the KDEF database. The Cohn-Kanade database is selected because of its popularity in previous publications. The KDEF database is selected due to its previous use at AAU. The two databases provides two quite different kinds of facial expressions. Even though both databases contains the seven basic facial expressions, they perform them different. The Cohn-Kanade database has very overplayed expressions whereas the KDEF database has more subtle and realistic expressions.

The rest of the report is organized so that it follows the flow of a typical computer vision system as illustrated in Figure 3.1. First, the preprocessing of the raw images is described in Chapter 4. Second, all four feature extractors are covered in Chapter 5. Third, Chapter 6 explains PCA together with SC. Fourth, the SVM classifier is explained in Chapter 7. At last, Chapter 8 provides details concerning the experiments developed to test which feature works better. The chapter also includes a discussion of each experiment and the results. The report ends with a conclusion in Chapter 9.

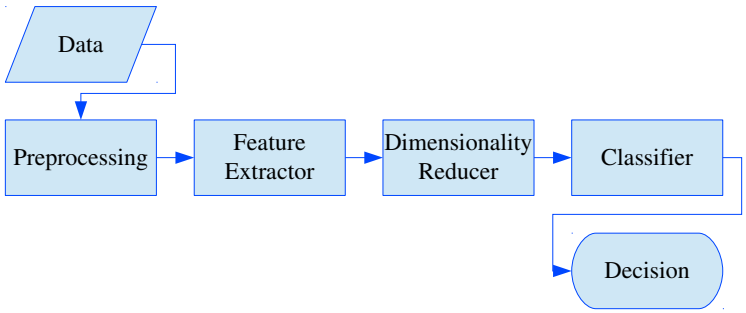

Figure 3.1. The flow of a typical computer vision system. The preprocessing optimizes the raw data prior to feature extraction. After the feature extraction, the feature descriptor dimensionality is reduced. Then, a classifier estimates the most likely class of the data. Finally, a class-decision is made based on the output of the classifier.

Preprocessing 4

This chapter describes the preprocessing done to the raw input images. As will become evident in Chapter 5, one of the benefits of local features is their robustness against illumination changes. In order to utilize this robustness, no lighting normalization such as histogram stretching is done. Two example images from the Cohn-Kanade and KDEF databases is illustrated on Figure 4.1 and 4.2.

Figure 4.1. Example of a surprised facial expression as illustrated by the Cohn-Kanade database.

Figure 4.2. Example of a surprised facial expression as illustrated by the KDEF database.

All faces in both databases are aligned similarly to the ones showed above. It is stated by Shan et al. [2009], that no face alignment normalization is necessary for such images. Both images contain elements which are not conveying information about the facial expression. This could be the background, the hair and the body. In this project, these elements can be considered as noise. The segmentation process proposed by Shan et al. [2009] removes this noise. The same process was also applied by Kabir et al. [2010] and Singh et al. [2012], among others. In this project, the same process is applied due to its simplicity and previous success.

The remainder of this chapter is organized as follows: Section 4.1 describes the segmentation process and Section 4.2 describes an image grid partitioning scheme.

4.1 Segmentation

In short, the segmenter must be able to remove as many unwanted parts of the images as possible. In other words, it is desired to crop the images so they only contain the faces.

First step is to detect the location of the face. There exist quite a number of ways to do so, but one method which has proven to be reliable yet simple is the Viola and Jones method [Viola and Jones, 2001]. They use a set of rectangular box features which can detect the face.

Actually, box features can be designed to detect a number of different objects. In both the KDEF and Cohn-Kanade databases there is only one person in each image. Therefore, the Viola-Jones features is used to detect just the eye-pairs and not the face. In the segmentation process proposed by Tian [2004], the size of the images is normalized by setting the distance between the eyes to a fixed number. In this project, the eye-distance is normalized to 100 pixels. Following Shan et al. [2009], the normalized images are cropped to a fixed size of 110 pixels by 150 pixels. The *Region Of Interest* (ROI) is placed so that the center-point between the eyes is placed vertically 1/4 down from the top of the image and in the middle horizontally. The size normalization is done to ensure, that the images has the same number of pixels. This will help the feature extractor to extract identical features from images containing identical facial expressions.

The eye-detection, normalization and cropping operation is done for all images in the databases. If however there should be an image where the eyes can not be found or if multiple eyes are found, an error routine steps into action. The segmentation process is implemented as a semi-automatic process. The error routine will ask the user to manually label the eyes if none were found. If multiple eyes were found, the system will judge by the size of the bounding box surrounding the eyes. First, bounding boxes which are more square than rectangular will be removed. Then boxes with a very small area is removed. If there are still more than one bounding box left, the user will again be asked to select the proper one or manually select the eyes if no bounding boxes are left.

The transformation done by the segmentation process is illustrated in Figure 4.3.

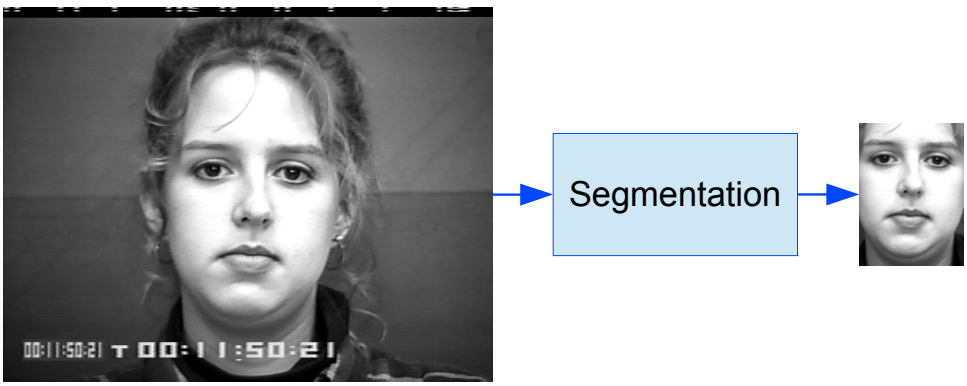

Figure 4.3. Example of the segmentation process. The image on the left is the input and the image on the right is the output. Note that the size of the output face is slightly smaller. This is due to the normalization of size.

4.2 Grid-partitioning of the images

As will be explained in Chapter 5, a general weakness of local features is their lack of spatial position information. The feature descriptors do not state anything about positions. Ahonen et al. [2006] states that the spatial layout of the face is important in face recognition and Shan et al. [2009] states that the layout is also important in FER.

Hadid et al. [2004] proposed to encode the locality information into local features by dividing the face images into four overlapping regions. One region placed over the mouth, one placed over the nose and two placed over either eye. Then they extracted local feature descriptors from each region, which they pooled together to form a final face descriptor. They used the system for face recognition and proved that the pooled descriptor works significantly better than the original descriptor. A couple of months after the first paper was published, the team published another one [Ahonen et al., 2004a]. The new paper introduced the idea of dividing the face image into a rectangular grid composed of non-overlapping cells. Both papers concerned LBP features. They constructed the pooled descriptor by concatenating the LBP descriptors from each cell together. The process is illustrated on Figure 4.4.

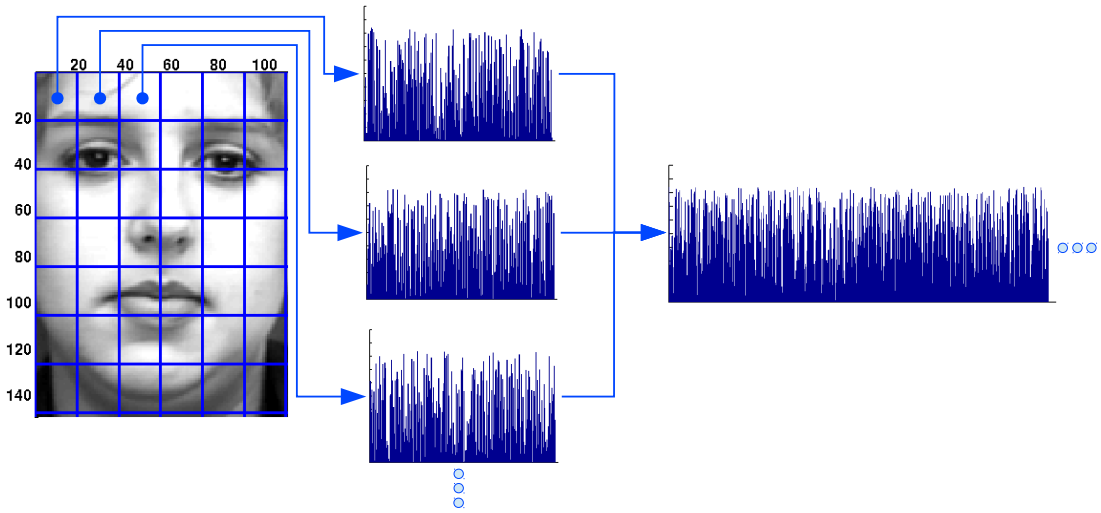

Figure 4.4. An illustration of how the image descriptor is constructed from several smaller descriptors. For convenience, only three local feature descriptors from three grid cells are shown. In reality, the local feature descriptors are extracted from every cell and then concatenated together to form one, large image feature descriptor. Note that the histograms displayed here is just for illustration. They do not originate from any real feature extractor.

The second paper also introduced the idea of given more weights to cells which contains most information. Consider the grid-separated image to the left on Figure 4.4. Intuitively, it makes good sense to weight the cells which contain the eye, nose and mouth parts higher, because they are expected to contain more facial expression information than the other cells. Indeed Ahonen et al. [2004a] reports a better recognition rate for faces when weighting these cells higher. They follow the method proposed by Ahonen et al.

[2004b], and use a nearest-neighbor classifier. The dissimilarity measurement between the histograms is done by Chi-square statistics, in which the cell weights are introduced as an extra parameter. The weights are chosen based on the class separation ability of each cell. Cells which poorly separated the classes are given a weight of 0, meaning that they do not contribute to the classification. Cells with an average separation ability is given a weight of 1. Cells with a separation ability above average is given a weight of either 2 or 4, depending on how good they perform.

This project uses SVMs to do the classification because they have proved to be superior to weighted Chi-statistics [Shan et al., 2009]. As far as the author is aware, no weight method has been proposed for SVMs. Therefore, this project uses unweighted image grids.

Lets clarify the grid settings used in this project and summarize at the same time. All of the four local features considered in this project uses histograms as their feature descriptor. By simply concatenating the histograms from all cells in a given image, a high dimensional feature descriptor is formed which contains some locality information. The spatial locality information is incoded into the different dimensions of the final descriptor. The optimal grid size depends on the size of the images. Naturally, a fine-grade grid will provide a high amount of locality information but also a very high dimensional feature descriptor. In addition, a very fine-grade grid can prevent the feature extractor from recognizing some features if the image resolution is low. This will result in a low recognition rate. If the grid is too coarse on the other hand, too little locality information is included and the performance will also be less than optimal. Some studies have tried to find the optimal grid size, but it is dependent on the size of the images. Jabid et al. [2010b] tested various grid sizes using the LDP feature and concluded that 7×6 was the most optimal. They used images similar in size to the ones used in this project, namely 150×110 pixels. Therefore, each cell in the optimal grid is roughly 21 pixels high and 18 pixels wide. This is the same optimal cell size as reported by Ahonen et al. [2004a]. As a consequence, this project will also rely on a grid size of 7×6.

Feature Extraction 5

This chapter describes the LBP, LDPv, LPQ and LFD feature extractors. First, a small summation is given on why local features has been chosen in general. Second, in the following four sections, each of the feature extractors are described. The descriptions includes details about why the features work, how the features work and their pros and cons.

Local features are a type of appearance features, which has been chosen due to versatility. By changing the classification scheme, the appearance features can easily be used for multiple recognition tasks. Moreover, appearance features tend to produce better recognition rates than geometric features.

LBP has been used extensively for FER with very promising results. They have proven superior to Gabor Wavelets features, which is another popular type of local feature. As a consequence, this project takes LBP as its starting point. Proposed in 2012, LDPv is a fairly new feature which has not yet received much attention. However, LDPv has already been reported to be superior to LBP. LDPv can be seen as an extension of LBP. LPQ and LFD has a notably different approach than LBP and LDPv. LBP and LDPv exists in the spatial domain whereas LPQ and LFD exists in the image frequency domain. LPQ has already provided better results for FER than LBP. Especially its robustness against image blur is a desired trait. LFD was developed as an extension of LPQ. It has provided promising results for face recognition, but as far as the author is aware, it has never been applied to FER.

5.1 Local Binary Pattern

This section describes the LBP feature. LBP was proposed by Ojala et al. [1994] and further refined by Ojala et al. [2002]. It continues to be a hot topic among texture recognition researchers. Hadid et al. [2004] first proposed using LBP for face recognition and Feng [2004] first proposed using LBP for FER.

As described by Shan et al. [2005], the LBP feature extractor describes a texture as a combination of micro-patterns. They also state that the texture of a face can be well described by these micro-patterns. The LBP descriptor is robust against illumination changes and the LBP feature extractor is computationally efficient. Therefore, they seem to be a good choice for FER.

As noted previously, LBP belongs to a special type of appearance features called local

features. This implies, that LBP considers local pixel regions in an image. In short, LBP performs a thresholding locally on all pixels in a grey-scale image, based on their surrounding pixels. From the thresholding of a given pixel, a set of binary values are formed which in combination acts as a unique code. The code describes the directions of the slopes formed by the pixels surrounding the center pixel. This process is explained on Figure 5.1.

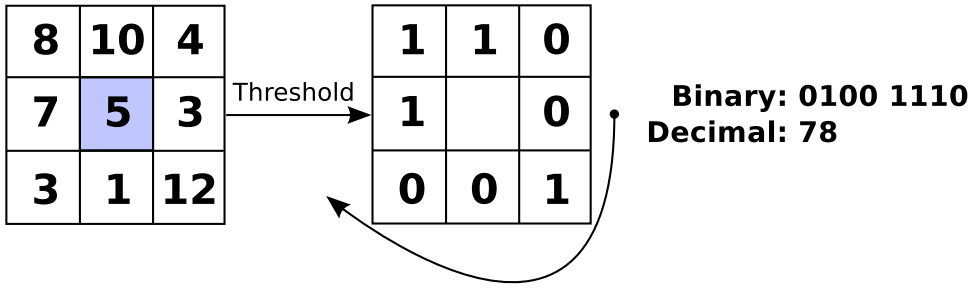

Figure 5.1. Illustration of how the LBP feature extractor calculates codes based on the local region around a pixel. The patch to the left illustrates a local region around a pixel. The center pixel is marked with blue. A thresholding operation is performed for each surrounding pixel, starting at the right, middle pixel and moving in a clockwise direction. If a surrounding pixel have a higher intensity than the center pixel, a 1 is placed in its place. Else a 0 is placed. The resulting 8-bit binary number is converted to a decimal number which is the "code" for that local region.

The LBP codes are calculated for all pixels. Thus, if a given image have size $N \times M$, the total number of codes is $NM - 2N - 2(M - 2)$. The $-2N - 2(M - 2)$ part is due to the pixels on the border of the image. They are not included because they are not surrounded by a complete local region. The LBP descriptor is formed by counting the number of occurrences of each pattern and convert that to a histogram.

The LBP codes are like the barcodes in the supermarket. In the supermarket, every product type have its own unique barcode. When the cashier scans a product, the barcode tells the cash register the products type and thereby its price. The barcodes does not code for specific products though. It can only tell that the customer bought a 33 cl Coca-Cola can, but not exactly which can the customer bought. The LBP codes work in a similar way. They can tell the general direction of the slopes surrounding a pixel, but not the exact amount of tilt. To understand this concept, please consider the spatial one-dimension toy-example on Figure 5.2

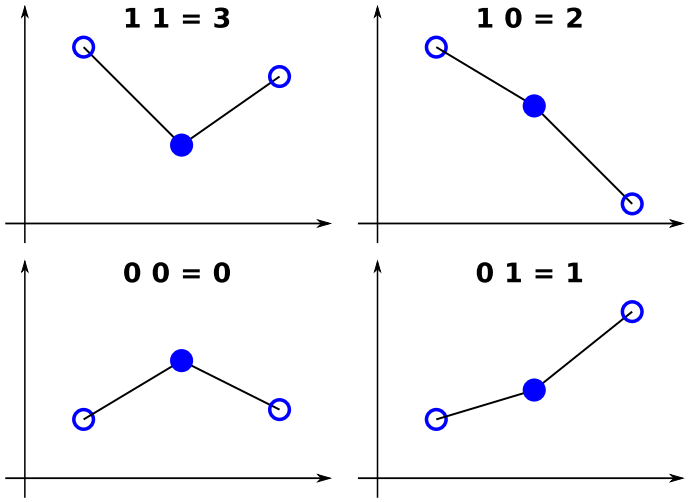

Figure 5.2. One dimensional illustrations of how the LBP features describes texture features of an image. The first-axis illustrates the spatial distribution and the second axis illustrates pixel intensity.

The four graphs shown in the above image are four different examples of a local region in one spatial dimension. The first axis illustrates the distribution along this axis and the second-axis illustrates the pixel intensity. The blue filled circle illustrates the center pixel and the two hollow circles illustrates two surrounding pixels. With only two surrounding pixels, the binary code can attain a total of $2^2 = 4$ different values, namely: 0, 1, 2 and 3. In the first graph at the top left, both of the surrounding pixels have an intensity higher than the center pixel. Therefore, the center pixel can be thought of as a valley. Because both surrounding pixels have a higher intensity, the threshold operation yields a binary number of 11_2. This corresponds of a decimal code of 3_{10}. Thus, every time three points line up to form a valley, they will be assigned the LBP code of 3. Said in another way, every time an LBP code of three is encountered, it is known that the underlying pixel forms the bottom of a valley. It is not possible to say anything about how deep the valley is, just that it is a valley.

LBP codes can be deduced from the rest of the graphs in a similar way. The center pixel on the graph at the top right forms a midpoint on a down going slope. The LBP code for this situation is 2. Like before, every time a 2 is encountered, it is known that the underlying pixel is situated on the middle of a down going slope. The center pixel on the graph at the bottom left is situated at a top point. The LBP code for a pixel located on a top is obviously 0. Last but not least, the center pixel on the graph at the bottom right forms a midpoint on an up going slope. The corresponding LBP code for up going slope is 1.

The simple toy-example above can be expanded to two spatial dimensions. Indeed, the LBP codes in 2D explain the slope layout of the local region around the pixels. In 2D, the possible number of codes is $2^8 = 256$ because there is eight surrounding pixels and thus eight bits to be converted to a decimal number. When working with the LBP codes, Ojala et al. [2002] discovered that the occurrences of the LBP codes are not equally distributed. Actually, it turns out that a particular type of codes accounts for about 90% of all codes.

They named these codes *uniform patterns* based on the layout of their bits.

Consider the following 8-bit binary number: $0110\ 0111_2$. It has three transitions from 0 to 1 or from 1 to 0. Let $U(\text{"patterns"})$ define a measure of how many transitions is in a given binary number. Thus, as was seen before $U(0110\ 0111_2) = 3$ and $U(0101\ 0101_2) = 7$. The uniform patterns are those that satisfies: $U(x) \leq 2$, where x is an 8-bit binary number.

Because of the abundance of uniform patterns, the LBP descriptor can safely be simplified to only consider uniform patterns without loosing a significant amount of descriptive power [Ojala et al., 2002].

Up until this point, the description of the LBP operator has considered only the eight pixels in immediate contact with the center pixel. Beside introducing the uniform patterns, Ojala et al. [2002] also introduced the concept of expanding the pixel region in consideration. In doing so, they defined the *radius* and *number of sampling points*. Consider Figure 5.3 for an example.

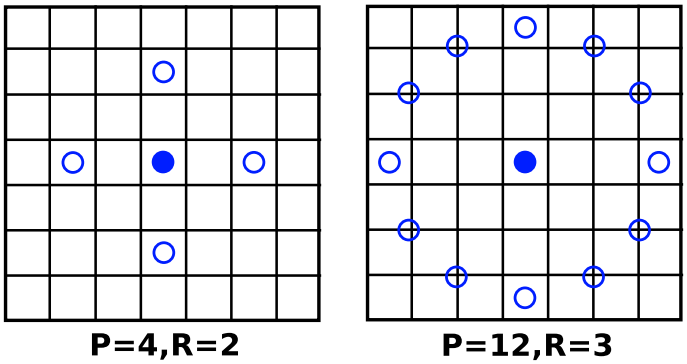

Figure 5.3. Example of circularly symmetric neighbor sets. The squares indicates pixels. The filled blue circle indicates the center pixel and the hollow circles illustrates the sampling points. Two different combination of number of patterns (P) and radius (R) is shown. Note that many more exists.

As seen on the above figure, the sampling points are aligned in a circular pattern with the center pixel at the center of the circle. If the center pixel is considered as location $(0, 0)$, the coordinate of sampling point n will be given by: $(-R\sin(2\pi n/P, R\cos(2\pi n/P))$. If a sampling point falls outside of the center of a pixel, the intensity of that location is found by interpolation. In principle, an infinite number of combinations of sampling points and radii exists. The only thing putting a constraint on the number is the size of the image.

The following notation is introduced to specify which kinds of LBP is used: $\text{LBP}^{u2}_{P,R}$, where P denotes the number of sampling points, R denotes the radius and $u2$ denotes that only the uniform patterns are used. Following this notation, the LBP operator on the left of Figure 5.3 is $\text{LBP}_{4,2}$ and the operator on the right is $\text{LBP}_{12,3}$. The operator showed in the beginning of the section on Figure 5.1 is $\text{LBP}_{8,1}$.

Finally, Ojala et al. [2002] also introduced the rotation invariant LBP descriptor. In $LBP_{P,R}$ and $LBP^{u2}_{P,R}$, the first bit is always the on to the left of the center at position $(0, R)$. Therefore the patterns will change if the underlying image is rotated. To make the

patterns rotation invariant, the following operator is defined:

$$LBP^{ri}_{P,R} = \min\{ROR(LBP_{P,R}, i) \quad | \quad i = 0, 1, \cdots, P-1\} \tag{5.1}$$

Where:

$ROR(\cdot, i)$ performs a bit-wise circular rotation i times.

The above operator rotates the extracted pattern until its value is minimum. This amounts to selecting the pattern with as many most significant bits equaling zero as possible. As a result, the number of different patterns in $LBP^{ri}_{P,R}$ is lower than the number of different patterns in $LBP_{P,R}$. As an example, the $LBP^{ri}_{8,1}$ extractor only has 36 different patterns. This number is further reduced if only the uniform patterns are considered. In fact, $LBP^{ri\,u2}_{8,1}$ only has 8 different patterns.

As mentioned earlier, the LBP descriptor is formed by counting how many times each LBP code occur. This corresponds to counting the number of edges, lines, spots and flat areas contained in the image. A histogram is formed based on the counting, which is used as the feature descriptor. The dimensionality of the descriptor depends on the choice of P.

The intensity robustness of the LBP descriptors comes from the thresholding procedure. If the overall intensity of an image is increased, the relative intensities between the pixel intensities will remain the same. The same is of cause true if the overall intensity is lowered.

In this project, $LBP^{u2}_{8,1}$ is used. The algorithm for extracting the LBP features is shown below.

Algorithm 1 Algorithm which extracts the $LBP^{u2}_{P,R}$ features from an image.

Require: Img,P,R
Ensure: $LBP^{u2}_{P,R}$
1: $D \in \mathbb{R}^{\text{Img}}$
2: **for all** pixels n in Img except the R border pixels **do**
3: **for all** pixels s in the P surrounding pixels at distance R **do**
4: S_I = Interpolated intensity at position s
5: **if** $S_I > Img(n)$ **then** $b = 1$
6: **else**$b = 0$
7: **end if**
8: $D(n) = D(n) + b^s$
9: **end for**
10: **end for**
11: H = histogram of D
12: $LBP^{u2}_{P,R}$ = histogram containing only the uniform patterns from H

5.2 Local Directional Pattern variance

This section describes the LDPv feature extractor. LDPv is an extension of LDP, which in turn can be considered as an extension of LBP. LDP was first proposed for face recognition

by Jabid et al. [2010a]. In another paper from the same year, Jabid et al. [2010b] proposed using LDP for FER. The same research group extended LDP and created LDPv which they applied directly to FER. Their findings was published by Kabir et al. [2010].

Where LBP extracts its codes directly from the pixel intensities, LDP starts by transforming the image into the directional edge responses using the eight Kirsch masks. After that, binary codes are extracted based on the magnitudes of the responses. LDPv enhances the descriptiveness by introducing the variance of the edge response as a way to weight the LDP descriptor.

The Kirsch masks are a set of eight image filters, which derives the directional edge responses. The eight directions are: east, north-east, north, north-west, west, south-west, south, and south-east. The kernels of the eight filters are defined as follows:

$$M_0 = \begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} \quad M_1 = \begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix} \quad M_2 = \begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \quad M_3 = \begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$$

$$M_4 = \begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix} \quad M_5 = \begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix} \quad M_6 = \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix} \quad M_7 = \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}$$

By closely investigating the kernels, it can be seen that their numbers rotate in a counter-clockwise direction. Each kernel derives an edge responses in one of the specified directions when they are convolved with the image. Take M_1 as an example. It has three 5's to the right and five -3's in the middle and to the left. Thus, it derives the east oriented edge response. Note that the sum of the -3's is $5 \cdot -3 = -15$ and the sum of the 5's is $3 \cdot 5 = 15$. Thus, the opposing sides of the kernels are equally weighted. Each 3×3 pixel patch in the image is convolved with each of the eight Kirsch masks. For a given image, the calculation is

$$R = (I * M_n) \quad \forall \quad n = 1, \cdots, 8 \tag{5.2}$$

Where:

I is the image.

M_n is Kirsch mask number n.

The convolution process for an example image patch is illustrated in Figure 5.4.

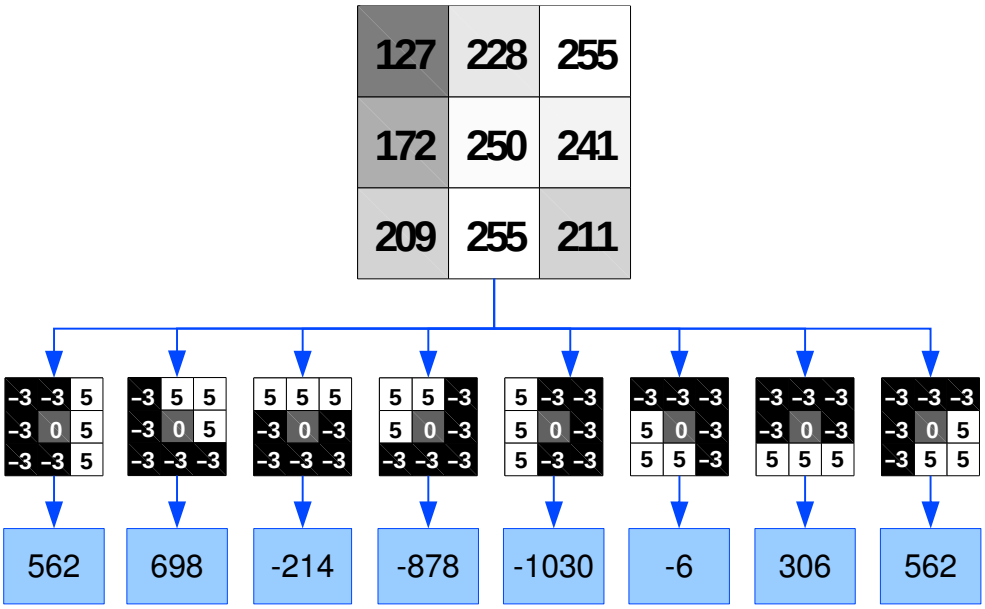

Figure 5.4. Example of a 3×3 pixel patch being convolved by eight Kirsch masks. The outputs of the convolution processes are the directional responses.

An eight bit binary code is formed based on the directional responses. This is done by setting the k largest responses to 1 and the rest to 0. The sign of a response tells if the slope in the direction of the respective Kirsch mask is up or down facing. This information is not a part of the LDPv descriptor. Therefore, the absolute values of the responses are used when forming the binary code. Thus, bit number n is given by:

$$b_n = \begin{cases} 1 & \text{if } |R_n| \geq \max\limits_i(|R_i|, k) \\ 0 & \text{otherwise} \end{cases} \tag{5.3}$$

Where:

$\max\limits_i(|R_i|, k)$ is absolute value of the k largest response.

Before using the LDP feature extractor, one has to select a value for k. Often $k = 3$ is used. In the example on Figure 5.4, the third largest response is 698. Therefore, the binary LDP code is 0101 1000 for $k = 3$. Note that there is three 1s in the binary code, the same number as k. Indeed, this is the case for all ks. The LDP code can therefore attain:

$$\begin{pmatrix} 8 \\ 3 \end{pmatrix} = \frac{8!}{3!(8-3)!} = 56 \tag{5.4}$$

different values. For a given image, the LDP codes are extracted for each pixel not located on the image border. As with LBP, the binary number is converted to a decimal number code. The number of occurrences of each code is counted and formed into a histogram which is used as the feature descriptor. According to the LDP definition, the length of the descriptor is 56 for $k = 3$. The LDP descriptor for a given k is designated as: LDP_k.

However, consider the change if $k = 4$ is chosen for the example in Figure 5.4. The fourth largest response is 562. Following the procedure of Equation 5.3, the binary LDP code is: 1101 1001 for $k = 4$. Because the responses contains two identical numbers, the resulting binary code contains not four but five 1s. Thus, the number of attainable values is higher than $\frac{8!}{4!(8-4)!} = 70$. This poses a problem when deciding the length of the descriptor. So far, it seems that there has not been published a solution to this problem. Two obvious solutions exists. Either, the feature descriptor could simply be 256 dimensions long and thus contain all possible values with eight bits. Most of the dimensions would however be 0. Else, only the first observed repeater could be set to 1 and other repeaters could be set to 0. In this case, the pattern on Figure 5.4 would be: 1101 1000 for $k = 4$. This would also solve the problem, but some information would be lost.

Like LBP, LDP is also robust against illumination changes. Furthermore, LDP is also more robust against random noise than LBP [Jabid et al., 2010a]. The LBP codes can change fairly easy if the images are exposed to noise. Therefore, noise has a large impact on the recognition rate when using LBP. Consider Figure 5.5 for an example of noise robustness.

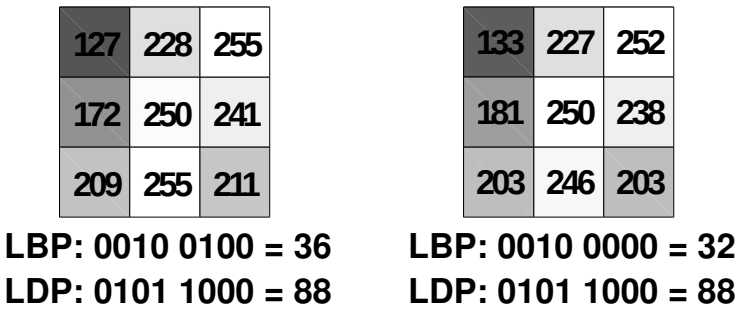

Figure 5.5. Example of how the LBP and LDP codes change if an image is exposed to random noise. To the left is the original image patch and its codes. To the right is the same image patch but with a little random noise added to the pixel intensities. As seen, the LDP code stays constant, but the LBP code changes. This is a general weakness of the LBP feature.

When considering texture it is usually so, that areas containing a lot of high-contrast texture also contains more information. Conversely, areas with low-contrast contains less information. Consider the face on Figure 5.6. The texture at the eyes obviously contain more information about the facial expression than the cheeks.

Figure 5.6. Example of a face which illustrates that areas with a lot of high-contrast texture like eye-brows, eyes, mouth, etc. has a larger pixel intensity variance than e.g. the cheeks. Thus, it is intuitively assumed that these areas contains more information.

The LDP codes will not take the amount of contrast into account. A low contrast area will attain the same amount of importance as a high contrast area. LDPv tries to assign more weight to high contrast areas and lower weight to low contrast areas. The variance of the edge responses will be large in a high contrast area and small in a low contrast area. Therefore, the variance, σ, is introduced as an adaptive weight of the LDP codes in the histogram generation process. For a given pixel at position (r, c), the variance of the edge responses can be calculated as:

$$\sigma(r,c) = \frac{1}{8}\sum_{n=0}^{7}(R(r,c)_n - \mu(r,c))^2 \tag{5.5}$$

Where:

$R(r,c)_n$ is edge-response number n at position (r,c).

$$\mu(r,c) = \frac{1}{8}\sum_{n=0}^{7} R(r,c)_n$$

Let T be the set of possible codes which LDP_k can attain. For a given image of size $M \times N$, the LDPv_k descriptor can be calculated as:

$$\text{LDPv}_k(\tau) = \sum_{r=1}^{M}\sum_{c=1}^{N} w(\text{LDP}_k(r,c),\tau) \quad \forall \quad \tau \in T \tag{5.6}$$

Where:

$$w(\text{LDP}_k(r,c),\tau) = \begin{cases} \sigma(r,c) & \text{if } \text{LDP}_k(r,c) = \tau \\ 0 & \text{otherwise} \end{cases}$$

The LDPv descriptor tries to incorporate both texture and contrast information. Kabir et al. [2010] proves that LDPv is superior to LDP for FER.

Following the procedure of Kabir et al. [2010], k is selected as $k = 3$ in this project. The algorithm for extracting the LDPv descriptor is presented as Algorithm 2 on page 32.

Algorithm 2 Algorithm which extracts the $LDPv_k$ descriptor from an image.

Require: Img,k
Ensure: $LDPv_k$
$M \in \mathbb{R}^{8\times3\times3}$
for $n \leftarrow 1$ to 8 **do** $M_n \leftarrow$ Kirsh mask n
end for
$D \in \mathbb{R}^{\text{Img}}$
$\sigma \in \mathbb{R}^{\text{Img}}$
for all rows, r, in Img except border **do**
    for all columns, c, in Img except border **do**
        $R \in \mathbb{R}^8$
        for $n \leftarrow 1$ to 8 **do**
            $R_n \leftarrow Img([r-1; r+1], [c-1; c+1]) * M_n$
        end for
        for $n \leftarrow 1$ to 8 **do**
            if $R_n \geq \max(R, k)$ **then** $b \leftarrow 1$
            else$b \leftarrow 0$
            end if
            $D(r, c) = D(r, c) + b^{n-1}$
        end for
        $\sigma(r, c) \leftarrow \text{var}(R)$
    end for
end for
$LDPv_k \in \mathbb{R}^{256}$
for $\tau \leftarrow 1$ to 256 **do**
    for all rows, r, in Img except border **do**
        for all columns, c, in Img except border **do**
            if $D(r, c) == \tau$ **then** $LDPv_k(\tau) \leftarrow LDPv_k(\tau) + \sigma r, c$
            end if
        end for
    end for
end for

5.3 Local Phase Quantization

This section describes the LPQ feature extractor. LPQ was developed by Ojansivu and Heikkilä [2008] as a highly robust texture descriptor. It is invariant to both illumination and blurring. Ahonen et al. [2008] proposed to use LPQ for face recognition. Yang and Bhanu [2011] and Dhall et al. [2011] simultaneously proposed using LPQ for FER. Yang and Bhanu [2011] works with video data from which they create what they call the *Emotion Avatar Image* (EAI) for each face. From the EAI, they extract both LBP and LPQ features which are fed into a linear kernel SVM for classification. The method proposed by Dhall et al. [2011] was submitted to the FERA 2011 competition. They use pyramid of histogram of gradients features together with LPQ. Therefore, their system combines geometric and appearance features. They use SVMs and largest margin nearest neighbor for classification.

The overall performance of the system achieved fifth place at the challenge.

The main idea behind LPQ is to transform the spatial input image into the frequency domain. By doing so, a very blur invariant local feature can be extracted. A blurred image, $g(\mathbf{x})$, can be regarded as a sharp image, $f(\mathbf{x})$, which has been convolved with a *Point Spread Function* (PSF), $h(\mathbf{x})$:

$$g(\mathbf{x}) = (f * h)(\mathbf{x}) \tag{5.7}$$

Where:

$\mathbf{x}$ is a two dimensional coordinate vector, $\mathbf{x} = [x, y]^T$.

Note that a PSF does precisely what the name implies. It takes a sharp point in an image and spread it over a larger area. Converted to the frequency domain by a Fourier transformation, Equation 5.7 becomes:

$$G(\mathbf{u}) = F(\mathbf{u})H(\mathbf{u}) \tag{5.8}$$

Where:

$\mathbf{u}$ is a two dimensional frequency vector, $\mathbf{u} = [u, v]^T]$.
G is the Fourier transform of g.
F is the Fourier transform of f.
H is the Fourier transform of h.

The magnitude and phase parts of Equation 5.8 can be separated into a multiplication and an addition [kim]:

$$|G(\mathbf{u})| = |F(\mathbf{u})||H(\mathbf{u})| \tag{5.9}$$
$$\angle G(\mathbf{u}) = \angle F(\mathbf{u}) + \angle H(\mathbf{u}) \tag{5.10}$$

The phase of the PSF measures its offset from the center of the image. If the PSF is defined such that $h(\mathbf{x}) = h(-\mathbf{x})$, it is called centrally symmetric. If the PSF is centrally symmetric, the angle of its phase will either be 0 or π. If the angle is 0, $H(\mathbf{u})$ will be real-valued and therefore its phase will make no contribution to $G(\mathbf{u})$. In short, if the PSF, $h(\mathbf{x})$, is centrally symmetric, the following applies:

$$\angle H(\mathbf{u}) = \begin{cases} 0 & \text{if } H(\mathbf{u}) \geq 0 \\ \pi & \text{otherwise} \end{cases} \tag{5.11}$$

$$\Updownarrow$$

$$\angle G(\mathbf{u}) = \angle F(\mathbf{u}) \quad \forall \quad H(\mathbf{u}) \geq 0 \tag{5.12}$$

From Equation 5.12 it is evident, that the phase of the observed, blurred image, $\angle G(\mathbf{u})$ is invariant to centrally symmetric blur at the frequencies where $H(\mathbf{u})$ is positive. According to Banham and Katsaggelos [1997], the $H(\mathbf{u})$ from motion blur and out-of-focus blur can be modeled as a sinc function which also contains negative values. However, the sinc function will be positive in the low frequency part before the first zero crossing. They

further state, that blur from atmospheric turbulence can be modeled as a Gaussian PSF. This results in a Gaussian $H(\mathbf{u})$, which has positive values over the entire spectrum. As noted by Ojansivu and Heikkilä [2008], is is impossible to achieve total blur invariance in practice. However, as long as the size of the PSF is significantly smaller than the image, blur invariance is achievable in the low frequency part of the spectrum.

LPQ relies on the 2D *Short-Term Fourier Transform* (STFT) for transforming the input image into the frequency domain. The 2D STFT computes the 2D DFT over a rectangular $M \times M$ neighborhood, $\mathcal{N}_{\mathbf{x}}$, at each pixel position, $\mathbf{x}$. For input image $f(\mathbf{x})$ at position $\mathbf{x}$ at frequency $\mathbf{u}$, the 2D STFT of size $M \times M$ is defined as follows:

$$F(\mathbf{u}, \mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{N}_{\mathbf{x}}} f(\mathbf{x} - \mathbf{y}) e^{-j2\pi \mathbf{u}^T \mathbf{y}} \tag{5.13}$$

$$= \mathbf{w}_{\mathbf{u}}^T \mathbf{f}_{\mathbf{x}} \tag{5.14}$$

Where:

$$\mathbf{w}_{\mathbf{u},\mathbf{y}} = e^{-j2\pi \mathbf{u}^T \mathbf{y}}$$
$$\mathbf{f}_{\mathbf{x},\mathbf{y}} = f(\mathbf{x} - \mathbf{y})$$

LPQ evaluates the 2D STFT at four frequency points, one for each direction of east, north, north-east and south-east. These frequencies are: $\mathbf{u}_1 = [a, 0]^T$, $\mathbf{u}_2 = [0, a]^T$, $\mathbf{u}_3 = [a, a]^T$ and $\mathbf{u}_4 = [a, -a]^T$. The parameter a is selected such that the frequency is below the point of the first zero crossing of $H(\mathbf{u})$. The following complex vectors is formed for each pixel position:

$$\mathbf{g}_{\mathbf{x}}^c = [F(\mathbf{u}_1, \mathbf{x}), F(\mathbf{u}_2, \mathbf{x}), F(\mathbf{u}_3, \mathbf{x}), F(\mathbf{u}_4, \mathbf{x})]^T \tag{5.15}$$

The above vector is used to form the following vector:

$$\mathbf{g}_{\mathbf{x}} = [\mathrm{Re}\{\mathbf{g}_{\mathbf{x}}^c\}, \mathrm{Im}\{\mathbf{g}_{\mathbf{x}}^c\}]^T \tag{5.16}$$

The calculation of the vector in Equation 5.16 can be written in matrix notation by:

$$\mathbf{g}_{\mathbf{x}} = \mathbf{W} \mathbf{f}_{\mathbf{x}} \tag{5.17}$$

Where:

$$\mathbf{W} = [\mathrm{Re}\{\mathbf{w}_{\mathbf{u}_1}, \mathbf{w}_{\mathbf{u}_2}, \mathbf{w}_{\mathbf{u}_3}, \mathbf{w}_{\mathbf{u}_4}\}, \mathrm{Im}\{\mathbf{w}_{\mathbf{u}_1}, \mathbf{w}_{\mathbf{u}_2}, \mathbf{w}_{\mathbf{u}_3}, \mathbf{w}_{\mathbf{u}_4}\}]^T$$

Because each of the $\mathbf{w}_{\mathbf{u}}$-vectors has as many elements as there are pixels in $\mathbf{f}_{\mathbf{x}}$, $\mathbf{W}$ is an $8 \times M^2$ matrix.

When defining the LPQ feature, Ojansivu and Heikkilä [2008] also proposed a way to decorrelate the coefficients in $\mathbf{g}_{\mathbf{x}}$. The details are not be covered here, but the general idea is presented. The method involves the assumption that the image, $f(\mathbf{x})$, is the result of a first order Markov process. It is further assumed that the correlation coefficient between adjacent pixel values are given by $\rho > 0$. This is used to establish the covariance matrix of $\mathbf{g}_{\mathbf{x}}$. The covariance matrix can then be used to form a whitening transform which will

rotate $\mathbf{g_x}$ so the samples becomes independent. Thus, the information of the quantization process described below will be maximally preserved.

For each pixel position, $\mathbf{x}$, $\mathbf{g_x}$ contains eight numbers. The LPQ binary pattern at position $\mathbf{x}$ is generated by the following quantizer:

$$q_{\mathbf{x},n} = \begin{cases} 1 & \text{if } g_{\mathbf{x},n} \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.18}$$

Where:

$$g_{\mathbf{x},n} \text{ is the } n\text{'th component of } \mathbf{g_x}. \tag{5.19}$$

Note that the pattern generator in Equation 5.18 encodes the phase information based on which quadrant of the complex plane the frequency response is in. This concept is illustrated on Figure 5.7.

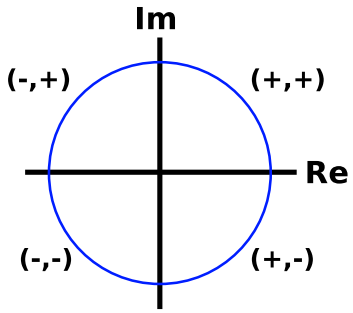

Figure 5.7. Illustration of the complex plane. The blue circle illustrates the unit-circle. Just outside the circle in each quadrant the sign for the real and imaginary part for that quadrant is illustrated. If e.g. a complex point is positioned in the second quadrant, its real number must be negative and its imaginary number must be positive.

Just as in LBP and LDP, the 8-bit binary patterns are converted to decimal numbers in the interval $[0; 256]$. The descriptor is a histogram formed by counting the occurrences of each pattern. There is three parameters which needs to be defined: the window size of the STFT, M, the frequency, a, and ρ. Both Ahonen et al. [2008] and Yuan et al. [2012] uses the following parameters: $M = 7$, $a = 1/7$ and $\rho = 0.9$. Both studies uses image sizes very similar to the one used here. Note that they advice defining $a = 1/M$.

In this project, the following parameters are used: $M = 7$, $a = 1/7$, $\rho = 0.9$. The length of the feature descriptor is 256. The algorithm for extracting the LPQ feature descriptor is presented below:

Algorithm 3 Algorithm which extracts the $LPQ^{s}_{M,a,\rho}$ descriptor from an image.

Require: Img,M,a,ρ,s
Ensure: $LPQ^{s}_{M,a}$
$\mathbf{u}_1 \leftarrow [a,0]^T$; $\mathbf{u}_2 \leftarrow [0,a]^T$; $\mathbf{u}_3 \leftarrow [a,a]^T$; $\mathbf{u}_4 \leftarrow [a,-a]^T$
$\mathbf{W} \in \mathbb{R}^{8\times M^2}$
for $n \leftarrow 1$ to 4 **do**
  for $r \leftarrow 1$ to M **do**
    for $c \leftarrow 1$ to M **do**
      $w \leftarrow \exp\left(-j2\pi\mathbf{u}_n^T\,[r-1,c-1]^T\right)$
      $\mathbf{W}(1+(n-1)2, c+(r-1)M) \leftarrow \mathrm{Re}(w)$
      $\mathbf{W}(5+(n-1)2, c+(r-1)M) \leftarrow \mathrm{Im}(w)$
    end for
  end for
end for
$\mathbf{D} \in \mathbb{R}^{Img}$
for all rows, r, in Img except the $M/2$ border rows **do**
  for all columns, c, in Img except the $M/2$ border columns **do**
    $\mathbf{f} \leftarrow \mathrm{RowConcat}(Img([r-M/2; r+M/2], [c-M/2; c+M/2]))$
    $\mathbf{g} \leftarrow \mathbf{Wf}$
    $\mathbf{g}_d \leftarrow \mathrm{Decorr}(\mathbf{g}, \rho)$
    for $n \leftarrow 1$ to 8 **do**
      if $\mathbf{g}_d(n) \geq 0$ **then**
        $\mathbf{D}(r,c) \leftarrow \mathbf{D}(r,c) + \mathbf{g}_d(n)^{n-1}$
      end if
    end for
  end for
end for
$\mathbf{H} \in \mathbb{R}^{256}$
$\mathbf{H} \leftarrow$ histogram of $\mathbf{D}$
$LPQ^{s}_{M,a} \leftarrow \mathbf{H}$

5.4 Local Frequency Descriptor

LFD was developed by Lei et al. [2011] to be used for face recognition. It is a further development of LPQ. LPQ throws away the magnitude information of the frequency response. LFD tries to incorporate this information into the descriptor. Lei et al. [2011] argues that the magnitude information is important when recognizing faces. A further advantage is that LFD is computed in a way which does not require the blur PSF to be positive to achieve blur robustness. It appears that LFD has not been used for FER so far.

The process of LFD is similar to LPQ all the way up until and including the calculation of the STFT. LPQ achieves its blur invariance by assuming that the phase of the Fourier transform of the blur PSF is 0 in the low frequency part of the spectrum. The following

relation for the phase and magnitude of the Fourier transform of the blur PSF was defined in Section 5.3:

$$|G(\mathbf{u})| = |F(\mathbf{u})||H(\mathbf{u})| \tag{5.20}$$
$$\angle G(\mathbf{u}) = \angle F(\mathbf{u}) + \angle H(\mathbf{u}) \tag{5.21}$$

Where:

G is the Fourier transform of the blurred image.
F is the Fourier transform of underlying, sharp image.
H is the Fourier transform of the blur PSF.

For some given frequency $\mathbf{u}$, the magnitude of $G(\mathbf{u})$ and $F(\mathbf{u})$ will be different. However, their relative relationship is preserved in the blurring process. Consider two pixel patches in a blurred image, $\mathcal{N}_{\mathbf{x}_1}$ and $\mathcal{N}_{\mathbf{x}_2}$. It is assumed that:

$$H(\mathbf{u})_{\mathbf{x}_1} = H(\mathbf{u})_{\mathbf{x}_2} = H(\mathbf{u}) \tag{5.22}$$

meaning that the two image patches are blurred by the same PSF. From Equation 5.20, the magnitudes of the two image patches are:

$$|G(\mathbf{u})_{\mathbf{x}_1}| = |F(\mathbf{u})_{\mathbf{x}_1}||H(\mathbf{u})| \tag{5.23}$$
$$|G(\mathbf{u})_{\mathbf{x}_2}| = |F(\mathbf{u})_{\mathbf{x}_2}||H(\mathbf{u})| \tag{5.24}$$

From the above equations it can be seen, that even though the magnitude of the two patches might be different, their relation will remain the same. This is because both magnitudes are multiplied by the same factor, $|H(\mathbf{u})|$. Thus, if $F(\mathbf{u})_{\mathbf{x}_1}|$ is larger than $|F(\mathbf{u})_{\mathbf{x}_2}|$, then $|G(\mathbf{u})_{\mathbf{x}_1}|$ must also be larger than $|G(\mathbf{u})_{\mathbf{x}_2}|$, and vice versa. Using this principle, a blur invariant feature can be formed by describing the relative magnitude responses between adjacent patches. This is done in a way similar to LBP.

First, the STFT is calculated for all pixel positions in the input image, $f(\mathbf{x})$. Note that the border pixels corresponding to half the window size of the STFT is omitted. Then, at all pixel positions, $\mathbf{x}$, the binary code vector $\mathbf{q}_m(\mathbf{x}, \mathbf{u})$ is defined. The magnitude responses of the pixels in a 3×3 region surrounding the pixel in question is considered. Each element in the binary pattern vector at position $\mathbf{x}$ for frequency $\mathbf{u}$, $\mathbf{q}_m(\mathbf{x}, \mathbf{u})$, is calculated as follows:

$$q_m(\mathbf{x}_c, \mathbf{u})_n = \begin{cases} 1 & \text{if } |G(\mathbf{u})_{\mathbf{x}_c}| \geq |G(\mathbf{u})_{\mathbf{x}_n}| \\ 0 & \text{otherwise} \end{cases} \tag{5.25}$$

Where:

$q(\mathbf{x}_c, \mathbf{u})_n$ is the n'th bit of the binary code at position $\mathbf{x}$ at frequency $\mathbf{u}$.
$|G(\mathbf{u})_{\mathbf{x}_c}|$ is the magnitude of the pixel position in question.
$|G(\mathbf{u})_{\mathbf{x}_n}|$ is the magnitude of the surrounding pixel position number n.

$\mathbf{q}(\mathbf{x}, \mathbf{u})$ has eight bits because there is eight surrounding pixels. The above quantizer operation is done for all pixel positions, $\mathbf{x}$, at all frequencies, $\mathbf{u}$. The number of occurrences

of each code is counted for all frequencies and formed into a histogram which is used as the descriptor. This descriptor is referred to as the *Local Magnitude Descriptor* (LMD).

As mentioned in the beginning, LFD encodes both the magnitude and the phase. The phase is also encoded by using the relativeness between different patches in the image, in a similar way as the magnitude. Instead of considering which position has the highest phase shift, it is considered if they lie in the same quadrant. At all pixel positions, $\mathbf{x}$, the binary code vector $\mathbf{q}_p(\mathbf{x}, \mathbf{u})$ is created. Each element in the vector is calculated as follows:

$$q_p(\mathbf{x}_c, \mathbf{u})_n = \begin{cases} 1 & \text{if } \angle G(\mathbf{u})_{\mathbf{x}_c} \text{ and } \angle G(\mathbf{u})_{\mathbf{x}_n} \text{ lie in the same quadrant} \\ 0 & \text{otherwise} \end{cases} \tag{5.26}$$

Like the magnitude quantizer, the above also forms an eight bit binary code for each frequency, $\mathbf{u}$. The number of occurrences of the phase codes are also counted for all frequencies and turned into a histogram, which forms the descriptor of the phase information. This descriptor is referred to as the *Local Phase Descriptor* (LPD).

As with LPQ, the LMD and LPD are also formed for the following four 2D frequencies: $\mathbf{u}_1 = [a, 0]^T$, $\mathbf{u}_w = [0, a]^T$, $\mathbf{u}_3 = [a, a]^T$ and $\mathbf{u}_4 = [a, -a]^T$. Each frequency yields a set of codes from the image. They are used to form two histograms, one for LMD and one for LPD. When forming one of the histograms, all patterns form all frequencies are counted and presented in the same histogram. Each of the two resulting histograms has a length of 256.

As an example of the extraction process, consider the example image at Figure 5.8.

Figure 5.8. Input image used for illustrating the LMD and LPD extraction process.

The four responses for the STFT at $a = 1/7$ is illustrated on Figure 5.9.

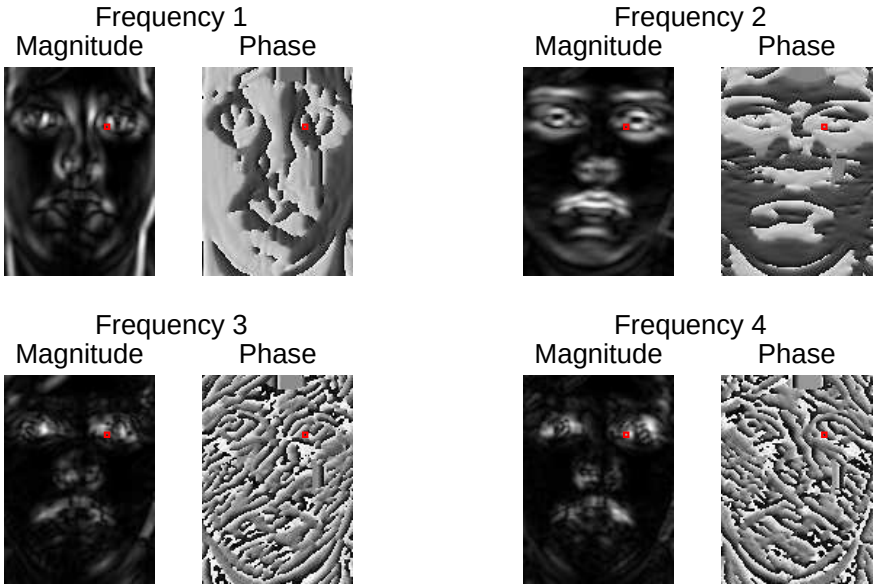

Figure 5.9. The response of the STFT at four frequencies. The red square marks a region which is used for a further description of the LFD feature.

To illustrate the code generating process, a closer look at the pixels inside the red square of the STFT response of the first frequency is provided on Figure 5.10.

Local frequency patch

Magnitude			Phase		
98	71	38	17	45	101
124	109	84	3	32	72
132	125	113	236	12	48

Figure 5.10. Example local region of size 3 from the STFT at frequency $\mathbf{u}_1$. The region is similar to the one marked by the red square at the upper left response shown on Figure 5.9. Note that the phase angles are in radians.

The LMD and LPD codes are generated using Equation 5.25 and 5.26, respectively. The resulting codes are:

$$\text{LMD: } 0000\ 1111_2 = 15_{10}$$
$$\text{LPD: } 0111\ 1010_2 = 122_{10}$$

This process is of cause done at all pixel positions. The LMD and LPD histograms are formed based on the codes from all frequencies. The final LFD descriptor is formed by concatenating the LMD and LPD histograms together.

Figure 5.11 illustrates the eight histograms which results from each of the four frequencies for both the magnitude and phase response.

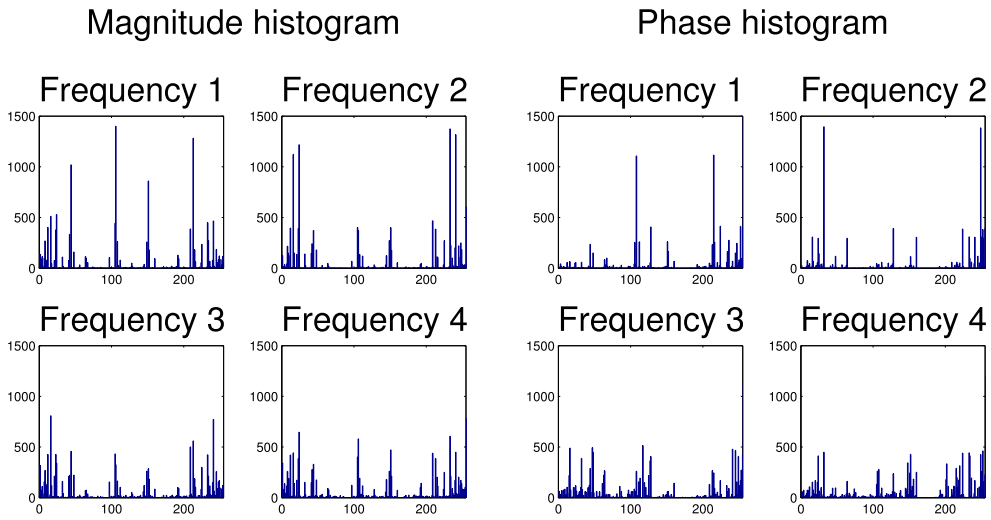

Figure 5.11. The eight histograms resulting from the STFTs shown on Figure 5.9. The four histograms to the left illustrates the LMD descriptors and the four histograms to the right illustrates the LPD descriptors.

As mentioned, the LMD histograms are summed together and the LPD histograms are summed together. Then two resulting histograms are concatenated together, which results in the histogram shown on Figure 5.12. Note that this is the LFD descriptor of the image shown on Figure 5.8.

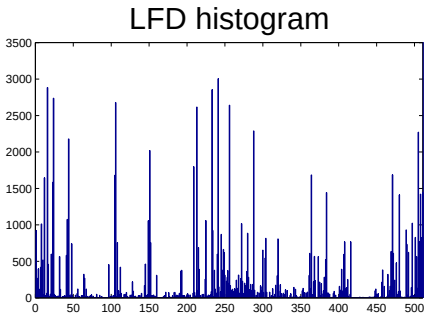

Figure 5.12. The Local Frequency Descriptor extracted from the image showed on Figure 5.8. It is formed by concatenating the LMD and LPD histograms together.

5.4.1 Statistical Uniform Patterns

With a length of 512, the LFD descriptor is significantly larger than the descriptors of the other local features considered in this report. To cope with this problem, Lei et al. [2011] propose to reduce the dimensionality by introducing statistical uniform patterns. As explained in Section 5.1, uniform patterns are patterns that have a certain number of bit transitions. Ojala et al. [2002] discovered that the patterns which had at maximum

two transitions where by far the most abundant type of LBP patterns. Therefore, they proposed to use only those patterns, thus reducing the LBP feature dimensionality.

The statistical uniform patterns are not defined based on the number of bit transitions. Instead, they are defined by the number of occurrences of each pattern over a large set of images. Thus, only the patterns which has the highest possibility of occurring are used.

Lei et al. [2011] extracts the statistical uniform patterns from LMD and LPD before they are concatenated into the LFD. They define an iterative algorithm for defining the patterns. First, both histograms are sorted. Then, in each step, the two bins with the lowest occurrence percentages in both histograms are combined and the histograms are resorted. The algorithm is inspired by Huffman coding. It can be iterated for as many steps as one like. Lei et al. [2011] use it to define 16 statistical uniform patterns for both LMD and LPD. The resulting LFD descriptor have a dimensionality of 32.

In this project, the same parameters as for LPQ are used: $M = 7$ and $a = 1/7$. As it is unknown exactly how many statistical uniform patterns should be used for FER, it is decided to use 50 patterns for both LMD and LPD. The resulting LFD descriptor has a dimensionality of 100. The LFD algorithm is shown below:

Algorithm 4 Algorithm which extracts the $LFD^{\text{sup}}_{M,a}$ descriptor from an image. The abbreviation s.u.p. is short for statistical uniform patterns. Note that the algorithm continues on the following page.

Require: Img,M,a,sup
Ensure: $LFD^{\text{sup}}_{M,a}$

$\mathbf{u}_1 \leftarrow [a, 0]^T$; $\mathbf{u}_2 \leftarrow [0, a]^T$; $\mathbf{u}_3 \leftarrow [a, a]^T$; $\mathbf{u}_4 \leftarrow [a, -a]^T$ $\mathbf{w} \in \mathbb{R}^{4,M^2}$
for $n \leftarrow 1$ to 4 **do**
    for $r \leftarrow 1$ to M **do**
        for $c \leftarrow 1$ to M **do**
            $w \leftarrow \exp\left(-j2\pi\mathbf{u}_n^T\ [r-1, c-1]^T\right)$
            $\mathbf{w}_n(c + (r-1)M) \leftarrow w$
        end for
    end for
end for
$\mathbf{G} \in \mathbb{R}^{4\times Img}$
for all rows, r, in Img except the $M/2$ border rows **do**
    for all columns, c, in Img except the $M/2$ border columns **do**
        $\mathbf{r} \in \mathbb{R}^4$
        for $n \leftarrow 1$ to 4 **do**
            $\mathbf{f} \leftarrow \text{RowConcat}(Img([r - M/2; r + M/2], [c - M/2; c + M/2]))$
            $\mathbf{G}_n(r, c) \leftarrow \mathbf{w}_n^T\mathbf{f}$
        end for
    end for
end for

20: $\mathbf{DM} \in \mathbb{R}^{4\times Img}$
21: $\mathbf{DP} \in \mathbb{R}^{4\times Img}$
22: **for all** rows, r, in Img except the $M/2$ border rows **do**
23: **for all** columns, c, in Img except the $M/2$ border columns **do**
24: **for** $n \leftarrow 1$ to 4 **do**
25: **for** $s \leftarrow 1$ to 8 **do**
26: $\mathbf{p}$ = neighboring pixel number s of $[r; c]$
27: **if** $|\mathbf{G}_n(r, c)| \geq |\mathbf{G}_n(\mathbf{p})|$ **then**
28: $b \leftarrow 1$
29: **else**
30: $b \leftarrow 0$
31: **end if**
32: $\mathbf{DM}_n(r, c) \leftarrow \mathbf{DM}_n(r, c) + b^{s-1}$
33: **if** $\angle\mathbf{G}_n(r, c)$ and $\angle\mathbf{G}_n(\mathbf{p})$ is in the same quadrant **then**
34: $b \leftarrow 1$
35: **else**
36: $b \leftarrow 0$
37: **end if**
38: $\mathbf{DP}_n(r, c) \leftarrow \mathbf{DP}_n(r, c) + b^{s-1}$
39: **end for**
40: **end for**
41: **end for**
42: **end for**
43: $\mathbf{HM} \in \mathbb{R}^{256}$
44: $\mathbf{HM} \leftarrow$ histogram of all values in $\mathbf{DM}$
45: $LMD \leftarrow$ histogram containing only s.u.p. from $\mathbf{HM}$
46: $\mathbf{HP} \in \mathbb{R}^{256}$
47: $\mathbf{HP} \leftarrow$ histogram of all values in $\mathbf{DP}$
48: $LPD \leftarrow$ histogram containing only s.u.p. from $\mathbf{H}$
49: $LFD^{\text{sup}}_{M,a} \leftarrow$ concatenation of LMD and LPD

Dimensionality Reduction 6

This chapter describes the PCA dimensionality reduction method and the SC clustering method. In the context of this project, both methods are used for dimensionality reduction.

PCA is a popular dimensionality reduction method which has been used in combination with LBP, LDPv and LPQ to provide promising recognition rates in FER. It appears that PCA has never been used in combination with LFD. SC is normally a clustering method. In this project it is used as an opposing method for reducing the data dimensionality.

In the following two sections, each of the two approaches are described in detail.

6.1 Principal Component Analysis

PCA was developed by Pearson [1901]. Over the years, it has been applied to many different fields of statistics. In brief, PCA seeks to uncover a set of basis which best describes the information carried by a dataset, by maximizing the variance explained by each basis. These basis are called Principal Components. The original data can be projected onto the Principal Components in order to align it with its directions of maximal variance. This principle is illustrated in Figure 6.1.

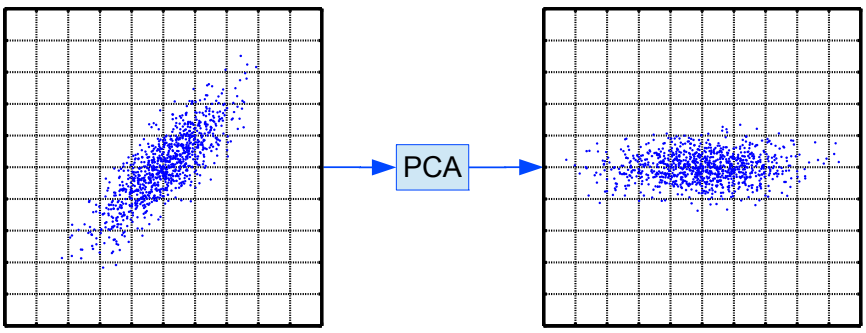

Figure 6.1. Illustration of a 2D dataset which is normally distributed. On the graph to the left, the data's direction of maximal variance is rotated by 45 deg relative to the first axis. Thus, each axis describes an equal amount of the variance. By using PCA, a new set of basis is defined where each basis maximizes the variance. The graph to the right displays the data transformed into the new set of basis. Note that the new set of basis is aligned with the orthogonal directions of maximum variance of the original dataset.

The Principal Components are sorted by the amount of variance they explain. Thus, the first Principal Component is the basis on which the distribution of the data is largest.

The following description of PCA is based on Bishop [2006]. Let $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^T$ be a dataset of N samples. Each sample is a point in a D-dimensional space, $\mathbf{x}_n \in \mathbb{R}^D$.

Now, lets find the first Principal Component of $\mathbf{X}$. To do so, it is assumed that all the data is to be projected onto a one-dimensional space. Thus, this space is a vector which points in some direction inside $\mathbb{R}^D$. Let this vector be $\mathbf{u}_1$. As $\mathbf{u}_1$ is a Principal Component and thus a basis, it has unit length. The projected data onto $\mathbf{u}_1$ is:

$$x'_n = \mathbf{u}_1^T \mathbf{x}_n \quad \forall \quad n = 1, \cdots, N \tag{6.1}$$

The mean of the projected data is:

$$\mu' = \mathbf{u}_1^T \boldsymbol{\mu} \tag{6.2}$$

Where:

$\boldsymbol{\mu}$ is the sample set mean of $\mathbf{X}$.

Knowing the mean, the variance of the projected data can be calculated by:

$$\sigma' = \frac{1}{N} \sum_{n=1}^{N} \left(x'_n - \mu'\right)^2 \tag{6.3}$$

$$= \frac{1}{N} \sum_{n=1}^{N} \left(\mathbf{u}_1^T \mathbf{x}_n - \mathbf{u}_1^T \boldsymbol{\mu}\right)^2 \tag{6.4}$$

Equation 6.4 can be simplified by introducing the data covariance matrix, $\mathbf{C}$:

$$\mathbf{C} = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n - \boldsymbol{\mu})(\mathbf{x}_n - \boldsymbol{\mu})^T \tag{6.5}$$

By substituting Equation 6.5 into Equation 6.4, the variance can be calculated by:

$$\sigma' = \mathbf{u}_1^T \mathbf{S} \mathbf{u}_1 \tag{6.6}$$

As mentioned previously, it is desired to make the variance of the projected data as large as possible w.r.t. $\mathbf{u}_1$. This could be done by making the length of $\mathbf{u}_1$ tend to infinity. However, this would not provide a meaningful solution. Therefore, the length of $\mathbf{u}_1$ is constrained to 1. The constraint is enforced by introducing a Lagrange multiplier:

$$L = \mathbf{u}_1^T \mathbf{S} \mathbf{u}_1 + \lambda_1 (1 - \mathbf{u}_1^T \mathbf{u}_1) \tag{6.7}$$

The $\mathbf{u}_1$ which makes the variance of the projected data largest can be found by: $\arg\max_{\mathbf{u}_1}(L)$. This maximization can be solved by finding stationary points via differentiation of Equation 6.7:

$$\frac{\partial L}{\partial \mathbf{u}_1} = 2\mathbf{S}\mathbf{u}_1 - 2\lambda_1 \mathbf{u}_1 \tag{6.8}$$

$$2\mathbf{C}\mathbf{u}_1 - 2\lambda_1 \mathbf{u}_1 = 0$$

$$\Updownarrow$$

$$2\mathbf{C}\mathbf{u}_1 = 2\lambda_1\mathbf{u}_1$$

$$\Updownarrow$$

$$\mathbf{C}\mathbf{u}_1 = \lambda_1\mathbf{u}_1 \tag{6.9}$$

From Equation 6.9 ii is obvious, that $\mathbf{u}_1$ must be an eigenvector of $\mathbf{C}$ with eigenvalue λ_1. This relation can be substituted into Equation 6.6, which reveals that:

$$\begin{aligned}\mathbf{u}_1^T\mathbf{S}\mathbf{u}_1 &= \mathbf{u}_1^T\lambda_1\mathbf{u}_1 = \lambda_1\mathbf{u}_1^T\mathbf{u}_1 \\ &= \lambda_1 = \sigma'\end{aligned} \tag{6.10}$$

Equation 6.10 implies, that the largest projected variance is obtained when λ_1 is largest. Thus, λ_1 must be the first eigenvalue of $\mathbf{C}$ and $\mathbf{u}_1$ must be the first eigenvector of $\mathbf{C}$.

Because the Principal Components forms a basis set, they must be orthogonal to each other. Thus, the second Principal Component can be found in a way similar to how the first was found. However, a constraint must be added to the optimization of Equation 6.7 to ensure that $\mathbf{u}_2 \perp \mathbf{u}_1$. Indeed, all Principal Components can be found like this. Therefore, it is obvious that all Principal Components are eigenvectors of $\mathbf{C}$.

The Principal Components are usually found by one of two ways. Either, the dataset covariance matrix is formed and eigenvalues are found via the eigenvalue decomposition. Or else, the singular value decomposition is calculated on $\mathbf{X}$ from which eigenvalues and eigenvectors of $\mathbf{C}$ can be extracted.

If the number of data points are lower than the number of dimensions, there is as many Principal Components as there are data points. Conversely, if the number of data points are higher than the number of dimensions, there is as many Principal Components as there are dimensions. As mentioned, the Principal Components are sorted by how much variance they explain. The dimensionality of the data can be reduced by simply removing some of the Principal Components which contains the lowest amount of variance. Then, the data can be projected into the reduced space, which keeps as much of the variance in the data as possible.

6.2 Spectral Clustering

SC is a type clustering method that comes in many different varieties. As far as the author is aware, SC has not been used to reduce the dimensionality of local features used in the FER problem.

The SC algorithm used here was proposed by Ng et al. [2001]. The following explanation of SC also includes elements from lux. In brief, SC derives clusters in a dataset from the eigenvectors with largest eigenvalue of a matrix which describes the similarities between the data points in the set. The eigenvectors are used to represent the data points in a new space. This new space is designed such that data points which has a high similarity clumps together. Consider the clustering example shown to the left on Figure 6.2. This problem is unsolvable with conventional clustering approaches like k-means clustering. SC can solve this clustering task.

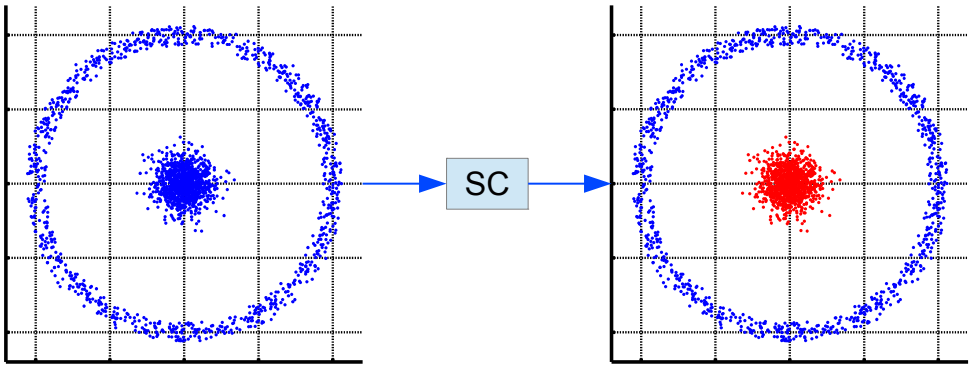

Figure 6.2. Example of a clustering problem. The plot shown to the left contains the unclustered data points. This would be an impossible task for ordinary clustering approaches, such as k-means-clustering. SC solves this problem as shown on the graph to the right. It does so by transforming the data into a space where similar points are grouped together. Clusters can easily be separated by e.g. k-means in the new space, and referred back to the original space.

As explained by lux, there is a multiple of ways to do SC. The following description of SC will not cover all of these ways, only the one which is used. The interested reader is recommended to read the tutorial by lux for an in-dept going description.

SC is a graph-based clustering approach. It tries to solve a relaxed version of the NP-hard problem of finding the optimal balanced cut of an undirected weighted graph. The edge-weights of the graph represents the similarities between points. A graph cut problem is illustrated on Figure 6.3

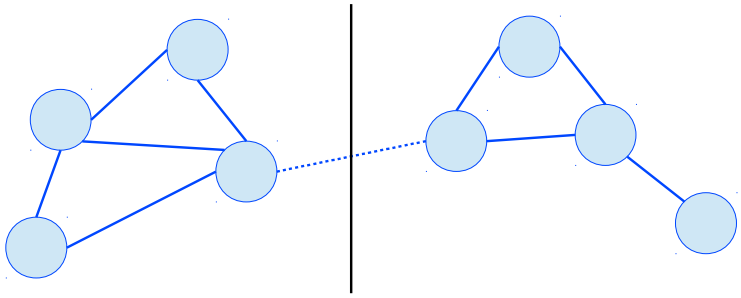

Figure 6.3. Illustration of a two-cut graph-cut problem. The distances between vertices illustrates their similarities. The two optimal cuts are separated by the black line.

Lets consider a given dataset of N samples in a D dimensional space: $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^T$. An undirected graph is formed by assigning a vertex to each data point: $V = \{v_1, \cdots, v_N\}$. Edges between the vertices are formed based on some form of similarity measure. The resulting graph is $G = (V, E)$.

In general, there are three ways of assigning edges to the graph: ϵ-neighborhood, k-nearest neighbor, and fully connected. Here, the fully connected graph is chosen. In the fully

connected graph, all pairs of vertices with positive similarity are connected. The Gaussian similarity is used as the similarity measure. Thus, the similarity between vertex v_n and v_m is defined as follows:

$$s(v_n, v_m) = \exp(\frac{-||\mathbf{x}_n - \mathbf{x}_m||^2}{2\sigma^2} \tag{6.11}$$

Where:

$$\begin{aligned} &\sigma \text{ defines the steepness of the roll-off of the similarity.} \\ &\mathbf{x}_n \text{ is the data point assigned to vertex } n. \\ &\mathbf{x}_m \text{ is the data point assigned to vertex } m. \end{aligned} \tag{6.12}$$

The plot of Equation 6.11 for $\sigma = 1$ is showed on Figure 6.4. Note that the similarity is decreasing with distance.

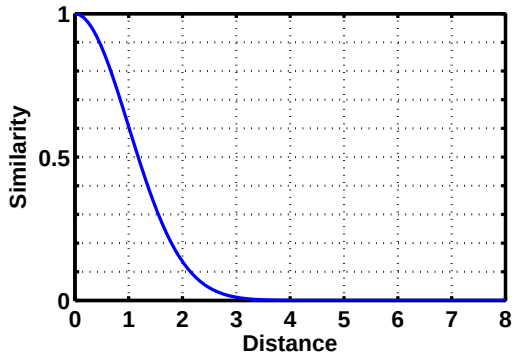

Figure 6.4. Plot of the Gaussian similarity measure for $\sigma = 1$. Note that the shape of the curve is similar to that of a low-pass filter. Following that line of thought, σ is a parameter which controls the roll-off of the filter.

The Gaussian similarity is always positive. Therefore, all vertices in the fully connected graph will be connected. Their pairwise similarity is assigned as weights on the edges between the vertices.

The task is now to partition the graph into K cuts. It is desired to have as high a similarity between the vertices inside the cuts as possible and as low a similarity as possible between vertices in different cuts. Thus, the similarity between two vertices grouped together should be high. Conversely, the similarity between two vertices grouped in different cuts should be low. Before defining a way to measure similarity between different groups, lets define the degree of a vertex:

$$d_n = \sum_{m=1}^{N} w_{n,m} \tag{6.13}$$

Where:

$$w_{n,m} \text{ is the weight between vertex } i \text{ and vertex } j.$$

For two subsets of vertices, $A \subset V$ and $B \subset V$, the total weight between them is defined as:

$$\mathrm{W}(A, B) = \sum_{n \in A, m \in B} w_{n,m} \tag{6.14}$$

The size of one of the subsets can be measured by summing up all the weights which are attached to vertices in that subset. This is called the volume and for subset A it is calculated by:

$$\mathrm{vol}(A) = \sum_{n \in A} d_n \tag{6.15}$$

Now, with these measures in place, it is possible to state the *Normalized Cut* problem. Some given graph, G, can be partitioned into K reasonably large cuts by minimizing:

$$\mathrm{Ncut}(A_1, \cdots, A_K) = \frac{1}{2} \sum_{n=1}^{K} \frac{\mathrm{W}(A_n, \bar{A}_n)}{\mathrm{vol}(A_n)} \tag{6.16}$$

Where:

$\bar{A}_n$ is the subset of vertices which is not in A, $V \setminus A = \bar{A}$.

In short, by minimizing Equation 6.16 with respect to $A_1, \cdots, A_K$, a set of partitions with low in-between similarity is obtained. The partitions will be balanced in size because of the division by the volume of the partitions. This is because $\sum_{n=1}^{K} 1/\mathrm{vol}(A_n)$ is smallest when the volumes are identical. Unfortunately, the Normalized Cut problem is NP-hard. Essentially, SC is a way to relax this problem.

The following description will explain how the Normalized Cut problem can be solved by SC as proposed by Ng et al. [2001]. The similarities between each vertex, V, can be explained by an affinity matrix, $\mathbf{A}$. Element A_{nm} in the affinity matrix describes the similarity between vertex n and vertex m. Thus, $\mathbf{A}$ is an $N \times N$ matrix. The elements of the affinity matrix is defined as follows:

$$A_{nm} = \begin{cases} \exp(\frac{-||\mathbf{x}_n - \mathbf{x}_m||^2}{2\sigma^2} & \text{if } n \neq m \\ 0 & \text{otherwise} \end{cases} \tag{6.17}$$

From Equation 6.17, it is evident, that $A_{nm} = A_{mn}$. Thus, $\mathbf{A}$ is a symmetric matrix. Now, the degree matrix is formed as a diagonal matrix with the sum of each row in $\mathbf{A}$ as its elements. The diagonal elements are defined as follows:

$$D_{nn} = \sum_{m=1}^{N} A_{nm} \tag{6.18}$$

The off-diagonal elements of $\mathbf{D}$ are all 0.

The degree matrix can be used to normalize the elements of $\mathbf{A}$ by introducing the Graph Laplacian. Multiple Graph Laplacians exists, but here the procedure of Ng et al. [2001] is followed and therefore the Normalized Graph Laplacian is used. Essentially, forming the

Normalized Graph Laplacian is a way to balance the affinity across different clusters. The Normalized Graph Laplacian is defined as:

$$\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} \tag{6.19}$$

Left multiplication by a diagonal matrix corresponds to scaling the rows of $\mathbf{A}$. Right multiplication of a diagonal matrix corresponds to scaling the columns of $\mathbf{A}$. Thus, Equation 6.19 scale each element in $\mathbf{A}$ as follows:

$$L_{nm} = \frac{A_{nm}}{\sqrt{\sum_{k=1}^{N} A_{nk}}\sqrt{\sum_{k=1}^{N} A_{mk}}} \tag{6.20}$$

Equation 6.20 states, that the similarity between vertex n and vertex m is scaled by the square root of the sum of all similarities assigned to vertex n and vertex m.

The Graph Laplacian can be used to identify partitions. Without loss of generality, it can be assumed that the entries in $\mathbf{A}$ are sorted based on which partition they belong to. Thus, elements in $\mathbf{A}$ corresponding to vertices which belongs to the same partition are put together. In that case, $\mathbf{L}$ will be block diagonal with K number of blocks, if K good partitions can be identified in G. That is:

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_1 & & & \\ & \mathbf{L}_2 & & \\ & & \ddots & \\ & & & \mathbf{L}_K \end{bmatrix} \tag{6.21}$$

Where:

$\mathbf{L}_n$ is a matrix smaller than $\mathbf{L}$ for $K > 1$.

The elements outside of the diagonal blocks will have low similarity because they are in different partitions. Thus, their similarity will tend toward 0. Each block $\mathbf{L}_n$ corresponds to a partition in G. Calculating the eigenvalues of $\mathbf{L}$ corresponds to finding the directions of largest similarity. Each of the K largest eigenvalues of $\mathbf{L}$ originates from exactly one of the blocks, $\mathbf{L}_n$. Each block will correspond to exactly one of the K largest eigenvectors. Thus, each block yields exactly one of the K largest eigenvalues of $\mathbf{L}$. Only the dimensions of the eigenvectors corresponding to non-zero elements in $\mathbf{L}$ are non-zero. E.g., the first eigenvector with largest eigenvalue will only have non-zero elements at the dimensions corresponding to samples belong to partition 1. Therefore, the K largest eigenvectors of $\mathbf{L}$ can be used as indicators to the optimal K partitions of G.

The precise flow of the SC algorithm will be illustrated through the following example. Lets consider the clustering problem showed on Figure 6.5.

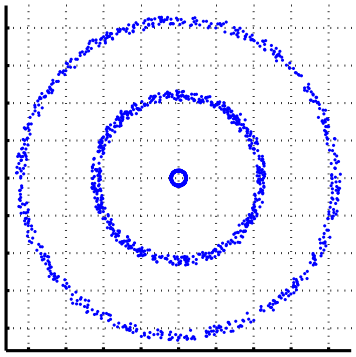

Figure 6.5. Example of 2D data used as input to illustrate the flow of the SC algorithm. The data contains a total of 1800 samples from three classes of equal size.

First of all, lets construct the affinity matrix given as by Equation 6.17. Because the data contains 2000 points, the resulting affinity matrix is displayed as a color-coded image on Figure 6.6.

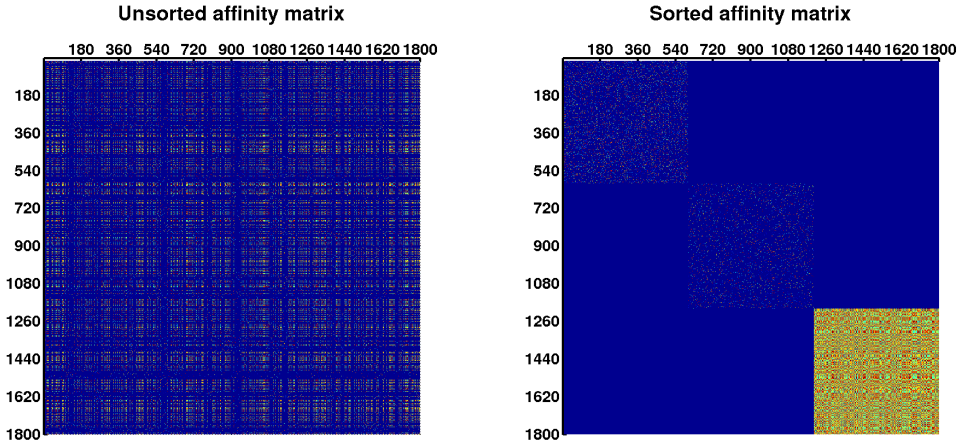

Figure 6.6. Affinity matrix corresponding to the input data shown on Figure 6.5. Blue color corresponds to 0, yellow to 0.5 and red to 1. The matrix shown on the left is unsorted. The matrix shown on the right is sorted relative to the ground truth clusters.

Then, the degree matrix is formed from the affinity matrix using Equation 6.18. Now, the Normalized Graph Laplacian is constructed from the affinity matrix and the degree matrix as defined by Equation 6.19. It is shown on Figure 6.7.

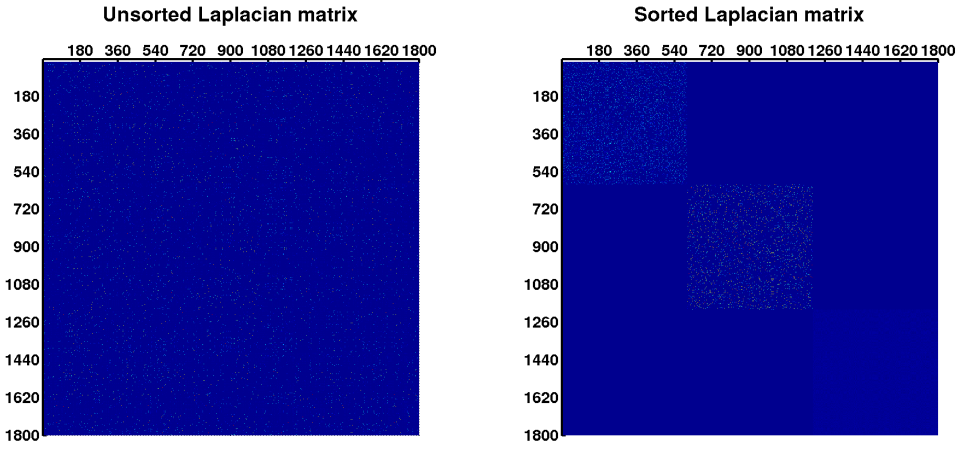

Figure 6.7. The Normalized Graph Laplacian matrix constructed from the affinity matrix from Figure 6.6. Like in the illustration of the affinity matrix, the left matrix is unsorted and the right is sorted relatively to the ground truth.

Then the eigenvalues and eigenvectors are extracted from the Graph Laplacian. Because there are three clusters, only the first three eigenvectors are used. These three eigenvectors span a three dimensional space in which data points from the same cluster are grouped closely together. This space is shown on Figure 6.8.

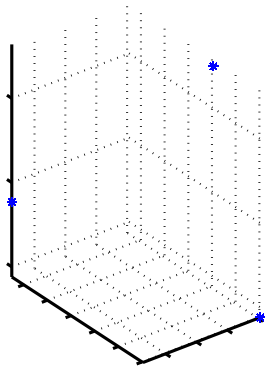

Figure 6.8. The data points transformed into a 3D space spanned by the first three eigenvectors. As seen, the data points forms very tight clusters. In total, 1800 points are displayed.

A k-means clustering algorithm is used to define clusters in the space spanned by the eigenvectors. Each point in the eigenvectors space corresponds to a vertex in the original graph. Therefore, the found clusters are simply referred back to the original space. The clustering result can be seen on Figure 6.9.

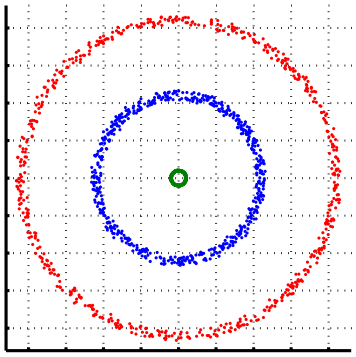

Figure 6.9. Result of the clustering in the space spanned by the first three eigenvectors of the Graph Laplacian. The clusters are labeled by the color of the data points.

SC can be used as a dimensionality reduction method. As seen on Figure 6.8, the data points from the example forms very tight clusters when presented in the space spanned by the first three eigenvectors. Indeed, the space spanned by the K first eigenvectors can be regarded as a reduced feature space if K is lower than the dimensionality of the original data. When using the method for dimensionality reduction, a K higher than the number of classes could be selected.

The outline of the SC algorithm for dimensionality reduction is as follows:

1. Form the Affinity matrix, $\mathbf{A}$, by Equation 6.17
2. Calculate the Degree matrix, $\mathbf{D}$, by Equation 6.18
3. Calculate the Graph Laplacian matrix, $\mathbf{L}$, by Equation 6.19
4. Find the K eigenvalues, $\lambda_1, \cdots, \lambda_K$, and eigenvectors, $\mathbf{e}_1, \cdots, \mathbf{e}_K$, of $\mathbf{L}$. Use the eigenvectors to form: $\mathbf{E} = [\mathbf{e}_1, \cdots, \mathbf{e}_K]^T$
5. Re-normalize the rows of $\mathbf{E}$ to unit length, forming $\mathbf{Y}$ where $Y_{nm} = E_{nm}/(\sum_r X_{nr}^2)^{1/2}$
6. Treat each row of $\mathbf{Y}$ as a point in $\mathbb{R}^K$

It should be noted that the affinity matrix has to be updated every time a new sample is obtained. This will most likely make the algorithm computationally more expensive than the PCA algorithm.

The SC algorithm needs two parameters to be defined: the number of clusters, K, and the variance of the similarity function, σ^2.

Classification 7

This chapter describes the SVM classifier. In order to compare LFD against the other feature descriptors, and to compare SC against PCA, something needs to be hold constant. Therefore, only a single type of classifier is considered in this project. SVM was chosen due to previous good recognition rates for FER with local features. In fact, to the extend of the knowledge of the author, SVM is currently the classifier which provides the highest recognition rates for FER with local features

7.1 Support Vector Machine

SVM was first proposed by Cortes and Vapnik [1995]. It is a binary classifier which creates a decision boundary between two separable classes. The decision boundary is learned in a supervised fashion from training data.

So far, SVMs has provided very promising results for FER with local features Caleanu [2013]. They have proved to be superior over other methods such as Chi Square statistics and Linear Programming when combined with LBP [Shan et al., 2009]. They have also been proved to yield a good recognition rate when combined with LDPv [Kabir et al., 2010] and LPQ [Yang and Bhanu, 2011]. It seems that SVMs has not been used in combination with LFD for FER so far.

The following description will explain how and why SVMs work. First, the linear classification model is covered. Second, it is described how a slack variable can be introduced to establish a better decision boundary and avoid over fitting. Third, the kernel trick is explained, which allows the decision boundary to be non-linear. Fourth, it is explained how multiple binary SVMs can be used in combination to do multi class classification. At last, it is explained how to find the optimal solution for the parameters in the classification model. This chapter is based on Bishop [2006].

SVMs are based on a linear decision function. For a given input sample, $\mathbf{x}$, the decision function is defined as follows:

$$y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b \tag{7.1}$$

Where:

$y(\mathbf{x})$ determines the class of $\mathbf{x}$ by its sign.

$\mathbf{w}$ is a weight vector which decides the orientation of the decision boundary.

$\phi(\mathbf{x})$ is a fixed feature space transformation.

b is an offset of the decision boundary along the orientation of $\mathbf{w}$.

The output of the decision function is a scaler, which indicates the class of $\mathbf{x}$ by the sign of $y(\mathbf{x})$. The goal is to estimate the parameters $\mathbf{w}$ and b which correctly classifies $\mathbf{x}$. This is done by fitting the model to a set of training data, $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$. Each sample in the training data, $\mathbf{x}_n$, has a target value: $t_n \in [-1; 1]$. The target value indicates the class of that sample.

Often when considering real datasets, the two classes are not directly separable. This problem can be solved by a fixed feature-space transformation $\phi(\mathbf{x})$.

It is desired to position the decision boundary in the exact middle between the two classes. By doing so, the most optimal division of the two classes is obtained. The term *margin* is introduced to measure the distance from the decision boundary to the classes. The margin is illustrated in a 2D example on Figure 7.1.

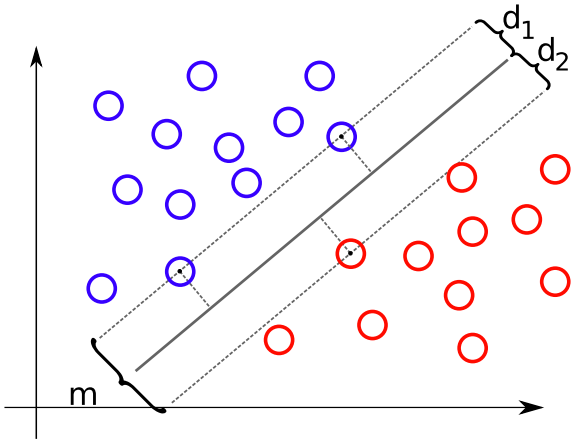

Figure 7.1. Example of a two class classification problem in 2D. Data points are illustrated by circles whose color identifies their class. The decision boundary is illustrated by the solid black line in between the two classes. The margin is illustrated as the dotted lines on both sides of the decision boundary. Note that $d_1 = d_2$.

The decision boundary is defined such that $y(\mathbf{x}) = 0$ if $\mathbf{x}$ is positioned on the boundary. By definition, it is decided that the closest training points must satisfy:

$$y(\mathbf{x}_{s+}) = \mathbf{w}^T \phi(\mathbf{x}_{s+}) + b = +1 \tag{7.2}$$

$$y(\mathbf{x}_{s-}) = \mathbf{w}^T \phi(\mathbf{x}_{s-}) + b = -1 \tag{7.3}$$

Where:

$\mathbf{x}_{s+}$ is a training point located on the positive-side margin.

$\mathbf{x}_{s-}$ is a training point located on the negative-side margin.

The above definitions can be met by scaling the length of $\mathbf{w}$ and b until they fit. Per definition, training points which satisfy Equation 7.2 are called Support Vectors of the first class. Likewise, training points which satisfy Equation 7.3 are called Support Vectors

of the second class. Thus, the Support Vectors are the training points nearest to the decision boundary. As will become evident in a little while, they are used to define the location of the decision boundary. There will always be at least two Support Vectors: one from each class. However, if multiple points shares the same shortest distance, there will be multiple Support Vectors. This is illustrated by the two training points from the blue class, marked with the black dot on Figure 7.1.

The distance from the decision boundary to a training point, $\mathbf{x}$, is given by:

$$d(\mathbf{x}) = \frac{|y(\mathbf{x})|}{||\mathbf{w}||} \tag{7.4}$$

The margin can be maximized by maximizing the distance defined in Equation 7.4 to all training points. Due to the division by $||\mathbf{w}||$, the distance in Equation 7.4 can be maximized by minimizing $||\mathbf{w}||^2$. The power 2 is introduced to avoid calculating the square root in the norm. Thus, the optimal values for $\mathbf{w}$ and b can be found by:

Solve:

$$\arg\min_{\mathbf{w},b} \left(\frac{1}{2}||\mathbf{w}||^2 \right) \tag{7.5}$$

Subject to:

$$t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1 \quad \forall \quad n = 1, \cdots, N \tag{7.6}$$

Because there is an objective function subject to a set of inequality constraints, the above problem is a quadratic programming problem. It can be solved by using a set of N Lagrange multipliers, one for each of the training points. The new unconstrained optimization problem is:

$$L_P(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} a_n \left(t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) - 1 \right) \tag{7.7}$$

The above problem is termed the *primal* problem. The Lagrange multipliers enforces the constraints by dragging $L_P(\mathbf{w}, b, \mathbf{a})$ towards $-\infty$ when $t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b)$ is larger than or equal to 1 for all n. The problem is to be minimized with respect to $\mathbf{w}$ and b, but maximized with respect to $\mathbf{a}$.

Stationary points can be found for $\mathbf{w}$ and b by setting the derivative of Equation 7.7 equal to 0:

$$\frac{\partial L(\mathbf{w}, b, \mathbf{a})}{\partial \mathbf{w}} = 0 \quad \text{and} \quad \frac{\partial L(\mathbf{w}, b, \mathbf{a})}{\partial b} = 0 \tag{7.8}$$

$$\Updownarrow$$

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n) \quad \text{and} \quad \sum_{n=1}^{N} a_n t_n = 0 \tag{7.9}$$

The expressions from Equation 7.9 can be substituted back into the primal form in Equation 7.7. By doing so, $\mathbf{w}$ and b is eliminated. The resulting optimization problem

is only dependent on **a**. This problem is called the dual form. The details regarding the derivation of the dual form can be found in Appendix A. The dual form optimization problem is as follows:

$$L_D(\mathbf{a}) = \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m K(\mathbf{x}_n, \mathbf{x}_m) - \sum_{q=1}^{N} a_q \tag{7.10}$$

Where:

$$K(\mathbf{x}_n, \mathbf{x}_m) = \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m)$$

The above problem is solved as a minimization problem. Usually, the dual form is stated as a maximization problem. This is done by multiplying Equation 7.10 by -1:

$$L_D(\mathbf{a}) = \sum_{q=1}^{N} a_q - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m K(\mathbf{x}_n, \mathbf{x}_m) \tag{7.11}$$

Subject to

$$a_n \geq 0 \; \forall \; n = 1, \cdots, N \tag{7.12}$$

The optimization problem in Equation 7.11 satisfy the Karush-Kuhn-Tucker conditions. Therefore, the following three propositions hold:

$$a_n \geq 0 \quad \forall \quad n = 1, \cdots, N \tag{7.13}$$

$$t_n y(\mathbf{x}_n) - 1 \geq 0 \quad \forall \quad n = 1, \cdots, N \tag{7.14}$$

$$a_n \left(t_n y(\mathbf{x}_n) - 1\right) = 0 \quad \forall \quad n = 1, \cdots, N \tag{7.15}$$

The proposition in Equation 7.15 can only be true if either $a_n = 0$ or $t_n y(\mathbf{x}_n) = 1$. This can be interpreted as follows: either $\mathbf{x}_n$ is a Support Vector or else a_n is zero. This will come in handy shortly.

First, the stationary point for $\mathbf{w}$ which was derived in Equation 7.9 is substituted into the decision function from Equation 7.1:

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n K(\mathbf{x}, \mathbf{x}_n) + b \tag{7.16}$$

Where:

a_n is the Lagrange multiplier of training point n.
t_n is the target class of training point n.
$\mathbf{x}_n$ is training point number n.
$\mathbf{x}$ is a new unknown sample. (7.17)

Note that the decision function above is invariable to training points whose a_n is zero. Therefore, only the training points that has an $a > 0$ influences the decision function. Because of the conditions explained above, these data points are the Support Vectors.

When the training has finished, all the training points with $a = 0$ can be tossed away and only the Support Vectors needs to be kept. The b parameter can be determined by setting $t_n y(\mathbf{x}_n) = 1$ for one of the Support Vectors and solving for b.

The procedure described above states how the standard SVM works. In this project however, the Soft Margin SVM is used. It is more durable and generally provides better classification accuracy than the standard SVM. The Soft Margin SVM is described in the following section.

7.1.1 Soft Margin SVM

In training of the standard SVM, all training points must be classified correctly by the boundary. If the two classes present in the training data is somehow entangled, it is impossible to meet this criterion.

A solution is to turn the hard decision boundary into a soft boundary. A soft boundary allows some of the training points to be misclassified. This can be achieved by introducing a slack variable on the definition of the margins:

$$t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1 - \xi_n \quad \forall \quad n = 1, \cdots, N \tag{7.18}$$

The slack variable is added as a constraint to the margin maximization problem:

Solve:

$$\arg\min_{\mathbf{w},b,\boldsymbol{\xi}} \left(\frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N} \xi_n \right) \tag{7.19}$$

Subject to:

$$t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1 - \xi_n \quad \forall \quad n = 1, \cdots, N \tag{7.20}$$

$$\xi_n \geq 0 \quad \forall \quad n = 1, \cdots, N \tag{7.21}$$

$$C > 0 \tag{7.22}$$

One parameter needs to be defined, namely C. A large C punishes misclassified points hard, thus forming a boundary close to that of the standard SVM. Conversely, a low C allows more points to be misclassified, thus creating a softer margin. Thus, the C parameter controls a trade of between a hard and a soft margin.

The above problem is converted into an unconstrained problem by introducing a Lagrange multiplier:

$$L_P(\mathbf{w}, b, \mathbf{xi}, \mathbf{a}, \mathbf{b}) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N} \xi_n - \sum_{m=1}^{N} a_m \left(t_m y(\mathbf{x}_m) - 1 + \xi_m\right) - \sum_{q=1}^{N} b_q \xi_q \tag{7.23}$$

Where:

a is enforcing the constrains in Equation 7.20.

$\mathbf{b}$ is enforcing the constrains in Equation 7.21.
$y(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x})$

Again, the dual form is used, which yields the following optimization problem:

Solve:

$$\arg\max_{\mathbf{a}} \left(\sum_{n=1}^{N} -\frac{1}{2} \sum_{m=1}^{N} \sum_{q=1}^{N} a_m a_q t_m t_q K(\mathbf{x}_m, \mathbf{x}_q) \right) \tag{7.24}$$

Subject to:

$$0 \leq a_n \leq C \quad \forall \quad n = 1, \cdots, N \tag{7.25}$$

$$\sum_{n=1}^{N} a_n t_n = 0 \tag{7.26}$$

The optimization problem in Equation 7.24 is used to find the Support Vectors of the soft margin. The same decision function as for the standard SVM is used to classify a new unknown sample:

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n K(\mathbf{x}, \mathbf{x}_n) + b \tag{7.27}$$

Like before, the class of $\mathbf{x}$ is found by:

$$I(\mathbf{x}) = \text{sign}(y(\mathbf{x})) \tag{7.28}$$

Where:

$I(\mathbf{x})$ is the class of $\mathbf{x}$.
$\text{sign}(\cdot)$ is a function which returns the sign of a number.

Even though the soft margin enhances the classification capability of the SVM, it is not always enough. Sometimes, the data is too entangled to be properly separated by a linear decision boundary. The dual form of the SVM Lagrangian optimization problem allow to use the kernel trick. The trick turns the linear SVM decision boundary into a non-linear boundary. The kernel trick is described in the following section.

7.1.2 The kernel trick

The decision function of the dual form given in Equation 7.27 removes the actual data vectors from the calculation. Only the scalar products between the unknown input vector and the Support Vectors are needed. Up until now, the kernel has been defined as:

$$K(\mathbf{x}_n, \mathbf{x}_m) = \phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m) \tag{7.29}$$

This is the inner product between two vectors in a space with the same dimensionality as $\mathbf{x}_n$ and $\mathbf{x}_m$. In principle, the vectors could be transformed into a higher dimensional

space which better separates the classes. The inner product between the vectors could then be calculated in this higher dimensional space. However, by using the kernel trick, the vectors does not have to be actually transformed into the higher dimensionality space. Only a function which computes the inner product in that higher dimensional space is needed.

The kernel trick allows the creation of a decision boundary which is linear in the higher dimensional space, but non-linear in the original space. As a result, SVMs implementing the kernel trick can often create a decision boundary between classes which are not linearly separable.

Multiple kernels have been proposed for SVMs. In this project, the Gaussian *Radial Basis Function* (RBF) kernel is used. It is defined as follows:

$$K_{\text{RBF}}(\mathbf{x}_n, \mathbf{x}_m) = \exp(-\gamma||\mathbf{x}_n - \mathbf{x}_m||^2) \tag{7.30}$$

Where:

$$\gamma = -\frac{1}{2\sigma^2} > 0 \tag{7.31}$$

As explained previously, the classification system should be able to classify the seven basic facial expressions. The SVM classifier considered up until this point can only do binary classification. Classification of multiple classes can be achieved by combining several binary SVMs. This procedure is explained in the following section.

7.1.3 SVM for multiple classes

A multi-class SVM can be created by combining multiple SVMs. In general, there are two approaches of combining the SVMs: *One-Against-All* (OAA) or *One-Against-One* (OAO).

The OAA approach trains one binary SVM for each class. Thus, if there are K classes, a total of K SVMs are trained. When training the SVM belonging to class n, the training points from class n is regarded as the first class and the training points from all other classes are regarded as the second class. By doing so, the n'th SVM can tell if an unknown sample belongs to class n or does not belongs to class n.

The OAO approach trains one binary SVM for each pair of classes. If there is K classes, the method trains $K-1$ SVMs for each class. This approach creates a total of $K(K-1)/2$ decision boundaries.

In this project, the OAO approach is used.

7.1.4 Parameter estimation

There is two parameters which needs to be defined for the Soft Margin SVM with the RBF kernel: C and γ. In this project, the grid search approach is used to estimate the optimal parameters, as defined by wei Hsu et al. [2010].

In short, every combination of parameters defined on a discrete grid is tried and the combination which yields the highest recognition rate is used. The recognition rate is calculated by a five fold cross-validation approach on the training data.

An example is shown on Figure 7.2. The figure shows a contour plot of the accuracy for different combinations of C and γ. The example is from the KDEF database.

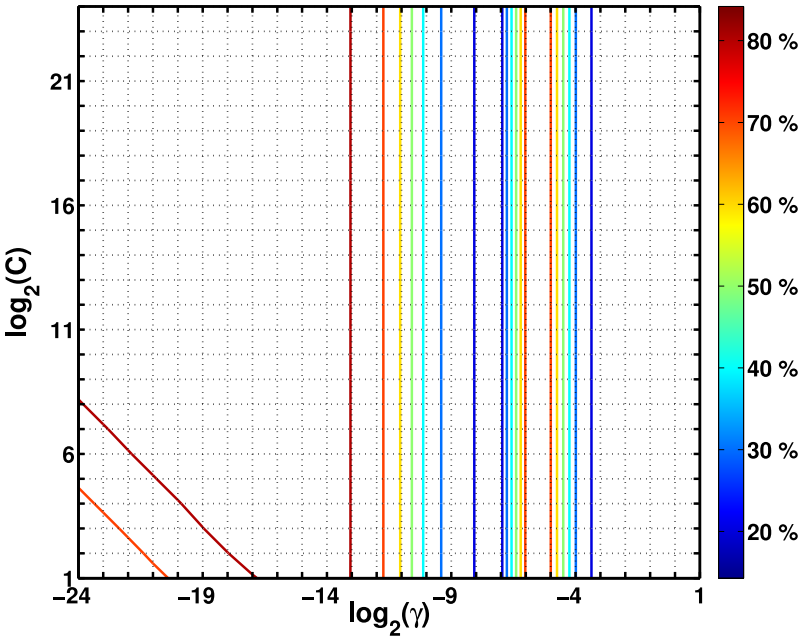

Figure 7.2. Contour plot of the accuracies obtained from different combinations of C and γ. The data used to generate this plot is from the KDEF database.

As noted by wei Hsu et al. [2010], the grid-search approach might not seem to be the most optimal approach for determining the parameters. However, the method has two main advantages: it is very likely that a global maximum within the search area is found and the search is easy to parallelize.

This is the end of the classification explanation, and hereby also the end of the technical explanations. The following chapter documents how the methods explained over the previous chapters was implemented and tested against each other.

8 Evaluation

This chapter describes the evaluation of the databases, features and dimensionality reduction methods presented in Chapter 5 to Chapter 7. The chapter is organized into four sections. The first section describes the evaluation data. The following three sections describe one experiment each. In order to make this chapter as pleasant to read as possible, most of the details of the experiments are described in Appendix B to D.

In combination, the three experiments can be used to determine which of PCA or SC is the better dimensionality reduction method and which of LBP, LPQ or LFD is the better feature descriptor for FER. Be aware, that the LDPv descriptor is unfortunately not included in the final conclusions. Despite an extensive debugging and multiple trials, the LDPv implementation failed to provide the expected recognition accuracies. This problem is detailed further in Section 8.2.

The chapter is divided into four sections. First, the details concerning the KDEF and Cohn-Kanade databases are presented. Second, a preliminary experiment is described which establishes a basis recognition accuracy for each of LBP, LDPv, LFD and LPQ. Third, a dimensionality reduction experiment is described, that seeks to uncover which of PCA or SC is better. Fourth and last, a blurring experiment is described which tries to determine which of LBP, LPQ or LFD is most robust against blurring of the test images.

Note that both the experimental results and their discussion are presented in this chapter.

8.1 Evaluation data

As specified in Chapter 3, the KDEF and Cohn-Kanade databases are used for evaluation. In this section, the specifications of the two databases are presented, based on the properties defined in Chapter 2. The KDEF database is considered first.

The KDEF database was compiled at Karolinska Institutet in Stockholm by Lundqvist et al. [1998]. It contains a total of 4900 2D images of 70 persons of Scandinavian ethnicity. 35 females and 35 males were used with an age span of 20 to 30 years. The database is based on the seven basic prototypic emotions: fear, anger, disgust, happy, neutral, sad, and surprised. All the images are labeled by the prototypic global expressions. The subjects received instructions on the facial expressions and then rehearsed the expressions for an hour prior to the photo shoot. No obstructions are visible in the images. All subjects wears a gray T-shirt. The backgrounds of the images are simple white and all subjects are positioned similarly in the images. A constant soft lighting is used, which illuminates the

entire face. The images are in full color. The subjects are photographed at five angles: $-90°$, $-45°$, $0°$, $+45°$, and $+90°$. All subjects are photographed twice for every facial expression at every angle. The database is free for scientific purposes but has a cost if used in a commercial setting.

In this project, only the frontal, $0°$ images from the database are used. There is $2 \cdot 70 \cdot = 140$ images for each facial expression which yields a total of $7 \cdot 140 = 980$ images. The images from the 13th female subject is shown below as an example:

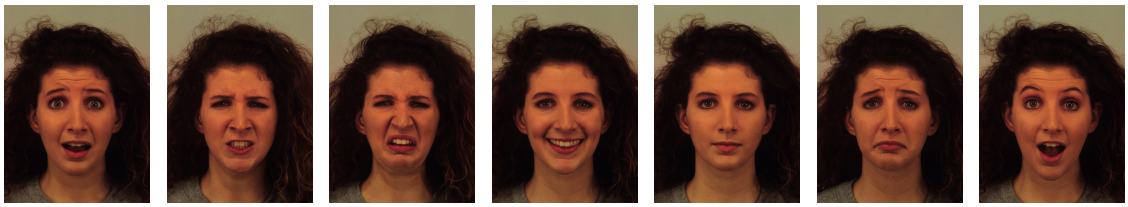

Figure 8.1. Frontal photographs of the 13th female subject showing all the seven basic expressions. The images are from the first photo session. The facial expressions left to right are: fear, anger, disgusted, happy, neutral, sad, and surprised.

Because of the two photo shoots, each person show the same facial expression twice. Therefore, there is a chance that the same person will be present in both training and test data if the database is randomly partitioned. As all of the feature descriptors used in this project operates on gray scale images, a color to gray scale conversion is done to the images as a part of the preprocessing step.

The Cohn-Kanade database was compiled at the Robotics Institute at Carnegie Mellon University, Pittsburgh by Kanade et al. [2000]. It comes in two versions: the basic and the extended version. A third version is planned for 2014. In this project, the basic version of the database is selected. The public part of the database contains a total of 487 image sequences from 97 different persons of varying ethnicity. The database is AU coded and no labels for the basic emotions are provided. Instead, a document accompanies the database when downloaded which explains how to do the translation from AUs to basic facial expressions. The subjects were instructed to perform 23 facial displays, each including single action units and action unit combinations. Six of these are based on the six basic prototypic expressions. No obstructions are visible in the sequences. 65% of the subjects are female. 15% of the subjects are African-American and 3% are Asian or Latino. The age span of the subjects is from 18 to 30 years. The backgrounds of the sequences are relatively simple, but contains some texture. The position of the subjects shift a bit from image sequence to image sequence. The lighting seems to be relatively constant. The subjects were filmed from $0°$ and $30°$, but only the $0°$ sequences are available to the public. The database is only available for non-commercial use. An example of one of the image sequence from subject 113 is shown below:

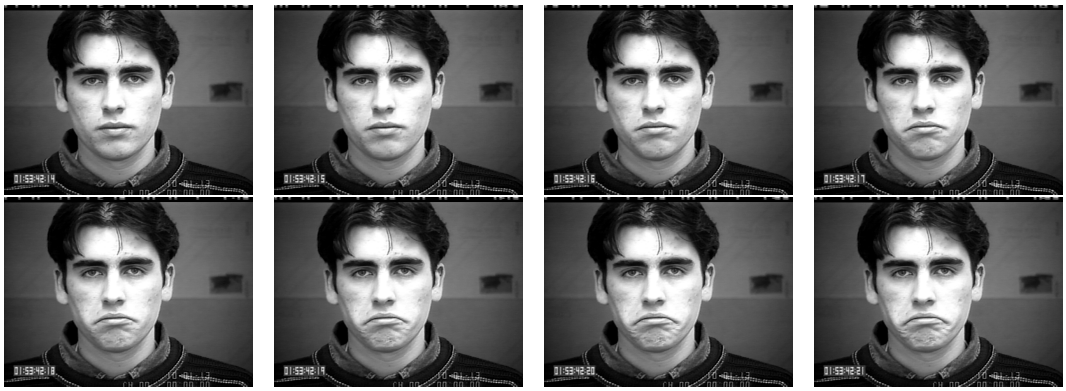

Figure 8.2. An example image sequence from the Cohn-Kanade database. This is sequence number 5 from subject 113. Note that the copyright is hold by ©Jeffrey Cohn.

This project only considers still images. Therefore, still images are extracted from the sequences following the procedure defined by Shan et al. [2009]. For every sequence, the last three frames showing peak expression is used. The neutral expression is drawn from the first image of all sequences. Facial expression examples from subject 55 and 121 is shown below:

Figure 8.3. Frontal photographs from the Cohn-Kanade database showing all the seven basic expressions. The images are from subject 55 and 121. The facial expressions left to right are: fear, anger, disgusted, happy, neutral, sad, and surprised. Note that the example of happy shown here is fairly subtle. Some of the other subjects opens their mouth wide and show teeth. Note that the copyright is hold by ©Jeffrey Cohn.

The Cohn-Kanade images are not labeled with basic expressions. Therefore, the basic expressions has been manually assigned to the images by judging from the look of the image and the AUs. It was not possible to label all images with satisfying certainty. As a consequence, a subset of 399 image sequences are used. From these, 399 neutral images and $3 \cdot 399 = 1197$ images of other basic expressions are extracted. In total, there are the following number of images for each expression:

Facial Expression	No. of images
Fear	183
Anger	114
Disgust	129
Happy	306
Neutral	399
Sad	231
Surprised	234
Sum	1596

Table 8.1. Number of images in each category from the subset extracted from the Cohn-Kanade database.

From Table 8.1 it is clear, that the number of samples in each class is not balanced. Further, because three peak frames are drawn from every image sequence, the same person occur multiple times for the same class. As a result, the same person could be present multiple times in both the test data partitioning and training data partitioning. Based on these observations, it is expected that the recognition rate obtained from the Cohn-Kanade database is considerably higher than that obtained from the KDEF database. The reader should be aware that this is not an error. It seems to be the normal way of partitioning the database [Shan et al., 2009]. Further more, the advantage of person repetition in training and test data is equal to all systems tested on the database. The results published in this report which uses the Cohn-Kanade database can only be recreated if the exact same partition of the original database is used with the same labels. It is not allowed to redistribute the database itself, but a list of used image sequences and their labels can be found at /`Database/CK/`.

8.2 Preliminary experiment

Prior to constructing an experiments which determines the best dimensionality reduction method, it is relevant to know what the basic performance of the features are. The preliminary experiment seeks to uncover the recognition accuracy of each feature before it has been dimensionality reduced.

In short, the main idea of this experiment is to approximate the recognition rate of the features before they have been tampered with by dimensionality reduction. Each of the features: LBP, LDPv, LPQ and LFD are extracted from both databases using the grid division approach. Every image is divided into 42 cells from which the feature descriptors are extracted. The final feature descriptor is made by concatenating the descriptors from each cell. The $LBP^{u2}_{8,1}$ descriptor is used as it performs similarly to the $LBP_{8,1}$ descriptor for FER [Shan et al., 2009]. No reduction or statistical uniform patterns are applied to the other feature descriptors.

After the features are extracted, the data samples are partitioned into five-folds which can be used for cross-validation. The partitioning is done 10 times with 10 different partitions

of the databases. This procedure was proposed by Doc. Zhanyu Ma due to the relatively small amount of data present in the databases.

For each repetition, for each of the five-folds, the training and test data is specified and inserted into an OAO multi-class SVM classifier. The final recognition accuracy is calculated as a mean of the means from all repetitions of the cross-validation[1].

The details of the experiment is documented in the measurement record found in Appendix B.

8.2.1 Results and discussion

The results of the preliminary experiment is presented in Table 8.2.

Feature	Database	Recognition rate
$LBP^{u2}_{8,1}$	KDEF	$85.58\% \pm 0.83$
$LBP^{u2}_{8,1}$	Cohn-Kanade	$96.04\% \pm 0.41$
LDPv	KDEF	$69.90\% \pm$ -.- -
LDPv	Cohn-Kanade	$67.00\% \pm$ -.- -
LPQ	KDEF	$89.48\% \pm 0.69$
LPQ	Cohn-Kanade	$98.35\% \pm 0.13$
LFD	KDEF	$86.60\% \pm 0.31$
LFD	Cohn-Kanade	$97.45\% \pm 0.16$

Table 8.2. Results of the preliminary experiment. The recognition accuracies obtained for each feature extracted from each database is reported together with the variance of the repetitions. Due to a server crash which caused a loss of data, the variance of the LDPv descriptor can not be reported.

One issue concerning the results needs immediate attention, namely the low recognition accuracy obtained with the LDPv descriptor. Kabir et al. [2010] proved that LDPv performs better than $LBP^{u2}_{8,1}$. This is in contradiction with the results obtained here. Another contradiction about LDPv is that the accuracy from KDEF is higher than the accuracy from Cohn-Kanade. This is unexpected, because there is more data in the Cohn-Kanade database and because the other features proved better at the Cohn-Kanade database. There must be a bug in the implementation of the feature extractor developed for this experiment. Unfortunately, an extensive debugging has not been able to resolve the issue. The author is still of the opinion, that the LDPv descriptor is interesting and promising. The idea of a local feature which is robust to noise is very appealing. The Matlab code of the LDPv implementation can be found in Appendix F. The interested reader can browse through the code and see if they can catch the error. Note that the code is also available at /`experiments/preliminary_LFD_LPQ_LDPv_LBP/`. As a result of the bad LDPv implementation, LDPv is disregarded in the final two experiments.

[1]The Matlab implementation of the experiment can be found at /`experiments/preliminary_LFD_LPQ_LDPv_LBP/`.

The results from the experiment indicates, that LPQ is better than LFD, which is better than LBP. The recognition rates are higher for LPQ and LFD than for LBP, and the variance is lower. Note that the statement of being better is solely in the context of FER. LFD was originally developed as an extension of LPQ. It was designed specifically with the face recognition problem in mind. It has been proved, that LFD performs slightly better than LPQ for face recognition [Lei et al., 2011]. Therefore, it is a noticeable result that LPQ seems to perform better than LFD for FER. This tendency is further supported by the results of the following experiments.

8.3 Dimensionality reduction experiment

The dimensionality reduction experiment seeks to uncover which of PCA or SC works better for dimensionality reduction of the local features considered in this report.

The LBP, LPQ and LFD features are extracted from both the KDEF and Cohn-Kanade databases using the grid division approach. Every image is divided into 42 cells from which the feature descriptors are extracted. The final feature descriptor is made by concatenating the descriptors from each cell. This yields three descriptors with a fairly high dimensionality. The dimensionality of the descriptors are reduced to a set of multiple dimensions ranging from 7 to 140 in steps of 7 dimensions at a time. The recognition accuracy of each dimension of each feature descriptor is then evaluated using OAO multi-class SVMs. Following the method described in the preliminary experiment, the accuracy is calculated by 10 runs of cross-validation on 10 different partitions of the databases[2].

Note that the $LBP^{u2}_{8,1}$ descriptor is used. Likewise, note that the LFD descriptors were reduced to 100 statistical uniform patterns. Lei et al. [2011] states that 32 statistical uniform patterns are optimal for face recognition. The 100 patterns selected here is well beyond what was proposed for face recognition. It is therefore expected, that the selection of 100 patterns will at least not hinder the descriptiveness of LFD when used for FER.

The details of the experiment is documented in the measurement record found in Appendix C.

8.3.1 Results and discussion

The results of the dimensionality reduction experiment are presented in the following graphs. One graph is presented for all combination of the three features and two databases, thus a total of six plots are shown.

The recognition accuracies obtained form the KDEF databases are as follows:

[2]The Matlab implementation of the experiment can be found at /experiments/dimensionality_LFD_LPQ_LBP/.

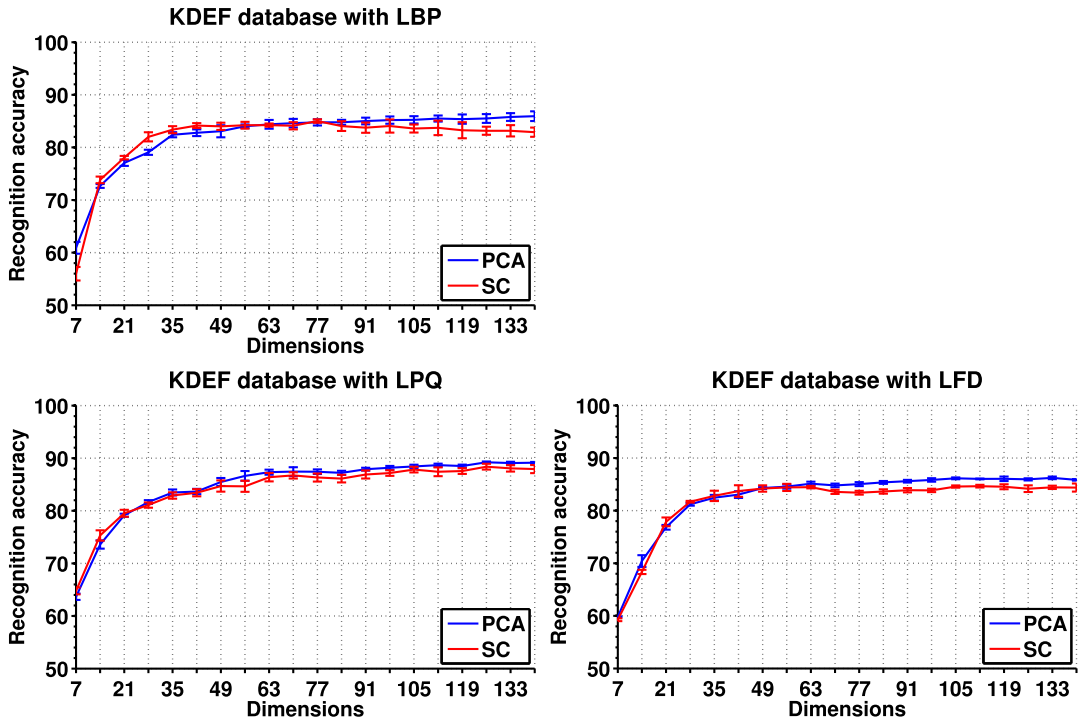

Figure 8.4. Results of the dimensionality reduction experiment performed on the KDEF database. The recognition rates are given as a percentage of correctly classified test samples obtained as a mean of 10 repetitions of five-fold cross-validation.

The recognition accuracies obtained form the Cohn-Kanade databases are as follows:

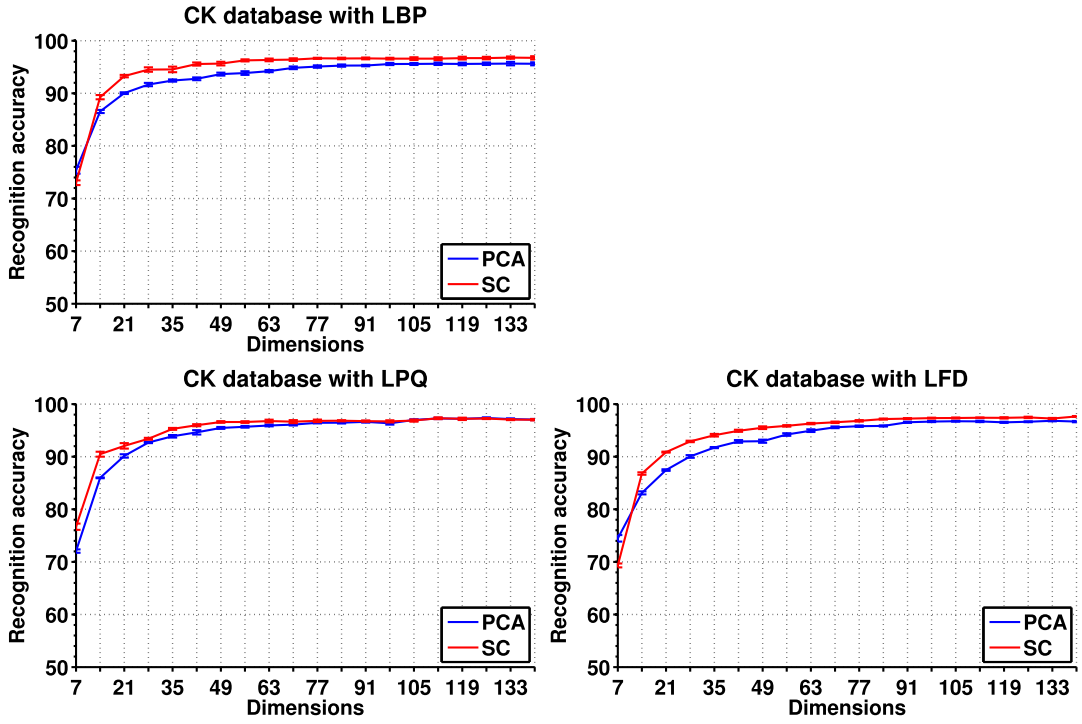

Figure 8.5. Results of the dimensionality reduction experiment performed on the CK database. The recognition rates are given as a percentage of correctly classified test samples obtained as a mean of 10 repetitions of five-fold cross-validation.

For the KDEF database, SC is better than PCA up until roughly 70 dimensions, at least for LBP. However, the variance of the SC results are larger than that of PCA. Thus, PCA can be considered more stable. PCA is better than SC after 63 dimensions for LBP, after 42 dimensions for LPQ and after 56 dimensions for LFD. The lower variance of PCA is shared between all feature extractors.

The results from the Cohn-Kanade database tells a quite different story. Except for the LPQ results above 77 dimensions, SC is the better reduction method for all three feature descriptors. The variance is equally low for PCA and SC.

In Appendix C, it is determined that the accuracy of PCA and SC is significantly different at 140 dimensions for all combinations of features and databases, except for LPQ in combination with the Cohn-Kanade database. Thus, it can be stated, that PCA is better than SC for all feature descriptors from the KDEF database reduced to 140 dimensions. Likewise, it can be stated that SC is better than PCA for the LBP and LFD feature descriptors from the Cohn-Kanade database reduced to 140 dimensions.

It could be argued, that the results are inconclusive and that PCA and SC seems to perform equally good. However, some arguments supports the use of PCA over SC. The SC approach requires the formation of a new affinity matrix every time a new, unknown sample is observed. Then, the eigenvectors of the new affinity matrix needs to be computed. PCA only has to do a transformation of basis every time a new sample is observed. Therefore, PCA is computationally simpler than SC. If the improvement of SC over PCA had been considerable, then the heavier computational load could be accepted. However, the improvement was only apparent in the larger of the two test databases. Further, the improvement was relatively subtle. Therefore, it is decided to only use PCA for the final blur experiment.

As a side observation, it can be noted that LPQ seems to perform better for FER than LBP and LFD.

8.4 Blur experiment

The blur experiment seeks to uncover if LFD is better than LBP and LPQ for blurred facial expression images. Three recognition systems are formed, one for each feature descriptor. The recognition systems are trained using sharp images. The systems is then used to recognize a set of test images which are blurred with different sizes of Gaussian blur. In the end, it is calculated how well the systems coped with the blurring.

The feature descriptors are extracted as explained in Section 8.3. However, the features are also extracted from blurred versions of the images. The images are blurred by Gaussian blurs with a variance ranging from 0 to 4 in steps of 0.25. From the dimensionality reduction experiment it was concluded, that all things considered, PCA was the better dimensionality reduction method. Therefore, only PCA is used in this experiment. The dimensionality of all feature descriptors from both the sharp and blurred images are reduced in steps from 7 to 140 dimensions with a step size of 7. The 10 repetitions of five-fold cross-validations is also done in this experiment. The recognition accuracy is

calculated by OAO multi-class SVMs. The SVM parameters are estimated using the grid search approach[3].

An example of one image blurred with the 17 Gaussian blurring variances tested in the experiment is presented in Figure 8.6.

Figure 8.6. Example of an image blurred by the 17 Gaussian blur kernels used in this experiment. The top left image is the original sharp image. The blurring variance increases by each column from left to right. The image is from female subject one from the KDEF database, showing the afraid facial expressions.

The details of the experiment are documented in the measurement record found in Appendix D.

8.4.1 Results and discussion

The results of the blurring experiment are presented in the graphs below. One graph is presented for all combination of the three features and two databases, thus a total of six plots are shown.

The recognition accuracies obtained form the KDEF databases are as follows:

[3]The Matlab implementation of the experiment can be found in /`experiments/blur_LFD_LPQ_LBP/`.

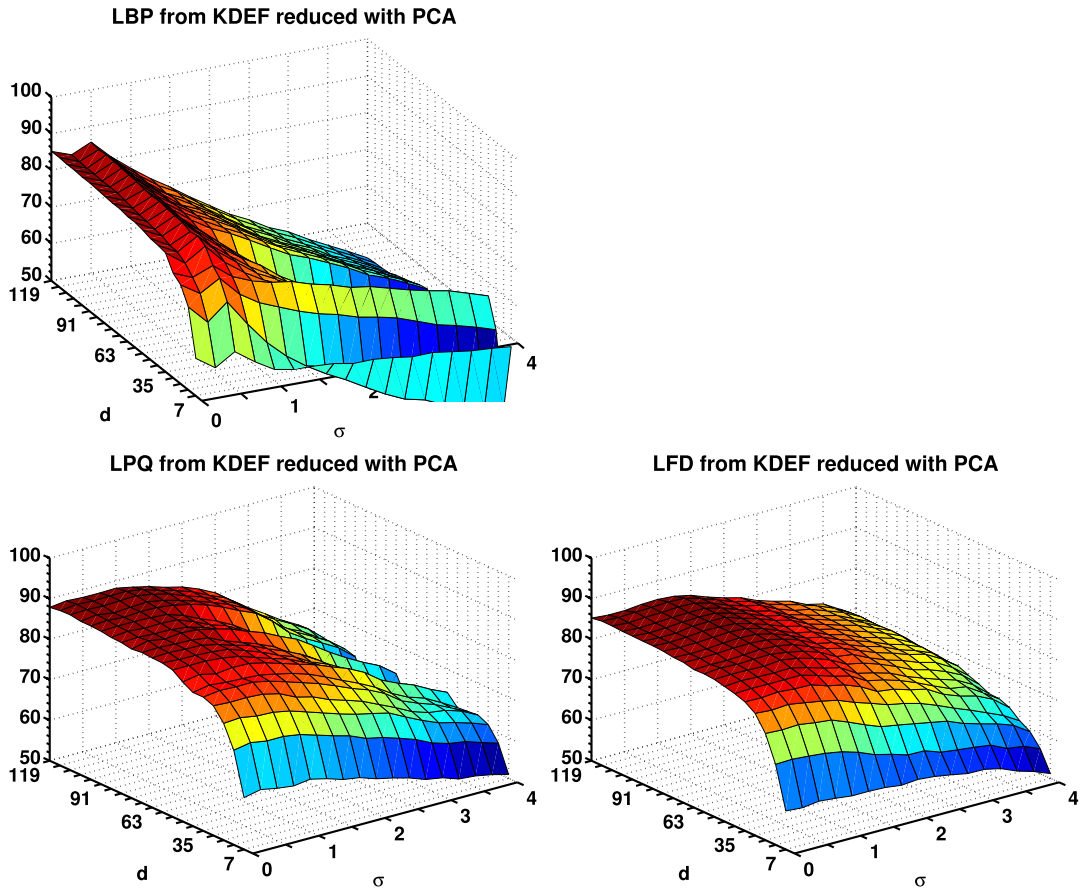

Figure 8.7. Results of the blur experiment performed on the KDEF database. The recognition rates are given as a percentage of correctly classified test samples obtained as a mean of 10 repetitions of five-fold cross-validation. Note that the z-axis describes the recognition accuracy in percentage. The irregular fluctuations are probably due to the sparsity of the SVM parameter estimation.

The recognition accuracies obtained form the Cohn-Kanade database are as follows:

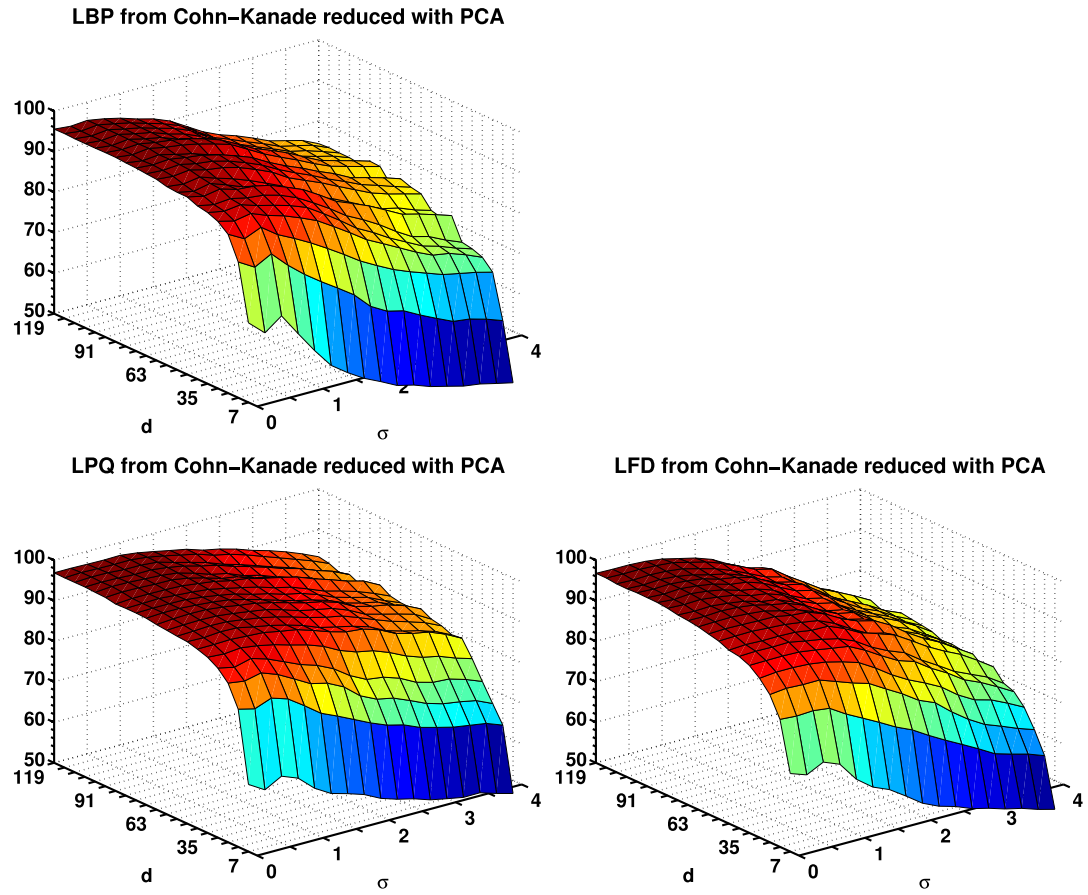

Figure 8.8. Results of the blur experiment performed on the Cohn-Kanade database. The recognition rates are given as a percentage of correctly classified test samples obtained as a mean of 10 repetitions of five-fold cross-validation. Note that the z-axis describes the recognition accuracy in percentage. The irregular fluctuations are probably due to the sparsity of the SVM parameter estimation.

On both figures, some irregular fluctuation of the recognition accuracy is present. Especially the LBP accuracy shown on the top left of Figure 8.7 shows large fluctuations. In the ideal world, it would be expected that the recognition rate increases as the number of dimension increase and decreases as the variance of the Gaussian blur increases. The source behind the irregular fluctuations is probably the sparsity of the grid search parameter estimation for the SVMs. The parameter combinations are tested in a grid with finite size. Thus, the optimal recognition point might be positioned in between two test points. Therefore, the accuracy could be different from the expected if the parameters are not optimal.

The accuracy of the LBP descriptor extracted from the KDEF database shows a lot of irregularity. Especially the ridge present at 28 dimensions is interesting. It seems unlikely that the recognition rate would have a steep drop off after 28 dimensions. Therefore, it is assessed, that the high ridge obtained at 28 dimensions originates from a lucky hit of a parameter top point. This suspicion is enhanced when considering the LBP accuracy plot from the Cohn-Kanade database shown at the top left of Figure 8.8. This plot is much smoother than the KDEF equivalent.

Judging from the plots from both databases, it seems that LPQ is in general the best of the features for both sharp and blurred images. The only case where this is not true is for the higher blur variances in the KDEF database. There, LFD actually provides a higher recognition rate than LPQ after a blur variance of 2.75. LPQ has a sudden drop off where LFD retains a flatter drop. The almost opposite effect is observed in the Cohn-Kanade database. There, LFD drops off faster than LPQ. As explained in Appendix D, the reason might be that LFD is not punished as hard as LPQ when the input images contains subtle facial expressions which are blurred. The higher number of samples in the Cohn-Kanade database might also have something do with it. In the appendix, it is calculated that on average, LFD is 0.35 percentage points better than LPQ over the entire accuracy surface from the KDEF database. For the Cohn-Kanade database, LPQ is on average 3.36 percentage points better than LFD. All in all, it seems that LPQ is the most descriptive feature, unless the input images contains subtle features and are severely blurred. In general, this observation matches the observations made in Section 8.1 and Section 8.3.

The reader should be aware, that the dimensionality of the LFD descriptors was reduced to 100 statistical uniform patterns. The dimensionality of the LPQ feature were not reduced by statistical uniform patterns. As described in Appendix E, a qualified guess for the optimal number of statistical uniform patterns for LFD is 22 for the magnitude part and 18 for the phase part. A recognition accuracy experiment should be performed to test whether these statistical uniform patterns could be used while retaining the recognition accuracy of the descriptor. If so, the computational cost of the subsequent dimensionality reduction and classification could be significantly lowered.

Conclusion 9

This chapter seeks to conclude and provide closure to the project. First, a small recap of the problem statement is provided. Second, the questions stated in the problem statement are answered. Finally, some observations concerning interesting subjects for future research are provided.

This project sought to answer the three following questions: Will the *Local Frequency Descriptor* (LFD) provide a higher recognition rate than *Local Phase Quantization* (LPQ) for facial expression recognition in blurred images? Will the LFD provide better results than the two popular feature extractors *Local Binary Patterns* (LBP) and *Local Directional Pattern variance* (LDPv) which has provided promising results for recognizing facial expressions in sharp images? Will *Spectral Clustering* (SC) reduce the dimensionality of the local feature descriptors to a reduced feature space which better discriminates the clusters of different classes than the so far popular *Principal Component Analysis* (PCA) reduction method?

The three questions was answered by constructing a set of recognition systems, all based on the *Support Vector Machine* (SVM) classifiers, but with different combinations of feature descriptors and dimensionality reduction methods. Due to an unresolved implementation problem, the LDPv descriptor was removed from the tests. Thus, a total of six recognition systems were implemented. The performance of the systems was measured by the use of two facial expression databases, namely the *Karolinska Directed Emotional Faces* (KDEF) database and a subset of the *Cohn-Kanade* database. When combined with the six recognition systems, a total of 12 experiments were performed.

The experiments showed, that for the facial expression recognition problem in blurred images, LFD yields a higher recognition rate than LBP but a lower recognition rate than LPQ. The same characteristic proved true for facial expression recognition in sharp images.

Further, the experiments showed that SC outperforms PCA for the Cohn-Kanade database but not for the KDEF database. The improvements observed when using the Cohn-Kanade database was deemed too subtle to justify the higher computational requirements of the SC approach.

The questions raised in the problem statement is answered as follows: Will LFD provide a higher recognition rate than LPQ for facial expression recognition in blurred images? In general: No, it will not. However, there seems to be more to the story. The results of the experiments conducted on the KDEF database showed, that LPQ was better than LFD at lightly blurred images. As the blurring increased, the recognition rate of LPQ dropped at a faster rate until LFD was actually better. When averaging the differences

between LFD accuracy surface and the LPQ accuracy surface, it turned out, that LFD was 0.35 percentage points better than LPQ on average. However, the recognition rate of LFD stayed below that of LPQ over the entire accuracy surface from the Cohn-Kanade database. The accuracy of LFD even dropped off faster than that of LPQ for increasing blur. On average, LPQ was 3.36 percentage points better then LFD for the Cohn-Kanade database. In general, the images contained in the Cohn-Kanade database showed more explicit facial expressions than those in the KDEF database. Therefore, the results seemed to indicate, that LPQ is better than LFD, except for subtle emotions in severely blurred images.

Will LFD provide better results than the two popular feature extractors LBP and LDPv which has provided promising results for recognizing facial expressions in sharp images? Due to implementation problems, this question could only be answered on behalf of LBP, and the answer seemed to be yes. LFD outperformed LBP for blurred as well as sharp images.

Will SC reduce the dimensionality of the local feature descriptors to a reduced feature space which better discriminates the clusters of different classes than the so far popular PCA reduction method? This question is a bit tricky to answer from the results. However, the answer seems to be yes, at least for the Cohn-Kanade database. However, the performance gain was so subtle, that the higher computational cost of SC compared to that of PCA was difficult to justify.

It should be noted, that the dimensionality of the raw LFD was reduced to 100 statistical uniform patterns for all experiments. This might have hindered its descriptiveness when compared to LPQ.

9.1 Future research

Future research should try to uncover an optimal number of statistical uniform patterns for the LFD descriptor when applied to facial expression recognition. After the main experiments documented in this report were done, another and better documented extraction of statistical uniform patterns were tried. This extraction is documented in Appendix E. It showed, that 40 statistical uniform patterns should be enough for LFD. In future research, it would also be interesting to test, if the dimensionality of the LPQ descriptor could be reduced by using only statistical uniform patterns without loosing a significant amount of its descriptiveness.

Previous research has documented good performance boosts by giving higher weight to the most descriptive parts of the face when using template matching for classification. However, it has also been proved that SVMs provide better recognition rates than template matching methods like Chi-square statistics. Therefore, future research should seek to uncover an image grid cell weighting procedure which can put higher weights to certain cells in the image grid while using SVMs. One solution to this problem could be to train a One-Against-One multi-class SVM for each cell in the image grid. Thus, with seven facial expressions, seven SVMs should be trained for each image cell. The classification would

be done by a voting scheme where the SVMs from all cells vote for the class they believe a new, unknown sample belongs to. The cell-weights would be used to put less emphasis on the votes from cells located on the rim of the face and more emphasis on the votes form the cells containing the eyes, nose and mouth.

A final remark for future research is the use of realistic datasets. The images contained in most of the available facial expression databases today are unrealistic. Therefore, it is uncertain if systems which uses these databases as their training data will be able to recognize true facial expressions at all. Future research should try to turn their research towards recognizing facial expressions in real environments on persons performing real tasks.

Derivation of the Lagrangian Dual form

A

This appendix describes how the dual form is derived from the primal form of the Lagrangian optimization problem involved in SVMs. The primal form of the problem is:

$$L_P(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} a_n \left(t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) - 1\right) \tag{A.1}$$

From the primal form, the following stationary points of $\mathbf{w}$ and b are found:

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n) \quad \text{and} \quad \sum_{n=1}^{N} a_n t_n = 0 \tag{A.2}$$

This can be substituted back into the primal form Lagrangian. By doing so, an optimization problem which only depends on $\mathbf{a}$ is achieved. The dual form can be derived as follows:

$$\begin{aligned}
L_D(\mathbf{a}) &= \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} a_n \left(t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) - 1\right) \\
&= \frac{1}{2}\mathbf{w}\cdot\mathbf{w} - \sum_{n=1}^{N} a_n \left(t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) - 1\right) \\
&= \frac{1}{2}\left(\sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n)\right)\cdot\left(\sum_{m=1}^{N} a_m t_m \phi(\mathbf{x}_m)\right) - \\
&\quad \sum_{q=1}^{N} a_q \left(t_q\left(\left(\sum_{p=1}^{N} a_p t_p \phi(\mathbf{x}_p)\right)^T \phi(\mathbf{x}_q) + b\right) - 1\right) \\
&= \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\left(a_n a_m t_n t_m \phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m)\right) - \\
&\quad \sum_{q=1}^{N}\left[a_q t_q\left(\left(\sum_{p=1}^{N} a_p t_p \phi(\mathbf{x}_p)\right)^T \phi(\mathbf{x}_q) + b\right) - a_q\right] \\
&= \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\left(a_n a_m t_n t_m \phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m)\right) - \\
&\quad \sum_{q=1}^{N} a_q t_q \sum_{q=1}^{N}\left(\left(\sum_{p=1}^{N} a_p t_p \phi(\mathbf{x}_p)\right)^T \phi(\mathbf{x}_q) + b\right) + \sum_{q=1}^{N} a_q
\end{aligned}$$

$$= \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m K(\mathbf{x}_n, \mathbf{x}_m) - \sum_{q=1}^{N} a_q \tag{A.3}$$

Where:

$$K(\mathbf{x}_n, \mathbf{x}_m) = \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m)$$

Measurement Record: Preliminary experiment

B

This measurement report documents the preliminary experiment. It is composed of four sub-experiments: one for each of LBP, LDPv, LPQ and LFD. The preliminary experiment is conducted to establish a basis performance. The basis performance reveals the ability of each of the four features without being manipulated by dimensionality reduction. The results can be used to reveal how much dimensionality reduction influences the performance in the subsequent experiments.

The implementation of the experiment can be found at /`Experiments/preliminary_LFD_LPQ_LDPv_LBP/`.

This measurement record is composed of the following sections: Methods, Experimental Setup, Results, Analysis and Discussion and Conclusion.

B.1 Methods

The four feature extractors: LBP, LDPv, LPQ and LFD is used in this experiment in combination with the SVM classifier. No dimensionality reduction is performed. The recognition accuracy of each feature is calculated based on two distinct databases: KDEF and Cohn-Kanade. In total, this results in eight sub-experiments which are defined by Table B.1.

No.	Database	Feature	Classifier
1.a	KDEF	LBP	SVM
1.b	Cohn-Kanade	LBP	SVM
2.a	KDEF	LDPv	SVM
2.b	Cohn-Kanade	LDPv	SVM
3.a	KDEF	LPQ	SVM
3.b	Cohn-Kanade	LPQ	SVM
4.a	KDEF	LFD	SVM
4.b	Cohn-Kanade	LFD	SVM

Table B.1. Specifications of the database, feature extractor and classifier involved in each sub-experiments documented by this measurement record.

Each experiment follows the following pipeline: Load database → segment each image → divide the images into grids → extract features from each image → calculate recognition performance.

The samples of both databases are divided into cross-validation groups before the experiment is executed. By during so, it ensured that each sub-experiment is run on the exact same data. This will eliminate any differences which could be present due to a different partitioning of the cross-validation groups.

The recognition accuracy is calculated as the mean recognition rate obtained by 10 repetitions of five-fold cross-validation with 10 different partitions of the databases. By doing so, the risk of a bad partitioning of the relatively small databases is lowered. The variances of the repetitions are calculated as well.

The images are segmented as explained in Chapter 4. The features are extracted as explained in Chapter 5. The classification is done by the OAO soft-margin SVM with RBF kernel, as explained in Chapter 7.

Every image is divided into a grid and the feature descriptors are extracted from each cell in the grids. The resulting descriptors from a given feature from a given image are concatenated together to form one descriptor for that image.

Note that no statistical binary patterns are used for LFD. This is because the raw recognition accuracy is desired, without influence of dimensionality reduction. However, it has been shown that LBP can be reduced to only the uniform patterns without significant loss in descriptiveness [Ojala et al., 2002]. Therefore, the $LBP^{u}_{8,1}2$ descriptor is used.

The recognition performance is calculated by five-fold cross-validation. The final recognition rate is calculated as the mean value of the recognition rate from each cross-validation.

The grid search approach is used to get optimal estimates of C and γ for the SVMs.

B.2 Experimental setup

The experiments are all implemented in Matlab. The used implementations are specified in Table E.1.

Method	**Implementer**	**Reference**
LBP	Other	[Heikkilä and Ahone, 2013]
LDPv	Self	-
LPQ	Other	[Rahtu et al., 2012]
LFD	Self	-
SVM	Other	LIBSVM [Chang and Lin, 2011]

Table B.2. Specification of the Matlab implementations of the used methods. The implementer state if the method is self-implemented or if it is implemented by others. In that case, a reference to the implementer is provided.

The parameters is defined as specified by Table E.2

Parameter	Designator	Value
No. of grid rows	k	7
No. of grid columns	l	6
Cross-validation	cv	5-fold
u.p. for LBP	LPQ_up	58
Max. pattern for LDPv	LDPv_k	3
Dim. for LDPv	LDPv_d	256
s.u.p. for LPQ	LPQ_sup	*256
s.u.p. for LFD	LFD_sup	*512
The window size for LPQ and LFD	M	7
The frequency for LPQ and LFD	a	1/7
The decorrelation parameter for LPQ	ρ	0.9

Table B.3. Specification of the settings of the parameters used in the experiment. The abbreviations are as follows: s.u.p. is statistical uniform patterns, u.p. is uniform patterns and dim. is dimensionality. *) no statistical uniform patterns were extracted.

The feature descriptor length will be $k \cdot l \cdot d$ where d is the length of one feature descriptor from one of the image grids. The resulting descriptor lengths are as follows: LBP) $42 \cdot 58 = 2436$, LDPv) $42 \cdot 256 = 10\,752$, LPQ) $42 \cdot 256 = 10\,752$, and LFD) $42 \cdot 512 = 21\,504$.

The KDEF database has 980 samples. A subset of 1596 samples are drawn from the Cohn-Kanade database[1].

The SVM parameter estimation grid search is done over the following range:

Parameter	Start	End	Step size
C	$\log_2(1)$	$\log_2(24)$	$\log_2(1)$
γ	$\log_2(-24)$	$\log_2(1)$	$\log_2(1)$

Table B.4. The grid search parameters used to find an optimal set of values for C and γ.

B.3 Results

This section documents the results of the experiment. The results are the mean recognition rates obtained as the mean of all repetitions of cross-validation, for all feature descriptors, for both databases. The recognition rate of the $LBP^{u2}_{8,1}$ descriptor is as follows:

[1]The indices to the samples drawn from the Cohn-Kanade database can be found in /databases/Cohn-Kanade/indices.mat.

Feature	Database	Recognition rate
$LBP^{u2}_{8,1}$	KDEF	$85.58\% \pm 0.83$
$LBP^{u2}_{8,1}$	Cohn-Kanade	$96.04\% \pm 0.41$

Table B.5. Result of sub-experiment 1. The result is the mean recognition accuracy obtained by LBP features extracted from each database.

Unfortunately, a runtime error caused a data loss when executing the experiment calculating the variance of the LDPv descriptor. As a result, only the mean accuracy is available but not the variance. The recognition rate of the LDPv descriptor for $k = 3$ is as follows:

Feature	Database	Recognition rate
LDPv	KDEF	69.90%
LDPv	Cohn-Kanade	67.00%

Table B.6. Result of sub-experiment 2. The result is the mean recognition accuracy obtained by LDPv features extracted from each database. Note that $k = 3$ for the LDPv descriptor. Unfortunately a runtime error prevented a calculation of the variances.

The recognition rate of the LPQ descriptor is as follows:

Feature	Database	Recognition rate
LPQ	KDEF	$89.48\% \pm 0.69$
LPQ	Cohn-Kanade	$98.35\% \pm 0.13$

Table B.7. Result of sub-experiment 3. The result is the mean recognition accuracy obtained by LPQ features extracted from each database.

The recognition rate of the LFD descriptor is as follows:

Feature	Database	Recognition rate
LFD	KDEF	$86.60\% \pm 0.31$
LFD	Cohn-Kanade	$97.45\% \pm 0.16$

Table B.8. Result of sub-experiment 4. The result is the mean recognition accuracy obtained by LFD features extracted from each database.

B.4 Analysis

This section analysis on the results presented in the previous section.

Based on the results obtained, the feature extractors sorted based on their recognition accuracy would be: LDPv, LBP, LFD and LPQ.

B.5 Discussion and Conclusion

This section provides a discussion and conclusion of the experiment.

It has been proven by Kabir et al. [2010] that the LDPv descriptor outperforms the LBP descriptor. They use a subset of the Cohn-Kanade database to get their results and they also use SVMs. Clearly, there is a problem regarding the implementation of the LDPv descriptor used in this experiment. Unfortunately, Kabir et al. [2010] did not publish their feature extraction code. Therefore, the feature extractor had to be implemented based on the description in their paper. An extensive debugging has taken place in order to enhanve the recognition rate, but the problem has not been solved so far. Something must have slipped under the eye of the author or maybe Kabir et al. [2010] apply a trick which were not documented in their paper. The author of this report still has a strong feeling for the LDPv descriptor. Its noise robustness and use of Kirch masks seems very cleaver. Therefore, the Matlab implementation of the feature extractor is provided in Appendix F. This allows the interested reader to run the code by themselves and possibly locate the bug. Note that the Matlab code can also be found in /`experiments/preliminary_LFD_LPQ_LDPv_LBP/`.

Besides the LDPv bug, it seems that LPQ outperforms both LFD and LBP.

Measurement Record: Dimensionality reduction experiment

C

This measurement record documents the dimensionality reduction experiment. It is composed of three sub-experiments: one for each of LBP, LPQ and LFD. The dimensionality reduction experiment is conducted to prove which of PCA or SC performs better for FER in combination with local features.

The LDPv descriptor is not a part of this experiment due to a bad implementation as documented in Appendix B.

The implementation of the experiment can be found at /Experiments/dimensionality_LFD_LPQ_LBP/.

This measurement record is composed of the following sections: Methods, Experimental Setup, Results, Analysis and Discussion and Conclusion.

C.1 Methods

The dimensionality of the three feature descriptors is reduced by both PCA and SC. Then, their recognition rate is calculated with SVM. The experiments are performed using both the KDEF and the Cohn-Kanade database. Therefore, a total of 12 experiments are conducted. They are specified in Table C.1.

No.	Database	Feature	Dimensionality reduction	Classifier
1.1.a	KDEF	LBP	PCA	SVM
1.1.b	KDEF	LBP	SC	SVM
1.2.a	Cohn-Kanade	LBP	PCA	SVM
1.2.b	Cohn-Kanade	LBP	SC	SVM
2.1.a	KDEF	LPQ	PCA	SVM
2.1.b	KDEF	LPQ	SC	SVM
2.2.a	Cohn-Kanade	LPQ	PCA	SVM
2.2.b	Cohn-Kanade	LPQ	SC	SVM
3.1.a	KDEF	LFD	PCA	SVM
3.1.b	KDEF	LFD	SC	SVM
3.2.a	Cohn-Kanade	LFD	PCA	SVM
3.2.b	Cohn-Kanade	LFD	SC	SVM

Table C.1. Specifications of the database, feature extractor, dimensionality reduction and classifier involved in each sub-experiment documented by this measurement record.

Each experiment follows the following pipeline: Load database → segment each image → divide the images into grids → extract feature descriptors from each cell in the image grid and form the final descriptors → reduce the dimensionality → calculate recognition performance. The samples are divided into cross-validation groups before the experiment is executed. By during so, it is ensured that each sub-experiment is run on the exact same data. This will eliminate any differences arising due to differences in the data partitions.

The recognition accuracy is calculated as the mean recognition rate obtained by 10 repetitions of five-fold cross-validation with 10 different partitions of the databases. By doing so, the risk of a bad partitioning of the relatively small databases is lowered. The variances of the repetitions are calculated as well.

The images are segmented as explained in Chapter 4. The features are extracted as explained in Chapter 5. The dimensionality reduction is done as explained in Chapter 6. The classification is done as explained in Chapter 7.

The grid search approach is used to get optimal estimates of C and γ for the SVMs.

Prior to running the experiments, a plot of the amount of variance explained by each Principal Component of PCA is obtained. The plot is used to choose a decent number of trial dimensions for the experiments.

In Appendix E, a possibly optimal number of statistical uniform patterns was reported to be 18 for LPQ and 40 for LFD. However, this information was unknown at the time of execution of this experiment. Therefore, the full LPQ descriptor is used and 100 statistical uniform patterns are extracted for LFD, 50 for the magnitude and 50 for the phase.

The grid search approach is used to get optimal estimates of C and γ for the SVMs.

C.2 Experimental setup

The experiments are all implemented in Matlab. The used implementations are specified in Table C.2.

Method	Implementer	Reference
LBP	Other	[Heikkilä and Ahone, 2013]
LDPv	Self	-
LPQ	Other	[Rahtu et al., 2012]
LFD	Self	-
PCA	Other	Std. Matlab function
SC	Self	-
SVM	Other	LIBSVM [Chang and Lin, 2011]

Table C.2. Specification of the Matlab implementations of the used methods. The implementer filed state if the method is self-implemented or if it was implemented by others. In that case, a reference to the implementer is provided.

The parameter values are defined in Table C.3

Parameter	Designator	Value
No. of grid rows	k	7
No. of grid columns	l	6
Cross-validation	cv	5-fold
Repetitions	rep	10
Dimensions	dims	7 to 140 in steps of 7
u.p. for $\text{LBP}^{u2}_{8,1}$	LPQ_up	58
s.u.p. for LPQ	LPQ_sup	*256
s.u.p. for LFD	LFD_sup	100
The window size for LPQ and LFD	M	7
The frequency for LPQ and LFD	a	1/7
The decorrelation parameter for LPQ	ρ	0.9

Table C.3. Specification of the settings of the parameters used in the experiment. The abbreviations are as follows: s.u.p. is statistical uniform patterns and u.p. is uniform patterns. *) no statistical uniform patterns were extracted.

The feature descriptor length is $k \cdot l \cdot d$ where d is the length of one feature descriptor from one of the image grids. The resulting descriptor lengths are as follows: LBP) $42 \cdot 58 = 2436$, LPQ) $42 \cdot 256 = 10\,752$, and LFD) $42 \cdot 100 = 4200$.

The KDEF database has 980 samples. A subset of 1596 samples are drawn from the Cohn-Kanade database[1].

[1]The indices to the samples drawn from the Cohn-Kanade database can be found in /databases/Cohn-Kanade/indices.mat.

The grid search approach is used to get optimal estimates of C and γ for the SVMs. The SVM parameter estimation grid search is done over the following range:

Parameter	Start	End	Step size
C	$\log_2(1)$	$\log_2(24)$	$\log_2(1)$
γ	$\log_2(-24)$	$\log_2(1)$	$\log_2(1)$

Table C.4. The grid search parameters used to find an optimal set of values for C and γ.

The following plots shows the amount of variance explained by each Principal Component, calculated for all feature descriptors for both databases:

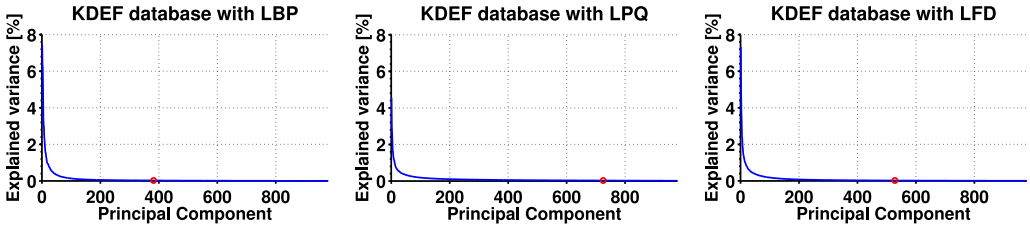

Figure C.1. Plots of the variance explained by each Principal Component. There is one plot for each feature extractor and all features were extracted from the KDEF database. The red dot mark the 95% variance point.

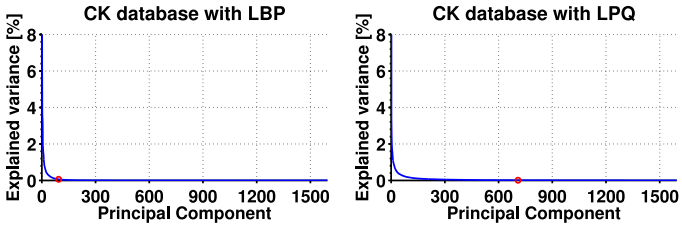

Figure C.2. Plots of the variance explained by each Principal Component. There is one plot for each feature extractor and all features were extracted from the Cohn-Kanade database. The red dot mark the 95% variance point. Unfortunately, a computer with the required amount of RAM to do PCA of the LFD extracted from the entire Cohn-Kanade database were not accessible when these figures where made. As a result, the explained amount of variance for each Principal Components of the LFD features from the Cohn-Kanade database is unknown.

It can be seen that the amount of variance is dropping off quite rapidly. The 95% variance point includes quite a large amount of Principal Components. As it is desired to reduce the number of dimensions as much as possible, it is chosen to try at max 140 dimensions. It can be seen from the plots above, that the individual components above 140 dimensions explains quite a low amount of variance compared to those below 140. Therefore, a maximum test dimensionality of 140 for the subsequent experiments seems to be a good compromise. The range of the tested dimensions in the subsequent experiments is specified in the following table:

Start dimension	End dimension	Step size
7	140	7

C.3 Results

This section presents the results from the experiment. In total, there is 12 sets of results: one for each of the experiments specified in Table C.1. As it would take up a lot of space to show all the detailed results here and convey very little understanding, only plots of the data are presented [2]. For each combination of database and feature extractor, the results from PCA and SC is presented in the same plot. This allows for easy comparison.

Each plot has the number of reduced dimensions on the first axis and the recognition accuracy on the second axis. Each point on the graph has an error bar which specifies the variance of the accuracy over all 10 repetitions. The plots are shown in the following order: KDEF+LBP, KDEF+LPQ, KDEF+LFD, Cohn-Kanade+LBP, Cohn-Kanade+LPQ, and Cohn-Kanade+LFD.

After the plots, the confusion matrices for all experiments obtained at 140 dimensions are presented. There is also six of those.

C.3.1 Recognition rates for the KDEF database

The following three plots documents the recognition rates for different dimensions of all feature descriptors extracted form the KDEF database:

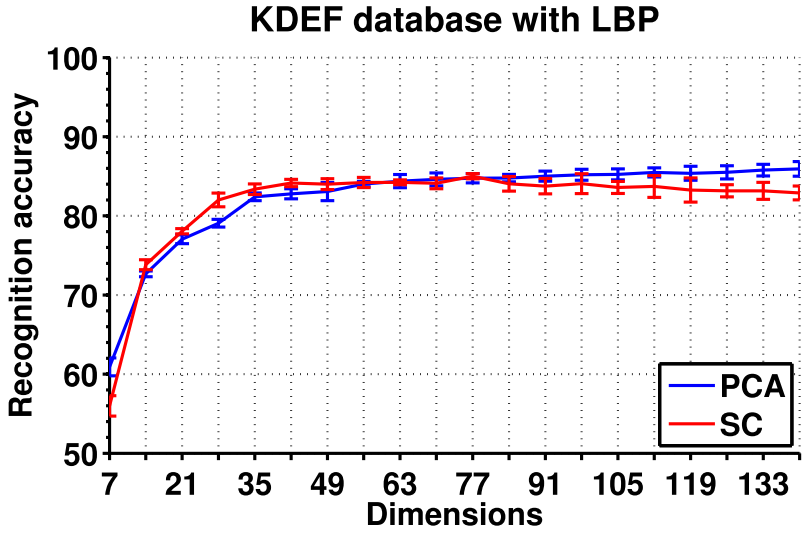

Figure C.3. Result of experiment 1.1: Plot showing the LBP recognition rates for different numbers of reduced dimensions. The results are obtained using the KDEF database.

[2] The complete data can be found in /Results/dimensionality/

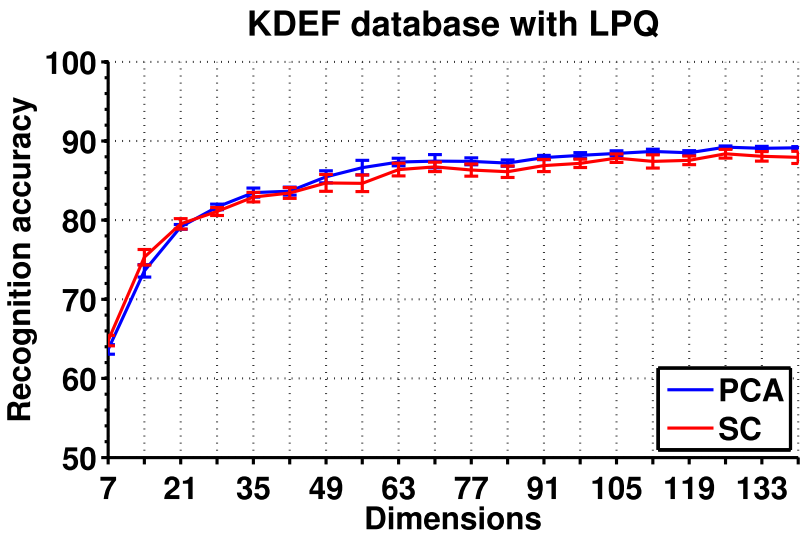

Figure C.4. Result of experiment 2.1: Plot showing the LPQ recognition rates for different numbers of reduced dimensions. The results are obtained using the KDEF database.

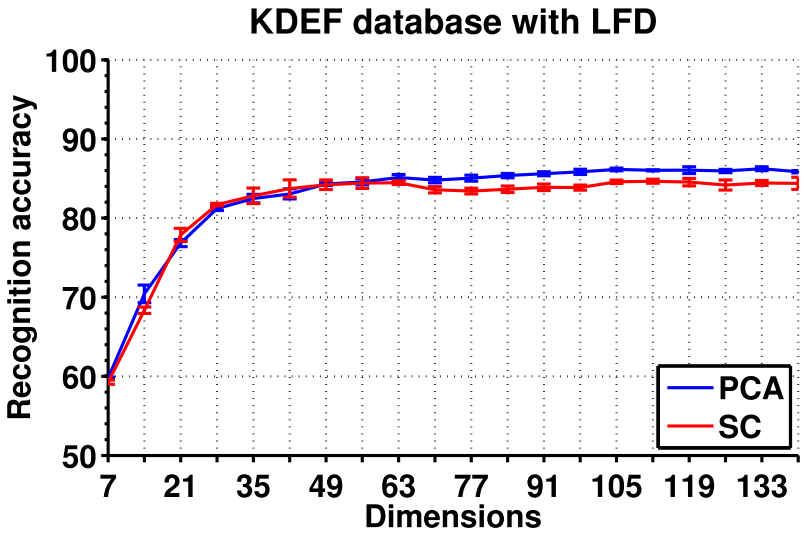

Figure C.5. Result of experiment 3.1: Plot showing the LFD recognition rates for different numbers of reduced dimensions. The results are obtained using the KDEF database.

C.3.2 Recognition rates for the Cohn-Kanade database

The following three plots documents the recognition rates for different dimensions of all feature descriptors extracted form the Cohn-Kanade database:

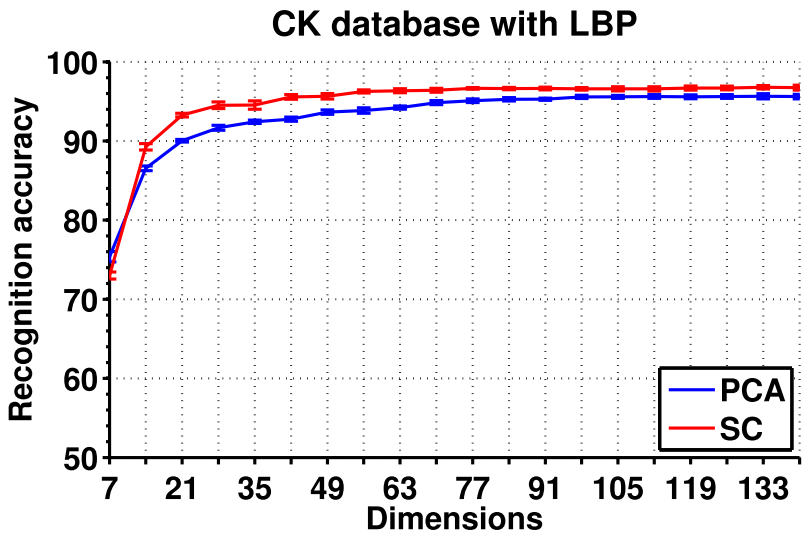

Figure C.6. Result of experiment 1.2: Plot showing the LBP recognition rates for different numbers of reduced dimensions. The results are obtained using the Cohn-Kanade database.

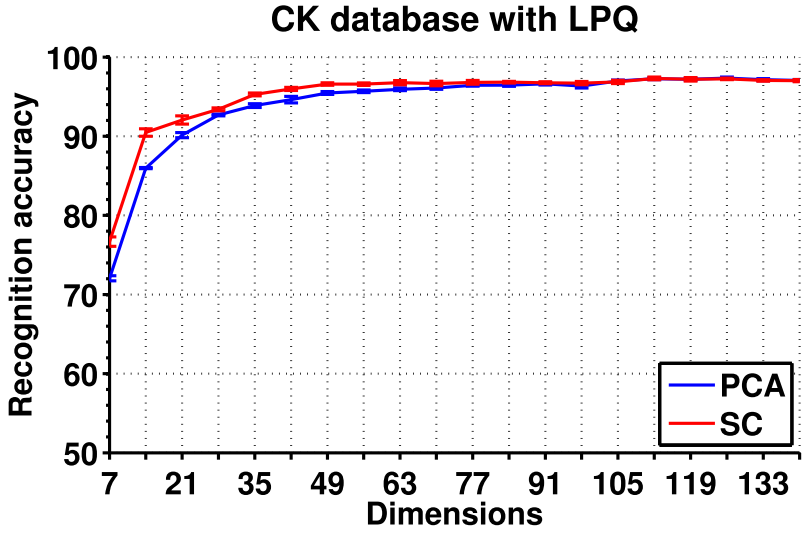

Figure C.7. Result of experiment 2.2: Plot showing the LPQ recognition rates for different numbers of reduced dimensions. The results are obtained using the Cohn-Kanade database.

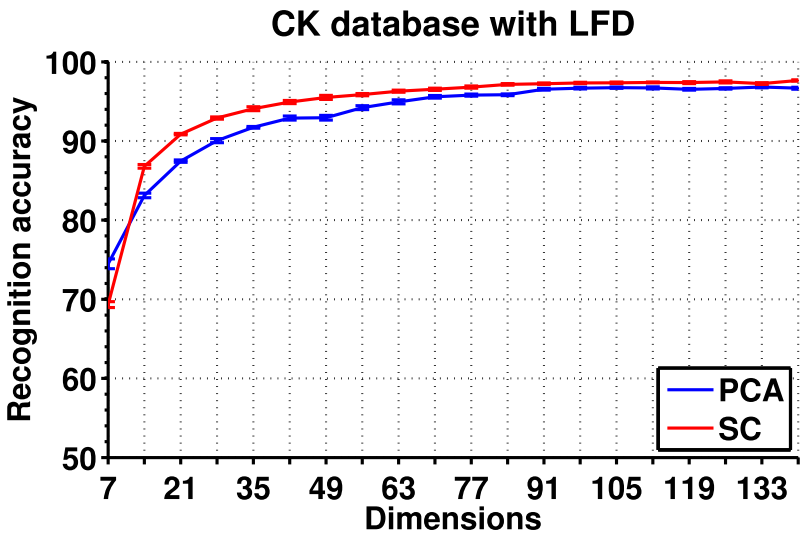

Figure C.8. Result of experiment 3.2: Plot showing the LFD recognition rates for different numbers of reduced dimensions. The results are obtained using the Cohn-Kanade database.

C.3.3 Confusion matrices resulting form the KDEF database

The following six tables presents the confusion matrices for 140 dimensions. They are presented in the following order: KDEF+LBP+PCA, KDEF+LBP+SC, KDEF+LPQ+PCA, KDEF+LPQ+SC, KDEF+LFD+PCA, and KDEF+LFD+SC.

	Afraid	**Angry**	**Disgusted**	**Happy**	**Neutral**	**Sad**	**Surprised**
Afraid	72.71	2.86	2.33	1.52	3.41	6.91	11.26
Angry	3.44	87.28	6.39	0.69	2.94	1.71	0.13
Disgusted	0.94	4.39	87.23	1.37	0.47	5.97	0.00
Happy	0.84	1.18	0.49	96.21	0.66	0.14	0.00
Neutral	1.94	2.21	0.07	0.00	89.05	1.18	1.50
Sad	9.24	2.01	3.49	0.20	2.81	83.93	0.21
Surprised	10.89	0.07	0.00	0.00	0.67	0.16	86.89

Table C.5. Confusion matrix of the LBP features extracted from the KDEF database, reduced to 140 dimensions by PCA.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	62.28	3.52	1.38	0.62	5.18	8.73	14.82
Angry	4.90	86.34	6.17	0.57	2.03	2.02	0.14
Disgusted	2.94	5.97	90.01	1.12	0.63	4.68	0.00
Happy	1.59	0.67	0.38	97.09	1.57	0.15	0.00
Neutral	2.40	0.98	0.13	0.07	85.33	3.10	0.65
Sad	11.96	2.51	1.93	0.54	3.71	80.09	0.92
Surprised	13.94	0.00	0.00	0.00	1.55	1.22	83.47

Table C.6. Confusion matrix of the LBP features extracted from the KDEF database, reduced to 140 dimensions by SC.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	78.15	2.36	0.43	1.17	2.01	7.37	7.72
Angry	2.99	89.54	6.56	0.21	1.72	1.65	0.00
Disgusted	0.96	3.76	92.09	0.27	0.00	3.15	0.00
Happy	1.01	0.00	0.21	98.15	0.81	0.13	0.00
Neutral	1.80	1.37	0.00	0.07	90.68	2.08	0.13
Sad	7.85	2.97	0.71	0.14	4.39	85.55	0.28
Surprised	7.26	0.00	0.00	0.00	0.39	0.06	91.87

Table C.7. Confusion matrix of the LPQ features extracted from the KDEF database, reduced to 140 dimensions by PCA.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	74.20	2.09	0.83	0.69	0.42	6.95	9.80
Angry	3.31	90.05	6.82	0.34	1.48	2.31	0.00
Disgusted	1.55	4.02	90.14	0.42	0.49	3.37	0.00
Happy	1.02	0.07	0.19	98.00	0.97	0.14	0.00
Neutral	1.99	2.17	0.28	0.34	92.63	2.03	0.56
Sad	8.95	1.60	1.73	0.21	3.27	84.81	0.49
Surprised	8.99	0.00	0.00	0.00	0.75	0.38	89.14

Table C.8. Confusion matrix of the LPQ features extracted from the KDEF database, reduced to 140 dimensions by SC.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	70.93	2.81	3.22	1.45	2.63	9.82	9.28
Angry	3.58	86.75	8.14	0.00	1.04	3.07	0.47
Disgusted	2.19	3.61	87.74	1.27	0.00	2.62	0.00
Happy	1.72	0.15	0.00	97.22	1.37	0.00	0.00
Neutral	1.77	1.96	0.00	0.00	89.19	2.05	0.07
Sad	11.07	4.35	0.90	0.07	5.58	82.09	0.96
Surprised	8.73	0.38	0.00	0.00	0.19	0.34	89.21

Table C.9. Confusion matrix of the LFD features extracted from the KDEF database, reduced to 140 dimensions by PCA.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	66.85	1.80	1.97	0.65	2.28	11.28	11.26
Angry	3.36	89.80	6.17	0.00	1.52	3.40	0.25
Disgusted	1.85	2.82	90.48	0.72	0.85	5.96	0.00
Happy	1.69	0.00	0.46	98.57	2.11	0.00	0.21
Neutral	2.35	1.94	0.07	0.07	87.02	2.46	1.18
Sad	13.99	3.27	0.86	0.00	5.03	76.38	0.82
Surprised	9.92	0.37	0.00	0.00	1.19	0.53	86.28

Table C.10. Confusion matrix of the LFD features extracted from the KDEF database, reduced to 140 dimensions by SC.

C.3.4 Confusion matrices resulting form the Cohn-Kanade database

The following 6 tables presents the confusion matrices for 140 dimensions. They are presented in the following order: Cohn-Kanade+LBP+PCA, Cohn-Kanade+LBP+SC, Cohn-Kanade+LPQ+PCA, Cohn-Kanade+LPQ+SC, Cohn-Kanade+LFD+PCA, and Cohn-Kanade+LFD+SC.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	93.83	3.31	1.16	0.76	0.60	6.47	0.63
Angry	1.48	94.13	1.07	0.05	0.12	0.46	0.00
Disgusted	0.94	0.97	96.82	0.12	0.10	0.56	0.00
Happy	0.57	0.35	0.16	97.30	2.05	0.20	0.00
Neutral	0.27	0.00	0.00	1.45	97.13	0.00	0.00
Sad	2.60	1.24	0.79	0.33	0.00	92.28	0.00
Surprised	0.30	0.00	0.00	0.00	0.00	0.04	99.37

Table C.11. Confusion matrix of the LBP features extracted from the Cohn-Kanade database, reduced to 140 dimensions by PCA.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	93.88	0.95	0.24	0.89	0.83	4.02	0.34
Angry	0.83	98.34	0.62	0.00	0.03	0.26	0.00
Disgusted	1.69	0.44	98.98	0.00	0.00	0.30	0.00
Happy	0.43	0.00	0.16	98.17	1.62	0.13	0.08
Neutral	0.43	0.00	0.00	0.59	97.52	0.00	0.00
Sad	2.12	0.27	0.00	0.35	0.00	95.30	0.00
Surprised	0.63	0.00	0.00	0.00	0.00	0.00	99.58

Table C.12. Confusion matrix of the LBP features extracted from the Cohn-Kanade database, reduced to 140 dimensions by SC.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	96.16	2.81	0.51	1.37	0.22	3.90	0.29
Angry	0.77	95.18	1.25	0.11	0.35	0.41	0.00
Disgusted	0.84	0.69	97.20	0.47	0.06	0.00	0.00
Happy	0.27	0.25	1.04	97.02	1.05	0.00	0.00
Neutral	0.29	0.48	0.00	0.98	98.31	0.00	0.00
Sad	1.35	0.59	0.00	0.05	0.00	95.65	0.00
Surprised	0.32	0.00	0.00	0.00	0.00	0.04	99.71

Table C.13. Confusion matrix of the LPQ features extracted from the Cohn-Kanade database, reduced to 140 dimensions by PCA.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	95.85	1.16	0.50	1.04	0.89	3.17	0.04
Angry	0.78	96.46	1.94	0.05	0.44	0.28	0.00
Disgusted	1.00	1.30	97.27	0.00	0.25	0.17	0.00
Happy	0.59	0.00	0.29	97.55	0.98	0.20	0.12
Neutral	0.59	0.49	0.00	1.15	97.44	0.00	0.00
Sad	0.80	0.59	0.00	0.21	0.00	96.06	0.00
Surprised	0.39	0.00	0.00	0.00	0.00	0.11	99.84

Table C.14. Confusion matrix of the LPQ features extracted from the Cohn-Kanade database, reduced to 140 dimensions by SC.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	95.32	2.65	0.77	1.18	0.41	5.09	0.50
Angry	0.88	94.29	0.91	0.25	0.26	1.04	0.00
Disgusted	0.78	0.83	97.46	0.11	0.00	0.00	0.00
Happy	0.38	0.44	0.85	97.74	0.64	0.00	0.00
Neutral	0.54	0.00	0.00	0.47	98.69	0.00	0.00
Sad	1.88	1.80	0.00	0.25	0.00	93.87	0.00
Surprised	0.23	0.00	0.00	0.00	0.00	0.00	99.50

Table C.15. Confusion matrix of the LFD features extracted from the Cohn-Kanade database, reduced to 140 dimensions by PCA.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	96.22	1.07	0.15	1.02	1.08	2.69	0.04
Angry	0.73	97.74	0.69	0.15	0.22	0.61	0.00
Disgusted	0.97	0.78	99.01	0.00	0.00	0.00	0.00
Happy	0.51	0.17	0.15	98.08	0.60	0.04	0.00
Neutral	0.38	0.00	0.00	0.46	98.09	0.04	0.00
Sad	1.10	0.24	0.00	0.30	0.00	96.58	0.00
Surprised	0.09	0.00	0.00	0.00	0.00	0.04	99.96

Table C.16. Confusion matrix of the LFD features extracted from the Cohn-Kanade database, reduced to 140 dimensions by SC.

C.4 Analysis

This section analysis on the results presented in the previous section. The analysis has the following three main focuses:

1. Which of LBP, LPQ or LFD performs better for recognition when their dimensionality is reduced?
2. Which of PCA or SC performs better with subsequent classification using SVMs?
3. Which of PCA or SC is most robust against different partitioning of the data?

Before the above questions are answered, another general observation of the recognition performance over different dimensions are made. On the plots shown on Figure C.3 to C.8 it can be seen that the recognition accuracy sometimes drops when the number of dimensions is increased. These fluctuations most likely arises due to the nature of the grid search parameter estimation of the SVMs. The grid is finite in size. Therefore, the true optimal global maximum might lie in between two trial points. It could also be placed outside of the search range. The effect of dropping recognition accuracy as the dimensionality increases is most pronounced at the LBP and LFD feature descriptors from the KDEF database using SC to reduce the dimensionality.

C.4.1 Focus point #1

Lets start by addressing the first focus point: which feature has the best performance under dimensionality reduction. Based on the six recognition rate plots presented in the previous section, it can be seen that LPQ outperforms both LBP and LFD using both dimensionality reduction methods. The performance of LFD comes extremely close to that of LPQ when extracted from the larger Cohn-Kanade database. However, the convergence towards the performance of LPQ is seen only from 70 dimensions and up when using SC and from 91 dimensions and up when using PCA.

C.4.2 Focus point #2

The second point of focus of this analysis is addressed by considering both the recognition rate plots and the confusion matrices. From the recognition plots of the KDEF database it can be seen, that the SC algorithm outperforms PCA for some of the lower dimensions. However, PCA seems to deliver a higher recognition accuracy and a smaller variance.

For the Cohn-Kanade database, SC seems to outperform PCA. The Cohn-Kanade database is roughly 1.6 times larger than the KDEF database. Further, the facial expressions on the Cohn-Kanade images are more pronounced than the expressions on the KDEF images. This might be the reasons why SC is better. The improvement is most evident for LBP. For the higher dimensions of LPQ, PCA and SC seems to be equally good. SC is only slightly better than PCA for the higher dimensions of LFD.

The results are ratio scaled. Therefore, it can be tested if the difference between the mean accuracy of PCA and SC is statistical significant. It is decided to only test at 140 dimensions because 140 dimensions generally yielded the best recognition rate. In order to do the test, it is assumed that the accuracy populations of the PCA and SC methods are both normally distributed with the same variance. Further, it is assumed that the drawn samples are independent of each other. It is hypothesized, that the means of the two populations are identical. Thus, the two-sample t-statistics can be calculated by:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sigma_{\bar{x}_1 - \bar{x}_2}} \tag{C.1}$$

Where:

$\bar{x}_1$ is the estimated mean of the first population.

$\bar{x}_2$ is the estimated mean of the second population.

$$\sigma_{\bar{x}_1 - \bar{x}_2} = \sqrt{\frac{\sigma_1^2}{N_1} + \frac{\sigma_2^2}{N_2}} = \sqrt{\frac{2\sigma^2}{N}}$$

$$\sigma = \frac{s_1^2 + s_2^2}{2}$$

s_1 is the estimated variance of the first population.

s_2 is the estimated variance of the second population.

The degrees of freedom can be calculated by:

$$dof = (N_1 - 1) + (N_2 - 1) \tag{C.2}$$

The degrees of freedom can be used to make a look-up in a precomputed table. The look-up will tell the probability of getting a t which is larger than or equal to or smaller than or equal to the t value computed by Equation C.1. In this test, a 98% confidence level is used. Thus, if the probability of the two means being equal is higher than 98%, the hypothesis that the two means are equal is accepted. This is the same as stating, that the means can be regarded as statistically different if the probability is below 2%. The following table states the probabilities of equal means for the PCA and SC accuracy experiment at 140 dimensions[3]:

Experiment no.	$\bar{\mathbf{x}}$		**s**		**D.O.F.**	**t**	**p(two-tailed)**
1.1	85.93	82.90	1.01	0.99	98	15.153	0.00
1.2	95.61	96.74	0.33	0.35	98	−9.769	0.00
2.1	89.13	87.93	0.17	0.85	98	8.426	0.00
2.2	97.06	97.02	0.12	0.16	98	0.436	0.664
3.1	85.85	84.39	0.16	0.86	98	10.202	0.00
3.2	96.67	97.64	0.14	0.10	98	−14.173	0.00

Table C.17. Results of the t-test for significant variance conducted on the accuracy of 140 dimensions for all experiments defined in Table C.1. Except for the LPQ experiment from the Cohn-Kanade database, the probability of the two means being identical is equal to 0. Thus, the means of all experiment except 2.2 can be regarded as significantly different.

From Table C.17, it can be seen that the accuracy of PCA and SC for 140 dimensions is different for all experiments except experiment 2.2.

C.4.3 Focus point #3

By summing the variance of each experiment together for PCA and SC, a measure of the total amount of variance for each method is obtained. The results are presented in the following table:

Database	**Dim. red.**	**Accumulated variance**	**Percentage of total**
KDEF	PCA	40.63	47.46%
KDEF	SC	44.99	52.54%
Cohn-Kanade	PCA	14.01	48.98%
Cohn-Kanade	SC	14.59	51.02%

Table C.18. Comparison of the variances of the PCA and SC methods, for both the KDEF and Cohn-Kanade databases. The leftmost column specifies how much of the total variance of the two reduction methods combined is explained by each method alone.

[3]Note that the calculator found at `http://onlinestatbook.com/2/calculators/t_dist.html` is used as a look-up table to find the accepted t-values for the degrees of freedom.

By judging from Table C.18, it seems that PCA is a little more robust than SC. However, the difference seems to be negligible.

C.5 Discussion and Conclusion

By judging from the results obtained from the KDEF database, it seems that PCA performs better than SC. However, by judging from the results from the Cohn-Kanade database, it seems that SC performs better than PCA. This contradiction might be because the feature descriptors drawn from the Cohn-Kanade database forms tighter clusters. Tighter clusters could be formed, because the facial expressions in the Cohn-Kanade database are generally more explicit than in the KDEF database. The Cohn-Kanade database is also significantly larger than the KDEF database. That fact might also help to form tighter clusters. The SC approach requires an update of the affinity matrix for every new, unknown sample. Nnew eigenvectors needs to be computed from the new affinity matrix at every new unknown sample. The PCA approach only requires a transformation of basis for every new sample. Thus, PCA is a computationally simpler approach. Had the improvement of SC been major, its heavier computational load could be accepted. However, the improvement was only observed at the larger of the two databases, and even for that database, the improvement was subtle. Therefore, PCA seems to be the better choice.

Regarding the number of dimensions, it seems that 140 dimensions are more than adequate for all feature descriptors. The recognition accuracy quickly flattens after 49 dimensions. Therefore, it would be redundant to use more than 140 dimensions and less than 140 dimensions would also do.

From the experiment, it seems that LPQ is the better choice of LBP, LPQ and LFD. Even though the difference is small between LPQ and LFD for the Cohn-Kanade database, LPQ does in fact outperform LFD at all dimensions for both databases, except for the higher dimensions of the Cohn-Kanade database.

The conclusions above are made based on the recognition accuracy plots. The confusion matrices shows that *Happy* is the easiest expression to detect in the KDEF database and that *Surprise* is the easiest expression to detect in the Cohn-Kanade database. However, they do not convey much information about which method is better.

Some sources of error should be noted. First of all, there is a question of implementation. The PCA algorithm which is build into Matlab was used. It is a fairly standard algorithm which has undergone multiple revisions by different users. It is therefore expected, that the implementation is fairly bug-free. The SC implementation was done by the author based on an implementation done by Doc. Zhanyu Ma. As a consequence, a bug could have been present in the SC implementation which might have lowered the recognition rate.

A source of error when judging the performance of the LFD descriptor against that of the LPQ descriptor is the statistical uniform patterns. Lei et al. [2011] report a better performance using LFD on face recognition than when using LPQ. They used only 32 statistical uniform patterns. In this experiment, 100 statistical uniform patterns were

used. It is expected, that more patterns equals more information, thus yielding a better recognition rate. However, even though 100 patterns were used, LFD still performed worse than LPQ for FER. It could be that the descriptive ability of the LFD descriptor is enhanced by using a lower number of uniform patterns. Surely, more experiments are needed on this matter, but so far it seems that LPQ is better than LFD.

Measurement Record: Blur experiment

D

This measurement record documents the blur experiment. This experiment seeks to determine which of LBP, LPQ or LFD is better when the test images are blurred. PCA is used for dimensionality reduction. Based on the dimensionality reduction experiment documented in Appendix C, it seems that SC and PCA provides more or less similar recognition accuracies. However, SC takes longer time than PCA because SC requires to update the affinity matrix for each new observed sample. Thus, PCA is better for a practical implementation. The recognition rate is calculated by SVMs.

This experiment is composed of three sub-experiments: one for each of LBP, LPQ and LFD.

The implementation of the experiment can be found at /Experiments/blur_LFD_LPQ_LBP/.

The measurement record is composed of the following sections: Methods, Experimental Setup, Results, Analysis and Discussion and Conclusion.

D.1 Methods

In brief, the three feature descriptors are extracted from the sharp images of both the KDEF and Cohn-Kanade databases. Then, the feature descriptors are extracted from several differently blurred versions of the images. The databases are partitioned into test and training data for five-fold cross-validation with a number of repetitions. For each repetition, for each cross-validation, the dimensionality of the training samples are reduced by PCA. Then, the test samples from the blurred images are mapped into the reduced Principal Component space. The test samples are drawn from the cross-validation group not used for training. A set of OAO SVMs are trained using the reduced training data. The recognition rate of each of the blurred images are then calculated by classifying each test sample by the SVMs.

The images are blurred using a Gaussian image blurring kernel. As described in Chapter 5, the Gaussian kernel is a good approximation to some of the natural occurring blurring distortions. The kernel is defined as:

$$G(r, c) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{r^2 + c^2}{2\sigma^2}\right) \tag{D.1}$$

The size of the kernel is usually selected as some rounded of multiple of the variance. In this context, a kernel size of round($6\sigma^2$) is selected. It is expected, that the filter has more or less reached 0 at $3\sigma^2$ away from the kernel origin.

There is two databases and three features. This corresponds to a total of six sub-experiments. The sub-experiments are specified in Table D.1

No.	Database	Feature	Dimensionality reduction	Classifier
1.1	KDEF	LBP	PCA	SVM
1.2	Cohn-Kanade	LBP	PCA	SVM
2.1	KDEF	LPQ	PCA	SVM
2.2	Cohn-Kanade	LPQ	PCA	SVM
3.1	KDEF	LFD	PCA	SVM
3.2	Cohn-Kanade	LFD	PCA	SVM

Table D.1. Specifications of the database, feature extractor, dimensionality reduction and classifier involved in each sub-experiments of the blur experiment.

The images are segmented as explained in Chapter 4. The features are extracted as explained in Chapter 5. The dimensionality reduction is done by PCA as explained in Chapter 6. The classification is done as explained in Chapter 7. The databases are divided into training and test data prior to executing the experiment. This ensures that the methods are tested on the exact same data.

The recognition accuracy is calculated as the mean recognition rate obtained by 10 repetitions of five-fold cross-validation with 10 different partitions of the databases. By doing so, the risk of a bad partitioning of the relatively small databases is lowered. The variances of the repetitions are calculated as well.

Every image is divided into a grid and the feature descriptors are extracted from each cell in the grids. The resulting descriptors are formed by concatenating the descriptors from each cell together.

In Appendix E, a possibly optimal number of statistical uniform patterns was reported to be 18 for LPQ and 40 for LFD. However, this information were unknown at the time of execution of this experiment. Therefore, the full LPQ descriptor is used and 100 statistical uniform patterns are extracted for LFD, 50 for the magnitude and 50 for the phase.

The grid search approach is used to get optimal estimates of C and γ for the SVMs.

D.2 Experimental setup

The experiments are all implemented in Matlab. The used implementations are specified in Table D.2.

Method	Implementer	Reference
LBP	Other	[Heikkilä and Ahone, 2013]
LDPv	Self	-
LPQ	Other	[Rahtu et al., 2012]
LFD	Self	-
PCA	Other	Std. Matlab function
SVM	Other	LIBSVM [Chang and Lin, 2011]

Table D.2. Specification of the Matlab implementations of the used methods. The implementer state if the method is self-implemented or if it was implemented by others. In that case, a reference to the implementer is provided.

For reasons explained in Appendix C, it is chosen to test the recognition rates of dimensions from 7 to 140 in steps of 7 dimensions. The tried variances of the Gaussian blurring kernel is chosen to run from 0 to 4 in steps of 0.25. The values of the parameters are defined in Table D.3

Parameter	Designator	Value
No. of image grid rows	k	7
No. of image grid columns	l	6
Cross-validation	cv	5-fold
Repetitions	rep	10
Dimensions	dims	7 to 140 in steps of 7
Blur variances	sigma	0 to 4 in steps of 0.25
u.p. for LBP	LPQ_up	58
s.u.p. for LPQ	LPQ_sup	*256
s.u.o. for LFD	LFD_sup	100
The window size for LPQ and LFD	M	7
The frequency for LPQ and LFD	a	1/7
The decorrelation parameter for LPQ	ρ	0.9

Table D.3. Specification of the settings of the parameters used in the experiment. The abbreviations are as follows: s.u.p. is statistical uniform patterns and u.p. is uniform patterns. *) no statistical uniform patterns were extracted.

The feature descriptor length will be $k \cdot l \cdot d$ where d is the length of one feature descriptor from one of the image cells. The resulting descriptor lengths are as follows: LBP) $42 \cdot 58 = 2436$, LPQ) $42 \cdot 256 = 10\,752$, and LFD) $42 \cdot 100 = 4200$.

The KDEF database has 980 samples. A subset of 1596 samples are drawn from the Cohn-Kanade database[1].

The SVM parameter estimation grid search is done over the following range:

[1]The indices to the samples drawn from the Cohn-Kanade database can be found in /databases/Cohn-Kanade/indices.mat.

Parameter	Start	End	Step size
C	$\log_2(1)$	$\log_2(24)$	$\log_2(1)$
γ	$\log_2(-24)$	$\log_2(1)$	$\log_2(1)$

Table D.4. The grid search parameters used to find an optimal set of values for C and γ.

D.3 Results

This section reports the results of the experiment. A 3D surface plot is shown for each of the sub-experiments specified in Table D.1. The plots are presented in the same order as the sub-experiments are presented in the table[2].

After the accuracy plots, the confusion matrices of each of the sub experiments for the highest tested dimension and highest tested blur variance are presented. The confusion matrices are created by calculating the mean the confusion matrix obtained at each repetition of the cross-validations.

D.3.1 Plots of the recognition accuracies from KDEF

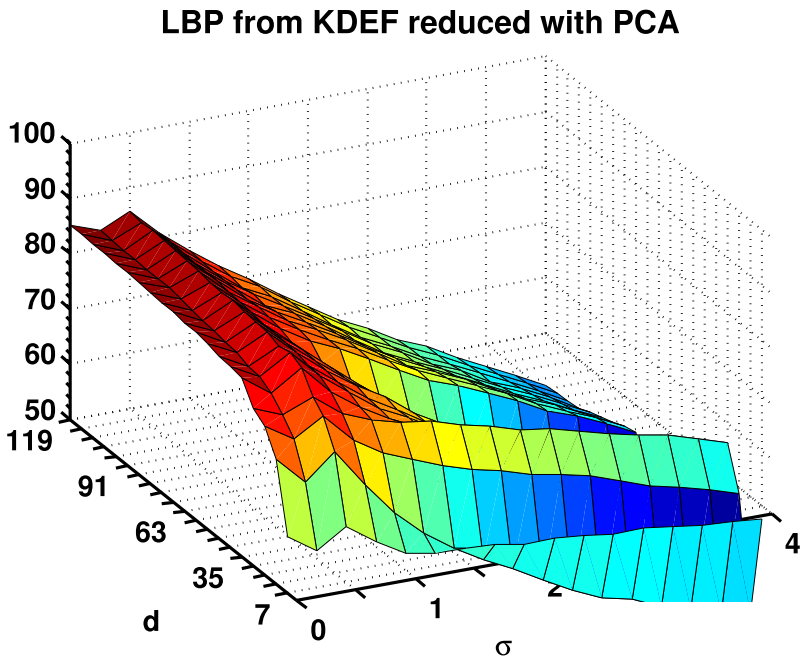

Figure D.1. Result of experiment 1.1: Plot showing the LBP recognition rates for different numbers of reduced dimensions at different variances of the image blurring. The results are obtained using the KDEF database. The z-axis describes the recognition accuracy.

[2]The complete data can be found in /Results/blur/

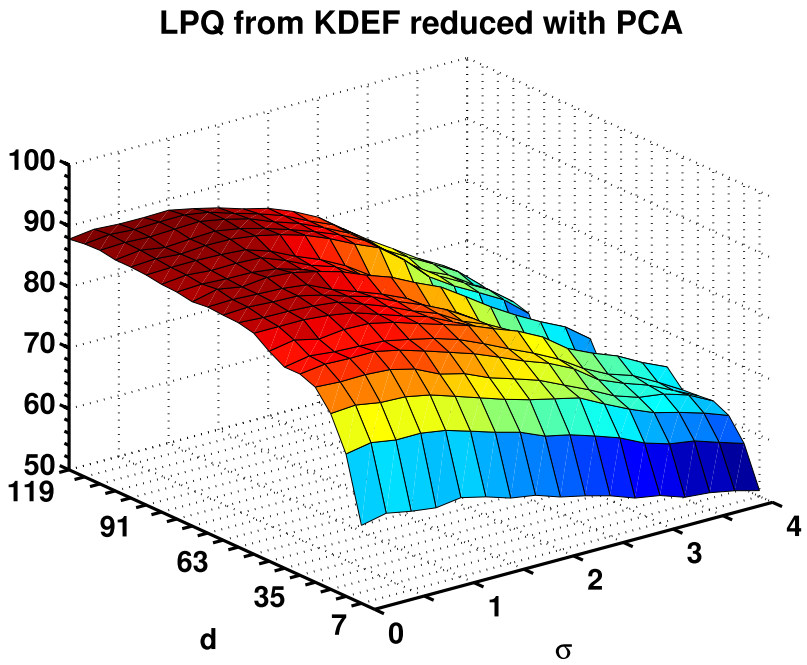

Figure D.2. Result of experiment 1.2: Plot showing the LPQ recognition rates for different numbers of reduced dimensions at different variances of the image blurring. The results are obtained using the KDEF database. The z-axis describes the recognition accuracy.

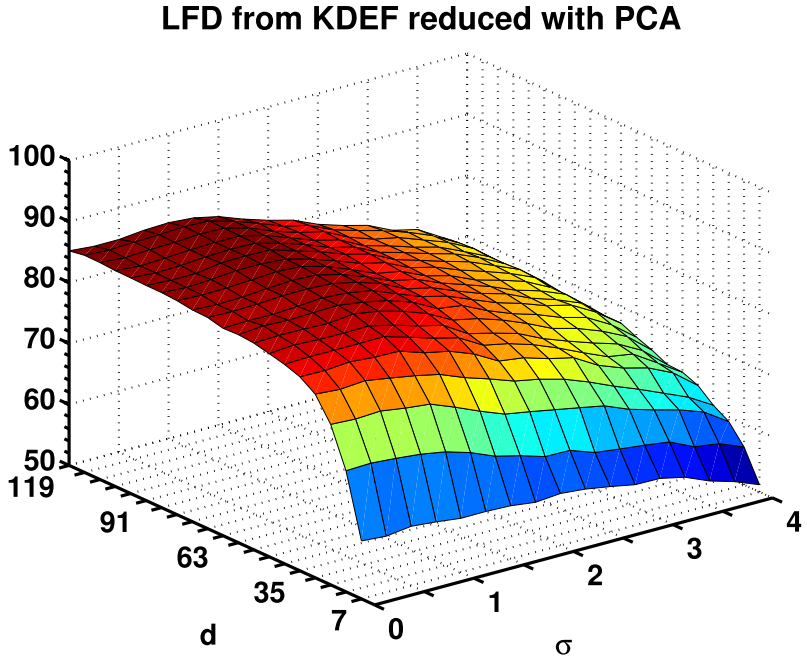

Figure D.3. Result of experiment 1.3: Plot showing the LFD recognition rates for different numbers of reduced dimensions at different variances of the image blurring. The results are obtained using the KDEF database. The z-axis describes the recognition accuracy.

D.3.2 Plots of the recognition accuracies from Cohn-Kanade

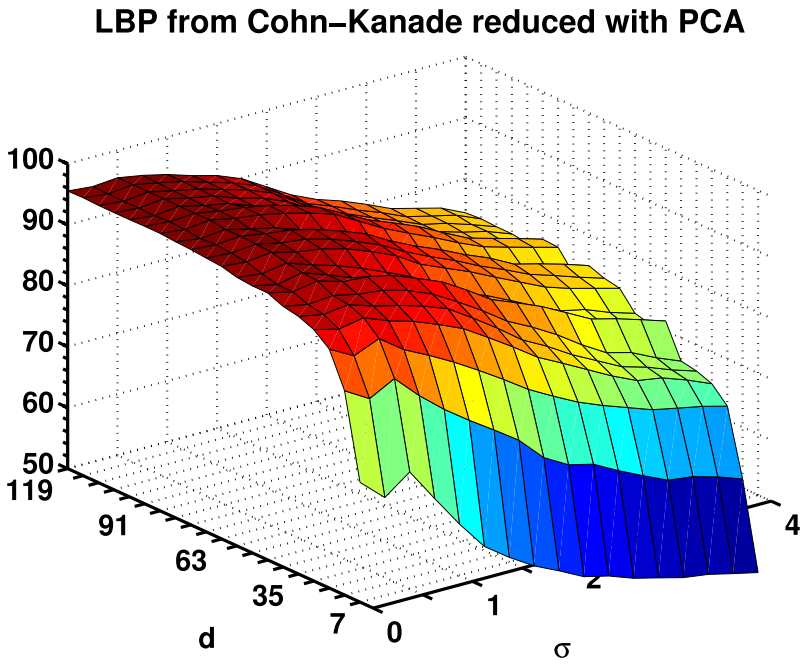

Figure D.4. Result of experiment 2.1: Plot showing the LBP recognition rates for different numbers of reduced dimensions at different variances of the image blurring. The results are obtained using the Cohn-Kanade database. The z-axis describes the recognition accuracy.

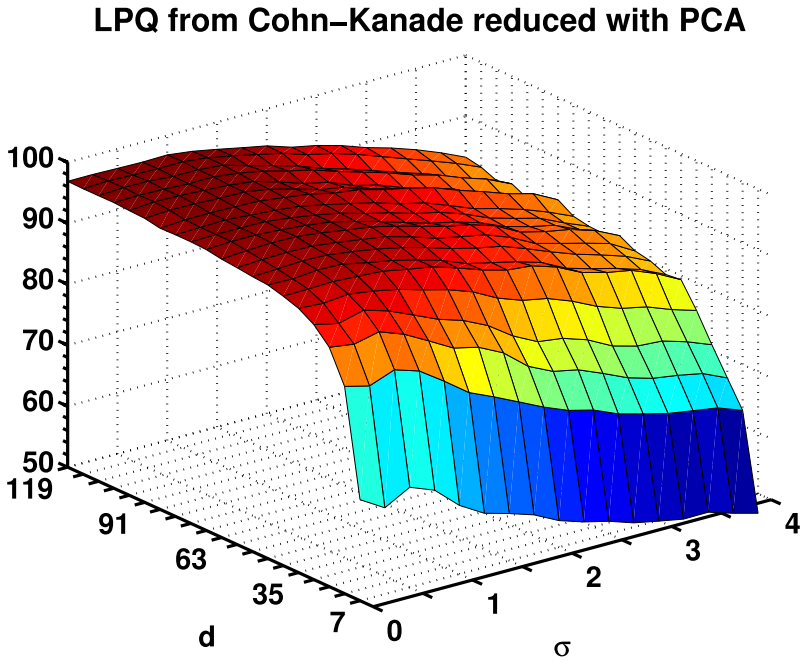

Figure D.5. Result of experiment 2.2: Plot showing the LPQ recognition rates for different numbers of reduced dimensions at different variances of the image blurring. The results are obtained using the Cohn-Kanade database. The z-axis describes the recognition accuracy.

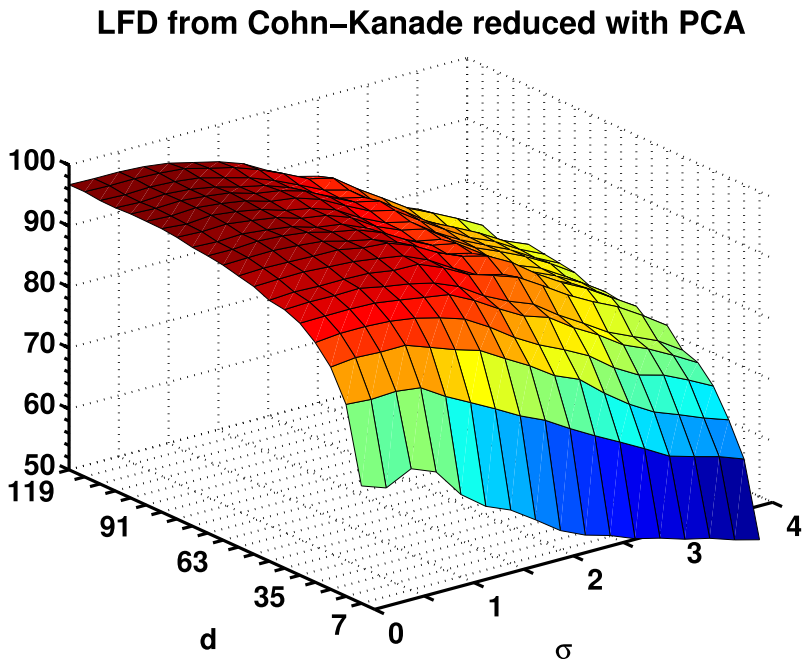

Figure D.6. Result of experiment 2.3: Plot showing the LFD recognition rates for different numbers of reduced dimensions at different variances of the image blurring. The results are obtained using the Cohn-Kanade database. The z-axis describes the recognition accuracy.

D.3.3 The confusion matrices from KDEF

The following three tables presents the confusion matrices for 140 dimensions and $\sigma^2 = 2$ for the image blurring. They are presented in the following order: KDEF+LBP+PCA, KDEF+LPQ+PCA and KDEF+LFD+PCA.

	Afraid	**Angry**	**Disgusted**	**Happy**	**Neutral**	**Sad**	**Surprised**
Afraid	57.38	11.04	7.85	0.75	6.80	19.26	6.87
Angry	0.00	39.73	7.72	0.07	1.84	2.75	0.00
Disgusted	0.00	15.49	76.77	1.16	0.00	4.12	0.00
Happy	1.00	4.21	0.66	97.87	0.26	2.22	0.00
Neutral	7.19	13.30	0.87	0.15	85.02	12.58	1.38
Sad	3.17	9.50	5.76	0.00	0.53	53.85	0.21
Surprised	31.26	6.73	0.37	0.00	5.55	5.23	91.54

Table D.5. Confusion matrix of the LBP features extracted from the KDEF database reduced to 150 dimensions by PCA. The input images were blurred with a Gaussian kernel where $\sigma^2 = 2$. The values are presented in percentages.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	69.64	3.17	1.20	3.85	2.59	8.91	8.55
Angry	4.11	87.38	8.28	0.85	3.87	3.62	0.00
Disgusted	1.54	4.57	88.94	1.29	0.00	5.10	0.00
Happy	0.61	0.00	0.00	93.22	0.06	0.13	0.00
Neutral	3.10	1.39	0.20	0.07	87.60	6.42	0.50
Sad	9.72	3.49	1.38	0.71	4.68	75.25	0.00
Surprised	11.29	0.00	0.00	0.00	1.19	0.57	90.95

Table D.6. Confusion matrix of the LPQ features extracted from the KDEF database reduced to 150 dimensions by PCA. The input images were blurred with a Gaussian kernel where $\sigma^2 = 2$. The values are presented in percentages.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	66.32	8.45	3.52	3.13	4.63	10.08	8.26
Angry	3.11	77.79	7.46	0.61	2.37	3.38	0.37
Disgusted	1.49	6.37	87.54	1.75	0.38	5.42	0.00
Happy	1.85	0.05	0.00	93.13	0.82	0.00	0.00
Neutral	3.20	2.28	0.28	0.00	87.64	9.38	1.67
Sad	10.66	4.35	1.21	1.39	3.21	71.24	0.08
Surprised	13.38	0.70	0.00	0.00	0.95	0.50	89.62

Table D.7. Confusion matrix of the LFD features extracted from the KDEF database reduced to 150 dimensions by PCA. The input images were blurred with a Gaussian kernel where $\sigma^2 = 2$. The values are presented in percentages.

D.3.4 The confusion matrices from Cohn-Kanade

The following three tables presents the confusion matrices for 140 dimensions and $\sigma^2 = 2$ for the image blurring. They are presented in the following order: Cohn-Kanade+LBP+PCA, Cohn-Kanade+LPQ+PCA and Cohn-Kanade+LFD+PCA.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	87.37	7.52	4.68	1.85	2.78	17.98	3.49
Angry	3.75	82.53	2.30	0.32	0.29	1.26	0.00
Disgusted	2.18	6.49	88.54	0.31	0.03	1.78	0.00
Happy	0.49	0.00	0.46	94.11	8.14	1.35	0.00
Neutral	0.24	0.00	0.00	2.15	88.59	0.00	0.00
Sad	5.08	3.37	4.02	0.90	0.00	77.15	0.93
Surprised	0.89	0.08	0.00	0.37	0.17	0.48	95.59

Table D.8. Confusion matrix of the LBP features extracted from the CK database reduced to 150 dimensions by PCA. The input images were blurred with a Gaussian kernel where $\sigma^2 = 2$. The values are presented in percentages.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	94.57	4.06	0.76	2.30	2.02	12.67	0.34
Angry	1.76	93.75	1.79	0.12	0.67	1.40	0.00
Disgusted	0.85	0.38	95.38	0.34	0.25	0.00	0.00
Happy	0.74	0.00	2.06	96.91	2.88	0.00	0.00
Neutral	0.31	0.00	0.00	0.00	94.19	0.00	0.00
Sad	1.45	1.82	0.00	0.33	0.00	85.92	0.42
Surprised	0.32	0.00	0.00	0.00	0.00	0.00	99.24

Table D.9. Confusion matrix of the LPQ features extracted from the CK database reduced to 150 dimensions by PCA. The input images were blurred with a Gaussian kernel where $\sigma^2 = 2$. The values are presented in percentages.

	Afraid	Angry	Disgusted	Happy	Neutral	Sad	Surprised
Afraid	90.32	12.04	0.16	1.44	2.76	17.95	2.32
Angry	1.82	80.87	0.07	1.02	0.34	3.82	0.00
Disgusted	1.33	0.00	97.66	0.00	0.09	0.33	0.00
Happy	2.06	2.46	2.11	96.49	2.97	0.10	0.04
Neutral	0.37	0.13	0.00	0.42	93.84	0.00	0.04
Sad	3.64	4.50	0.00	0.62	0.00	77.79	0.00
Surprised	0.45	0.00	0.00	0.00	0.00	0.00	97.60

Table D.10. Confusion matrix of the LFD features extracted from the CK database reduced to 150 dimensions by PCA. The input images were blurred with a Gaussian kernel where $\sigma^2 = 2$. The values are presented in percentages.

D.4 Analysis

This section analyses on the results presented in the previous section.

First, some general observations are made from the plots. Then, an analysis of the difference between the recognition rates of LPQ and LFD is performed.

The LBP plot for the KDEF database shown on Figure D.1 indicates that LBP has a hard time when the test images are blurred. The recognition rate quickly drops off as the variance of the blurring kernel increases. It is assessed, that the irregular fluctuations observed in the recognition rate is due to the sparsity of grid search approach used for estimating the SVM parameters. When considering the confusion matrix in Table D.5, it is clear that especially the subtle expressions such as afraid and angry disappears from LBP when the images are blurred. LBP handles the blurring of the Cohn-Kanade images a lot better. The recognition rates for high blur variance shown on Figure D.4 are substantially better than those observed for the KDEF database. By judging from the confusion matrix in Table D.8, LBP is much better at describing the subtle expressions such as afraid and angry in Cohn-Kanade than in KDEF. This might be due to the explicitness of the facial expressions in the Cohn-Kanade database. An over-acted facial expression will be easier

to recognize on a blurred image, because the face markers affiliated with the expression are large.

For the KDEF database, the LPQ recognition accuracy is substantially better than that of LBP. This is seen in Figure D.2. The surface is much smoother and only a small drop-off is observed as the blur variance increases. As seen in Figure D.5, LPQ is also better than LBP for the Cohn-Kanade database, although the improvement is not quite as stunning as for KDEF. Both the overall recognition rate and the blur robustness is enhanced by LPQ relative to LBP. Again, it is assessed that the fluctuations observed in both the KDEF and Cohn-Kanade plots originates from the parameter estimation of the SVMs.

In Figure D.3, it is seen that the LFD recognition accuracy surface is quite smoother than the one for LPQ from the KDEF database. However, this phenomenon is not observed in Figure D.6, where LFD is extracted from the Cohn-Kanade database. In both plots it is seen, that LFD outperforms LBP. When it comes to the comparison with LPQ, it is seen that LPQ is better than LFD for most combinations of dimensionality and blur variance. However, when comparing Figure D.6 against Figure D.5 it is seen, that LFD actually outperforms LPQ as the blur variance grows large. After a variance of 2.75, the accuracy of LFD is higher than that of LPQ for all dimensions. This tendency is however reversed when considering the Cohn-Kanade database. There, LFD drops off faster than LPQ.

Judging from the confusion matrices for LPQ and LFD from KDEF, it seems that LPQ is better at describing all the facial expressions. However, judging from the confusion matrices of the Cohn-Kanade database, it seems that LPQ is better at describing afraid, angry happy, neutral, sad and surprised. LFD on the other hand seems better at describing disgusted.

To better establish which of LPQ or LFD is better, every point on the accuracy surfaces for LPQ is compared to every point on the accuracy surfaces of LFD. By judging which method performs best at all point, a conclusion on the overall winner can be made. The calculation is done as follows:

$$p = \frac{1}{SD} \sum_{s=1}^{S} \sum_{d=1}^{D} \mathbf{Q}_{sd} - \mathbf{F}_{sd} \tag{D.2}$$

Where:

$\mathbf{Q} \in \mathbb{R}^{S \times D}$ is the accuracies for the LPQ descriptor.
$\mathbf{F} \in \mathbb{R}^{S \times D}$ is the accuracies for the LFD descriptor.
S is the maximum number of blur variances.
D is the maximum number of dimensions.
p is a scalar telling how many percentage points LPQ is better or worse than LFD.
(D.3)

By performing the above calculation on the results from the KDEF database, the mean difference between LPQ and LFD is found to be:

$$p = -0.35\%\text{-points} \tag{D.4}$$

Therefore, LFD is on average 0.35 percentage points better than LPQ at the KDEF database. The same calculation on the results from the Cohn-Kanade database yields:

$$p = 3.36\%\text{-points} \tag{D.5}$$

Therefore, LPQ is on average 3.36 percentage points better than LFD for the Cohn-Kanade database.

D.5 Discussion and Conclusion

By judging from the results of the experiment documented by this measurement record, it seems that LPQ is the better choice of LBP, LPQ or LFD for FER in blurred images. However, LFD provided better results than LPQ at the KDEF database for high blur variances. The facial expressions contained in the Cohn-Kanade database are in general more explicit than the ones in the KDEF database. The results could indicate, that LPQ is being punished harder than LFD when the facial expressions (and thus the facial features) are subtle. If this is true, it can be stated that LPQ is in general better than LFD, unless the facial features are subtle and the images are severely blurred.

The conclusion that LPQ outperforms LFD is backed by the results obtained in Appendix C and B. It should be noted, that the LFD descriptors were reduced to 100 statistical uniform patterns while the complete LPQ descriptor were used. This might have given LPQ an advantage over LFD.

Measurement Record: Statistical uniform patterns

E

This measurement report documents the experiment which seeks to establish a decent number of statistical uniform patterns for LPQ and LFD.

The implementation of the experiment can be found at /`experiments/sup_LFD_LPQ/`.

The record is composed of the following sections: Methods, Experimental Setup, Results, Analysis and Discussion and Conclusion.

E.1 Methods

This record is based on the explanation of statistical uniform patterns presented in Section 5.4. The flow of the experiment is as follows: load the segmented images from both the KDEF and Cohn-Kanade databases → extract the LFD and LPQ binary patterns from all images → count the total number of occurrences of each pattern over all images → use an iterative procedure inspired by Huffman coding to sort the patterns based on their number of occurrences → output the sorted list of patterns and a list of how many patterns where combined in each step.

The algorithm used in this experiment is presented below. It is run three times, one for LPQ, one for the magnitude part of LFD and one for the phase part of LFD:

Algorithm 5 Algorithm which uncovers the statistical uniform patterns. D is a data array containing all images from both the KDEF and the Cohn-Kanade databases. The f parameter indicates which feature to extract. $\mathbf{P}$ is an array holding a list of which patterns were combined in each step. The $\mathbf{u}$ vector hold a list of how many patterns were combined in each step.

Require: $\mathbf{D} \in \mathbb{R}^{N \times M \times K}$, f
Ensure: $\mathbf{P}, \mathbf{u}$
$\mathbf{P} \in \mathbb{R}^{256 \times 2}$
$\mathbf{u} \in \mathbb{R}^{256}$
$\mathbf{F} \in \mathbb{R}^{K \times 256}$
$\mathbf{F} \leftarrow$ extract feature descriptor f from all images in $\mathbf{D}$
$\mathbf{H} \in \mathbb{R}^{256 \times 2}$
$\mathbf{H}(1 \rightarrow 256, 1) \leftarrow \sum_{n=1}^{K} \mathbf{F}_n$ where $\mathbf{F}_n$ is the n'th row of $\mathbf{F}$
$\mathbf{H}(1 \rightarrow 256, 2) \leftarrow 1 \rightarrow 256$
for $t \leftarrow 1$ to 255 **do**
    $\mathbf{H}_s \leftarrow \mathbf{H}$ sorted by the first column
    $\mathbf{P}(t, 1) \leftarrow \mathbf{H}_s(end, 2)$
    $\mathbf{P}(t, 2) \leftarrow \mathbf{H}_s(end - 1, 2)$
    $\mathbf{u}(t) \leftarrow \mathbf{H}_s(end, 1)$
    $\mathbf{H}_s(end - 1) \leftarrow \mathbf{H}_s(end - 1, 1) + \mathbf{H}_s(end, 1)$
    $\mathbf{H} \leftarrow \mathbf{H}_s(1 \rightarrow end - 1)$
end for

E.2 Experimental setup

The experiment is implemented in Matlab. The used implementations are specified in Table E.1.

Method	Implementer	Reference
LPQ	Other	[Rahtu et al., 2012]
LFD	Self	-

Table E.1. Specification of the Matlab implementations of the used methods. The implementer-field state if the method is self-implemented or if it was implemented by others. In that case, a reference to the implementer is provided.

The values of the parameters are specified in Table E.2

Parameter	Designator	Value
The window size	M	7
The frequency	a	1/7
The decorrelation parameter	ρ	0.9

Table E.2. Specification of the settings of the parameters used in the experiment.

The KDEF database has 980 samples. A subset of 1596 samples are drawn from the Cohn-Kanade database[1].

E.3 Results

The amount of combined patterns in each iteration:

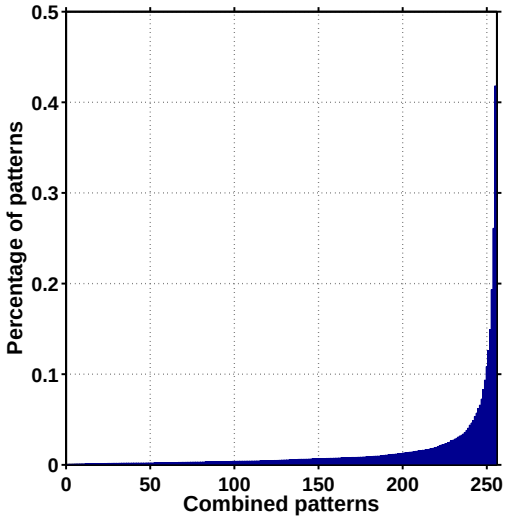

Figure E.1. The amount of combined patterns in each step of the iterative coding algorithm for LPQ. The y-axis is the percentage of the total amount of patterns.

[1]The indices to the samples drawn from the Cohn-Kanade database can be found in /databases/Cohn-Kanade/indices.mat.

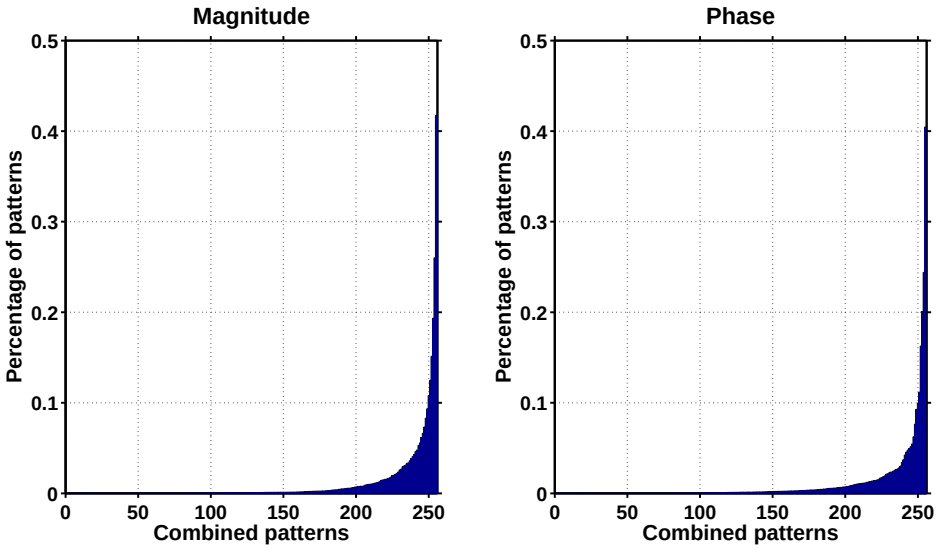

Figure E.2. The amount of combined patterns in each step of the iterative coding algorithm for LFD. The y-axis is the percentage of the total amount of patterns.

E.4 Analysis

To select a decent number of statistical uniform patterns to use, an elbow point is sought in the plots showing the amount of stepwise combined patterns.

In Figure E.1, such an elbow point seems to be located at 238 combined patterns. Thus, the $256 - 238 = 18$ most frequent patterns are used[2].

In Figure E.2, two elbow points are sought. One for the magnitude and one for the phase. The magnitude patterns seems to have an elbow point at 234 patterns. Thus, the $256 - 234 = 22$ most frequent magnitude patterns are used. The phase patterns seems to have an elbow point at 238. Thus, the $256 - 238 = 18$ most frequent patterns are used[3].

E.5 Discussion and Conclusion

The experiment indicates, that 18 statistical uniform patterns should be enough for LPQ and that 40 patterns should be enough for LFD. However, an experiment holding the actual recognition rate against the number of patterns should be performed before any conclusion could be drawn. Just because some patterns are more frequent than others does not necessarily mean that they convey more information. Indeed, it could be so, that the most rare patters are actually the ones holding most of the descriptive information about the facial expressions.

[2]The indexes of these patterns can be found at /SUP/LPQstatisticalPatterns.mat.

[3]The indexes of both the magnitude and phase patterns can be found at /SUP/LFDstatisticalPatterns.mat.

This experiment has specified a basis for selecting a decent number of statistical uniform patterns for LPQ and LFD. However, it is concluded, that a follow-up experiment is needed to uncover the optimal number of patterns.

Matlab implementation of LDPv

F

This appendix contains the Matlab source code of the implemented feature extractor of the LDPv feature descriptor. The source code is included in this report because it does not provide nearly as high recognition rates as would be expected from previous publications using the LDPv descriptor. By including the source code here, transparency is ensured because subsequent authors can easily review the code. Hopefully, that will allow them uncover if there is an error in the implementation or if there is a general problem with the LDPv descriptor. Note that the function can be found in /Experiments/preliminary_LFD_LPQ_LDPv_LBP/feature_extractor/ldpv.m.

It should be noted, that it was tried to establish contact to the authors behind the paper by Kabir et al. [2010] which proposed LDPv. Unfortunately, it tuned out to be impossible to establish contact.

```
1  function [LDPv, LDP, LDP_img] = ldpv(imgIn, varargin)
2  % LDPV Extracts the LDPv features of a given set of images
3  %    [LDPv] = LDPV(Img) returns the LDPv features of image set Img.
4  %    The images are divided into 7x6 blocks and by using the 3 most
5  %    dominant gradients (K = 3).
6  %    If Img contains multiple images, they most be stacked in the
7  %    third dimensions, such that the size of Img is n by m by l,
8  %    where l is the number of images.
9  %
10 %    [LDPv] = LDPV(imgIn, [nvb, nhb]) divides the images into nvb
11 %    vertical blocks and nhb horizontal blocks.
12 %
13 %    [LDPv] = LDPV(imgIn, [nhb, nvb], k) sets K = k.
14 %
15 %    [LDPv, LDP] = LDPV(imgIn) outputs both the LDPv features and
16 %    the LDP features.
17
18 %% Setup
19 nhb = 6; % Number of horizontal blocks
20 nvb = 7; % Number of vertical blocks
21 k = 3;
22 if length(varargin) == 1
23     blck = varargin{1};
24     nvb = blck(1); % Number of vertical blocks
25     nhb = blck(2); % Number of horizontal blocks
26 elseif length(varargin) == 2
27     blck = varargin{1};
28     nvb = blck(1); % Number of vertical blocks
29     nhb = blck(2); % Number of horizontal blocks
```

```
30      k = varargin{2};
31  end
32
33  % Define the Kirsch masks and combine them into one 3D array
34  M = zeros(3,3,8);
35  M(:,:,1) = [-3,-3,5;-3,0,5;-3,-3,5];
36  M(:,:,2) = [-3,5,5;-3,0,5;-3,-3,-3];
37  M(:,:,3) = [5,5,5;-3,0,-3;-3,-3,-3];
38  M(:,:,4) = [5,5,-3;5,0,-3;-3,-3,-3];
39  M(:,:,5) = [5,-3,-3;5,0,-3;5,-3,-3];
40  M(:,:,6) = [-3,-3,-3;5,0,-3;5,5,-3];
41  M(:,:,7) = [-3,-3,-3;-3,0,-3;5,5,5];
42  M(:,:,8) = [-3,-3,-3;-3,0,5;-3,5,5];
43
44  % Set histogram size
45  histSize = 256;
46
47  % Create matrix with all possible outcomes of LDP for the chosen k
48  v = zeros(8,1);
49  v(1:k) = 1;
50  T = unique(perms(v),'rows');
51
52  % Allocate space for output variables
53  img_nr_ub = size(imgIn,1)-2;     % Img size without border
54  img_nc_ub = size(imgIn,2)-2;     % Img size without border
55  LDP_img = zeros(img_nr_ub, img_nr_ub, size(imgIn,3), 'uint8');
56  LDPv = zeros(size(imgIn,3), histSize*nvb*nhb, 'double');
57
58  % Allocate space for computation variables
59  % Directional responses
60  Mres = zeros(size(LDP_img,1), size(LDP_img,2), 8);
61  MresSort = zeros(size(Mres));
62  % Variance of directional repsonses
63  S = zeros(size(Mres,1), size(Mres,2));
64  % LDP descriptors
65  LDP_temp = zeros(size(LDP_img,3), histSize,nhb*nvb);
66  LDP = zeros(size(LDP_img,3), nhb*nvb*histSize);
67
68  % Allocate space for the current operating image in the stack
69  currentImg = zeros(size(imgIn,1),size(imgIn,2));
70
71  % Loop through all images in the input
72  for imgNr = 1:size(imgIn,3)
73      % Cast the current image to double for calculation purposes
74      currentImg(:,:) = cast(imgIn(:,:,imgNr), 'double');
75
76      % Calculate the Directional Responses for all directions
77      % (for all pixels because of matrix operation)
78      for dir = 1:8
79          Mres(:,:,dir) = rot90(abs(...
80              conv2(rot90(currentImg,2), M(:,:,dir), 'valid')),2);
81      end
82
83      % Calculate the LDP bit response for all pixels
84      % Sort the Mres to find the k highest value
85      MresSort(:,:,:) = sort(Mres,3);
86      for i = 1:8
```

```
87          LDP_img(:,:,imgNr) = LDP_img(:,:,imgNr) + cast(...
88              unitstep(Mres(:,:,i) - ...
89              MresSort(:,:,1+8-k)) * 2^(8-i),'uint8');
90      end
91
92      % Calculate the variances of the Directional Responses
93      S(:,:) = var(Mres,1,3);
94
95      % Calculate the block size
96      verBS = floor(size(S,1)/nvb);
97      horBS = floor(size(S,2)/nhb);
98
99      % Calculate the LDPv descriptor
100     for v = 0:nvb-1
101         for h = 0:nhb-1
102             col = (1+h*horBS):(1+h*horBS)+horBS-1;
103             row = (1+v*verBS):(1+v*verBS)+verBS-1;
104             % Calculate LDP
105             LDP_temp(imgNr, :, v*nhb+h+1) = sum(hist(...
106                 cast(LDP_img(row, col, imgNr),'double'), ...
107                 histSize),2);
108             % Calculate LDPv
109             for tau = 1:size(T,1)
110                 LDPv(imgNr,(v*nhb+h)*histSize+tau) = sum(...
111                     S( LDP_img(row, col,imgNr) == bit2dec( T(tau,:) )) ...
112                     );
113             end
114         end
115     end
116     % Concatenate the histograms from each image cell
117     LDP(imgNr,:) = LDP_temp(imgNr, :);
118 end
119
120
121
122 %% Local functions
123
124 % Function for converting from bit sequence to decimal number
125 function b = bit2dec(a)
126     b = sum(a.*2.^(linspace(7,0,8)));
127 end
128
129 end
```

Bibliography

T. Ahonen, M. Pietikainen, A. Hadid and T. Maenpaa, Aug 2004a. T. Ahonen, M. Pietikainen, A. Hadid and T. Maenpaa. Face recognition based on the appearance of local regions. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 153–156 Vol.3, Aug 2004a. doi: 10.1109/ICPR.2004.1334491.

Ahonen et al., Dec 2006. T. Ahonen, A. Hadid and M. Pietikainen. *Face Description with Local Binary Patterns: Application to Face Recognition.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 28(12), 2037–2041, 2006. ISSN 0162-8828. doi: 10.1109/TPAMI.2006.244.

T. Ahonen, E. Rahtu, V. Ojansivu and J. Heikkila, Dec 2008. T. Ahonen, E. Rahtu, V. Ojansivu and J. Heikkila. Recognition of blurred faces using Local Phase Quantization. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4, Dec 2008. doi: 10.1109/ICPR.2008.4761847.

Timo Ahonen, Abdenour Hadid and Matti Pietikäinen. Face Recognition with Local Binary Patterns. In Tomás Pajdla and Jiří Matas, editor, *Computer Vision - ECCV 2004*, volume 3021 of *Lecture Notes in Computer Science*, pages 469–481. Springer Berlin Heidelberg, 2004b. ISBN 978-3-540-21984-2. doi: 10.1007/978-3-540-24670-1_36. URL `http://dx.doi.org/10.1007/978-3-540-24670-1_36`.

Banham and Katsaggelos, Mar 1997. M.R. Banham and A.K. Katsaggelos. *Digital image restoration.* Signal Processing Magazine, IEEE, 14(2), 24–41, 1997. ISSN 1053-5888. doi: 10.1109/79.581363.

P.N. Belhumeur, D.W. Jacobs, D. Kriegman and N. Kumar, June 2011. P.N. Belhumeur, D.W. Jacobs, D. Kriegman and N. Kumar. Localizing parts of faces using a consensus of exemplars. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 545–552, June 2011. doi: 10.1109/CVPR.2011.5995602.

Bettadapura, March 2012. Vinay Bettadapura. *Face Expression Recognition and Analysis: The State of the Art.* ArXiv e-prints, 2012.

Bishop, 2006. C.M. Bishop. *Pattern Recognition and Machine Learning.* Information Science and Statistics. Springer, 2006. ISBN 9780387310732.

Cătălin-Daniel Caleanu, May 2013. Cătălin-Daniel Caleanu. Face expression recognition: A brief overview of the last decade. In *Applied Computational Intelligence and*

Informatics (SACI), 2013 IEEE 8th International Symposium on, pages 157–161, May 2013. doi: 10.1109/SACI.2013.6608958.

Chang and Lin, **2011**. Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: A library for support vector machines*. ACM Transactions on Intelligent Systems and Technology, 2, 27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Ira Cohen, Nicu Sebe, Ashutosh Garg, Michael S. Lew and Thomas S Huang, *2002*. Ira Cohen, Nicu Sebe, Ashutosh Garg, Michael S. Lew and Thomas S Huang. Facial expression recognition from video sequences. In *Multimedia and Expo, 2002. ICME '02. Proceedings. 2002 IEEE International Conference on*, volume 2, pages 121–124 vol.2, 2002. doi: 10.1109/ICME.2002.1035527.

Cortes and Vapnik, **1995**. Corinna Cortes and Vladimir Vapnik. *Support-Vector Networks*. Machine Learning, 20(3), 273–297, 1995. ISSN 0885-6125. doi: 10.1023/A:1022627411411. URL `http://dx.doi.org/10.1023/A%3A1022627411411`.

Dahmane and Meunier, **2014**. M. Dahmane and J. Meunier. *Prototype #x2013; Based Modeling for Facial Expression Analysis*. Multimedia, IEEE Transactions on, PP(99), 1–1, 2014. ISSN 1520-9210. doi: 10.1109/TMM.2014.2321113.

A. Dhall, A. Asthana, R. Goecke and T. Gedeon, *March 2011*. A. Dhall, A. Asthana, R. Goecke and T. Gedeon. Emotion recognition using PHOG and LPQ features. In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 878–883, March 2011. doi: 10.1109/FG.2011.5771366.

Abhinav Dhall, *Sept 2013*. Abhinav Dhall. Context Based Facial Expression Analysis in the Wild. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, pages 636–641, Sept 2013. doi: 10.1109/ACII.2013.111.

Du et al., **2014**. Shichuan Du, Yong Tao and Aleix M. Martinez. *Compound facial expressions of emotion*. Proceedings of the National Academy of Sciences, 2014. doi: 10.1073/pnas.1322355111. URL `http://www.pnas.org/content/early/2014/03/25/1322355111.abstract`.

Ekman and Rosenberg, **1997**. P. Ekman and E.L. Rosenberg. *What the Face Reveals: Basic and Applied Studies of Spontaneous Expression Using the Facial Action Coding System (FACS)*. Series in affective science. Oxford University Press, 1997. ISBN 9780195104462. URL `http://books.google.dk/books?id=fFGYs079-7YC`.

Ekman and Friesen, **Feb 1971**. Paul Ekman and Wallace V. Friesen. *Constants across cultures in the face and emotion*. Journal of Personality and Social Psychology, 12(2), 124–129, 1971. ISSN 1939-1315. doi: 10.1037/h0030377.

Ekman and Friesen, **1977**. Paul Ekman and Wallace V. Friesen. *Manual for the Facial Action Coding System*. Consulting Psychologists Press, 1977.

T. Fang, X. Zhao, O. Ocegueda, S.K. Shah and I. A. Kakadiaris, March 2011. T. Fang, X. Zhao, O. Ocegueda, S.K. Shah and I. A. Kakadiaris. 3D facial expression recognition: A perspective on promises and challenges. In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 603–610, March 2011. doi: 10.1109/FG.2011.5771466.

Fasel and Luettin, 2003. Beat Fasel and Juergen Luettin. *Automatic facial expression analysis: a survey.* Pattern Recognition, 36(1), 259–275, 2003. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/S0031-3203(02)00052-3. URL `http://www.sciencedirect.com/science/article/pii/S0031320302000523`.

Xiaoyi Feng, Sept 2004. Xiaoyi Feng. Facial expression recognition based on local binary patterns and coarse-to-fine classification. In *Computer and Information Technology, 2004. CIT '04. The Fourth International Conference on*, pages 178–183, Sept 2004. doi: 10.1109/CIT.2004.1357193.

A. Hadid, M. Pietikainen and T. Ahonen, June 2004. A. Hadid, M. Pietikainen and T. Ahonen. A discriminative feature space for detecting and recognizing faces. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–797–II–804 Vol.2, June 2004. doi: 10.1109/CVPR.2004.1315246.

Shan He, Shangfei Wang, Wuwei Lan, Huan Fu and Qiang Ji, Sept 2013. Shan He, Shangfei Wang, Wuwei Lan, Huan Fu and Qiang Ji. Facial Expression Recognition Using Deep Boltzmann Machine from Thermal Infrared Images. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, pages 239–244, Sept 2013. doi: 10.1109/ACII.2013.46.

Heikkilä and Ahone, 2013. Marko Heikkilä and Timo Ahone. *A general Local Binary Pattern (LBP) implementation for Matlab*, 2013. Downloaded: 28-05-2014.

Fu-Song Hsu, Wei-Yang Lin and Tzu-Wei Tsai, Oct 2013. Fu-Song Hsu, Wei-Yang Lin and Tzu-Wei Tsai. Automatic facial expression recognition for affective computing based on bag of distances. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific*, pages 1–4, Oct 2013. doi: 10.1109/APSIPA.2013.6694238.

Izard, Mar 1994. Carroll E. Izard. *Innate and universal facial expressions: Evidence from developmental and cross-cultural research.* Psychological Bulletin, 115(2), 288–299, 1994. ISSN 1939-1455. doi: 10.1037/0033-2909.115.2.288.

T. Jabid, M.H. Kabir and O. Chae, Jan 2010a. T. Jabid, M.H. Kabir and O. Chae. Local Directional Pattern (LDP) for face recognition. In *Consumer Electronics (ICCE), 2010 Digest of Technical Papers International Conference on*, pages 329–330, Jan 2010a. doi: 10.1109/ICCE.2010.5418801.

Jabid et al., Oct 2010b. Taskeed Jabid, Md. Hasanul Kabir and Oksam Chae. *Robust Facial Expression Recognition Based on Local Directional Pattern.* ETRI Journal, 32 (5), 784–794, 2010. doi: 10.4218/etrij.10.1510.0132. Downloaded: 20-05-2014.

Jack et al., Sept 2009. Rachael E. Jack, Caroline Blais, Christoph Scheepers, Philippe G. Schyns and Roberto Caldara. *Cultural Confusions Show that Facial Expressions Are Not Universal.* Current Biology, 19(18), 1543–1548, 2009. doi: 10.1016/j.cub.2009.07.051.

M.H. Kabir, T. Jabid and Oksam Chae, Aug 2010. M.H. Kabir, T. Jabid and Oksam Chae. A Local Directional Pattern Variance (LDPv) Based Face Descriptor for Human Facial Expression Recognition. In *Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on*, pages 526–532, Aug 2010. doi: 10.1109/AVSS.2010.9.

T. Kanade, J.F. Cohn and YingLi Tian, 2000. T. Kanade, J.F. Cohn and YingLi Tian. Comprehensive database for facial expression analysis. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 46–53, 2000. doi: 10.1109/AFGR.2000.840611.

N.U. Khan, Feb 2013. N.U. Khan. A comparative analysis of facial expression recognition techniques. In *Advance Computing Conference (IACC), 2013 IEEE 3rd International*, pages 1262–1268, Feb 2013. doi: 10.1109/IAdCC.2013.6514409.

Rizwan Ahmed Khan, Alexandre Meyer, Hubert Konik and Saida Bouakaz, June 2012. Rizwan Ahmed Khan, Alexandre Meyer, Hubert Konik and Saida Bouakaz. Exploring human visual system: Study to aid the development of automatic facial expression recognition framework. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 49–54, June 2012. doi: 10.1109/CVPRW.2012.6239186.

Kotsia et al., 2008. Irene Kotsia, Ioan Buciu and Ioannis Pitas. *An analysis of facial expression recognition under partial facial image occlusion.* Image and Vision Computing, 26(7), 1052 – 1067, 2008. ISSN 0262-8856. doi: http://dx.doi.org/10.1016/j.imavis.2007.11.004. URL `http://www.sciencedirect.com/science/article/pii/S0262885607002107`.

Zhen Lei, T. Ahonen, M. Pietikainen and S.Z. Li, March 2011. Zhen Lei, T. Ahonen, M. Pietikainen and S.Z. Li. Local frequency descriptor for low-resolution face recognition. In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 161–166, March 2011. doi: 10.1109/FG.2011.5771391.

D.G. Lowe, 1999. D.G. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150–1157 vol.2, 1999. doi: 10.1109/ICCV.1999.790410.

Patrick Lucey, Jeffrey F. Cohn, Takeo Kanade, Jason Saragih and Zara Ambadar, June 2010. Patrick Lucey, Jeffrey F. Cohn, Takeo Kanade, Jason Saragih and Zara Ambadar. The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 94–101, June 2010. doi: 10.1109/CVPRW.2010.5543262.

Lundqvist et al., **1998**. Daniel Lundqvist, A Flykt and A Öhman. *The Karolinska Directed Emotional Faces - KDEF*. CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet, 1998.

M. Lyons, S. Akamatsu, M. Kamachi and J. Gyoba, Apr 1998. M. Lyons, S. Akamatsu, M. Kamachi and J. Gyoba. Coding facial expressions with Gabor wavelets. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 200–205, Apr 1998. doi: 10.1109/AFGR.1998.670949.

Martinez and Benavente, **June 1998**. A.M. Martinez and R. Benavente. *The AR Face Database*, 1998.

Mehrabian, **1968**. A. Mehrabian. *Communication without words.* Psychology Today, 2 (4), 53–56, 1968.

Andrew Y. Ng, Michael I. Jordan and Yair Weiss, 2001. Andrew Y. Ng, Michael I. Jordan and Yair Weiss. On Spectral Clustering: Analysis and an algorithm. In *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS*, pages 849–856. MIT Press, 2001.

T. Ojala, M. Pietikainen and D. Harwood, Oct 1994. T. Ojala, M. Pietikainen and D. Harwood. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. In *Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision amp; Image Processing., Proceedings of the 12th IAPR International Conference on*, volume 1, pages 582–585 vol.1, Oct 1994. doi: 10.1109/ICPR.1994.576366.

Ojala et al., **Jul 2002**. T. Ojala, M. Pietikainen and T. Maenpaa. *Multiresolution gray-scale and rotation invariant texture classification with local binary patterns.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 24(7), 971–987, 2002. ISSN 0162-8828. doi: 10.1109/TPAMI.2002.1017623.

Ville Ojansivu and Janne Heikkilä. Blur Insensitive Texture Classification Using Local Phase Quantization. In Abderrahim Elmoataz, Olivier Lezoray, Fathallah Nouboud and Driss Mammass, editors, *Image and Signal Processing*, volume 5099 of *Lecture Notes in Computer Science*, pages 236–243. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-69904-0. doi: 10.1007/978-3-540-69905-7_27. URL `http://dx.doi.org/10.1007/978-3-540-69905-7_27`.

M. Pantic, M. Valstar, R. Rademaker and L. Maat, July 2005. M. Pantic, M. Valstar, R. Rademaker and L. Maat. Web-based database for facial expression analysis. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 5 pp.–, July 2005. doi: 10.1109/ICME.2005.1521424.

Parrott, **2001**. W. Gerrod Parrott. *Emotions in Social Psychology.* ISBN-13: 978-0863776830, Key Readings in Social Psychology. Psychology Press, 2001.

Pearson, **1901**. Karl Pearson. *LIII. On lines and planes of closest fit to systems of points in space.* Philosophical Magazine Series 6, 2(11), 559–572, 1901. doi: 10.1080/14786440109462720. URL `http://dx.doi.org/10.1080/14786440109462720`.

Rahtu et al., **2012**. Esa Rahtu, Janne Heikkilä and Ville Ojansivu. *Matlab codes for Local Phase Quantization*, 2012. Downloaded: 28-05-2014.

M. Ranzato, J. Susskind, V. Mnih and G. Hinton, *June 2011.* M. Ranzato, J. Susskind, V. Mnih and G. Hinton. On deep generative models with applications to recognition. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2857–2864, June 2011. doi: 10.1109/CVPR.2011.5995710.

Anwar Saeed, Ayoub Al-Hamadi and Robert Niese, *Nov 2012.* Anwar Saeed, Ayoub Al-Hamadi and Robert Niese. Neutral-independent geometric features for facial expression recognition. In *Intelligent Systems Design and Applications (ISDA), 2012 12th International Conference on*, pages 842–846, Nov 2012. doi: 10.1109/ISDA.2012.6416647.

Samal and Iyengar, **1992**. Ashok Samal and Prasana A. Iyengar. *Automatic recognition and analysis of human faces and facial expressions: a survey.* Pattern Recognition, 25(1), 65–77, 1992. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/0031-3203(92)90007-6.

Sebe et al., **2007**. N. Sebe, M.S. Lew, Y. Sun, I. Cohen, T. Gevers and T.S. Huang. *Authentic Facial Expression Analysis.* mage and Vision Computing, 25, 1856–1863, 2007.

Caifeng Shan, Shaogang Gong and Peter W. McOwan, *Sept 2005.* Caifeng Shan, Shaogang Gong and Peter W. McOwan. Robust facial expression recognition using local binary patterns. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 2, pages II–370–3, Sept 2005. doi: 10.1109/ICIP.2005.1530069.

Shan et al., **2009**. Caifeng Shan, Shaogang Gong and Peter W. McOwan. *Facial expression recognition based on Local Binary Patterns: A comprehensive study.* Image and Vision Computing, 27(6), 803 – 816, 2009. ISSN 0262-8856. doi: http://dx.doi.org/10.1016/j.imavis.2008.08.005. URL `http://www.sciencedirect.com/science/article/pii/S0262885608001844`.

Sim et al., **2002**. Terence Sim, Simon Baker and Maan Bsat. *The CMU Pose, Illumination, and Expression (PIE) Database*, 2002.

S. Singh, R. Maurya and A. Mittal, *Dec 2012.* S. Singh, R. Maurya and A. Mittal. Application of Complete Local Binary Pattern Method for facial expression recognition. In *Intelligent Human Computer Interaction (IHCI), 2012 4th International Conference on*, pages 1–4, Dec 2012. doi: 10.1109/IHCI.2012.6481801.

Susskind et al., **2010**. J.M. Susskind, A.K. Anderson and G.E. Hinton. *The Toronto Face Database*, 2010. Downloaded: 18-05-2014.

Suwa et al., **1978**. M. Suwa, N. Sugie and K. Fujimora. *A preliminary note on pattern recognition of human emotional expression.* Proceedings of the 4th International Joint Conference of Pattern Recognition, pages 408–410, 1978.

Tan et al., **2013**. Zheng-Hua Tan, Søren Holdt Jensen, Børge Lindberg, Nicolai Bæk Thomsen and Xiaodong Duan. *Durable Interaction with Socially Intelligent Robots (iSocioBot)*. `http://www.socialrobot.dk/` and `http://kom.aau.dk/~zt/iSocioBot/index.htm`, 2013. Accessed: 08-05-2014. Grant DFF – 1335-00162.

Ying-Li Tian, June 2004. Ying-Li Tian. Evaluation of Face Resolution for Expression Analysis. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, pages 82–82, June 2004. doi: 10.1109/CVPR.2004.60.

Tian et al., **Feb 2001**. Ying-Li Tian, Takeo Kanade and Jeffrey F. Cohn. *Recognizing action units for facial expression analysis*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 23(2), 97–115, 2001. ISSN 0162-8828. doi: 10.1109/34.908962.

Valstar et al., **Aug 2012**. M.F. Valstar, M. Mehu, Bihan Jiang, M. Pantic and K. Scherer. *Meta-Analysis of the First Facial Expression Recognition Challenge*. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 42(4), 966–979, 2012. ISSN 1083-4419. doi: 10.1109/TSMCB.2012.2200675.

Michael F. Valstar, Bihan Jiang, Marc Mehu, Maja Pantic and Klaus Scherer, March 2011. Michael F. Valstar, Bihan Jiang, Marc Mehu, Maja Pantic and Klaus Scherer. The first facial expression recognition and analysis challenge. In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 921–926, March 2011. doi: 10.1109/FG.2011.5771374.

S. Velusamy, V. Gopalakrishnan, B. Anand, P. Moogi and B. Pandey, Jan 2013. S. Velusamy, V. Gopalakrishnan, B. Anand, P. Moogi and B. Pandey. Improved feature representation for robust facial action unit detection. In *Consumer Communications and Networking Conference (CCNC), 2013 IEEE*, pages 681–684, Jan 2013. doi: 10.1109/CCNC.2013.6488525.

P. Viola and M. Jones, 2001. P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1, 2001. doi: 10.1109/CVPR.2001.990517.

D. Vukadinovic and M. Pantic, Oct 2005. D. Vukadinovic and M. Pantic. Fully automatic facial feature point detection using Gabor feature based boosted classifiers. In *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, volume 2, pages 1692–1698 Vol. 2, Oct 2005. doi: 10.1109/ICSMC.2005.1571392.

wei Hsu et al., **2010**. Chih wei Hsu, Chih chung Chang and Chih jen Lin. *A practical guide to support vector classification*, 2010.

Songfan Yang and B. Bhanu, March 2011. Songfan Yang and B. Bhanu. Facial expression recognition using emotion avatar image. In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 866–871, March 2011. doi: 10.1109/FG.2011.5771364.

Baohua Yuan, Honggen Cao and Jiuliang Chu, March 2012. Baohua Yuan, Honggen Cao and Jiuliang Chu. Combining Local Binary Pattern and Local Phase Quantization for Face Recognition. In *Biometrics and Security Technologies (ISBAST), 2012 International Symposium on*, pages 51–53, March 2012. doi: 10.1109/ISBAST.2012.14.

Zhao and Pietikainen, June 2007. Guoying Zhao and Matti Pietikainen. *Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 29(6), 915–928, 2007. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1110.

Wang Zhen and Ying Zilu, May 2012. Wang Zhen and Ying Zilu. Facial Expression Recognition Based on Local Phase Quantization and Sparse Representation. In *Natural Computation (ICNC), 2012 Eighth International Conference on*, pages 222–225, May 2012. doi: 10.1109/ICNC.2012.6234551.