VERY LOW BITRATE FACIAL VIDEO CODING
BASED ON PRINCIPAL COMPONENT ANALYSIS

Ulrik Söderström

Department of Applied Physics and Electronics,
Umeå University
Umeå 2006
Digital Media Lab
Department of Applied Physics and Electronics
Umeå University
SE-901 87 Umeå, Sweden

Thesis for the degree of licentiate of engineering in applied electronics at Umeå University

Vetenskaplig uppsats för avläggande av teknologie licentiatexamen i tillämpad elektronik vid Umeå universitet

ISSN 1652-6295:7
ISBN 91-7264-172-X

Copyright © 2006 by Ulrik Söderström

Author's email: ulrik.soderstrom@tfe.umu.se
Typeset by Ulrik Söderström \LaTeX
Printed by Print & Media, Umeå University, Umeå, 2006
Till May.

Du var den bästa.
## Contents

**Acknowledgement** 5  
**Abstract** 7  
**Publications** 9  

### 1 Introduction and motivation

1.1 General introduction ........................................... 11  
1.2 Compact face modelling ....................................... 13  
   1.2.1 Principal component analysis ............................... 13  
   1.2.2 Compact facial representation ............................ 14  
   1.2.3 The face space and personal mimic space ............... 15  
1.3 Low bitrate facial representation ............................ 16  
1.4 Research process .............................................. 17  

### 2 Principal Component Analysis video coding

2.1 Introduction .................................................... 19  
2.2 Principal component video coding ............................. 19  
   2.2.1 Chroma subsampling ...................................... 19  
   2.2.2 Hands-free video system .................................. 21  
   2.2.3 Video sequences .......................................... 21  
   2.2.4 Eigenspace construction .................................. 22  
      2.2.4.1 Singular Value Decomposition ......................... 23  
   2.2.5 Encoding with Eigenspace ................................. 23  
   2.2.6 Decoding with Eigenspace ................................. 24  
2.3 Encoding and decoding time .................................. 24
CONTENTS

2.4 Practical results ............................................. 24
2.5 Theoretical bounds ........................................... 27
  2.5.1 Distortion bound ........................................ 28
  2.5.2 Rate-Distortion bound .................................. 29
  2.5.3 Distortion measurement comparison ...................... 30
2.6 Compression of the Eigenimages .............................. 32
  2.6.1 Quantization - uniform or pdf-optimized? ............... 32
  2.6.2 JPEG compression ....................................... 33
  2.6.3 Compression of the mean image .......................... 33
  2.6.4 Loss of orthogonality ................................... 33
  2.6.5 Compression methods .................................... 34
2.7 Re-use of Eigenspaces ...................................... 38
  2.7.1 Sensitivity tests .......................................... 39
  2.7.2 Adjustment of different pixel intensities ............... 39
  2.7.3 Normalization of feature position ....................... 40
  2.7.4 Selecting original coefficients ........................ 41

3 User test on conversational Multimedia ........................ 45

4 Conclusions and contributions ................................ 49

Bibliography .................................................. 51
Acknowledgement

First I would like to thank my supervisor and guide, professor Haibo Li. It is due to his help, insight, advice and guidance that I have been able to pursue my research.

Thanks to all the members of Digital Media Lab who have helped me in one way or another. Special gratitude goes to Zhengrong Yao who helped me in many ways in the beginning of my research career.

Tack till Sara, Ville, Peter och Niklas.utan våra fika- och lunchraster hade min tillvaro varit mycket tråkigare.


Ett stort tack till Leif och Sven som byggde min ”hjälm”. Den skulle bara användas till mitt examensarbete men den har förföljt mig under hela min forskarkarriär.

Tack till alla som jag har samarbetat med under min tid som forskare, speciellt Greger Wikstrand och Anders Broberg.

Jag vill tacka Per Levén och Jerry Eriksson som gav mig en projektanställning innan jag blev doktorand.

Tack till min familj som stöttat och hjälpt mig genom många saker i livet. Det glömmer jag aldrig.

Tack till Bo Kaspers orkester, Lars Winnerbäck och Kent, vars musik tagit mig genom många arbetspass.
Ett stort tack går till Smilla. Det är i princip omöjligt att inte vara på bra humör när man möts av 30 kilo kärlek när man kommer hem.

Elin, du förtjänar ett tack som är större än ord. Eftersom jag är begränsad till just ord i den här uppsatsen får Tack av hela mitt hjärta duga.
Abstract

This thesis introduces a coding scheme for very low bitrate video coding through the aid of principal component analysis. Principal information of the facial mimic for a person can be extracted and stored in an Eigenspace. Entire video frames of this person's face can then be compressed with the Eigenspace to only a few projection coefficients. Principal component video coding encodes entire frames at once and increased frame size does not increase the necessary bitrate for encoding, as standard coding schemes do. This enables video communication with high frame rate, spatial resolution and visual quality at very low bitrates. No standard video coding technique provides these four features at the same time.

Theoretical bounds for using principal components to encode facial video sequences are presented. Two different theoretical bounds are derived. One that describes the minimal distortion when a certain number of Eigenimages are used and one that describes the minimum distortion when a minimum number of bits are used.

We investigate how the reconstruction quality for the coding scheme is affected when the Eigenspace, mean image and coefficients are compressed to enable efficient transmission. The Eigenspace and mean image are compressed through JPEG-compression while the while the coefficients are quantized. We show that high compression ratios can be used almost without any decrease in reconstruction quality for the coding scheme.

Different ways of re-using the Eigenspace for a person extracted from one video sequence to encode other video sequences are examined. The most important factor is the positioning of the facial features in the video frames.

Through a user test we find that it is extremely important to consider secondary workloads and how users make use of video when experimental setups are designed.
List of publications

Peer review papers

Ulrik Söderström and Haibo Li, *Very low bitrate full-frame facial video coding based on principal component analysis*, Signal and Image Processing Conference (SIP’05), Honolulu, August 15-17 2005


Hung-Son Le, Ulrik Söderström and Haibo Li, *Ultra low bit-rate video communication, video coding = facial recognition*, Proc. of 25th Picture Coding Symposium, Beijing, April 2006


Other papers


Ulrik Söderström and Haibo Li, *Re-use of Eigenspaces to encode new facial video sequences*, Manuscript
Introduction and motivation

1.1 General introduction

High quality video communication is a very popular feature for many applications. "High quality" is a term that can be translated into three major factors; high spatial resolution, high visual quality and high framerate. A high spatial resolution means that there are many pixels in the image, but the pixels must also be unique. It is possible to reach a high spatial resolution with interpolation techniques where new pixels are created from the surrounding pixels. Interpolation only increases the spatial resolution at the expense of image quality. High quality means that there is both high spatial resolution and image quality at the same time. With high framerate we refer to a framerate that is at least 15 frames per second. When a framerate below 12-15 is used the sense of having eye-contact and lip synchronization is lost [1].

The creation of high quality video will, with standard coding methods, require large bandwidth. With standard methods it is impossible to create high quality video at very low bitrates. There are standard coding methods that allow video at low bitrates with reasonable quality [2]. Some standards can provide video at very low bitrates [3,4] but the quality they provide is quite poor. There is a large gap in the technology when it comes to video over very low bitrates. A description of the standards is found in section 1.3. Most standards for video communication are based on block-based encoding through transform coding, most often Discrete Cosine Transform (DCT). The video frame is divided into small blocks, usually each block is 8x8 pixels large. Each block is then encoded through DCT. This encoding does not provide enough compression by itself so these methods also make use of block matching to increase the compression and reduce the bitrate. With block matching you only need to encode the first frame in a sequence through DCT. For the following frames it is enough to encode how each block has moved and changed from the first frame. When very low bitrate is desired the number of DCT-coefficients that are used for the first frame is too low and the possible movements of the blocks are too crude; these methods simply do not reach very
INTRODUCTION AND MOTIVATION

low bitrates while retaining a reasonable visual quality. The number of separate blocks is also increased when the spatial resolution is increased, so these methods can only provide large resolution at the expense of higher bitrate or reduced visual quality. There are methods that are based on other techniques than DCT that are better at reaching low bitrates. However, to reach low bitrates it is important to make the assumption that there is a certain type of video that should be encoded. To this date there is no technique that is better than DCT for encoding arbitrary video sequences. Video sequences of a face or the head and shoulders of someone are widely used in video conference and video telephone applications. Assuming that a video sequence mainly consists of a face, or head and shoulders, it is possible to use other techniques than DCT for successful encoding of this video sequence. A review of some of these techniques can be found in section 1.3. All of these methods have drawbacks; some can’t provide real-time performance, some are animation-like in their appearance and some does not provide a frame-rate that is high enough. Any technique based on animation or texture-mapping to a model is not sufficient. Solutions with high visual quality at the expense of low framerate also fall short. The sense of having eye contact is sensitive to low resolution while lip synchronization is sensitive to low framerate [1]. Therefore, it is not sufficient to provide either high framerate or spatial resolution; both are important. The conclusion is that there is a need for a video coding technique that can provide high visual quality, high spatial resolution and high framerate at very low bitrates.

When a coding scheme is to be created it is important to consider the intended usage of the coding scheme. When we are referring to very low bitrates it does not say anything about the scenario which the scheme will be used in. There are very different demands depending on which system that the coding scheme is used on. The coding scheme that we present in this thesis can be used for both video telephone communication on mobile devices and for video conference applications on standard PC’s. The use of mobile devices imposes problems on the computational and memory capacity which often are low on mobile devices. There are also much higher restrictions for bandwidth for wireless connections compared to wired connections. Our video coding scheme can cope with all of these issues.

Wearable computers can provide some interesting features wanted for video communication. Fickas et.al. [5] state that wearable computers provide hands-free operation and mobility. A wearable computer can be used with one or no hands and it is not tethered, allowing the user to roam freely. To move freely and communicate without the use of hands is easily realized by using a hands-free microphone and a cellular phone, but this doesn’t include video communication. In several environments there is a need for both video communication and hands-free operation. Eye contact is important to the communication between two persons [6]. Kraut et.al. [7] found that bicycle repairs are completed twice as fast when a remote expert provides assistance through video communication. Video communication can hence improve the working conditions. The use of hands-free equipment allows the user to roam more freely but it also gives benefits for video coding that can be exploited. When hands-free equipment is used it ensures that the face of
the user is filmed at a near frontal view. The large changes in such a video will be the changes in facial appearance; changes in the facial mimic. This is a benefit that we make use of.

In this thesis we present a solution for encoding video sequences of faces into very low bitrates while still retaining a high spatial resolution, image quality and framerate. Theoretical bounds that show how efficient this coding can be are presented along with practical results. We show what happens with the performance of the coding scheme when the principal components are compressed to allow easier transmission and storage. We also present how existing Eigenspaces can be used for encoding of new video sequences, the difficulties this creates and how these can be solved. Furthermore, we present the results and conclusions from a subjective user test on video quality parameters in computer-mediated-communication.

The disposition of the thesis is as follows. In the rest of this section we present the background and motivation for the development of principal component video coding. We also describe how the research process has evolved. In chapter 2 the coding scheme is described. This chapter contains information about the usage of the coding scheme, practical results, theoretical bounds, compression of Eigenspaces and re-use of Eigenspaces. A user test on conversational Multimedia is described in chapter 3 and the contributions and conclusions of this thesis are presented in chapter 4.

1.2 Compact face modelling

The human face is able to exhibit several facial expressions. All of the expressions are often referred to as the facial mimic. The changes in the face are the only thing that changes when we communicate visually and verbally through our faces. We consider changes in appearance, such as shading and lightning, to be negligible; only changes of facial features exist. Consequently, a model of the entire human face only needs to be a model of the facial mimic.

1.2.1 Principal component analysis

Principal component analysis (PCA) is originally a dimension reduction technique [8]. PCA can be used to reduce the number of dimensions that is needed to model a space and it can detect structures in the model that are not visible at a first sight.

PCA is a linear transformation that transforms a number of correlated variables into a number of uncorrelated variables, called principal components or basis vectors. This procedure means that the coordinate system is changed so that the greatest variance by any projection of the data is aligned along the first coordinate axis. The second greatest variance is aligned along the second coordinate axis, etc. Dimension reduction is achieved by retaining fewer principal components than the number of dimensions in the original data set.
INTRODUCTION AND MOTIVATION

PCA is also called the Karhunen-Loève transform (KLT) and is sometimes referred to as the optimal linear transform. It is optimal in the sense that no other linear transform is able to compact the data as efficiently. By compacting data we mean representing as much information as possible with the use of only a subset of the principal components. This advantage over other transforms comes at a price; PCA is data dependent and each data set will have its own basis vectors. Other transforms that are used for image compression, e.g. DCT, have the same basis vectors independently of the data set.

There are some different algorithms that perform PCA and in our work we have chosen to use Singular Value Decomposition (SVD) to perform PCA. A description on SVD is found in section 2.2.4.1.

1.2.2 Compact facial representation

The American psychologist Paul Ekman has performed research on the human face, human facial expressions and facial mimic. [9,10]. He has shown that the visual expressions in a human face can be modelled with only little amount of information. According to Ekman, all possible facial emotions can be modelled with six basic emotions. The basic emotions are happiness, sadness, surprise, fear, anger and disgust (figure 1.1). By blending the basic emotions in different ways it is possible to create any other expression. Following his theory it is possible to model the facial expressions with a model that contains only six dimensions, one for each basic expression. It is however not so straightforward to say that you can use six images of the basic expressions to model all possible images of facial expressions. There is a difference between the psychological representation of expressions and the representation in digital images. Therefore it is reasonable to assume that more than six images are needed to represent the facial expressions through digital images.

As we have explained in the previous section, PCA can be used to reduce the dimensionality of a data set. PCA can also be used to represent facial images. This use of PCA was introduced by Kirby and Sirovich [11,12]. They used PCA to create a very compact model of human faces, by applying PCA to a series of facial images. They used normalized facial images as input to a PCA and used the compact description to model the faces. The space that the input faces is distributed in can be seen as very high-dimensional, with one dimension for each facial image. After PCA has been applied, the number of dimensions can be vastly reduced while the representation quality is almost retained. Kirby and Sirovich found that fewer than 100 principal faces are needed to model male Caucasian faces [12]. This is a very compact representation since it is enough to use 100 facial images to model millions of faces.
1.2.3 The face space and personal mimic space

The face space can be seen as a high-dimensional space which human faces are distributed within, consisting of one dimension for each face that exists. The face space is often used for facial recognition, where it is desirable to reduce the dimensionality while retaining the difference, or distance, between the faces in the space. This is where PCA has a major role to play; a compact representation extracted through PCA will have exactly these two features. Through PCA the dimensionality of the face space can be reduced sufficiently to enable real-time use of the face space for facial recognition while still retaining a sufficient difference between the faces. To perform facial recognition within the face space it is enough to calculate the Euclidian distances between the faces in the space and the face that should be recognized. The face that yields the smallest distance is the best match for the new face and is assumed to be the same person.

The Euclidian distance between the faces in the face space represents the difference in appearance that exists between the faces. What happens if these faces are not different peoples faces, but instead different facial expressions of the same person? Then this space models the facial expressions and mimic that this person can express through her face. We call this personal mimic space and it is also high-dimensional. Ohba et.al. presents a good introduction to the personal face (mimic) space [13,14]. Just as the face space can be reduced in dimensionality through PCA, the personal mimic space can as well. The separation of the different faces in the personal mimic space corresponds to the separation of the facial

![Figure 1.1: The six basic emotions](image-url)
expressions. This distribution in the first 3 dimensions for two personal mimic spaces is shown in figure 1.2. All of the facial mimic will be distributed in the high-dimensional space but it is impossible to visualize more than three dimensions so we only show the first three dimensions. Each point is a possible face and every image that is close to these points is a face. By modelling this space it is possible to model the entire facial mimic.

1.3 Low bitrate facial representation

Many different methods and standards have been developed for very low bitrate encoding and representation of facial video sequence. Video coding standards like MPEG-4 [15] and h.264 [3,4] have been defined. Both coding standards use DCT encoding of blocks and block-matching of areas in the video images. Instead of coding the content of each video frame these techniques make use of the fact that frames adjacent in time to each other share large similarities. Reference frames are encoded as still images. Block matching means that the reference frame blocks are matched against a new frame and the only thing that needs to be encoded is the positional change and slight block changes between the frames. In this way, the compression can be very high for long sequences of approximately the same content. Therefore these coding techniques should be suitable to use for encoding of facial video sequence but when very low bitrates are wanted, the reconstruction result is poor. With block-wise DCT, each block is compressed separately. When high compression ratios are used the boundaries between the blocks can become very apparent in the image, causing a common block error. High compression ratio also means that the quality in each individual block is low and the block-matching is crude. The total result is images of low quality with block patterns apparent in the images.

Other techniques have also been proposed to reach very low bitrates. Model-based coding is a technique where a facial model is used with texture mapping to create video-like animations at very low bitrates [16]. Model-based coding was
first proposed by Forchheimer et. al. in 1983 [17]. The facial model often consists of a wire-frame model that can be driven through the movement of the points in the wire-frame. This wire-frame structure is also used in facial animation. Facial animation is a technique that enables visual communication at very low bitrates [18]. Both facial animation and model-based coding reaches very low bitrate representation of human faces since only the changes of the wire-frame points has to be encoded. Facial animation is defined in the MPEG-4 standard but is not widely used. Model-based coding is used for one of the most impressive low bitrate encodings that has been performed on facial video. Eisert and Girod used model-based coding to create video of very high quality at very low bitrates [19]. Matching pursuit is another technique that reaches very low bitrate encoding [20]. Matching Pursuit uses features from an alphabet to reconstruct video sequences. Only information about which features that are used in a video frame has to be transmitted.

Almost all of the above mentioned methods can represent the facial mimic with little amount of information but they all have limitations in visual quality and/or frame rate. Pighin et. al. provides a good explanation why high visual quality is important and why video is superior to animations [21]. The face simply exhibits so many tiny creases and wrinkles that it is impossible to model with animations or low spatial resolution. That means that any technique based on animation or texture-mapping to a model is not sufficient. Some approaches have focused on retaining the spatial quality of the video frames at low bitrates at the expense of frame rate. Wang and Cohen presented a solution where high quality images are used for teleconferencing over low bandwidth networks with a framerate of one frame each 2-3 seconds [22]. The idea of using low framerate is however not acceptable since both high framerate and high spatial resolution is important for many visual tasks. According to Lee and Eleftheriadis different facial parts have different encoding needs [1]. The sense of having eye contact is sensitive to low resolution while lip synchronization is sensitive to low framerate. Therefore it is not sufficient to provide either high framerate or spatial resolution; both are important.

There exists no technique that is good enough for displaying high quality facial images with high framerate over low bitrate networks. That is the main motivation for the development of a new coding scheme. The findings of Paul Ekman show that it should be possible to model facial mimic in a very sparse space. This finding also motivates us to make use of a model for facial mimic to reach very low bitrates. The use of PCA for sparse modelling of faces and facial expressions provides motivation for using PCA to extract the most prominent facial expressions.

1.4 Research process

The research presented in this thesis has evolved through extensive discussions with my supervisor, Prof. Haibo Li. First, we found that no video coding tech-
INTRODUCTION AND MOTIVATION

Principal component video coding functions well for low bitrate video with high quality. We decided to work with facial video sequences since they are becoming popular and have some features that make it possible to use high compression. Haibo Li had already tried to use a low bitrate representation of facial video when he was a student, back in 1993 [23]. We found that Paul Ekman had shown that facial expressions can be modelled in a very efficient way [9,10]. I developed a first practical implementation of principal component video coding and examined how it worked. I found several problematic issues and choose to use the hands-free video equipment that I had previously used for my master thesis (section 2.2.2). I recorded more video sequences and made further practical experiments. I then performed a user test together with Greger Wikstrand to examine how video can be used in conversational multimedia. The result implied that video has such high impact that it must be used wisely, otherwise video might actually degrade the user experience. Principal component video coding has a purpose to serve there since it can enable good visual communication. Then I became interested in knowing how efficient the coding scheme could be. Many researchers had used PCA for encoding of human faces but no one had derived bounds that show exactly how efficient this can be. First I tried to derive theoretical bounds assuming that the facial mimic was distributed according to a Gaussian mixture, but I limited myself to bounds where I assume that the facial mimic is Gaussian distributed. The next step was to focus on the practical usage of the coding scheme. If the Eigenspace should be transmitted over a low capacity network it must be compressed. I examined how compression affects the reconstruction quality and if compression can be performed. If it is not possible to compress the Eigenspace and update or send it over a low capacity network it must be possible to re-use an Eigenspace. I therefore examined how the Eigenspace can be re-used to encode new video sequences.
2.1 Introduction

The motivation for the development of a coding scheme based on principal component analysis (PCA) is described in detail in chapter 1. The main reason is however that high compression of video sequences with standard techniques lead to a reduction in either resolution, image quality or frame rate, often in at least two of the mentioned factors. Since high spatial resolution and quality is important there is an apparent need for video coding that can provide all three factors at very low bitrates. In this chapter we will describe a coding scheme based on PCA that is the main part of this thesis.

The coding scheme is introduced in section 2.2, where the different parts or the system are described. Performance and practical results are discussed in section 2.3 and 2.4, respectively. Theoretical bounds for the efficiency of the coding are presented in section 2.5 and compression of the Eigenspaces is discussed in section 2.6. How an existing Eigenspace can be used to encode new video sequences is handled in section 2.7.

2.2 Principal component video coding

Principal component analysis (PCA) can be used to create a compact model of a person’s facial mimic. Since PCA has such high energy compactness it is possible to encode the entire frame at once with PCA. The compact model is improved through a number of techniques which are described in the following section.

2.2.1 Chroma subsampling

The first step is to reduce the size of each video frame through chroma subsampling. The original video frames are represented in RGB color space. This color space
can without loss be converted into YUV color space. The three components of YUV color space are created from RGB values according to equation 2.1.

$$Y = 0.299R + 0.587G + 0.114B$$
$$U = -0.147R - 0.289G + 0.436B$$
$$V = 0.615R - 0.515G - 0.100B$$

(2.1)

The YUV color space has a very important feature; it is based on luminance, Y, and chrominance, U and V. The human eye is much more sensitive to changes in luminance than changes in chrominance [24], meaning that the YUV color space can be compressed without any visible effect for a human observer. This is done through subsampling of the chrominance channels, U and V. There are many different ways to perform chroma subsampling and we have chosen to use YUV 4:1:1 subsampling. With this subsampling method only every fourth value of U and V is stored together with all Y values (figure 2.1). The horizontal color resolution is reduced with this method.

The pixel information that is retained is organized into a single frame according to figure 2.2. In this way the frame size is reduced to half the original size almost without any loss in quality visible to a human observer. The decrease in quality is still measurable with peak signal-to-noise ratio.

The frame size for the videos that are used in the experiments are 240x176x3 pixels. After YUV subsampling the frame size is 360x176 pixels. This compression is used for video frames, the principal components and the mean image.

The organization of the Y, U and V components have importance when the images later will be compressed with the JPEG standard. This standard is more
efficient at compressing areas which have similar values. By organizing all Y pixels adjacent to each other while all U and V pixels are adjacent to each other the change in pixel value is minimized. In this way the compression of the images is made as efficient as possible.

The subsampled color information can be used to create a full YUV frame. From this frame a reconstructed video frame in RGB format can be attained.

2.2.2 Hands-free video system

In chapter 1 we introduced the idea that wearable computers provide features that can be exploited for video compression. Therefore we have used a hands-free video equipment to make encoding with the scheme more efficient.

To enable the use of PCA for modelling of the face the facial features must be positioned at the same pixel position in all frames. This is usually ensured through a normalization of the faces before PCA is applied. To circumvent such a computationally heavy normalization we make use of a hands-free camera solution that ensures that the facial features are positioned at approximately the same pixel position from the beginning. To emulate a hands-free solution for video communication we use a construction helmet and a web camera. A light metal arm is attached to the helmet and the web camera is attached to the metal arm (figure 2.3). The camera is positioned so that it films the face of the person wearing the helmet.

2.2.3 Video sequences

Most experiments in this thesis are performed with the same video sequences. 10 video sequences are used to evaluate the performance of the principal component coding scheme. The same 10 video sequences are used to calculate theoretical bounds and examine how compressed Eigenspaces perform. Four of these sequences show the same person recorded on different time instances. These sequences are used to examine how existing Eigenspaces can be used to encode new video sequences.
Each of the video sequences show one person when he/she is displaying the six basic emotions proposed by Ekman [10]. After each emotion the subject returns to a neutral state. The video sequences are approximately 30 seconds long, an emotion is displayed for 2-4 seconds and a new emotion is displayed approximately every 5 seconds. The framerate for all the video sequences is 15 fps and the resolution is 240x176 pixels. The resolution of the video sequences is chosen to approximately match the resolution used in video telephony over 3G networks. Two different backgrounds are used for the video sequences. Eight of the sequences are shot against a white background and two of the sequences are shot against a non-homogenous background. Example frames from some of the video sequences are shown in figure 2.4.

2.2.4 Eigenspace construction

When PCA is applied to a series of images a number of principal components can be extracted. There are as many principal components as there are images used as input to the PCA. Each principal component is called Eigenimage since each principal component in fact is an image. A collection of Eigenimages is referred to as an Eigenspace. An Eigenspace is created through a series of operations.

First, all video frames, $N$ in a video sequence $X_n(x,y)$ $n=(1,2,...N)$ are converted into YUV color space and YUV compressed. The frames are then transformed into column vectors and stored in a matrix $O$. The mean value for each pixel position is calculated according to equation 2.2 and stored as an image; the mean image, $X_{mean}$.

\[
X_{mean} \approx \frac{1}{N} \sum_{n=1}^{N} X_n \tag{2.2}
\]

where $N$ is the number of frames in the video sequence and $n$ is the frame number.

The mean image is then subtracted from all column vectors of $O$. A new matrix is created from $O$, consisting of $O^T$ multiplied with $O$. A singular Value Decomposition (SVD) is performed on the matrix $O^T O$. This yields the three matrices,
U, S and V (The matrices are explained in section 2.2.4.1). O is multiplied with V to create the matrix E.

\[ \mathbf{O} \cdot \mathbf{V} = \mathbf{E} \]  

(2.3)

This algorithm removes the need to perform a SVD for the much larger matrix \( \mathbf{OO}^T \). Each column of \( \mathbf{E} \) is an Eigenimage.

\[ \mathbf{E} = \begin{bmatrix} \Phi_{1,1} & \Phi_{2,1} & \cdots & \Phi_{N,1} \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_{1,x+y} & \Phi_{2,x+y} & \cdots & \Phi_{N,x+y} \end{bmatrix} \]  

(2.4)

Only the chosen number M of principal components in \( \mathbf{E} \), \( m=(1,2,\ldots,M) \) with the largest Eigenvalues \( \Lambda_m \) have to be stored as the Eigenspace \( \Phi \).

\[ \Phi = \begin{bmatrix} \Phi_{1,1} & \Phi_{2,1} & \cdots & \Phi_{M,1} \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_{1,x+y} & \Phi_{2,x+y} & \cdots & \Phi_{M,x+y} \end{bmatrix} \]  

(2.5)

The Eigenimages are then normalized so they have the norm of 1. This is done by calculating the original norm of each Eigenimage and dividing the Eigenimages with the respective norm.

\[ \Phi_n = \frac{\Phi_n}{|\Phi_n|} \]  

(2.6)

The Eigenspace \( \Phi \) is now orthonormal. Each Eigenimage have the norm of 1 and the Eigenimages are orthogonal to each other. After the extraction of the Eigenspace \( \Phi \) and the mean image \( \mathbf{X}_{\text{mean}} \) they can be used to encode video sequences.

### 2.2.4.1 Singular Value Decomposition

Singular value decomposition (SVD) is a factorization of a rectangular real or complex matrix. Any matrix \( \mathbf{P} \) can be decomposed into three matrices \( \mathbf{U}, \mathbf{S} \) and \( \mathbf{V} \).

\[ \mathbf{USV} = \mathbf{P} \]  

(2.7)

\( \mathbf{V} \) contains a set of orthonormal input basis vectors for \( \mathbf{P} \). \( \mathbf{U} \) contains a set of orthonormal output basis vectors for \( \mathbf{P} \). \( \mathbf{S} \) contains the singular values, ordered in descending order.

### 2.2.5 Encoding with Eigenspace

When an Eigenspace is created according to the section above it is possible to use this Eigenspace to encode video frames. A video frame that should be encoded will be preprocessed in the same way as the Eigenimages. The frame is converted
into YUV color space and YUV compressed, according to section 2.2.1. It is transformed into a column vector and the mean image is subtracted from the video frame.

The video frame is then projected on the Eigenspace. This is simply a vector dot multiplication between the video frame and the Eigenimages. The result is a single value for each Eigenimage, called projection coefficient $\alpha^x$.

$$\alpha^x_m = X^T \Phi_m$$

where $\alpha^x$ is the projection coefficient, $m$ is the number of the Eigenimage $\Phi$ and $X$ is the video frame that should be encoded.

### 2.2.6 Decoding with Eigenspace

The Eigenimages $\Phi$ and the projection coefficient $\alpha^x$ are multiplied to decode the video frames. By adding the result from the multiplication of each Eigenimage and respective projection coefficient with the mean image the video frame is reconstructed.

$$\hat{X} = X_{mean} + \sum_{m=1}^{M} \alpha^x_m \Phi_m$$

where $\alpha^x$ is the projection coefficient, $m$ is the current number of the Eigenimage $\Phi$, $X_{mean}$ is the mean image and $\hat{X}$ is the reconstructed image.

### 2.3 Encoding and decoding time

The coding scheme is somewhat computationally expensive. To make certain that the coding scheme can be used on mobile devices with limited computational and memory capacity the first experiments were carried out on a Pentium III 1100 MHz workstation with 256 MB RAM. The computational and memory capacity of this workstation is similar to the capacity of mobile devices.

The average time for projecting one video frame onto the Eigenspace (encoding) was 10.9 ms for an Eigenspace of 10 Eigenimages and 26.6 ms for 25 Eigenimages. On average it took 13.8 ms to reconstruct one frame from the projection coefficients (decoding) for 10 Eigenimages and 34.9 ms for 25 Eigenimages. The encoding and decoding time is equal to a framerate of 40 fps for 10 Eigenimages and 16 fps for 25 Eigenimages. The video sequences have a framerate of 15 fps so it is possible to encode them with up to more than 25 Eigenimages in real-time.

### 2.4 Practical results

The Eigenspace and mean image extracted from a video sequence according to section 2.2.4 is used to encode and decode videos sequences according to section
2.2.5 and 2.2.6, respectively. The reconstruction quality is measured in peak signal to noise ratio (PSNR). PSNR is calculated from the mean square error (MSE) between the original and reconstructed frames.

\[
MSE = \sum_{i=1}^{x*y} \frac{(X_i - \hat{X}_i)^2}{x*y}
\]  
\[
PSNR = 20 \times \log\left(\frac{255}{\sqrt{MSE}}\right)
\]

where \(x*y\) are the number of pixels in each frame, \(i\) is the pixel position, \(X\) is the original video frame and \(\hat{X}\) is the reconstructed video frame. 255 is the maximum pixel value.

The reconstructed video sequence have different quality depending on the number of Eigenimages \(\Phi\) that are used for reconstruction. The average peak signal to noise ratio (PSNR) values for all the videos are noted in table 2.1. The reconstruction quality for a video sequence is dependent on the person in the video sequence since the level of expression a person uses to show emotions affect the Eigenspace creation and the coding performance. The more expressive emotions a person uses, the lower reconstruction quality. In figure 2.5 reconstructed images and difference images for the basic emotions are shown for one of the video sequences. The video sequences with non-homogenous background had slightly lower reconstruction quality, measured in PSNR, than the video sequences with white background. The background is blurred for the video sequences with a non-homogenous background but the quality of the reconstructed facial image is maintained.

A detailed description of the practical experiments is found in paper I.
Figure 2.5: Left: Original images. Center: Reconstructed images using 10 Eigenimages. Right: Error images when using 10 Eigenimages (scaled to highlight the error.)
2.5 Theoretical bounds

Exactly how compact can the representation of the facial mimic for a human be? This question has been addressed within the psychological community. According to American psychologist Paul Ekman all facial emotions consist of blended versions of only six basic emotions [9,10]. By modelling these six emotions it is possible to model all possible facial emotions. It is however not so straightforward to say that you can use 6 images of the basic expressions to model all images of the facial expressions. There is a difference between the psychological representation of expressions and the representation in digital images. So, our goal is to evaluate exactly how compact representation of the facial mimic that can be created with the aid of PCA.

Kirby and Sirovich found that fewer than 100 principal faces is needed to model male Caucasian faces [12]. This can be considered to be very compact; by using only 100 facial images you can model millions of faces. All faces do however have the same expression so this does not model any information about the facial mimic. Ekman’s claim that the facial mimic for one person can be modelled with six basic emotions isn’t considered by Kirby and Sirovich. There are implementations that consider the personal mimic space. All of these attempts of using PCA to encode images, both still images and video sequence, show that images can be encoded at very low bitrates with high reconstruction quality, measured in PSNR. No one has derived a bound for the efficiency of this kind of coding. Crowley et.al. show that they can produce a reconstruction quality of almost 35 dB (PSNR) when they use 15 basis images for reconstruction [25]. Torres et.al. have an average reconstruction quality of 29.6 dB when they use 16 basis images [26]. We have shown that when 10 basis images are used an average reconstruction quality above 31 dB can be achieved [27,28]. There are many results presented, but what is the performance limit?

We want to derive theoretical bounds for modelling facial mimic through PCA. The task is however not so straightforward. The representation bound is affected by multiple factors, e.g., the spatial resolution, the quality of the original video sequence and how expressive the facial mimic of the person is. Therefore, we limit the task to the given circumstances:

- A spatial resolution of 240x176 pixels
- A color depth of 8 bits per pixel
- RGB color space in the original video
- Theoretical bounds are calculated individually for each person

The spatial resolution is equal to high resolution for mobile phones. The standard with the highest quality used today is called QVGA and has a total pixel number of approximately 77000 [29]. A resolution of 240x176 pixels is equal to 42420 pixels.
Two different bounds can be derived, a rate-distortion bounds and a distortion bound. Rate-distortion bound refers to the pair of minimum rate and minimum distortion. For a given distortion, the source cannot be modelled correctly by a rate lower than the rate described in the derived function. The distortion bound describes the minimum distortion that can be achieved for a specific source.

The distortion that is measured in these bounds is measured in mean square error, where the pixel intensities are compared between the original and reconstructed images. This measurement is based on the signal level. Lower mean square error does not guarantee that the reconstructed image has a higher visual quality. Subjective evaluation gives a better measurement of the actual visual quality. Since it is subjective it is impossible to derive a theoretical bound for this kind of measurement. Therefore we focus on objective quality measurement based on mean square error.

### 2.5.1 Distortion bound

The representation of the facial mimic is affected by the number of dimensions that are used for the representation but it is also affected by the mean value for each pixel position. The mean image is subtracted from all video frames before they are projected on the principal components.

The signal can be modelled without any loss if all linearly independent basis functions and the mean values are used for representation (equation 2.12).

\[
X = X_{\text{mean}} + \sum_{n=1}^{N} \alpha_n^x \Phi_n = X_{\text{mean}} + \Phi \alpha^x
\]  

(2.12)

where

\[
\Phi = [\Phi_1...\Phi_N]
\]  

(2.13)

and

\[
\alpha^x = [\alpha_1^x...\alpha_N^x]^T
\]  

(2.14)

The values in the vector \(\alpha^x\) are called projection coefficients. If less features (\(M<N\)) are used for representation, the signal can still be approximated but then with an error. This error is:

\[
mse(\text{opt}) = \sum_{i=M+1}^{N} \lambda_i
\]  

(2.15)

Fukunaga explains how the mean square error can be calculated for a high-dimensional source when the mean has been subtracted [30]. We have extended this calculation to include the mean image.

A mean square error bound for the number of Eigenvectors \(\Phi\) that are used for reconstruction is calculated from equation 2.15 by varying the number \(M\) of...
projection coefficients $\alpha^x$ that are used for reconstruction. This is the distortion bound for representing the signal with the selected number of Eigenvectors.

The distortion measurement in equation 2.15 is measured as mean square error. From the mean square error it is possible to calculate the corresponding PSNR.

Table 2.2 presents the average PSNR for all 10 facial mimic video sequences when different number of Eigenvectors $\Phi$ are used for reconstruction. As few as approximately 10 Eigenvectors can model the facial mimic in an efficient way.

### 2.5.2 Rate-Distortion bound

For the distortion bound there was an exact representation of the coefficients $\alpha^x$. In any real implementation, the coefficients $\alpha^x$ should be compressed to enable a more efficient transmission. In most low bandwidth applications we are limited by the number of bits that we can use for transmission, and not by the quality that can be reached. Often the quality has to be maximized dependent on a given bitrate. When quantization is used for representation of any source the question is how to assign the bits to model the source in the most efficient way. A theoretical bound for this assignment can be calculated with a Rate-Distortion function. Rate-Distortion functions refer to the minimum rate for representing a source at a given distortion, or the lowest possible distortion at a given rate.

A high-dimensional source represented through PCA by independent principal components can be assumed to consist of several independent Gaussian variables. For this kind of source the rate distortion function can be calculated through reverse "water-filling" where the rate and distortion is controlled by the variable $\gamma$ [31]. The variable $\gamma$ controls how many principal components that are used for representation of the source. An example of reverse "water-filling" when six principal components are used is shown in figure 2.6. Only the principal components who have a variance higher than $\gamma$ is represented with bits. All others are ignored.

The rate distortion function controlled by $\gamma$ is given by

$$R(D) = \sum_{i=1}^{M} \frac{1}{2} \log \frac{\sigma_i^2}{D_i}$$

(2.16)

where

$$D_i = \begin{cases} \gamma, & \text{if } \gamma < \sigma_i^2, \\ \sigma_i^2, & \text{if } \gamma \geq \sigma_i^2, \end{cases}$$

(2.17)
Figure 2.6: Reverse water-filling for independent Gaussian principal components. Only the components who have a variance larger then $\gamma$ is allocated bits in the quantization process.

Figure 2.7: Mean Rate-distortion bound for all 10 video sequences of facial mimic.

where $\gamma$ is chosen so that $\sum_{i=1}^{M} D_i = D$

This distortion $D$ is measured in mean square error. From the mean square error we calculate PSNR. The mean results for all videos are shown in figure 2.7.

2.5.3 Distortion measurement comparison

The two theoretical bounds are bounds for the same facial mimic video sequences but they are still different. The first bound describes the reconstruction quality that is achievable when a number of Eigenimages $\Phi$ are used to describe the facial mimic. The number of bits for representing these coefficients can be considered to be unlimited. The second bound describes the reconstruction quality when an optimal bit assignment is used between the Eigenimages to represent the facial mimic. If unlimited bits are used in the second bound it is equal to the first bound. The difference between the two bounds is:
where $M$ is the number of Eigenvectors used for reconstruction and $\gamma$ is the level from equation 2.17.

If the number of bits used for quantization of the coefficients is increased the gap between the two bounds is reduced. If an unlimited number of bits is used for quantization, i.e., exact representation of the coefficients is used, the gap is eliminated.

More information about the theoretical bounds is found in paper II.
2.6 Compression of the Eigenimages

The Eigenspace takes up a lot of space, originally an Eigenspace of 10 Eigenimages need 5 MB for storage. After chroma subsampling the storage need is reduced to 2.5 MB. The storage need is actually of less importance than the transmission cost since this scheme is aimed at low capacity networks. For example, it would take more than 35 minutes to transmit 2.5 MB over a GSM network, and approximately 3 minutes over a GPRS network. Therefore the Eigenimages must be compressed if they should be transmitted from the encoder to the decoder in a convenient way. Since the Eigenimages are images of faces the straightforward way is to compress them with JPEG encoding [32,33], which is the standard encoding for natural images. The mean image can also be compressed through JPEG compression since it has to be present at both encoder and decoder for the coding scheme to work. The projection coefficients are a small amount of data and can be compressed sufficiently through quantization. How will this compression affect the reconstruction quality of the coding scheme?

To make the JPEG-compression more efficient the values in the Eigenspace are transformed to integer values through quantization. The integer values are JPEG-encoded while the quantization reconstruction levels and boundaries are stored. The JPEG-encoded integer values are used with the reconstruction levels to inverse quantize the image and construct the resulting image.

1. Quantization of the Eigenspace. The quantization values are stored in an image. The reconstruction values are stored without loss.

2. JPEG-encoding of the quantization values.

3. Inverse quantization mapping of the JPEG-encoded values with the quantization reconstruction values.

2.6.1 Quantization - uniform or pdf-optimized?

The quantization for the Eigenspace and mean image can be performed in many different ways. We have examined the use of probability density function (pdf) optimized quantization and uniform quantization [34]. A pdf-optimized quantization is more likely to yield a quantization with good performance. The breakpoints and reconstruction levels of the pdf-optimized quantizer follow the distribution of the data and the step size is different for each step. This requires that information about each step size and breakpoint is transmitted to the decoder as well as the quantized data. A uniform quantization has the same step size for all steps and it is enough to send the starting value, number of steps and step size to the decoder.

We find that when approximately 7 bits are used for quantization there is no significant difference between uniform and pdf-optimized quantization. A quantization with 8 bits does not result in a large error between the quantized and original Eigenspace, for both uniform and pdf-optimized quantization. 8 bits are
equal to 1 Byte and is convenient to use for digital media since 1 Byte is the standard size.

The projection coefficients are also compressed through uniform quantization. Uniform quantization is sufficient compression for the coefficients. They are compressed 4 times, from 32 to 8 bits and still produce an exact representation of the Eigenspace.

2.6.2 JPEG compression

The Eigenspace is represented by integer values after quantization. Since we have chosen to use a quantization resolution of 8 bits (section 2.6.1) the values range between 0 and 255. The quantized Eigenimages are compressed with JPEG-encoding and the compressed, quantized Eigenimages are then used with the quantization reconstruction values to reconstruct the Eigenimages. The quantization reconstruction values are stored without compression since it is enough to store the starting value and the step size when we are using uniform quantization. This is only two floating point values, 64 bits total.

2.6.3 Compression of the mean image

The mean image has the same size as the Eigenimages and can be compressed in the same manner. The mean image has a large effect on the reconstruction result (equation 2.9). This should mean that degradation of the mean image quality will degrade the reconstruction quality more than if an Eigenimage is degraded equally much. The mean image is however subtracted before the Eigenspace is created; before the Eigenimages become orthogonal. When a compressed mean image is used the Eigenspace is still orthogonal, and this means that degradation of the mean image has less effect on the reconstruction quality than degradation of Eigenimages. The mean square error between the original and reconstructed image is almost unchanged when the mean image is compressed more heavily.

It is possible to choose almost any compression ratio and still reach a good reconstruction result. Compression to 6 kB is sufficient for our needs and further compression slightly increases the error. This is equivalent to using a JPEG quality factor of 25.

2.6.4 Loss of orthogonality

The principal components in the original, uncompressed Eigenspace are orthogonal, meaning that they follow

$$\Phi_i^T \Phi_j = \begin{cases} 1 & \text{for } i = j \\ 0 & \text{for } i \neq j \end{cases} \quad (2.19)$$

The orthogonality between the principal components ensures that the information in the different components is independent of each other. This increases
the energy compactness since no information is described in more than one principal component. When the Eigenspace is quantized the orthogonality is affected and the efficiency of the coding is reduced. We examine how large the loss of orthogonality is and how it affects the coding performance. The loss of orthogonality is measured as the average of the sum of the inner product of all principal components that should be orthogonal, i.e., $\Phi_i^T\Phi_j=0$.

The loss of orthogonality through compression of the Eigenspace can be regained. We have examined two different methods to ensure orthogonality between the Eigenimages:

- Least-square calculation of projection coefficients
- Re-orthogonalization of the Eigenspace through modified Gram-Schmidt projection

With least-square calculation of projection coefficients the coefficients are calculated with the aid of a matrix that is dependent on the Eigenimages. This matrix can be calculated once and used for encoding of several video frames. Use of Gram-Schmidt projection [35] means that a compressed Eigenspace is transformed into an Eigenspace that is orthogonal again. No storage space is saved but the Eigenspace can be compressed for transmission, which is extremely important. We actually make use of modified Gram-Schmidt projection since this method is more stable than regular Gram-Schmidt projection and performs well [36].

2.6.5 Compression methods

We have compared five different methods for handling compressed Eigenspaces and mean images. As a reference we use original Eigenspace and mean image for encoding and decoding of video sequences. The different methods that we compare are noted in table 2.3.

<table>
<thead>
<tr>
<th>Encoder</th>
<th>Decoder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eigenspace</td>
<td>Mean image</td>
</tr>
<tr>
<td>1</td>
<td>Original</td>
</tr>
<tr>
<td>2</td>
<td>Original</td>
</tr>
<tr>
<td>3</td>
<td>Compressed</td>
</tr>
<tr>
<td>4</td>
<td>Least-Square</td>
</tr>
<tr>
<td>5</td>
<td>Gram-Schmidt</td>
</tr>
</tbody>
</table>

Table 2.3: The five different methods that are evaluated.
calculation of the coefficients and use of Gram-Schmidt modified Eigenspaces increase the storage need again. Table 2.4 depicts the storage need each Eigenimage has when it is compressed.

From a storage point of view it is more efficient to use least-square calculation of the coefficients over Gram-Schmidt Eigenspace. The storage need is slightly higher at the encoder side but much lower at the decoder side. Compressed Eigenspaces that do not use either re-orthogonalization technique has the lowest storage need of all methods.

The different methods also require different number of operations for encoding and decoding and will have different computational complexity. The reference coding requires the least number of operations. When the mean image and Eigenspace are compressed they must be inverse quantized before encoding and decoding of frames can be performed. This procedure increases the complexity for both the encoder and decoder. Least-square calculation increases the complexity of the encoder side but the decoder side is unchanged. When a Gram-Schmidt re-orthogonalized Eigenspace is used there is no change in complexity for actual encoding and decoding. The increased complexity comes from performing modified Gram-Schmidt projection on both the encoder and decoder side.

The coding schemes are awarded points between 1 and 3 for their reconstruction quality, orthogonality loss, complexity and encoding need. 3 points mean that the coding scheme have small storage need, low loss of orthogonality, low computational complexity or high reconstruction quality. 2 points is equal to medium need and 1 point is the opposite of 3 point. The comparison is visualized in figure 2.8.

The loss of orthogonality is not an important factor except when very high compression is used. In figure 2.9 we present a comparison of the different methods. Along the horizontal axis we present the compressed size of the Eigenimages. The left side of the y-axis depicts the reconstruction quality, measured in PSNR. The right side of the y-axis depicts the loss of orthogonality. The vertical line in the figure symbolizes the compression ratio that we have chosen to use. It is clear to see that the loss of orthogonality is not an important factor for the coding scheme. The loss of orthogonality only becomes large when the Eigenspace is compressed heavily but at this compression ratio the reconstruction of the video frames are not at a satisfying level. This means that there is no need to employ re-orthogonalization techniques to avoid the loss of orthogonality. The best method

<table>
<thead>
<tr>
<th>Method</th>
<th>Storage need [kB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>256</td>
</tr>
<tr>
<td>Quantized [8 bits]</td>
<td>64</td>
</tr>
<tr>
<td>JPEG-compressed</td>
<td>1-37</td>
</tr>
</tbody>
</table>

Table 2.4: Storage need for one Eigenimage.
Figure 2.8: Comparison of the different coding schemes.
to use is compressed Eigenspace and mean image at both encoder and decoder without any re-orthogonalization technique. This method provides high reconstruction result with low storage need and moderate computational complexity.

When the Eigenimages are compressed to approximately 8 kB and 10 Eigenimages are used for encoding and decoding these Eigenimages can be transmitted over a GPRS network in less than 6 seconds. On a GSM network it would take approximately 1 minute for this transmission. If the Eigenimages are compressed more than 30 times and the mean image is compressed more than 40 times the reconstruction results is on average only lowered 2 dB. The average reconstruction value is still above 34 dB, both measured in PSNR. A detailed description of the compression is found in paper III.
2.7 Re-use of Eigenspaces

If the Eigenspaces cannot be transmitted over a network in an efficient way there must be another way to allow practical usage of the Eigenspaces. So, if both the encoder and decoder have access to an Eigenspace used for a previous communication event, how can this Eigenspace be used to encode new video frames?

Suppose there are two video sequences recorded with the same hands-free equipment. They show the same person but they are recorded at different occasions and are different in viewing angle of the face, facial size and illumination of the face. The first video sequence $X$ can be used to extract an Eigenspace $\Phi$ and mean image $X_{\text{mean}}$. The second video $Y$ can then be encoded with the Eigenspace $\Phi$ and mean image $X_{\text{mean}}$ extracted from $X$.

Encoding of $Y$ with $\Phi$ and $X_{\text{mean}}$ is performed similar to the encoding of $X$, as we have described before. Each video frame of the new video sequence $Y$ is treated as a column vector and YUV compressed. The mean image $X_{\text{mean}}$ is subtracted from the video frame and the video frame is projected onto the Eigenspace $\Phi$, yielding a number of projection coefficients. These projection coefficients are denoted $\alpha^y$ since they come from projection of video sequence $Y$.

$$
\alpha^y_m = Y^T \Phi_m \quad (2.20)
$$

The projection coefficients $\alpha^y$ is multiplied with the Eigenspace $\Phi$ and the mean image $X_{\text{mean}}$ is added to reconstruct the image $\hat{Y}$.

$$
\hat{Y} = X_{\text{mean}} + \sum_{m=1}^{M} \alpha^y_m \Phi_m \quad (2.21)
$$

The resulting video $\hat{Y}$ will resemble $X$ since it is the mean image and Eigen-images extracted from this video sequence that are used for reconstruction.

The result for this procedure is quite poor. The facial features are not positioned at the same pixel position for $X$ and $Y$. When the mean image $X_{\text{mean}}$ is subtracted from $Y$ the result is a image that actually isn’t a image of a face. When this image is projected on the Eigenspace $\Phi$ the coefficients $\alpha^y$ are optimized to reconstruct the image which isn’t a face. The lighting and shading are also different between $X$ and $Y$. Even if the facial features are positioned on the same pixel position they may have quite different pixel intensity. The result is a reconstructed video sequence $\hat{Y}$ with blurred frames and frames that are not correct facial images. Table 2.5 show the confusion matrix for original and decoded emotions. The numbers are presented in percentage. The confusion matrix show how the emotions in the original video sequence is reconstructed. The "mixed" column means that there is no clear reconstructed emotion or that the reconstructed image is blurred.
Orig. em.
neutral happy sad surprise anger fear disgust mixed
neutral 100
happy 65
sad 5
surprise 35
anger 65
fear 75
disgust 10

Table 2.5: Confusion matrix for mapping of emotions between original and reconstructed video, measured in percentage.

Figure 2.10: Reconstruction results when shift in position and intensity is applied.
- No shift
- Shift in intensity
- Position shift to the left
- Position shift up and to the left

2.7.1 Sensitivity tests
Since the result is not satisfactory we examine why. We examine the sensitivity of shift in position and pixel intensity between the videos sequences $X$ and $Y$. The pixel position of the facial features is much more important than the pixel intensity of the facial features. This can be seen from figure 2.10, where the sensitivity of change in position and lightning is shown in the same figure.

2.7.2 Adjustment of different pixel intensities
The coding scheme is not as sensitive when it comes to changes in pixel intensity as it is for changes in position, but it is still moderately sensitive. Since the mean image $X_{\text{mean}}$ is subtracted from all the new video frames $Y_N$ before they are
projected on the Eigenspace $\Phi$ this might affect the projection coefficients and consequently the reconstruction result. The mean intensity of the mean image $X_{\text{mean}}$ and each new video frame $Y_N$ can be calculated. The difference between them can then be adjusted for. This ensures that the video frames in $Y$ have the same mean intensity as the mean image $X_{\text{mean}}$, improving the possibility that each facial feature will have the same intensity in both videos. Figure 2.11 shows the effect of adjusting for a shift in mean. It is apparent from the figure that this has no effect for the change of pattern of the projection coefficients; the pattern is the same but slightly shifted. This change is so moderate that we consider it to be negligible.

2.7.3 Normalization of feature position

Different pixel positions between the frames can be solved by transforming either the Eigenimages $\Phi$ or the new video sequence $Y$. Transformation of $\Phi$ will lead to a reduction in orthogonality between the Eigenimages $\Phi$ and reduced encoding efficiency. Therefore it is much more efficient to transform $Y$ instead. After $Y$ has been transformed the pixel position for each facial feature will be the same in $Y$ as in $X$ and $\Phi$. One way to perform this transformation is to use affine transform.

Affine transformation can be performed through multiplication with a rotation matrix $A$ and addition of a translation vector $b$.

$$\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = A \begin{bmatrix} x \\ y \end{bmatrix} + b \quad (2.22)$$

where matrix $A$ refers to a rotation and/or scaling and vector $b$ refers to a translation.

Pixel positions from the eye corners, nostrils and mouth edges are collected for
The position of these features is extracted manually. With the corresponding positions for both video sequences it is possible to calculate the rotation matrix $A$ and translation vector $b$. The new video sequence $Y$ is transformed with rotation matrix $A$ and translation vector $b$ according to equation 2.22. The transformed video sequence $Y$ is then ready to be encoded with Eigenspace $\Phi$ and mean image $X_{\text{mean}}$. The result for this encoding is better than the result without affine transformation. Still there are some errors visible in the individual frames. Table 2.6 show how emotions in $Y$ are decoded in $\hat{Y}$. The table shows that the mapping of emotions are much better when affine transformation is used.

It seems that normalization through affine transformation does not solve the problem of poor reconstruction quality entirely. There are still both visually bad frames and wrongfully mapped emotions from the original to reconstructed video. Both these problem can be solved by choosing coefficients that ensure natural looking video frames.

### 2.7.4 Selecting original coefficients

The problems of decoded video frames with incorrect visual appearance and emotions can be solved by selecting projection coefficients that ensure good looking frames. The coefficients from encoding the original video $X$ can be used for decoding of the new video $Y$. Through comparison of the Euclidean distance between the coefficients from $Y$ and $X$ the set of coefficients for $X$ that is closest can be chosen.

$$d_n = \sum_{i=1}^{K} \sqrt{(x_{n,1} - x_{Y,i})^2 + (y_{n,i} - y_{Y,i})^2}$$  \hspace{1cm} (2.23)$$

where $d_n$ is the Euclidian distance to frame $n$, $i$ is the feature-number, $x$ and $y$ are the positions of the coefficients and $K$ is the number of features in each frame.

<table>
<thead>
<tr>
<th>Orig. em.</th>
<th>neutral</th>
<th>happy</th>
<th>sad</th>
<th>surprise</th>
<th>anger</th>
<th>fear</th>
<th>disgust</th>
<th>mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>neutral</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>happy</td>
<td></td>
<td>95</td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sad</td>
<td></td>
<td></td>
<td>75</td>
<td></td>
<td>25</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>surprise</td>
<td></td>
<td></td>
<td></td>
<td>80</td>
<td></td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>anger</td>
<td></td>
<td></td>
<td></td>
<td>85</td>
<td></td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fear</td>
<td></td>
<td></td>
<td></td>
<td>60</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>disgust</td>
<td></td>
<td></td>
<td></td>
<td>60</td>
<td></td>
<td>40</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2.6: Confusion matrix for mapping of emotions between original and reconstructed video when affine transformation has been applied (measured in percentage).
Decoding will then be performed with the coefficients \( \alpha_x \) from the original frames. Each frame is therefore assured of individually better appearance but the risk is that the transition between the frames will be very jerky and unnatural. The result is good looking frames but the emotions are still not correctly translated from the original to the reconstructed video.

When we plot the distributions of the coefficients along the first two Eigen-images it is easy to understand why. The projection coefficients have a similar pattern but they are also far from each other. This also answers the question why the emotions in the decoded videos do not match the emotions in the original video. The position of emotions for \( \alpha_x \) and \( \alpha_y \) are not very similar (figure 2.12). The positions for \( \alpha_y \) are very centered and they are hardly discriminated from each other.

By comparing the Euclidian distance and choosing the closest match it is highly probable that the incorrect emotion will be chosen. This can be improved through normalization of the projection coefficients. The projection coefficients along each Eigenvector are divided with its norm so that the new norm for the vector is 1. The mean is also shifted to 0. The normalized coefficients provide much more similar patterns between the coefficients for \( X \) and \( Y \). Still there is a clear differ-
The conclusion is that it is not enough to use Euclidian distance to compare the coefficients, not even when the coefficients are normalized. In [37] we show how video sequences can be given a natural and smooth transition between frames. By using dynamic programming and locally linear embedding we remove jitter and jerky transitions in a video sequence. A similar approach can be used here. The difference in pattern between $\alpha^x$ and $\alpha^y$ must be improved by a better normalization technique than affine transformation.

Paper IV handles the re-use of Eigenspaces.
The principal component video coding allows the use of high quality video over low bitrate networks. The high quality means high framerate and high spatial resolution, both being important for communication. Through a user test we examine how important spatial resolution and framerate is in practice. Since the principal component video coding is not fully developed we use the h.261-codec to evaluate the two factors.

The usual assumption for video is that more is better, up to a certain level. Higher bitrates, framerates etc. will improve the user experience. This is held to be true both in streaming video [38] and conversational multimedia (CMM). In CMM, and at least for certain tasks, having video is better than not having it [39]. It has also been shown that a certain framerate is required to perform many conversation related tasks such as lip-reading [1]. In general, there seems to be a threshold at around 15 or 16 frames per second (fps) [40]. However, there are some indications that higher bitrates etc. are not always better. In one experiment, Wikstrand and Sun found that subjects judge low-bandwidth video as better and high-bandwidth video as worse when they were given harder secondary visual tracking tasks [41]. Other researchers have found that higher framerates were worse both subjectively and objectively in terms of task performance in computer supported cooperative work [42].

The purpose of the user tests was to examine the effects of differential video quality on the user in a CMM-mediated lie detection situation. We chose to use the card game "Bluffstopp", which is prone both to lying and lie-detection.

The goal in "Bluffstopp" is to get rid of all your cards as fast as possible. If a player cannot play a higher card he/she can pass. When a player lays down a card he/she will also say which card that was laid. Since the cards are played face down it is possible to lie. If a player suspects that another player lied or cheated he/she says "bluff" and looks at the top card in the pile. If it was a bluff, the bluffer picks up three cards from the deck, or if the deck is empty, from the opponents. If it was not a bluff, the caller picks up three cards in the same way.

A computerized, networked version of the game was developed. Players inter-
acted with the game by clicking on cards. The users interacted through a video stream of all the participants’ faces and through audio.

Two different framerates were used, 10 fps and 20 fps to bracket the limit of 15 fps. The resolutions that were used were QCIF (160x120) and CIF (320x240) since they are widely used on video conference applications.

Measures were gathered from the participants in two categories using self-reports; media quality and usability. Media quality was measured by asking users to rate the experience compared to playing face to face, the framerate, the frame quality, the overall video quality and the audio quality.

First, the game of Bluffstopp was introduced to the participants and they play the game with a real deck, sitting around a table. Then the participants fill in the first part of the evaluation form. Each participant is then placed in an office, in front of a workstation with a web camera and a headset. The web-camera is placed on the bottom right side of the computer screen for all users. After each of the four games the participants fill out the self-report form while the video parameters are changed. Framerate and spatial resolution are varied according to a Latin square design. A Latin square is an table filled with symbols in such a way that each symbol occurs exactly once in each row and exactly once in each column. The Latin square minimizes order effect from the test result.

The results show that the participants did not enjoy higher framerates, as expected. On the contrary, they enjoyed the lower framerates more. The first question that must be asked when faced with a result like this is if it reflects reality, i.e. that a lower framerate might actually be better in this application and that the effect is just not a fluke. If the results are a fluke then the problem might be that the framerate did not actually achieve its intended value or the framerate was too unstable. If the results on the other hand are genuine it should be possible to provide an explanation of them.

Through examination we found that the results are in fact genuine and that in this case higher system performance leads to lower perceived quality of service. The actual framerate is more fluctuating when the target framerate is increased from 10 to 20, but the mean is also highly increased. The actual framerate was always higher when a target framerate of 20 fps was used, compared to a target framerate of 10 fps. Watson et.al. found that if the framerate is sufficiently high the variance in framerate do not affect the task performance [43]. If the average framerate is 17 fps and the variance is high this reduces the task performance only if the framerate drops below 15 fps for a long time. In our experiment the framerate only drops below 15 fps for a quick period and on a few occasions. This leads us to the conclusion that the variance of the framerate shouldn’t degrade the user’s subjective experience of the high framerate video compared to the low framerate video.

If the movement in the videos would be large and most of the frames needed to be intra-coded for several adjacent frames the bitrate needed to encode this would be high. The lower subjective ratings for videos with higher framerates could then be explained with the fact that the quality of each individual frame actually was
lower because of block errors in the frames. We find that this is not the case; almost no block errors are visible in the videos, ruling out this theory.

The conclusion is that it is extremely important to consider secondary workloads and how users make use of video when experimental setups are designed. In this case it should be better with higher system performance but due to our experimental setup the video is only disturbing the users and degrade their experience. Watching the video becomes a secondary task and a so difficult task that it is a disturbing factor, instead of an improving factor.

The user test is described in detail in paper V.
Conclusions and contributions

The main contribution of this thesis is the introduction of a full-frame video coding scheme based on principal component analysis. The scheme can encode entire video frames to very low bitrates and reconstruct them with high spatial and visual quality. A large advantage for the full-frame coding compared to standard video coding is that the bitrate needed for transmission is not increased when higher spatial resolution is used. The encoding and decoding time is increased but the bitrate is not affected. For any standard video coding technique a higher spatial quality results in higher bitrate or lowered quality.

We have presented two theoretical representation bounds for human facial mimic, one rate-distortion bound and one distortion bound. These bounds can be useful in many fields that use facial modelling since they show how much information that is, at least, needed to represent facial mimic.

The principal components (Eigenspace) and the mean image used for encoding must be present at both the encoder and decoder to enable the use of encoding and decoding. Transmission of the Eigenspace will be performed over the network and must therefore be compressed to enable this transmission at a reasonable time. We have shown that it is possible to compress the Eigenspace and mean image so they can be transmitted in 6 seconds on a GPRS network without losing more than 2 dB in reconstruction quality. We have also shown that the projection coefficients can be compressed 4 times almost without any loss of quality.

That the loss of orthogonality within the Eigenspace is not an important factor when compression is applied has been shown. Methods that ensure orthogonality only increase the reconstruction quality moderately and at the same time increase the complexity and storage need for the encoder and decoder.

Existing Eigenspaces can be re-used to encode new video sequences of the same person. The largest problem for re-use is normalization. It is crucial that the facial features are positioned at the same pixel positions in both the Eigenspace and the new video.

Through a user test it has been shown that it is extremely important to consider secondary workloads and how users make use of video when experimental setups
CONCLUSIONS AND CONTRIBUTIONS

are designed. When watching the video becomes a secondary task it is sometimes a so difficult task that video becomes a disturbing factor, instead of an improving factor.
Bibliography


52  BIBLIOGRAPHY


face-to-face teleconference,” in DCC ’05: Proceedings of the Data Compress-


[24] G.J.C. van der Horst and M.A. Bouman, “Spatiotemporal chromaticity dis-
1488, 1969.


[27] Ulrik Söderström and Haibo Li, “Full-frame video coding for facial video
sequences based on principal component analysis,” in Proceedings of Irish
MachineVision and Image Processing Conference 2005 (IMVIP 2005), Au-

[28] Ulrik Söderström and Haibo Li, “Very low bitrate full-frame facial video
coding based on principal component analysis,” Signal and Image Processing
Conference (SIP’05), August, 15-17 2005, online: www.medialab.tfe.umu.se.

[29] J. Rasmusson et al., “Multimedia in mobile phones - the ongoing revolution,”


[31] Thomas M. Cover and Joy A. Thomas, Elements of Information Theory,
Wiley Series in Telecommunications. John Wiley & Sons, New York, NY,
USA, 1991.


[33] W.B. Pennebaker and J.L. Mitchell, JPEG Still Image Data Compression

[34] Khalid Sayood, Introduction to data compression, Morgan Kaufmann Pub-


