**University of Karlsruhe**
**Faculty of Computer Science**

Interactive System Labs (ISL)
Prof. Dr. Alexander Waibel


# Video-based Face Recognition Using Local Appearance-based Models

**Diploma thesis**

by

**Johannes Stallkamp**

NOVEMBER 2006

Advisors:

Prof. Dr. Alexander Waibel
Dr.-Ing. Rainer Stiefelhagen
MSc. Hazım Kemal Ekenel

# Statement of Authorship

I hereby declare that this thesis is my own original work which I created without illegitimate help by others, that I have not used any other sources or resources than the ones indicated and that due acknowledgement is given where reference is made to the work of others.

Karlsruhe, November 2006 ..... Johannes Stallkamp .....

# Contents

# List of Figures

# List of Tables

# Abbreviations

| | |
|---|---|
| ROC | Receiver operating characteristc |
| DCT | Discrete cosine transform |
| EM | Expectation-maximization |
| fps | Frames per second |
| FRS | Face recognition system |
| KLT | Karhunen-Loève transform |
| normalized-rg | Normalized-red-green color space |
| PCA | Principal component analysis |
| RGB | Red-green-blue color model |
| ROI | Region of interest |
| SVM | Support vector machine |

**Evaluation Models**

| | |
|---|---|
| DT2ND | Distance-to-second-closest, a weighting scheme |
| DTM | Distance-to-model, a weighting scheme |
| Frame-GMM | Frame-based evaluation of the GMM approach |
| Frame-KNN | Frame-based evaluation of the KNN approach |
| GMM | Gaussian mixture model |
| KNN | K-nearest neighbors |
| Smooth-GMM | Video-based evaluation of the GMM approach using identity smoothing |

| | |
|---|---|
| Video-GMM | Video-based evaluation of the GMM approach using posterior probabilities |
| Video-KNN | Video-based evaluation of the KNN approach using simple sum |
| Weighted-KNN | Video-based evaluation of the KNN approach using weighted sum |

**Performance measures**

| | |
|---|---|
| CCR | Correct classification rate |
| CDR | Correct detection rate |
| EER | Equal error rate |
| FAR | False acceptance rate |
| FCR | False classification rate |
| FDR | False detection rate |
| FRR | False rejection rate |

# 1 Introduction

In human-human interaction, it is advantageous to recognize one's counterpart because this is the key to access available background knowledge about this person, originating from previous encounters. It can comprise a person's preferences and dislikes, way of thinking, credibility or reliability. The knowledge about each other affects how people treat each other and to what extent they are able to estimate each other's intentions. A barkeeper who serves the customer's favorite drink as soon as he sits down, a shop which chalks up for its regular customers or a friend who is able to greet one by name are simple examples. Many more are imaginable as everybody experiences similar situations every day.

Ordinary computer systems are not able to do this. While it is easy for them to store background knowledge, the user has to communicate his identity actively in order to allow the computer to access this knowledge. Biometric systems endow computers with the required perceptual abilities to recognize the user autonomously. In addition to widely used and developed security applications like video surveillance and access control, this opens up many new possibilities. Smart cars, that recognize legitimate drivers and adjust the settings of seat, mirror and radio station accordingly, are as imaginable as smart homes, that route phone calls automatically to the current location of the desired recipient. When implementing such systems, it is crucial that the complexity the user is confronted with does not practically destroy the benefits and comfort the system was designed to provide.

The goal of building smart environments is to give a benefit to the user without restricting him or her. The person must be allowed to move freely and naturally without the need to express certain behavior patterns to suit the system. Computer systems in such environments *"have to fit naturally within the pattern of normal human interactions"* (Pentland and Choudhury, 2000, p. 55). Otherwise, it will be restricted to a technology-affine group of people since general acceptance will be low.

Identification of a user can be achieved by exploiting different cues. Depending on the nature of the cue, more or less interaction on part of the user is necessary. For fingerprint identification, the user needs to touch a sensor, for speech recognition, the user needs to provide speech samples by uttering one or more words. The third popular approach to biometric identification is the use of facial features. Alongside speech, it is a very natural approach and mimics human recognition. Even though current face identification systems often still require interaction — look at the camera — or at least a certain behavior pattern — do not tilt your head — as well, the nature of the cue inherently allows for unobtrusive, interaction-free recognition, as the visibility of the

face does not need any specific action. On the contrary, specific action is necessary to hide it.

This work is part of *CHIL - Computers in the Human Interaction Loop*, an Integrated Project under the Sixth Framework Programme of the European Commission (Waibel et al., 2004). The goal of this project is to support human-human interaction by creating perceptual computer systems that release the human of the need to explicitly interact with the system in order to use a certain service. Requiring minimum human attention to provide maximum service, these systems are designed to be helpful assistants that allow humans to concentrate on interactions among themselves. To build such systems and offer personalized and context-dependent services, it is crucial to know the identity of the participants. This work presents an approach to unobtrusive person identification by face recognition together with the arising challenges.

In the following, this chapter will first explain the objectives for the development of this system in Section 1.1 before an overview of already existing face recognition systems will be given in Section 1.2. Subsequently, the scenario in which the developed system is evaluated is introduced in Section 1.3. The design of the system is outlined in Section 1.4. Finally, Section 1.5 summarizes the scientific contribution made by this work.

## 1.1 Motivation

The goal of this work is to build a real-time capable face recognition system (FRS) for unconstrained environments. It is supposed to handle robustly real-life situations with all the challenges they bring along that make the task harder.

The key difficulties in real-world applications arise from

**Unobtrusive Recognition** The face recognition system is supposed to work in the background in an unobtrusive manner. The people to be recognized are not to be disturbed or interrupted in their actions by the presence of the computer vision system. While the inconvenience emerging from a necessity to interact with the system, e. g., to look straight into the camera for some time, would still be acceptable for security scenarios in areas with restricted access, it is annoying in environments like smart homes, where the system might be passed several times over a short period of time. A major share of the following problems arise from this central goal and challenge.

**Changing illumination** While it is already laborious and energy-consuming to establish constant illumination in window-less rooms, it gets practically impossible if daylight is involved. Daylight leads to very different illumination depending on the time of day, time of year and weather conditions. However, in spite of this hardly controllable natural influences, even the artificial light sources are withdrawn from the system's control if unobtrusive recognition as postulated above

is to be implemented. Since the user, i. e., the person to be recognized, is not supposed to be restrained by the system, he is free to switch on and off any light sources that might be available. This leads to a wide variety of illumination configurations in terms of light intensity, direction and even color.

**Varying appearance** Every recognition system that is deployed in a real-world setting is necessarily trained on a very small amount of data compared to the quantity encountered during a long period of operation. In terms of a face recognition system, the problem arises that people can change their appearance on a daily basis. They may or may not wear different kinds of glasses, or facial hair can change the aspect of large areas of the face. Parts of the face may be occluded by clothing or accessories — like sunglasses, scarves, hats or caps — or simply by the person's hand because he or she needs to yawn or cough. Generally, the facial expressions change when the person is laughing, talking, shouting or crying. Since it is impossible to capture every kind of variation within the training data, the system itself needs to be designed general enough to make up for them.

**Pose variations** In addition to not to look the same all the time, people do not stiffly hold their head in a fixed position, facing the camera upright, either. Following the postulated unobtrusiveness, people are free to move. They might turn away from the camera or rotate the head arbitrarily.

Numerous face recognition systems have been developed over the past years. Starting with recognition of frontal faces under fixed conditions (Belhumeur et al., 1997; Turk and Pentland, 1991), more recent systems approached the above mentioned difficulties arising from real-world data as well. Unfortunately, these systems generally work under fully controlled conditions (Georghiades et al., 2001; Phillips et al., 2003; Wang et al., 2004). The subject is usually aware of the camera and actively cooperating with the FRS, displaying a fixed set of different facial expressions if necessary and rotate the head at certain angles. If varying lighting conditions apply, the single light source is positioned as well at fixed angles like left, right and in front of the person. In occluded images, usually the same part of all faces is covered in the same or a similar manner (e. g., Martinez and Benavente, 1998). As detailed above, this does not reflect a realistic situation. The FRS developed in this work takes the combination of these issues into account in order to process real-world data.

## 1.2 Previous work

This section gives an overview of previously developed face recognition systems. The first part concentrates on methods for face detection, which is a vital step preceding recognition. Afterwards, different approaches to face recognition are presented, where the focus is put on video-based methods.

No numbers are reported for the results, as testing conditions and evaluation concepts vary largely, so that the results would not be comparable in many cases. As detailed explanation of the experimental setup is beyond the scope of this work, the reader is referred to the original publications.

### 1.2.1 Face detection

Face detection has attracted a lot of research effort in the past, as it is the key to more sophisticated systems that allow face tracking or recognition. A large variety of different methods has been developed which can be divided into two categories: feature-based and image-based (Hjelmas and Lee, 2001).

Image-based approaches directly evaluate pixel values in some way, e. g., by feeding them into a neural network. Rowley et al. (1998) used a set of retinally connected neural networks to classify input samples as face or non-face. In order to compensate for translation, a window is shifted pixelwise over the image and each location is evaluated. To account for scale variations, this approach is applied to a pyramid of stepwise downsized versions of the input image. Both the sliding window and the scaling lead to a high computational effort.

As this is generally a concern for image-based approaches, it applies as well to the work of Menser and Müller (1999), which is based on the eigenfaces approach introduced by Turk and Pentland (1991) for face detection and recognition (see below, Section 1.2.2). Every subwindow is projected onto the face space by means of principal component analysis (PCA). Instead of processing the original image, Menser and Müller use a skin probability image to increase robustness towards complex backgrounds and illumination. In addition to the residual reconstruction error, the distance to the mean face in the face space is used as a measure for "faceness" which increases the robustness in uniform backgrounds.

Feature-based methods exploit certain characteristics of a face. These can be low-level features like edges, skin color or motion, the position of eyes, nose and mouth or the geometric relations between them, for example.

One representative of this class is the component-based approach by Heisele et al. (2001b), which is applied to synthetic face images derived from textured 3-dimensional head models. The system uses 15 support vector machines (SVM), one for each of the 14 components, and one to classify their geometric configuration. Each component is selected by growing a rectangular region around a seed point as long as the upper bound on the expected probability of error of the corresponding SVM decreases. This approach yields a discriminating set of components and is not restricted to faces. Heisele et al. report an increased performance over a holistic approach, because the single components are less affected by in- and out-of-plane rotations.

Papageorgiou et al. (1998) introduced a general framework for object detection based on an overcomplete dictionary of three types of Haar wavelets. These are used to compute differences between image intensities in neighboring regions of the image. A sta-

tistical analysis of the normalized resulting values allowed them to reduce the number of features, that are necessary for a successful detection, by nearly two orders of magnitude. Classification of pre-aligned faces is done with SVMs. During training, false detections in non-object samples are iteratively used to refine the model until a satisfying decision surface is found. Selection of these problematic samples for training overcomes the problem that the non-object class is extremely large and heterogeneous compared to the object class. A similar experiment was conducted for pedestrian detection in which an extension incorporating motion information was able to improve the results.

The idea of using Haar basis functions to compute intensity differences was taken up by Viola and Jones (2001) to build a real-time capable face detection system. They extended the set of Haar-like features and introduced a data structure called *integral image*, which allows efficient computation. Thus, faces of different sizes can be detected by scaling the features instead of the image, as done by Papageorgiou et al., which results in a speed-up. Viola and Jones used a modified AdaBoost algorithm to select relevant features. These are arranged in a classifier cascade to further speed up processing. In analogy to the training method used by Papageorgiou et al., Viola and Jones use false detections of one classifier to train its successor in the cascade. As this approach is widely employed throughout this work, a more detailed explanation can be found in Section 2.1. Later, Jones and Viola (2003) added a decision tree for pose estimation so that an appropriate cascade could be selected to allow for multi-view face detection.

Since a detailed survey of the wide variety of face detection techniques is beyond the scope of this work, the interested reader is referred to the work of Hjelmas and Lee (2001) and Yang et al. (2002).

### 1.2.2 Face recognition

Over the last years, video-based face recognition approaches have come more and more into focus of research. They can help to overcome the difficulties arising from changing illumination, pose, expression and temporary occlusion which cannot be resolved by frames-based approaches.

The most popular holistic frame-based approach to face recognition is called *eigenfaces* and was introduced by Turk and Pentland (1991). It describes faces as a linear combination of principal components, i. e., the eigenvectors of the covariance matrix of the training set that are associated to the largest eigenvalues. In face recognition context, these are generally referred to as eigenfaces. The distance of a test sample from feature space (DFFS, Moghaddam and Pentland, 1995), i. e., the energy difference between the sample and its projection, can be used to determine whether it represents a face. If so, the class with the smallest distance is selected under the restriction that this distance is not too large. Otherwise, the sample is classified as unknown. This property can be used to implement automatic learning of new faces. The holistic approach, how-

ever, is very sensitive to occlusion and other local variations because they can affect the whole feature vector (Ekenel and Stiefelhagen, 2006).

Ekenel and Stiefelhagen (2005) choose a local appearance-based approach in order to be independent from the detection of salient facial features and less affected by local variations. This is realized using the discrete cosine transform which allows efficient dimensionality reduction and computation. In order to account for local variations in the image, the transform is applied to local blocks. The transformation results of all blocks are fused on feature and decision level, where the former is found to perform slightly superior. The local model outperforms the global version of the approach as well as other holistic models like principal component (PCA), independent component (ICA) and linear discriminant(LDA) analysis. Further details are given in Sections 2.3, 3.3 and 3.4 because this approach is fundamental in this work.

Zhou et al. (2003) are concerned with face recognition in video sequences rather than still images. Instead of applying the common two-step approach of tracking followed by recognition, they develop a single-step approach which is able to do both simultaneously. To achieve this, the underlying probabilistic framework incorporates a motion equation to model a person's movement, an identity equation to model the development of this person's identity and an observation equation to relate motion and identity to each other. Sequence importance sampling (SIS) is used to propagate the joint posterior probability distribution of identity and motion over time. Zhou et al. present results in two categories. In one, the system is trained on one single still image per person, whereas in the second, an exemplar-based approach is used to train the system with video data.

Since head poses and facial expressions change continuously rather than discretely, Lee et al. (2003) represent the appearance of a person by the means of manifolds. Since a person's appearance manifold is non-linear and complex, it is divided into disjoint *pose manifolds* which are connected by transition probabilities. Applying PCA to exemplars, which are extracted from training videos with k-means clustering, yields an affine plane which approximates the pose manifolds. The transition probabilities are learned from the temporal dependencies between pose manifolds in training sequences. A Bayesian probabilistic framework estimates the closest manifold to a given sequence of test samples. An iteratively computed weight mask allows to handle partial occlusions.

In order to model person-specific appearance and dynamics, Liu and Chen (2003) train individual *hidden Markov models (HMM)* on eigenface image sequences. During classification, the identity of a person is determined in maximum-likelihood manner. If the likelihood difference between the top two candidates is larger than a certain threshold, the sequence is used to adapt the best candidate's model accordingly.

Arandjelovic and Zisserman (2005) developed a system to retrieve faces in feature-length movies based on single or multiple query images. This implies a large variety of pose and illumination changes as well as complex background and partial occlusions. SVM-based mouth and eye detectors are used in conjunction with a gradient-based face

boundary detector to perform automatic face registration. To suppress coarse variations of ambient illumination, the registered face image is band-pass filtered, resulting in a so-called *signature image*. Classification is based on a modified Euclidean distance between the query's and film characters' signature images. The modification of the distance measure increases the robustness against partial occlusions.

Using feature films as input data as well, Sivic et al. (2005) create a person retrieval system. Single faces are represented with scale invariant (SIFT) descriptors of facial features. Face detections within the same shot are automatically grouped into *face-tracks*, which are represented by a histogram over the corresponding faces. A feature-based vector quantization allows to build meaningful histograms. A query consists of a single face selected in one of the scenes. The system automatically extends the query to all faces in that scene to compute the face-track. A chi-square goodness-of-fit test is then used to retrieve all matching face-tracks and, thus, faces.

Face recognition systems that are to be deployed in a real-life scenario, usually encounter the problem that they are confronted with unknown people. To reject those, Li and Wechsler (2005) make use of transduction, an inference method that derives the class of a test sample directly from a set of training samples, instead of trying to induce some generalizing classification function over the training set. To reject a test sample, its k-nearest neighbors are used to derive a distribution of credibility values for *false* classifications. Subsequently, the credibility of the test sample is computed by iteratively assigning it to every class in the k-neighborhood. If the highest achieved credibility does not exceed a certain level, defined by the previously computed distribution, the face is rejected as "unknown". Otherwise, it is classified accordingly.

As for face detection, a detailed survey of available face recognition techniques, especially of frame-based approaches, is beyond the scope of this work, the reader is referred to the work of Zhao et al. (2003).

## 1.3 Scenario

The face recognition system developed in this work is deployed at the entrance door to a seminar room. As can be seen in Figure 1.1, the camera is located opposite the door with a distance of several meters. Individuals are recognized when they enter the room. Depending on their intention, they turn sideways to get to the seminar area or collect a print-out, walk straight through to the next room or just stand in the door frame for some time before they leave. Different light sources (fluorescent lamps, daylight lamps and two smaller windows) cause varying illumination. The exact configuration is in full control of the people using the room and no specific lighting policy is enforced.

This setup accounts for the broad variety of changing conditions caused by real-world data. The camera position ensures unobtrusive recognition of people, as it allows them to enter the room as usual. Nevertheless, the direction of the camera enables the FRS to generally capture at least some frontal or close-to-frontal views of a person which

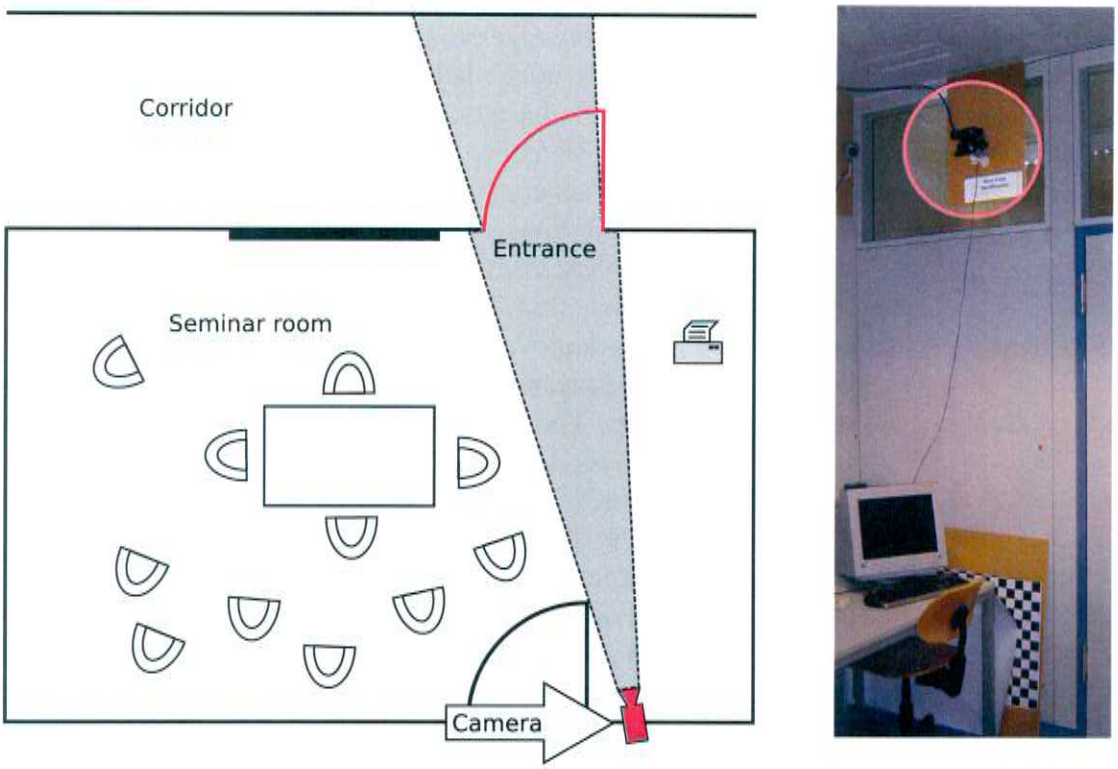



***Figure 1.1:*** *Layout of the seminar room. The camera is facing the door from a distance of several meters, capturing a part of the corridor as well.*

are necessary for successful recognition. Figure 1.2 shows some examples of the varying recording conditions encountered during the evaluation of the system. The system is currently restricted to detect and recognize single individuals rather than groups of people.

## 1.4 System overview

This section will give an overview of the system and the general ideas to solve the problems arising from real-world data. A detailed description of the methodology is given in Chapter 3.

As clarified above, the recognition of faces under real-life conditions grants the user many degrees of freedom, which in turn make the classification problem harder. In order to achieve a real-world deployable system, these difficulties need to be tackled. The solution developed in this work is characterized by two key techniques:

**Local appearance-based model** A local appearance-based model — in contrast to holistic approaches like eigenfaces as introduced by Turk and Pentland (1991) —

***Figure 1.2:*** *Exemplary recognition situations showing a variety of different lighting, pose and occlusion conditions. No individual explicitly looks into the camera.*

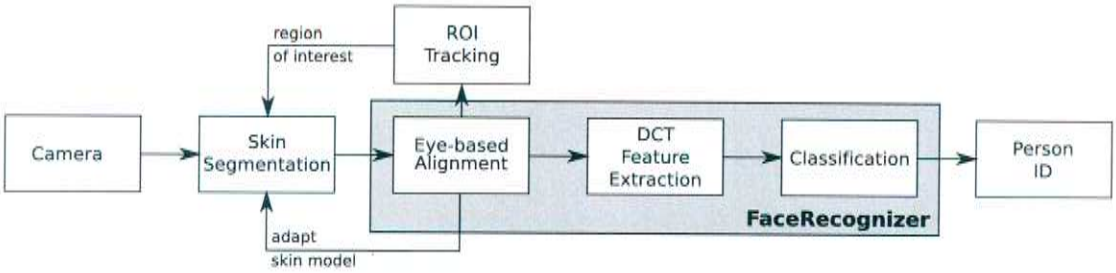



***Figure 1.3:*** *Overview of the recognition system.*

processes the image information locally in order to reduce impairments caused by occlusion and variations of illumination and expression. Based on research by Ekenel and Stiefelhagen (2005), local feature vectors are computed using the block-based discrete cosine transform. These local features are then fused to allow classification of the image. The locality of the model allows for individual processing of image regions, e. g., in terms of illumination normalization, and control of how much each block contributes to the final classification. The local model still takes spatial dependencies between blocks into account, but in contrast to component-based approaches, recognition using local appearance-based models does not rely on the detection of specific features prior to classification.

**Video-based recognition** Taking into account that a person stays in the camera's field of view for some time when entering the room, it is reasonable to use all available views of this person for recognition purposes instead of a single one, even if this single frame is considered to be the "best" one by some quality measure. The plus of data being available for evaluation is able to compensate for poor quality frames. It makes classification more robust because the combination of several frames can lead to confident decisions, even if every single frame is ambiguous. Additionally, it leads to better quality input data as video-based image processing allows to use tracking techniques. These can be employed to track registration-specific cues and, as a consequence, make alignment, and therefore feature extraction, more robust since in-plane pose variations can be handled.

To understand how these central ideas are integrated into the developed system, the process from data acquisition to classification is outlined in the following. An overview of this process is depicted in Figure 1.3.

In the given scenario, appearing faces are comparatively small with respect to the input image. To avoid unnecessary processing of non-relevant data, the image data needs to be reduced to "interesting" areas, i. e., areas which are likely to contain a face. The door scenario would allow to concentrate on the center region of the image, as a face should appear here when somebody enters the room. This approach, however, would be specific to this scenario. In a more general approach, skin color information

is used to determine so called *regions of interest (ROI)*. With an appropriate skin color model, these *face candidates* can be quickly detected. The reduced amount of data leaves more computational resources to the actual recognition task and allows real-time processing. In this work, an adaptive histogram-based skin color model, as explained in Section 3.1, is used to segment skin-colored areas in the image.

Subsequently, the detected face candidates are inspected whether they actually do represent a face or not. This is achieved based on the face detection framework introduced by Viola and Jones (2001) which uses Haar cascade classifiers. Using these classifiers, the ROIs are checked for a face as well as for two eyes. The eye positions are then tracked over time with a Kalman filter to compensate for failures of the eye detector. It is important to get robust estimates of the eye positions because these are exploited to extract and normalize the face image to a fixed orientation and scale (see Section 3.2).

If face candidates are confirmed by the alignment process, this information is used for subsequent frames in two ways. First, the color information of a face is used to adapt the skin color model to the current individual in order to improve skin segmentation in the next frames. Second, the location of the face is exploited to reduce the search area in the following frame to the region surrounding it. This is another step towards a real-time capable system.

The aligned face is divided into blocks to account for local feature extraction which is based on the *discrete cosine transform (DCT)*. DCT has been especially chosen for its compact data representation, data-independency and computational efficiency.

The resulting frame-based feature vectors are then classified using *k-nearest neighbors* and *Gaussian mixture models*. Several temporal fusion schemes that will be introduced in Chapter 3 are evaluated in Chapter 4 in order to assess their performance.

As criticized above, the major part of recent face recognition research is still based on data which is collected in a controlled environment, even if it already includes certain variations of the recording conditions. Alongside this, publicly available face databases, like the FRGC (Phillips et al., 2005) and AR (Martinez and Benavente, 1998) face databases, are recorded under very controlled environmental conditions with the individual being aware of the camera and of being recorded. The discrete nature of the captured variations does not reflect real-life situations with continuous changes. The difference can be seen in Figure 3.5 in Chapter 3 which contrasts face images from these databases with some obtained with the proposed system. As a consequence, these databases cannot provide data to evaluate the developed system. This leads to the necessity of collecting one's own data. Using a simplified version of the system outlined above called *face recorder*, this is easily achieved. In the reduced system in Figure 1.4, a successful detection of a face triggers a recording instead of the feature extraction and the classification process. The recording continues until the person leaves the camera's field of view. Like this, the system can run unattendedly and automatically collect separate video sequences of individuals entering the room over an arbitrary period of time until the resulting video database captures a sufficient number of variations and people.

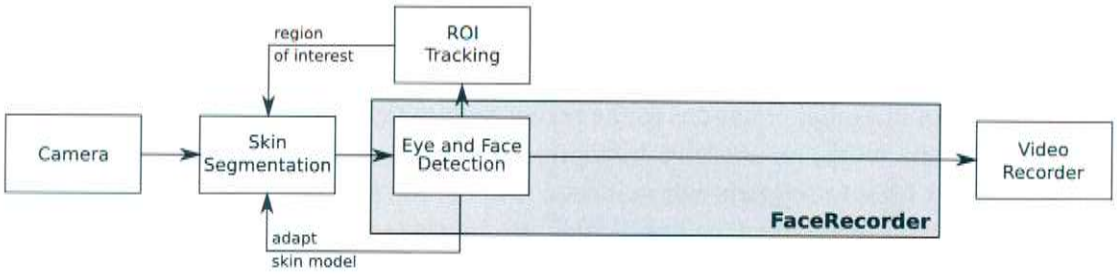



*Figure 1.4: Overview of the data collection system.*

Manual segmentation of a continuously recorded video stream would be tedious and far exceed any reasonable amount of time and memory, especially since the door scenario implies long periods of time in which nothing happens.

Concluding this overview, a remark about terminology is necessary. The term *face detection* can be interpreted in two ways, technically and functionally. The former refers to the low level application of Haar classifier cascades to detect a face pattern rather than an eye pattern. The latter, in contrast, corresponds to a more abstract view, referring to the confirmation of a face candidate by means of Haar-feature-based eye and face detection. In most parts of this work, the functional interpretation is used. Exceptions ensue from context if a discrimination between the different classifier cascades is necessary.

## 1.5 Contribution

The contribution made by this work comprises two major aspects:

**Fully automatic data collection of real-life data** The system is able to automatically record segmented video data of people entering a room. This allows to record data continuously over a long period of time rather than during designated recording sessions only. People behave naturally, since they are not required to interact with the recording system in any special way. As a consequence, the system is able to capture continuous real-life variations of illumination, pose and appearance of a variable number of subjects. This is a major contrast to existing public databases, which are recorded under controlled conditions and contain only a pre-defined set of discrete variations. From the collected data, training samples can be extracted in an unsupervised manner.

**Real-time face recognition in real-life video data** Due to the locality of the model and the exploitation of temporal dependencies between frames, the developed system robustly handles strong variations in the data. The underlying models are easily extendable or reducible to recognize more or less people without the necessity to retrain everybody else. The system successfully extends the frame-based

approach by Ekenel and Stiefelhagen (2005) to video-based data. Furthermore, it adds automatic model generation with varying granularity for each person caused by the heterogeneity of the training data, which contains largely different numbers of samples per person.

# 2 Basic Principles

This chapter will introduce the theoretical foundations of the major techniques which are employed within this work. Implementational details are omitted, as they are presented in Chapter 3.

## 2.1 Haar cascade classifiers

Haar cascade classifiers represent a framework for rapid object detection in images as proposed by Viola and Jones (2001). This framework is based on a set of Haar-like rectangular features which can be efficiently computed using an image representation called *integral image*. A cascaded architecture trained with the AdaBoost boosting algorithm (Freund and Schapire, 1997) allows rapid evaluation of these features in order to detect learned objects or, in this case, faces and eyes.

### 2.1.1 Haar-like features

As mentioned above, the detection framework makes use of a large number of rectangular features which are reminiscent of Haar basis functions. Some examples of these features are depicted in Figure 2.1.

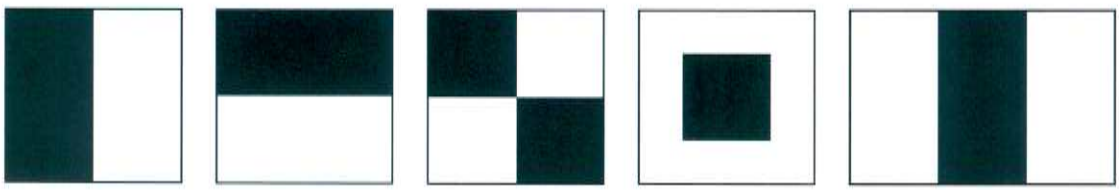
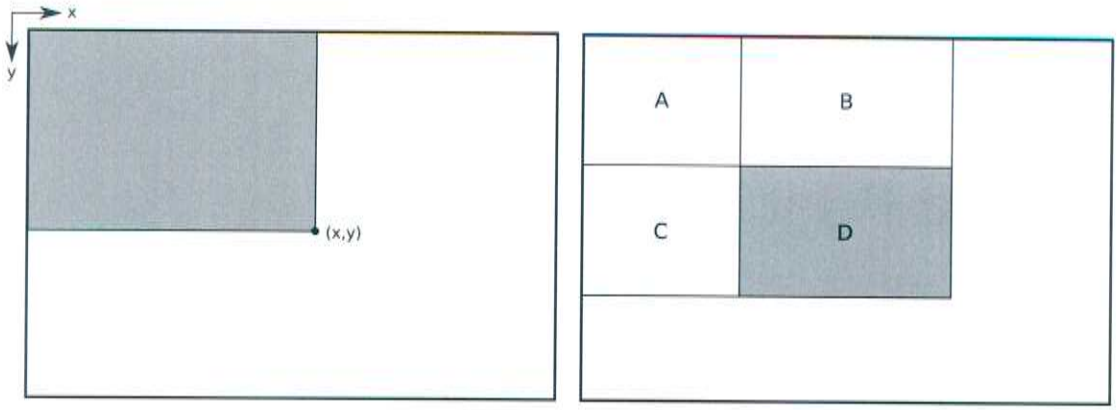


***Figure 2.1:*** *Examples of Haar-like features. Their values represent the intensity differences between the black and the white areas.*

Each feature is basically computed as an intensity difference between adjacent regions of an image. Although not being invariant to rotation, a single feature can easily be evaluated at an arbitrary location or scale. This is made possible by representing the image as an integral image.




(a) (b)

***Figure 2.2:*** *Integral image. (a) The integral at $(x,y)$ is the sum of all pixels up to $(x,y)$ in the original image. (b) The area of rectangle $D$ results to $ii(A+B+C+D) - ii(A+C) - ii(A+B) + ii(A)$. Each $ii(\cdot)$ can be determined with a single array reference (taken from: Viola and Jones, 2001).*

### 2.1.2 Integral image

Similar to an integral in mathematics, pixel $ii(x,y)$ in the integral image represents the sum of the pixels above and left of pixel $i(x,y)$ in the original image, including $i(x,y)$ (see Figure 2.2 (a)). Therefore, the integral image is defined as

$$ii(x,y) = \sum_{\substack{x' \leq x \\ y' \leq y}} i(x',y') \tag{2.1}$$

with $0 \leq x < X$ and $0 \leq y < Y$, $X$ and $Y$ being the width and height of the original image, respectively. Since at each location $(x,y)$ all pixels above and left of it have to be accessed to compute $ii(x,y)$, this basic formulation is computationally very expensive. Taking into account that $ii(x-1,y)$ already contains the sum up to $(x-1,y)$, a pair of recurrences

$$s(x,y) = s(x,y-1) + i(x,y) \qquad \text{with } s(x,-1) = 0 \tag{2.2}$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \qquad \text{with } ii(-1,y) = 0 \tag{2.3}$$

allows to compute the integral image efficiently with one pass over the original data. The term $s(x,y)$ denotes the cumulative sum of the elements in column $x$ up to row $y$.

In the integral image, the integral of an arbitrary rectangle can be computed with a maximum of four array references, one for each vertex of the rectangle. Please refer to Figure 2.2 (b) for an example. Looking at the example features in Figure 2.1, it is obvious that the most complex one, the center one, can be determined with as few as nine accesses to the integral image. Hence, based on this image representation, the

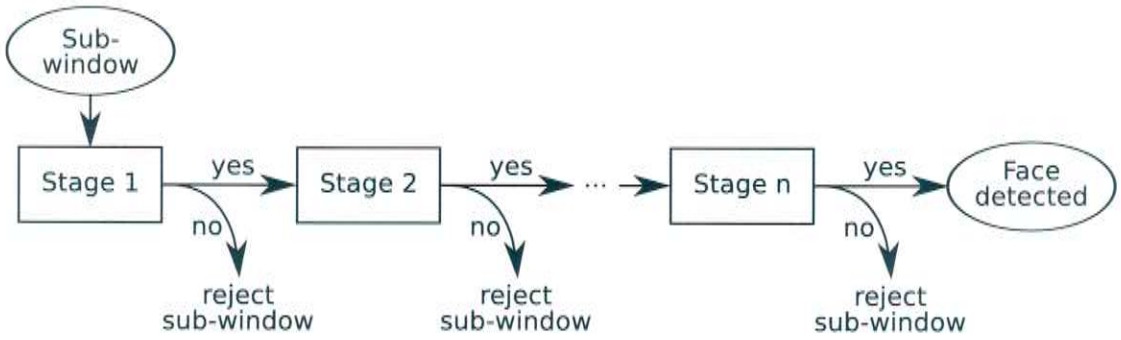


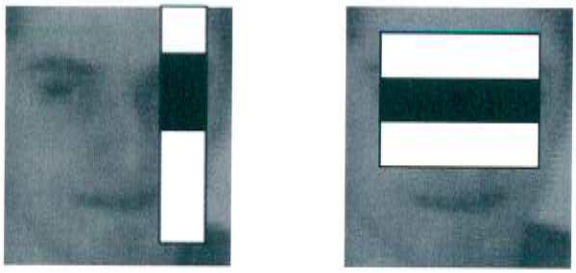

*Figure 2.3: Structure of the classifier cascade. "Yes" and "no" denote if the sub-window successfully passed the previous stage (Cf. Viola and Jones, 2001).*

Haar-like features described above can be evaluated at any location or scale in constant time. In comparison, computation of a feature of size $X' \times Y'$ in the original image requires $X' \cdot Y'$ accesses.

### 2.1.3 Classifier training

Even though any single feature can be computed very rapidly, the exhaustive set of possible features in an image is very large. In the work of Viola and Jones (2001), it consists of approximately 160,000 features per $24 \times 24$ pixel sub-window. Since the input image is scanned with a sliding window at different scales, evaluation of the full feature set leads to a very high computational effort. To reduce the number of features and to obtain an efficient detection algorithm, the most discriminant ones are selected using a modified version of the AdaBoost boosting algorithm by Freund and Schapire (1997). The thresholded single features are considered as weak learners which are then weighted and combined to form a stronger classifier, which takes the form of a perceptron. Within this classifier, discriminating features, i. e., good classification functions, obtain a high weight, whereas less differencing features and therefore ones with poor classification performance get a low weight. In the framework by Viola and Jones, AdaBoost is used to greedily select a small number of distinctive features from the vast set of available ones.

It is obvious that the number of features in the strong classifier directly affects computation time as well as the correct and false detection rates. A smaller number of features leads to a faster classifier with fewer correct and more false detections. In order to keep the number of evaluated features small but still obtain good detection results, a *cascade* of several of the strong classifiers outlined above is constructed. A cascade is essentially a degenerate decision tree as depicted in Figure 2.3. Each stage hands on its detections — both correct and false — to its successor, which is trained to discriminate these more difficult cases using additional features. Negative sub-windows are



(a) (b)

***Figure 2.4:*** *Two out of four features evaluated in the first stage of the face detection cascade used in this work. Both features embody the observation that the forehead and cheeks of a person are usually brighter than the eye region because the eyes are located further aback.*

discarded immediately. A sub-window which successfully passes the whole cascade is considered a correct detection.

Consequently, the *entire* set of selected features has only to be evaluated for the small number of positive sub-windows — compared to the overall number of sub-windows in an image. The majority of negative sub-windows is discarded early in the detection process using only a small subset of the selected features. Figure 2.4 shows two sample features of the four features in the first stage of the face detector that was used in this work.

To train this system, all sub-windows that pass one stage are used as training samples for the next one. This stage is then trained to discriminate these more difficult cases using a different set of features. Each strong classifier has to solve a harder problem than its predecessor. For each stage, limits for acceptable correct and false detection rates are defined. Features are added to these classifiers until these requirements are met. If the overall detection rates are not yet satisfying, another classifier is trained and added to the cascade. Given its sequential structure, the correct detection rate $D$ and the false detection rate $F$ of the final cascade with $K$ stages can be computed using

$$D = \prod_{i=1}^{K} d_i \tag{2.4}$$

$$F = \prod_{i=1}^{K} f_i \tag{2.5}$$

where $d_i$ is the correct detection rate and $f_i$ is the false accept rate of classifier $i$. These rates are computed on the samples that are passed on from the classifier $i-1$, where $i = 0$ is the image itself and therefore yields all possible sub-windows.

The power of this detection framework is stressed by the fact that Viola and Jones were able to reject 50 % of the negative sub-windows while detecting 100 % of the faces with as few as *two* features.

## 2.2 Kalman filter

The Kalman filter (KF) is a linear state estimator and was initially introduced by Kalman (1960). Since then, the KF and several variants like the Extended Kalman filter for non-linear estimation are commonly used in tracking tasks. A detailed introduction to these topics can be found in (Bar-Shalom and Fortmann, 1988; Jazwinski, 1970; Welch and Bishop, 2001).

In this work, the system contents itself with the basic implementation of the Kalman filter for linear prediction. It is based on a discrete-time dynamic system which is described by a process model

$$x(t) = A(t) \cdot x(t-1) + B(t) \cdot u(t) + v(t) \tag{2.6}$$

and an observation model

$$z(t) = H(t) \cdot x(t) + w(t) \tag{2.7}$$

The system state at time $t$ is denoted by $x(t)$. $A(t)$ and $H(t)$ stand for the known state transition and measurement matrices, while matrix $B(t)$ allows to model the influence of some optional control input $u(t)$. The vectors $v(t)$ and $w(t)$ represent the process noise and the observation or measurement noise, respectively. They are assumed to be independent, white Gaussian random processes with zero-mean and covariances $Q(t)$ and $R(t)$, respectively.

Equations (2.6) and (2.7) allow to infer the usually not directly observable current system state $x(t)$ from a sequence of measurements $\{z(t)\}_t$. The recursivity of Equation (2.6) is a key property of the Kalman filter, as it avoids the need to process all measurements $\mathcal{Z}_t = \{z(i)\}_{i=0}^{t}$ in every time step.

When estimating the system state, let $\hat{x}(t|\mathcal{Z}_{t-1})$ denote the *a priori* or *predicted state estimate* at time $t$ taking into account the measurements $\mathcal{Z}_{t-1} = \{z(i)\}_{i=0}^{t-1}$ up to time $t-1$, and $\hat{x}(t|\mathcal{Z}_t)$ the *a posteriori* or *filtered state estimate* derived from all measurements $\mathcal{Z}_t$. The predicted state estimate is given by

$$\hat{x}(t|\mathcal{Z}_{t-1}) = A(t)\hat{x}(t-1|\mathcal{Z}_{t-1}) + B(t) \cdot u(t) \tag{2.8}$$

and the resulting *state prediction error* is

$$\tilde{x}(t|\mathcal{Z}_{t-1}) = x(t) - \hat{x}(t|\mathcal{Z}_{t-1}) \tag{2.9}$$

From that, the *state prediction error covariance* can be computed as

$$\begin{aligned} P(t|\mathcal{Z}_{t-1}) &= E[\tilde{x}(t|\mathcal{Z}_{t-1})\tilde{x}^T(t|\mathcal{Z}_{t-1})] \\ &= A(t)P(t-1|\mathcal{Z}_{t-1})A^T(t) + Q(t) \end{aligned} \tag{2.10}$$

Concerning the actual observations, the *predicted measurement*

$$\hat{z}(t) = H(t)\hat{x}(t|\mathcal{Z}_{t-1}) \tag{2.11}$$

allows to compute the *innovation* or *measurement residual*

$$\alpha(t) = \tilde{z}(t) = z(t) - \hat{z}(t) \tag{2.12}$$

and its covariance

$$\begin{aligned} S(t) &= E[\tilde{z}(t)\tilde{z}^T(t)] \\ &= H(t)P(t|\mathcal{Z}_{t-1})H^T(t) + R(t) \end{aligned} \tag{2.13}$$

The innovation describes the difference between the predicted measurement and the actual observation. Together with the *Kalman gain*, which is defined as

$$K(t) = P(t|\mathcal{Z}_{t-1})H^T(t)S^{-1}(t) \tag{2.14}$$

the filtered state estimate can be updated following the *state update equation*

$$\hat{x}(t|\mathcal{Z}_t) = \hat{x}(t|\mathcal{Z}_{t-1}) + K(t)\alpha(t) \tag{2.15}$$

and the *filtered state error covariance* according to the *covariance update equation*

$$P(t|\mathcal{Z}_t) = [I - K(t)H(t)]P(t|\mathcal{Z}_{t-1}) \tag{2.16}$$

Since a Kalman filter uses measurements to correct its state estimates, it can be thought of as a predictor-corrector algorithm as it is commonly used to numerically integrate differential equations (Welch and Bishop, 2001). The set of equations concerning the prediction of the current state, and therefore the next measurement, is made up of Equations (2.8) and (2.10). After the actual observation has been made, the correction step leads to an update of the filter state according to this observation using the measurement update equations (2.12),(2.14),(2.15) and (2.16). Figure 2.5 summarizes this process.

## 2.3 Discrete cosine transform

High-dimensional data can pose many challenges. Analysis of images of size $X \times Y$ on pixel level, for example, would result in a feature space with $X \cdot Y$ dimensions. This grows easily into thousands of dimensions, which makes it very difficult to model the data since many traditional statistical methods break down due to the enormous number of variables. Furthermore, larger feature vectors both require more memory and increase processing time.

The good news is that, most of the time, not all dimensions are necessary in order to build a model which captures the underlying characteristics of the data. In fact, those can often be suitably represented using only a small fraction of the initial number of dimensions. Unfortunately, the essential dimensions are usually not axially parallel to

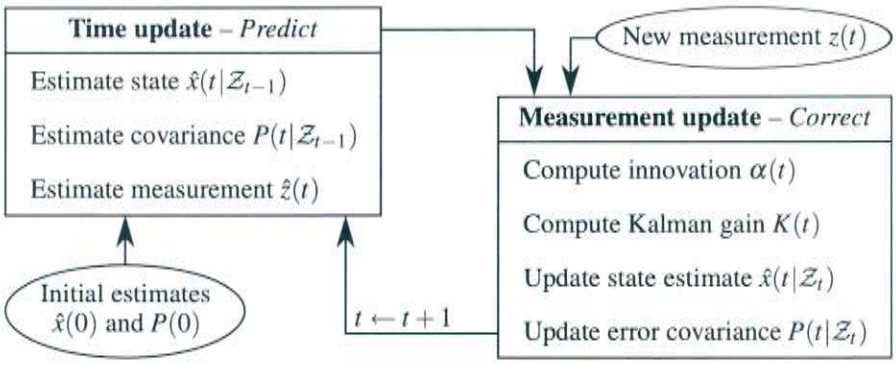



***Figure 2.5:*** *Overview of the Kalman filter as a predictor-corrector algorithm (Cf. Welch and Bishop, 2001).*

the dimensions of the original data, as the variables can be highly correlated. Therefore, it is crucial to move the data to a different representation which is more appropriate in these terms.

One of the methods to achieve this is the *discrete cosine transform (DCT)*. It is widely used in signal processing, especially in image processing where it is well-known as the basis of the widespread JPEG still image compression standard (Wallace, 1991). It interprets the data as superimposition of cosine oscillations and transforms it to frequency domain. Since this work deals with computer vision problems, the input signal is considered to be 2-dimensional image data. For a 2-dimensional signal $f(x,y)$, the DCT is defined as

$$F(u,v) = C(u)C(v)\frac{2}{X}\sum_{x=0}^{X-1}\sum_{y=0}^{Y-1} f(x,y)\cos\left(\frac{\pi u(2x+1)}{2X}\right)\cos\left(\frac{\pi v(2y+1)}{2Y}\right) \tag{2.17}$$

where the input is of size $X \times Y$ and $C(\cdot)$ is defined as

$$C(i) = \begin{cases} \frac{1}{\sqrt{2}} & i = 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.18}$$

The cosine basis functions connected to the resulting coefficients are depicted in Figure 2.6. Coefficient (0,0) represents the average signal, i. e., in case of image processing, the average gray value of the image. It is called DC coefficient by analogy with *direct current* in electricity. Similarly, the other coefficients are called AC by analogy with *alternating current*.

The DCT has several advantages that makes its use appealing:

**Orthonormality** The DCT is orthonormal and therefore lossless. This way, one has full control which part of the signal is to be discarded to reduce the dimensionality. No information is inherently lost by the transformation itself. As a consequence, it is fully invertible.

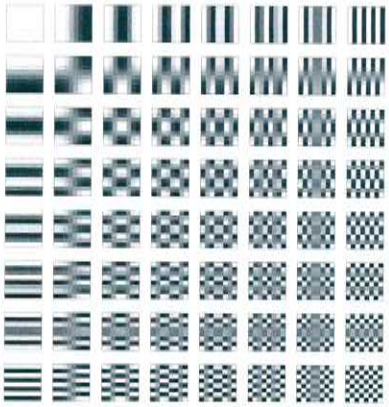
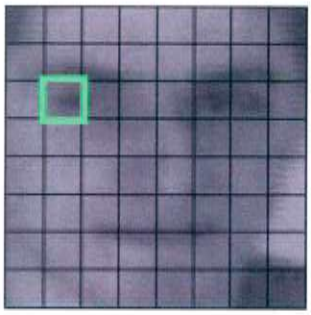


*Figure 2.6: Cosine basis functions of the discrete cosine transform for input size $8 \times 8$. The frequency of the basis functions increases from top left (0,0) to bottom right (8,8).*



(a)

| 674 | 55 | 10 | 3 | 1 | -1 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| -2 | -6 | -10 | -4 | 3 | 0 | 0 | -1 |
| 50 | -20 | -2 | 0 | -2 | -1 | -1 | -1 |
| -13 | 3 | -1 | 1 | 0 | 0 | 0 | 0 |
| -6 | -2 | -1 | 2 | -1 | 1 | 0 | 0 |
| -2 | -1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 3 | 1 | -1 | -1 | -1 | 0 | 0 |

(b)

*Figure 2.7: Discrete cosine transform of an $8 \times 8$ pixels image patch. The coefficients represent the basis functions depicted in Figure 2.6.*

**Compactness of representation** The DCT approximates the Karhunen-Loève transform (KLT) which is optimal in terms of representational compactness under certain conditions (Goyal, 2001). Applying DCT to images generally leads to high-valued coefficients for low-frequency basis functions and to low-valued coefficients for high frequency ones as can be seen in Figure 2.7. Obviously, the major part of the signal energy is encoded in a small number of low-frequency coefficients and therefore dimensions. This is the key to reducing the dimensionality of the data. The DCT itself is lossless, as mentioned above, and the dimensionality of the transformed signal is still the same as the one of the input signal. But high-frequency coefficients can be removed without any or, at most with negligible effects on the input signal, thus reducing its dimensionality. Essentially, this low-pass filters the original data. Figure 2.8 visualizes this compaction by showing the average energy of all 64 blocks in an input image of size

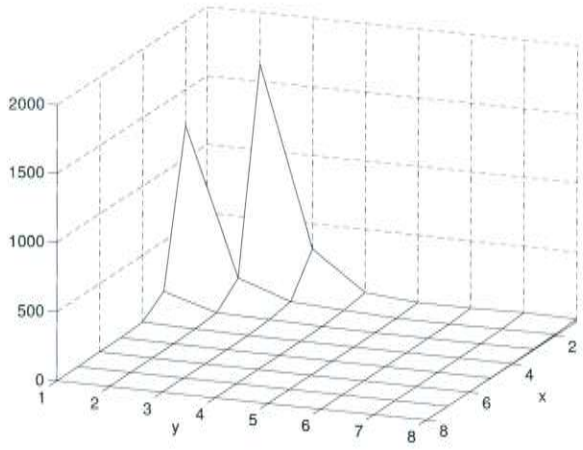



***Figure 2.8:*** *Average energy of all 64 blocks of the image in Figure 2.7 (a). The DC coefficient has been removed to allow meaningful scaling.*

$64 \times 64$ pixels. The image has been split into blocks of $8 \times 8$ pixels to allow the DCT to capture enough local detail while still providing sufficient compaction. This size is based on the JPEG standard.

**Data-independency** The basis functions of the DCT are independent from the data to be transformed. This is in contrast to PCA, the discrete realization of the KLT. Since these transforms rely on the covariance of the data, the basis of the new vector space has to be computed from a representative training set. This leads to additional efforts both in terms of computation and construction of the training set. The DCT always uses the basis functions shown in Figure 2.6 for input of size $8 \times 8$. Hence, the representation of already processed data does not change as it would with PCA, if new and unforeseen data arrived due to a non-representative training set, which would make recomputation of the basis functions necessary.

In order to represent the coefficients of a 2-dimensional DCT as a 1-dimensional vector, the transformed signal is scanned following a zig-zag pattern as shown in Figure 2.9.

## 2.4 K-means clustering

*K-means*, introduced by MacQueen (1967, see also Tan et al., 2005), is an unsupervised learning method which partitions the data into $k$ clusters. Each cluster is represented by its centroid. The approach uses complete and hard clustering, which means that each sample belongs to exactly one cluster. It is widely used for its simplicity and efficiency.

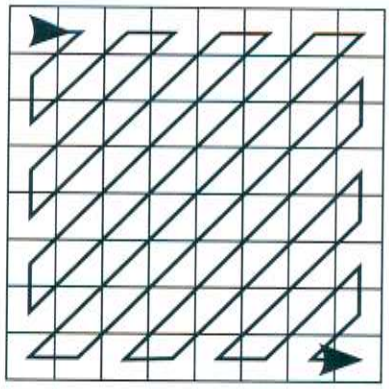


*Figure 2.9: The DCT coefficients are serialized according to a zig-zag pattern.*

### 2.4.1 Definition

The basic outline of this algorithm is rather simple. To determine cluster association, an appropriate distance metric $d(x_n, k_m)$ is necessary. Common ones are, for example, city block (Equation (3.5)) and Euclidean distances. Furthermore, the number of clusters $k$ has to be chosen in advance. This has to be done carefully in order to achieve meaningful clusters.

The system is initialized by selecting $k$ samples as initial cluster centroids. Depending on the available knowledge of the data, these can be chosen randomly from the data, by iteratively selecting data points that lie maximally apart or by running the algorithm on a small subset with random initialization and using the resulting centroids to cluster the complete data set.

Afterwards, each point is assigned to the closest centroid according to $d(\cdot,\cdot)$. Subsequently, the centroids are recomputed as mean vector of all assigned points. These two steps, assignment and centroid update, are repeated until the cluster means do not change any more.

### 2.4.2 Discussion

K-means can be regarded as a hill-climbing algorithm which minimizes an objective function. For city block and Euclidean distances, these are commonly the sum of errors (SE) and the sum of squared errors (SSE), respectively. They are defined as

$$\mathrm{SE} = \sum_{i=1}^{k} \sum_{x \in k_i} d(x,k_i) \qquad \mathrm{SSE} = \sum_{i=1}^{k} \sum_{x \in k_i} d(x,k_i)^2 \tag{2.19}$$

With each iteration, the error decreases but since the error surface is seldom unimodal, the algorithm can converge to a local optimum. A common way to reduce the risk to end up with a locally good result only, is to run the algorithm several times with

different random initializations of the centroids. Afterwards, the solution yielding the lowest error is selected.

Several extensions of the original algorithm have been developed. Split and merge techniques allow for adaptation of the number of clusters and selecting the medoid, i. e., the data point which is closest to the cluster center, instead of the mean increases robustness against outliers. Zhang et al. (2000) proposed to use the harmonic mean to determine soft cluster assignments. As a result, they report a higher robustness against bad initialization.

## 2.5 K-nearest neighbors classification

The *k-nearest neighbors (KNN)* approach is a type of *discriminative model*.This family of learning techniques derives the classification result directly from a set of training samples instead from an abstract model of the characteristics of the data (Tan et al., 2005).

### 2.5.1 Nearest neighbor

The elements of the training data are called *representatives* or *prototypes*. Representatives with $n$ features are considered as points in an $n$-dimensional vector space. A new sample $x \in \mathbb{R}^n$ is labeled with the class of the closest representative, the nearest neighbor, according to a distance metric $d(x,y)$. Although the resulting error will be greater than the optimal Bayes error rate, it is never more than twice, given an unlimited number of representatives. The proof is omitted here but can be found in (Duda et al., 2001, pp. 182–184). Please note that differently scaled features can bias the distance metric to overemphasize large-valued features at the cost of small-valued ones. This effect can be mitigated with appropriate normalization techniques.

### 2.5.2 K-nearest neighbors

If the class of $x$ depends on a single prototype only, it is easily affected by noise. This can be avoided by selecting $k$ nearest neighbors and derive the classification decision from their class labels. The simplest way to do this is a majority vote which assigns the most common among the class labels in question. This leads to equal contribution of every neighbor, independent of its distance to $x$ and, thus, renders the approach unnecessary sensitive to the choice of $k$. Individual weights $w_i$ for every selected prototype $k_i$, $i = 1,\ldots,k$, can be derived by taking the actual distance $d(x,k_i)$ into account. For example, using $w_i = \frac{1}{d(x,k_i)^2}$ greatly reduces the influence of distant training samples (Tan et al., 2005).

## 2.6 Gaussian mixture models

A *Gaussian mixture model (GMM)*, also known as *mixture of Gaussians*, exploits the fact that any probability distribution can be approximated with a combination of multivariate normal distributions. This section will introduce the underlying assumptions and answer the question of how to determine the necessary parameters. Since it is possible to generate new samples with this kind of model, it is also referred to as *generative model*.

### 2.6.1 Definition

A mixture model is defined as a weighted combination of $M$ probability distributions $P_i(X;\theta_i)$ with parameter sets $\theta_i$, $i = 1,\ldots,n$, of random variable $X$. For Gaussian mixture models, these distributions are chosen to be normal with parameters $\mu_i$ and $\Sigma_i$, where $\mu_i$ denotes the mean and $\Sigma_i$ the covariance of $P_i$.

A GMM of $M$ $N$-dimensional Gaussian distributions is defined as

$$\begin{aligned} P(X = x|\Theta) &= \sum_{i=1}^{M} \alpha_i \cdot P_i(X = x;\theta_i) \\ &= \sum_{i=1}^{M} \alpha_i \cdot \mathcal{N}(X = x;\mu_i,\Sigma_i) \\ &= \sum_{i=1}^{M} \alpha_i \cdot \frac{1}{(2\pi)^{\frac{N}{2}} |\Sigma_i|^{\frac{1}{2}}} e^{\frac{1}{2}(x-\mu_i)\Sigma^{-1}(x-\mu_i)} \end{aligned} \tag{2.20}$$

where $\Theta = \{\alpha_1,\ldots,\alpha_M,\mu_1,\ldots,\mu_M,\Sigma_1,\ldots,\Sigma_M\}$ symbolizes the complete parameter set of the mixture. $|\Sigma_i|$ represents the determinant of the covariance matrix of the respective normal distribution and $\alpha_i$ the contribution of $P_i(X = x;\mu_i,\Sigma_i)$ to the overall likelihood $P(x = X)$. The mixing parameters $\alpha_i$ must satisfy the condition $\sum_{i=1}^{M} \alpha_i = 1$.

### 2.6.2 Parameter estimation by expectation-maximization

To determine the free parameters of the GMM, an expectation-maximization (EM) algorithm (Bilmes, 1997; Dempster et al., 1977) is used. Given a set of samples $\mathcal{X} = \{x_1,\ldots,x_K\}$ independently drawn from the mixture model (2.20), the EM algorithm estimates the set of parameters $\Theta^\star$ of this model that maximizes the likelihood that $\mathcal{X}$ is observed. With the knowledge on which mixture component generated which sample, this computation would be straightforward since the parameters of each component could be directly calculated based on the samples it generated. Unfortunately, this data is not observable when drawing samples from an unknown distribution.

To solve this problem, the EM algorithm iteratively estimates the mixture parameters based on an initial guess $\Theta^g$. A common way to determine $\Theta^g$ is to process the samples in $\mathcal{X}$ with the k-means clustering algorithm.

Each iteration of the algorithm consists of two steps: the *Expectation* or *E-step* and the *Maximization* or *M-step*. The algorithm monotonically approximates the maximum-likelihood and is therefore guaranteed to converge. Since the search space is multi-modal, the algorithm can converge to a local optimum. To reduce this risk, a reasonable initialization of $\Theta^g$ is crucial.

### Expectation step

The E-step computes the expected value of the hidden parameters $\alpha_m$ based on the current set of parameter estimates $\Theta$ and the observed data $\mathcal{X}$.

$$E[\alpha_{m,k}] = P(X = P_m | x_k, \Theta) = \frac{\alpha_m \cdot \mathcal{N}(X = x_k; \mu_m, \Sigma_m)}{\sum_{j=1}^{M} \alpha_j \cdot \mathcal{N}(X = x_k; \mu_j, \Sigma_j)} \tag{2.21}$$

### Maximization step

During the M-step, the estimated hidden parameters are used to modify $\Theta$ in order to maximize $P(\mathcal{X}|\Theta)$. The optimized parameters are computed using

$$\alpha_m^{\text{new}} = \frac{1}{N} \sum_{k=1}^{K} E[\alpha_{m,k}] \tag{2.22}$$

$$\mu_m^{\text{new}} = \frac{\sum_{k=1}^{K} x_k E[\alpha_{m,k}]}{\sum_{k=1}^{K} E[\alpha_{m,k}]} \tag{2.23}$$

$$\Sigma_m^{\text{new}} = \frac{\sum_{k=1}^{K} E[\alpha_{m,k}](x_k - \mu_m^{\text{new}})(x_k - \mu_m^{\text{new}})^T}{\sum_{k=1}^{K} E[\alpha_{m,k}]} \tag{2.24}$$

# 3 Methodology

After the main theoretical foundations of this work have been introduced on a general level, this chapter will focus on their integration into the developed face recognition system. In addition to that, it will give detailed insight into its functionality and the underlying design decisions. Starting with the skin color segmentation in Section 3.1, the detection of faces and their registration is covered in Section 3.2. Details about the feature extraction using the discrete cosine transform are given in Section 3.3. Following the structure of the system, Section 3.4 then introduces the models used for person classification. Section 3.5 completes the chapter with a description of the region-of-interest tracking algorithm.

The implementation of the face recognition system is based on the Open Computer Vision Library (OpenCV, Intel Corporation, 2006).

## 3.1 Skin color segmentation

Not only in the given scenario, but in all recognition scenarios involving wider than close-up views, the face to be recognized is comparatively small with respect to the image dimensions. In order to avoid unnecessary processing of the background, it is crucial to concentrate on meaningful areas of the image. In order to identify these, the image is searched for skin-like colors. This is a reasonable and well-explored approach, as faces generally expose larger areas of skin depending on already described appearance variations. Skin-color has been intensively used as a low-level cue to detect faces and people in images since the mid-nineties (e. g., Hunke and Waibel, 1994; Kjeldsen and Kender, 1996; Raja et al., 1998; Soriano et al., 2000). This is because it is invariant to orientation and scale and efficient to compute.

### 3.1.1 Skin color representation

In this study, a histogram-based model is used to represent the skin color distribution. It is learned from a representative training set of skin samples. This kind of model has been used in a variety of works (e. g., Jones and Rehg, 2002; Soriano et al., 2000). It is non-parametric and makes no prior assumption about the actual distribution of skin colors. Extensive research efforts have been spent on the question which color space would be the most suitable for this kind of model. Terrillon et al. (2000) compared nine different skin chrominance models showing that *Tint-Saturation-Value* (TSL) and

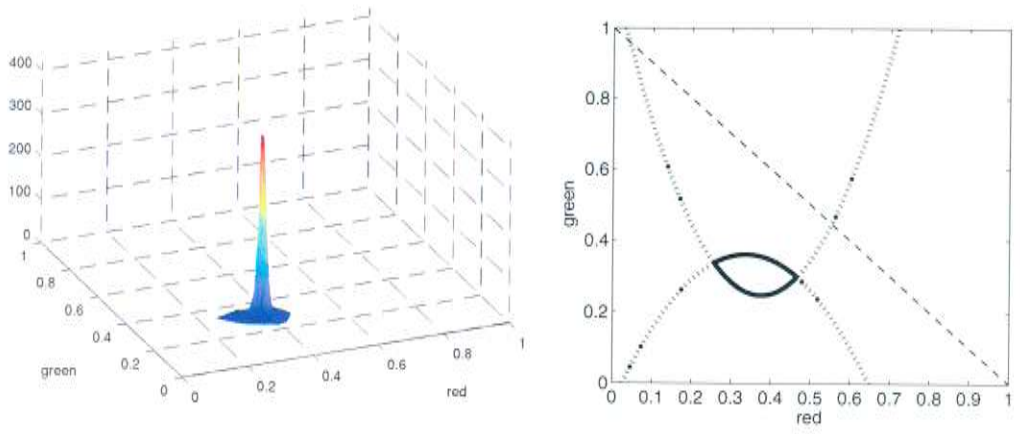
*normalized red-green* (normalized-rg) color spaces yielded the best performances in their face detector setup. The model utilized in this work is located in the normalized-rg color space because conversion to TSL requires additional effort. It is based on the widely used RGB color model, in which each pixel is represented by a tuple with one value for each of the colors red, green and blue. It emerges from the RGB color model by normalizing each pixel value $(R,G,B)$ over its intensity $I = R+G+B$, i. e., $R = \frac{R}{I}$ and $G = \frac{G}{I}$. Hence, the normalized-rg color space describes proportions of the three single colors. The information on the blue channel has been dropped. It is redundant because the normalized values sum up to 1. This reduces the dimensionality of the problem and leads to a 2-dimensional model histogram. This simpler model can be computed and applied easily. Since the basic idea of the skin segmentation is to enable the system to run in real-time, it needs to be as efficient and fast as possible.

The advantage of choosing a chrominance-based color space is a reduced sensitivity to illumination influences. At the same time, different skin tones, due to different ethnic backgrounds, get more similar to each other in this representation, forming a compact cluster in color space. Nevertheless, a study by Shin et al. (2002) suggests that disregarding the illumination information leads to inferior skin detection rates because skin and non-skin separation is reduced. The proposed system, however, follows the conclusion of the survey by Vezhnevets et al. (2003, p. 90) which states *"that dropping luminance is a matter of training data generalization"*. Thus, fewer training images with less variation are necessary to build a reasonable skin representation. Easier training is traded for a reduced rate of detected skin pixels, but since the skin segmentation is only a preprocessing step to locate relevant portions of the image, its success does not depend on the detection of every single skin-like colored *pixel* but on the detection of such *regions*.

The system uses a skin model histogram with $128 \times 128$ bins. This reduces computational requirements and influence by noise compared to larger histogram sizes. At the same time, it still ensures the necessary resolution of different colors, which degrades with a decreasing number of histogram bins.

### 3.1.2 Skin locus

Representing skin in the 2-dimensional normalized-rg color space yields another advantage. In addition to the fact that all skin tones populate a closed region in color space, this region is describable using two quadratic functions. The general shape reminds of an eye-brow but the actual shape is camera-dependent (Martinkauppi, 2002; Martinkauppi et al., 2001). According to Störring (2004), this area is referred to as *skin locus*. Figure 3.1 visualizes the skin model and skin locus derived from 242 training samples or, to be more precise, 799,785 training pixels, captured with a Canon VC-C1 camera. The skin locus in Figure 3.1 (b) shows that the camera tends to overemphasize




(a) (b)

***Figure 3.1:*** *(a) The skin color distribution as determined from a training set. (b) The skin locus in normalized-rg color space, described by two functions of quadratic order.*

*blue* at cost of *green* under certain conditions, resulting in a locus that extends towards the *red* axis. A certain color $(r, g)$ is part of the locus if

$$g > f_{min}(r) \quad \wedge \quad g < f_{max}(r) \tag{3.1a}$$

with

$$f_{min}(r) = \quad 6.38r^2 - 4.79r + 1.15 \tag{3.1b}$$

$$f_{max}(r) = -3.51r^2 + 2.53r - 0.06 \tag{3.1c}$$

The boundary functions $f_{min}$ and $f_{max}$ are computed by fitting second-order polynomials of $r$ to the boundary points of the skin distribution, i. e., to the outer-most histogram bins with non-zero count.

The samples used to build the model are manually cropped from images by selecting large skin areas in faces in a set of input images. Rectangular sample areas are chosen to contain as many skin pixels as possible while including as little distracting ones originating from, for example, glasses, lips or hair. Prominent regions are the forehead or the central region comprising the nose and both cheeks (see Figure 3.2) but other parts of the face are included as well to achieve a more general model. Accepting a small number of distractors into the model does not severely affect the model performance but greatly eases the skin sample selection.

### 3.1.3 Segmentation

The segmentation process is based on *histogram backprojection*, a technique that highlights colors in the image which are part of a histogram-based color model (Soriano

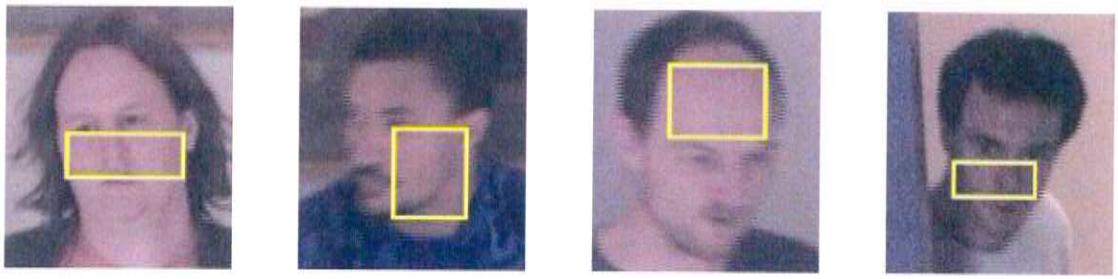


***Figure 3.2:*** *Skin samples are chosen to contain many skin colored pixels but no or only few distractors.*

et al., 2000; Swain and Ballard, 1991; Yoo and Oh, 1999). As a first step, the ratio histogram $R$ is computed from the skin model histogram $S$ and the image histogram $I$

$$R(r,g) = \min\left(\frac{S(r,g)}{I(r,g)}, 1\right) \tag{3.2}$$

where $r$ and $g$ denote the histogram bin. Next, $R$ is *backprojected* onto the original image, which means, that each pixel $i(x,y)$ is replaced by $R(r_{x,y}, g_{x,y})$, where $r_{x,y}$ and $g_{x,y}$ denote the normalized color values of $i(x,y)$. This results in a gray scale image which can be interpreted as a probability map of skin presence. Bright values denote a high, dark values a low probability. Backprojecting the ratio histogram instead of the model histogram itself emphasizes colors that are characteristic for the model. In turn, colors which are part of the model but which are also common in the background are weakened.

So, if the background contains as many pixels of a certain color as the model or less, these pixels will get a skin probability of 1, according to Equation (3.2). If there more pixels of that color in the background than in the model, skin probability will be lower. Let $\Sigma_S$ denote the total number of pixels captured in the histogram. Obviously, the model histogram encodes implicitly the size of the target skin area by $\Sigma_S$. Therefore, it is fundamental, that $\Sigma_S$ is normalized to a sensible value. It is observed, that the face sizes in this scenario range roughly from $45 \times 45$ pixels to $100 \times 100$ pixels. If $\Sigma_S$ is chosen at the lower bound of this range, small faces will be segmented successfully, but large faces result in low skin probabilities because the image contains more skin-colored pixels than the model "allows", i. e., the denominator exceeds the numerator in Equation (3.2). Due to the reduced skin probability the pixels are likely to fall below the segmentation threshold. The larger $\Sigma_S$ is chosen, the more skin-colored pixels can the image contain before the resulting skin probability is reduced. In addition to allowing larger faces to be segmented, this increases the amount of segmented skin-colored background as well, if no large face is visible.

As a trade-off, the initial skin model is normalized to a pixel count of $70 \times 70$ pixels. This size allows for reasonable initial segmentation, both for large and small faces. As

*Figure 3.3: Skin segmentation process (a) input image (b) backprojection (c) thresholded backprojection (d) result*

soon as a face is detected, the model size is adapted accordingly (see Section 3.1.4 for details).

The result of the backprojection is shown in Figure 3.3. Yoo and Oh (1999) complain that the background remains noisy in cluttered environments but this issue is successfully addressed with a two-stage thresholding algorithm based on region-growing. The first stage is a basic binary threshold at level $T_{\text{high}}$ . The second one is a hysteresis threshold similar to the one introduced by Canny (1986) for edge detection. It uses a lower threshold value $T_{\text{low}}$ than the initial one but it only adds those pixels to the previously created binary image which are 8-connected to already selected pixels. Application of this method to Figure 3.3(b) results in Figure 3.3(c). White areas denote the result of the first, gray areas the result of the second thresholding step. Figure 3.3(d) shows the final result after noise removal. The thresholded image is less cluttered, if the backprojection is smoothed using a Gaussian kernel because this mitigates interlacing effects and noise. Morphological operators have been omitted for speed reasons.

Possible *face candidates* are extracted from the thresholded image using a connected components algorithm (Rosenfeld and Pfaltz, 1966).

The lower threshold $T_{\text{low}}$ is determined adaptively. It is chosen as the average gray level of the non-black pixels of the backprojection, i. e., as the mean probability of all skin-like colored pixels. This approach has a major advantage over a constant value of $T_{\text{low}}$. If the skin of an entering person is only poorly represented by the current model, due to color, size or both, only a small percentage of the skin pixels will be larger than $T_{\text{high}}$ while the majority will have comparatively small values. If a constant $T_{\text{low}}$ is chosen too large, these pixels will not be segmented. Choosing $T_{\text{low}}$ small enough to successfully segment the badly modeled skin pixels, problems arise when a well-modeled face is encountered. The skin pixels of such a face will, to a large extent, get high probabilities of being skin. As a consequence, application of $T_{\text{high}}$ already leads to reasonable segmentation. The small $T_{\text{low}}$ from before will then add unnecessary clutter to the segmented image.

### 3.1.4 Model adaptation

The model generated from the skin samples, $M_0$, is very general, both in terms of size and color distribution. Therefore, it is only used for initial detection and is then adapted to the current illumination situation and the person's specific skin color. Whenever a face is successfully detected in a skin-colored area, the histogram $H_{\text{face}}$ of this area is used to update the current model $M_t$.

$$M_{t+1}(r,g) = M_t(r,g) + \alpha H_{\text{face}}(r,g) \tag{3.3}$$

with update parameter $\alpha$ and bin indexes $r$ and $g$. With $\alpha = 0.4$, this ensures fast adaptation to every specific case. Due to the Gaussian smoothing, the thresholding process described above leads to segmentation of non-skin pixels close to skin-colored ones, e. g., eyes, lips and hair. In order to avoid adaptation to these colors, only colors inside the skin locus are used to compute $H_{\text{face}}$. Every time the face detection *fails*, $H_{\text{face}}$ is replaced by $M_0$ in Equation (3.3). On the one hand, this simply ensures that the model is reset after a person left the field of view. On the other hand, it allows to recover from misadaptations and sudden changes of the illumination situation.

Every time the skin color model is updated, its size is adapted to the size of the segmented skin region. Since the given scenario aims at people entering the room, the model size is actually chosen slightly larger. This is done to account for the increasing scale of the face over time as the person walks towards the camera.

## 3.2 Eye-based alignment

Since not all skin areas detected in the input image originate necessarily from faces but as well from arms, hands, the back of a bald head or simply some skin-like colored

element in the background, the determined skin areas are considered *face candidates* that need to be confirmed.

Additionally, actual faces need to be transformed to a normalized orientation and size. This is necessary because the feature extraction process is not invariant to variations of translation, rotation and scale.

Both the confirmation and the alignment rely on eye locations. Confirmation is supported by a face detector. Eyes and face are detected using Haar cascade classifiers. The processed image regions are slightly enlarged compared to the plain skin areas because the face detector takes the face outline into account. As stable eye detections are crucial, eye locations are tracked over consecutive frames using Kalman filters. A critical part in using such a filter is proper initialization, which is performed as follows.

### 3.2.1 Eye tracking

Both eyes are tracked separately. The state of each of the two Kalman filters covers the $x$- and $y$-position of one of the eyes, together with its speed of motion in these directions, $v_x$ and $v_y$. The state estimates are supported by measurements of the $(x, y)$ location of the eyes as determined by eye detectors.

The problem that arises with eye detection is, that an eye detector with a reasonable detection rate produces quite a few false positives. This is due to the fact that the intensity distribution of an eye, as captured by the classifier, is rather simple. Therefore, it can be observed in other parts of the processed area as well, e. g., on curly hair. This is especially true since the detector is trained with input data which is rotated up to 30 degrees. In order to initialize the Kalman filters, it is necessary to decide on the "true" detection among all available ones. It is observed that the majority of false positives only show up in single frames or pairs of frames. Nevertheless, some of them are detected more consistently whereas eyes are not necessarily detected in every single frame.

To solve this problem, the approach depicted in Figure 3.4 is implemented (Bar-Shalom and Fortmann, 1988). The detections of each eye cascade are used to generate track hypotheses over consecutive frames. Close detections in consecutive frames are associated to each other to form a track. Tracks that do not get updated with a new measurement are extrapolated based on previous observations. If several detections are associated with one track, it gets split into two. If two tracks overlap for several frames, one of them is discarded. Algorithm 1 gives an overview of the construction of track hypotheses.

To decide, whether a detection $d$ is close to another, a *validation region* $V_t$ is defined (Bar-Shalom and Fortmann, 1988). If $d$ is in this region, it is considered close, otherwise not. Following the notation in Chapter 2, a degenerate version of the Kalman filter is used to determine $V_t$. The last measurement and the derived velocity of the track in question is assumed to form the last state estimate $\hat{x}(t-1)$. This allows to estimate a detection $\hat{z}(t)$. Assuming that the distribution of the true measurement $z(t)$ given $\mathcal{Z}(t-1)$

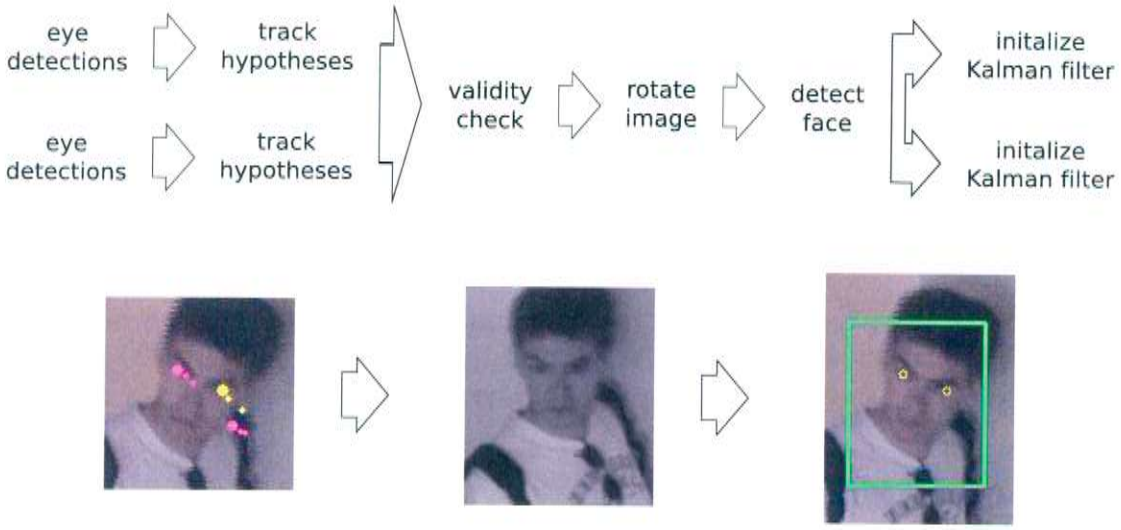



*Figure 3.4: Initialization of the Kalman filters for eye tracking*

**Algorithm 1** Building track hypotheses

```
n ← 3                                          ▷ Maximum extrapolation and overlap
for each eye detection d do                    ▷ Associate detections
    for each track t do
        if d is close to last detection in t then
            associate d with t
    if d unassociated then
        create new track
for each track t do                            ▷ Extend and split tracks
    if t has newly associated detections then
        for each newly associated detection d do
            t' ← t                             ▷ Copy track
            extend t' with d
    else
        extrapolate new measurement e
        extend t with e
for each track t1 do                           ▷ Delete bad tracks
    if t1 was extrapolated the last n times then
        delete t1
        continue
    for each track t2 do
        if t1 and t2 overlap for the last n measurements then
            delete t2
```

is normal with mean $\hat{z}(t)$ and variance $S(t)$, an ellipsoidal region around $\hat{z}(t)$ can be defined as

$$\begin{aligned} V_t(\gamma) &= \{z : [z - \hat{z}(t)]^T S^{-1}(t)[z - \hat{z}(t)] \leq \gamma\} \\ &= \{z : \alpha^T(t) S^{-1}(t) \alpha(t) \leq \gamma\} \end{aligned} \tag{3.4}$$

The parameter $\gamma$ is derived from the chi-square distributed left hand side of the inequality in Equation (3.4). It describes the probability mass captured by $V_t$. The chi-square distribution has two degrees of freedom, which is due to the dimensionality of the measurements. Choosing $\gamma = 16$, the probability that the true measurement $z(t)$ will fall into $V_t$ is larger than 99.9 %. As a consequence, only detections within the validation region are considered as "close" and are associated with the corresponding track.

From the set of tracks, eye pairs are generated with the following constraints:

- Left eye is left of right eye.
- Eye distance is larger than a minimum of 15 pixels.
- Left and right eye move into a similar direction, i. e., the angle between their tracks is smaller than 30 degrees.
- Left and right eye move at similar speed, i. e., the average speed over the whole track length does not differ by more than 3 pixels per frame .

At this point, the number of possible eye candidates is already greatly reduced. To verify the eye pair hypotheses, the image is first rotated, so that the eye positions are on horizontal line. Next, a face detector is used to confirm or discard the hypothesis. The rotation is necessary because the face detector is restricted to upright faces. Without that restriction, the false positive rate would strongly increase as in the eye detector case. If the face detector is successful, the Kalman filters are initialized accordingly. As a fallback solution, eye candidates trigger the Kalman filter initialization if they appear consistently over a long time. On the one hand, this is necessary because the face detector may still fail on an upright face. On the other hand, it is possible because normally only the true eye locations are consistently detected over a longer period of time. The face detector approach is able to succeed within three frames while the fallback solution is triggered after successful detection of a valid eye pair over 15 frames.

Since the data collection process does not need to register the faces, at least not at the lowest level of recording video sequences, the face recorder is backed up with a face detector cascade. In addition to successful eye tracking initialization, a hit of this detector triggers a recording as well. This accounts for people, whose eyes can not be detected for some reason, e. g., due to occlusion by hair. Nevertheless, these variations are included into the database to challenge future systems.


***Figure 3.5:*** *Sample face images (a) generated with the proposed system and data and (b) four images from the AR and three images from FRGC v2.0 face databases for comparison of data quality (see Section 1.4; Martinez and Benavente, 1998; Phillips et al., 2005)*

### 3.2.2 Registration

Despite the fact that the eye detector is trained to account for some amount of rotation, it still works best on horizontal eyes, i. e., upright faces. Therefore, the detection results can be greatly improved if the image is rotated based on the Kalman filter prediction prior to detection. If eye detection fails nevertheless, the prediction can be used as substitute. For registration, the face image is rotated to bring the detected or predicted eye locations into horizontal position. Afterwards, the image is scaled and cropped to a size of $64 \times 64$ pixels, so that the eyes are located at certain coordinates in the resulting image. Figure 3.5 shows some samples obtained with this method.

### 3.2.3 Training set augmentation

In order to increase the amount of training data and to reduce the effect of possible registration errors caused by imprecisely detected eye locations, the training data is augmented with so-called *virtual* samples. These are generated by artificial perturbations of the originally detected eye locations by $\pm 2$ pixels in $x$- and $y$-direction. The face is then aligned according to these new coordinates and stored in the training database. Since nine locations per eye are evaluated, this increases the training set size by factor 81, which allows to build meaningful models even for people with little training data.

The preprocessing of the video data is complete at this point and the following section will explain how feature vectors are extracted from the registered faces.


***Figure 3.6:*** *Reconstruction of a face using different numbers of AC coefficients. DC is included for better visualization. The right-most image corresponds to the original image.*

## 3.3 Local appearance-based face model

Appearance-based models are characterized by the usage of pixel intensities as features for recognition purposes rather than geometric features. For the reasons explained in Section 2.3, the discrete cosine transform was chosen to implement this approach.

Research by Ekenel and Stiefelhagen (2005) showed that traditional holistic approaches like PCA are outperformed by local models as these are less affected by local variations. Therefore, DCT is applied to non-overlapping blocks of $8 \times 8$ pixels in size. This size allows sufficient compression while offering a reasonable constancy towards local variations within each block. Due to the locality of the approach, proper alignment of the face image is crucial to ensure that the same parts of each face get encoded in the same block. Local feature vectors are built by zig-zag scanning the coefficients. The DC coefficient is omitted, as it only represents the average gray value of the block and is therefore strongly affected by illumination changes. From the remaining, only the first five coefficients are selected. These capture a major part of the block's information while the dimensionality of the global feature vector is kept in reasonable bounds. Figure 3.6 visualizes image reconstructions from reduced numbers of coefficients to confirm this decision. Normalization of the local feature vectors to unit norm further increases robustness against lighting variations (Ekenel and Stiefelhagen, 2006b). The global feature vector, finally, is simply the row-wise concatenation of the local ones. Thus, the input dimensionality is reduced from $64 \times 64 = 4096$ to 320.

It is important not to confuse these features with structural features which are used, for example, in component-based face models like eye, mouth and nose positions (e. g., Heisele et al., 2001a). The DCT only transforms the data to a different representation which is more compact than raw image intensities. Therefore, the resulting feature vector leads to an appearance- rather than a feature- or component-based model.

## 3.4 Classification

This section will present the two concepts of classification: A k-nearest-neighbor model (Section 3.4.1) and a Gaussian mixture model (Section 3.4.2). In parallel, the ap-

proaches to temporal fusion are introduced. While feature fusion is performed on the frame level, decision fusion is used to classify the whole sequence. Both classifiers are trained person-wise, i. e., one mixture model and one set of representatives are generated for each subject individually. This allows easy extension or, if necessary, reduction of the database without the need to retrain the whole system.

### 3.4.1 K-nearest neighbor model

The K-nearest neighbor (KNN) model is based on representatives as introduced in Section 2.5. These are selected from the training set using the well-known k-means clustering algorithm (see Section 2.4). The number of clusters for a certain person depends on the number of training samples available. Like this, more accurate models can be built for people who "use" the system more often because more training data implies a larger bandwidth of captured variations. The resulting cluster means are then used as representatives.

At runtime, the ten representatives $S_i$, $i = 1,2,\ldots,10$, which are closest to the test vector $x$, are selected with score $s_i = d(x,S_i)$. The $L_1$-Norm or *city block distance*

$$d(x,y) = \|x-y\|_1 = \sum_{j=1}^{D} |x_j - y_j| \tag{3.5}$$

with $D$ being the dimensionality of the feature vectors, is used. Because distances can differ largely between frames, they need to be normalized. This is achieved with linear *min-max normalization* (Snelick et al., 2005),

$$s_i' = 1 - \frac{s_i - s_{\min}}{s_{\max} - s_{\min}} \qquad i = 1,2,\ldots,10 \tag{3.6}$$

which maps the scores to $[0,1]$. To have equal contribution of each frame, these scores are re-normalized to $\sum_{i=1}^{10} s_i' = 1$. Of course, among the ten closest representatives, there can be several ones of the same class. Since some people have far fewer representatives than others, care must be taken that their scores are not dominated by those. Individual scores are selected by a simple max-rule, which only selects the maximum score for each class. The popular sum-rule (Kittler et al., 1998), which adds up the scores for each class, is no alternative due to the heterogeneity of individual training set sizes. It can distort the classification results and prevent small classes from being recognized.

An example shall help to clarify this. Assuming a simple database consisting of one representative of person A and nine representatives of person B, a test vector is presented to the classifier. The test sample is assumed to have a distance of 80 to class A, and of $90,95,\ldots,130$ to class B, respectively. These distance values are chosen in anticipation of Figure 3.7(a), which shows the observed distribution of distances to correct and false classes. It is obvious, that the selected values express a reasonable dissimilarity between person A and B. The expected classification is "A". The normalized

score of class A is 0.22, while the scores for class B range from 0.17 to 0. The max-rule would now select class A over class B, while the sum-rule would clearly decide in favor of class B with a total score of 0.78.

### Temporal fusion

In order to fuse the scores of multiple frames, three approaches are evaluated. Except for the first, the decision is based on the final score after all frames have been processed.

**No fusion.** In this case, every single frame is evaluated on its own. It is used to determine the baseline performance of the system. It will be referred to as *Frame-KNN*.

**Simple sum** This approach accumulates frame scores over time. The decision will then be made on the final score. It will be referred to as *Video-KNN*.

**Weighted sum** Since not all frames are of the same quality and some might be more ambiguous than others, a weighting is introduced in order to penalize uncertain frames. Two observations are used to determine a frame's weight.

1. For wrong classifications, the distance to the closest representative is, on average, larger than for correct ones. Moreover, badly aligned frames result in larger distances as well. To account for this, the frames $f_i$, $i = 1,2,\ldots$, are weighted with respect to the closest representative $c$ with

$$w_{\mathrm{DTM}}(f_i) = \begin{cases} 1 & \text{if } d(f_i,c) < \mu \\ e^{-\frac{d(f_i,c)-\mu}{2\sigma^2}} & \text{otherwise} \end{cases} \tag{3.7}$$

   This weighting function is chosen according to the observed distribution of frame distances $d(f_i, c_{f_i,\mathrm{correct}})$, the distances of all frames $f_i$ to the closest representative $c_{f_i,\mathrm{correct}}$ of the corresponding correct class. The distribution, determined on a parameter estimation set, resembles a normal distribution $\mathcal{N}(x;\mu,\sigma^2)$. To increase robustness against outliers, $\mu$ is chosen as sample median, $\sigma^2$ as median absolute deviation (MAD, Huber, 1981) An example distribution and weight function is shown in Figure 3.7. Using the weight function $w_{\mathrm{DTM}}$, the influence of frames which are not sufficiently close to the model is reduced. This weighting scheme will be referred to as *distance-to-model (DTM)*.

2. In case of misclassification of frame $f_i$, the difference of the distances $\Delta(f_i)$ to the closest and second closest representatives is generally smaller than in the correct case. The distribution of these distances follows approximately an exponential distribution

$$\varepsilon(x;\lambda) = 0.1\lambda e^{-\lambda x} \qquad \text{with } \lambda = 0.5 \tag{3.8}$$

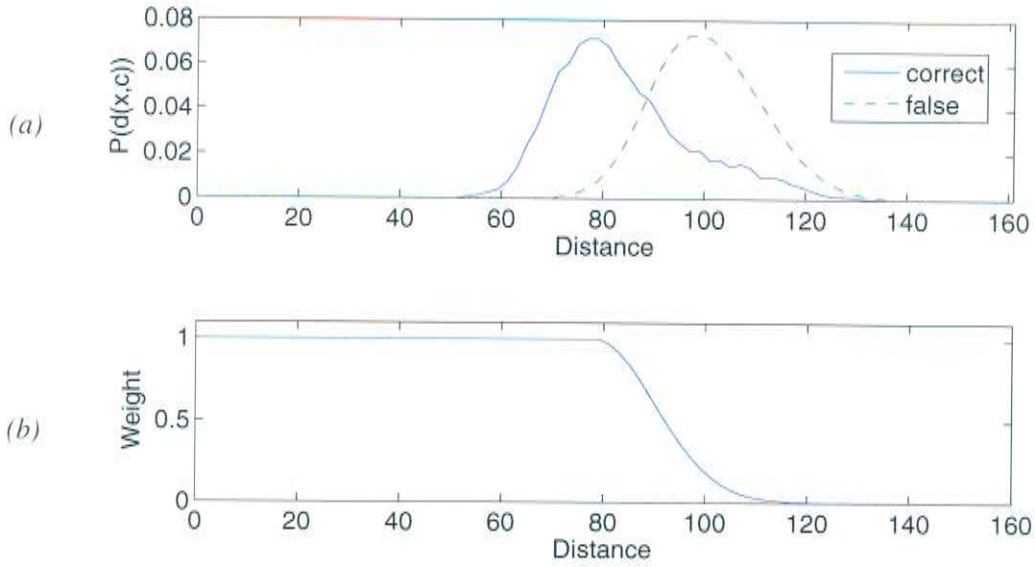



***Figure 3.7:*** *DTM weight function. (a) Distribution of the distances to the closest representative of the correct class (blue, solid) and to all other classes (green, dashed) and (b) the actual weight function.*

The weights are then computed as the cumulative distribution function of $\varepsilon(\cdot)$

$$w_{\mathrm{DT2ND}}(f_i) = \mathcal{E}(\Delta(f_i)) = 1 - \mathrm{e}^{-\lambda\Delta(f_i)} \tag{3.9}$$

An example distribution and weight function is shown in Figure 3.8. This weighting scheme will be referred to as *distance-to-second-closest (DT2ND)*.

This fusion approach will use the product of the so-computed weights to modify a frame's share in the final score. It will be referred to as *Weighted-KNN*.

### 3.4.2 Gaussian mixture model

The Gaussian mixture model approach trains one GMM per class using the EM algorithm. Likewise the KNN model, the number of components per mixture depends on the number of training samples available for a person. At runtime, person $x$ is classified as one of the $N$ registered individuals in a maximum log-likelihood manner using

$$\arg\max_{i \in N} \log P(x|i) = \arg\max_{i \in N} \log \sum_{j=1}^{k_i} \alpha_{ij} \cdot \mathcal{N}(x; \mu_{ij}, \Sigma_{ij}) \tag{3.10}$$

To keep the computational effort within reasonable bounds, only a diagonal rather than the full covariance matrix is used.

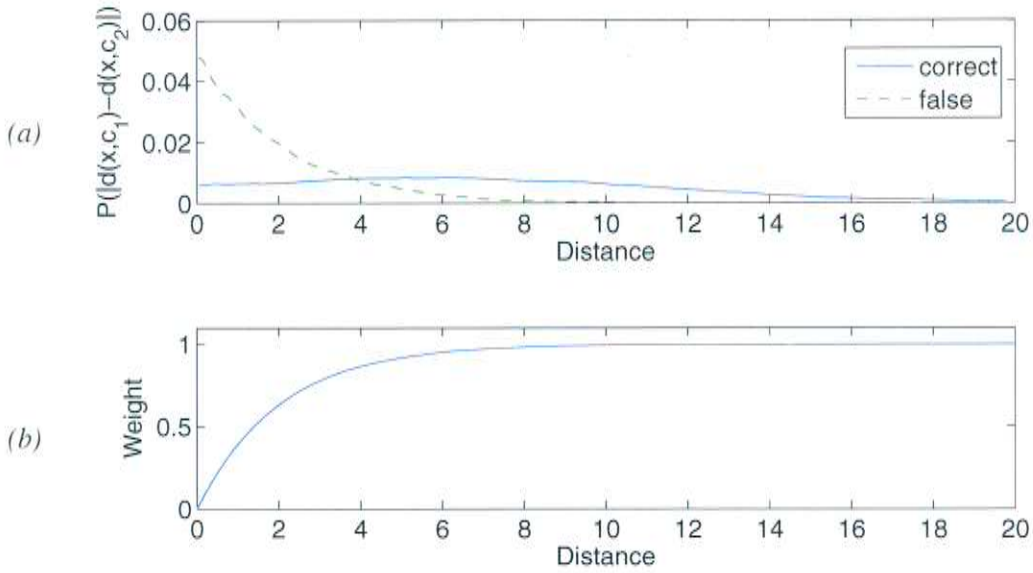



***Figure 3.8:*** *DT2ND weight function. (a) Distribution of the distances between the closest and second closest representatives for correct (blue, solid) and false classifications (green, dashed) and (b) the actual weight function.*

### Temporal fusion

Again, three approaches are employed to evaluate the classification performance of the GMM setup on video input. As above, the classification of a sequence is made on the final score.

**No fusion** Similar to the KNN model, this approach determines the baseline performance of the model. Every frame is evaluated on its own based on a min-max normalization. The GMM outputs represent a proximity instead of a distance measure, i. e., the more similar the input is to the modeled class, the higher are the GMM outputs. Thus, only the fraction in Equation (3.6) is used, instead of the difference, to perform the normalization. It will be referred to as *Frame-GMM*.

**Bayesian inference** Using Bayes' rule, posterior probabilities are computed for each class. These posteriors are used as priors in the next frame. The posterior probability $P(i_t|x_{0:t})$ of person $i$ at frame $t$ given the all the previous observations $x_{0:t}$ is calculated as

$$P(i_t|x_{0:t}) = \frac{P(x_t|i_t) \cdot P(i_t|x_{0:t-1})}{P(x_t)} \tag{3.11}$$

The conditional observation likelihood $P(x_t|i_t)$ is computed by the GMM for person $i$, the unconditional one by

$$P(x_t) = \sum_{i=1}^{N} P(x_t|i_t) \cdot P(i_t|x_{0:t-1}) \tag{3.12}$$

with $N$ being the number of individuals. The priors are initialized uniformly, i. e.,

$$P(i|x_0) = \frac{1}{N} \tag{3.13}$$

This approach takes into account the temporal dependency by computing the probability to observe a given sequence of input frames. It will be referred to as *Video-GMM*.

**Bayesian inference with smoothing** Based on the previous approach, the idea of a consistent identity is introduced as suggested by Zhou et al. (2003). The identity of an entering person does not change but depending on frame and model quality the classification of single frames can differ from previous ones. As a consequence, the influence of frames which are not consistent with the current sequence hypothesis, i. e., the current classification for a given sequence, is reduced. Extending Equation (3.11), the smoothed posteriors are calculated as

$$P(i_t|x_{0:t}) = \frac{P(x_t|i) \cdot P(i_t|i_{t-1}) \cdot P(i_t|x_{0:t-1})}{P(x_t)} \tag{3.14}$$

with

$$P(i_t|i_{t-1}) = \begin{cases} 1-\varepsilon & \text{if } i_t = i_{t-1} \\ \frac{\varepsilon}{N} & \text{otherwise} \end{cases} \tag{3.15}$$

The amount of smoothing is determined by the smoothing parameter $\varepsilon$, where smaller values denote stronger smoothing. With a value of 0, the sequence is basically classified solely based on the first frame. Nevertheless, values *close* to 0 lead to a stabilization of the sequence hypothesis while still allowing a change to a different identity as the experiments in Section 4.2.8 will show. Further on, this approach will be referred to as *Smooth-GMM*.

## 3.5 Region-of-interest tracking

When somebody enters the room, it is likely that his or her face and especially the eyes are not visible in every single frame. Most of the time, this is due to the person turning sideways or looking down for a moment, but it can also be caused by fast movement which blurs the person's appearance. Both for data collection and recognition, it would be unfortunate, if the system lost track of the person at that point. As far as data collection is concerned, this would lead to a premature end of the recording, resulting in an incomplete video sequence. Additionally, if the person faced the camera again, a new recording would be triggered, yielding two video fragments of the individual entering the room. In the case of recognition, this leads to two classification results and depending on the combination of good and poor frames, the classification may get unstable

since less data per sequence is available for evaluation.
The Kalman filters, which track the eyes, cannot save this situation because the person might change the direction of movement while facing away from the camera. The time that passed since the last correction of the Kalman filters will become too large, and the tracking results will become unusable.
Nevertheless, there is a simple method to resolve the stated problem. Since the camera runs at a speed of 25 frames per second (fps), the positional difference of a person's location between frames is quite small. Complementary, some skin from cheeks, ear, neck or forehead usually remains visible. This is even true when the person turns around completely, given the neck is not occluded by hair or clothing.
So, if both the eye prediction and detection, and therefore the confirmation of the face candidate, fail, but the skin region overlaps the last confirmed detection more than a certain extent, this skin region is considered to represent the formerly successfully detected face. Although this frame can not be further evaluated in terms of recognition, at least the position of the last confirmed face can be updated accordingly. To account for the uncertainty introduced by this approach, the search area is slightly increased into all directions. With this basic tracking approach, an interruption of the recording or recognition procedure can be avoided in most cases. Furthermore, it is still possible to concentrate processing on a small region of the subsequent frame instead of the whole image to save processing time.

# 4 Experiments

This chapter will give insights into the structure of the utilized data sets and evaluate the performance of the system in the given scenario. The first part, Section 4.1, examines the quality of the automatic data collection process. Afterwards, Section 4.2 gives details about the collected data and analyzes the system's recognition performance from different perspectives.

## 4.1 Face recorder

The crucial part in evaluating the performance of the face recorder is to determine how many of the people entering the room are actually detected and recorded. Besides, it is necessary to examine how much of the genuine sequence is finally captured. Like the recognition task, subjects are non-cooperative and are allowed to move and behave naturally.

### 4.1.1 Experimental setup

Due to the unobtrusive manner in which data was collected, single or groups of persons can face the camera while just standing in or passing through the camera's field of view. This triggers the detection and recording system in addition to individuals that enter the room. Since the focus of this system lies on the latter, other face appearances need to be filtered. To achieve this, continuous video is manually labeled to flag sequence parts in which people enter the room. This is the ground truth for the following evaluation.

The performance of the face recorder is assessed with two measures: first, the number of detected sequences and second, the average overlap of the detected sequences with the genuine ones. This overlap is a percentage with respect to the length of the ground truth labeled data, i. e., an overlap of 100 % means that the detected sequence covers at least all frames of the genuine one.

#### Data set I

For this evaluation, four continuous video streams were recorded on three different days and manually labeled for ground truth. They cover a time frame of 16.5 hours and consist of approximately 1.5 million frames. Table 4.1 gives a detailed overview. Looking at the share of less than one percent of relevant data within the recorded video,

| Sequence | Duration (hh:mm:ss) | Total no. of frames | No. of sequences | No. of relevant frames |
|---|---|---|---|---|
| A | 02:53:16 | 259,910 | 42 | 2,929 |
| B | 04:04:13 | 366,318 | 12 | 3,233 |
| C | 03:25:25 | 308,124 | 12 | 989 |
| D | 06:07:38 | 551,443 | 63 | 6,220 |
| Total | 16:30:32 | 1,485,795 | 129 | 13,371 |

***Table 4.1:*** *Overview of data set I. The number of sequences refers to situations in which somebody is actually entering the room.*

it is obvious that continuous recording is not an option for sensible data collection, not only concerning memory requirements but especially in terms of effort and time-consumption of tedious manual segmentation.

### 4.1.2 Detection results

The results in Table 4.2 are given as *correct detection rate (CDR)* and *false detection rate (FDR)*. CDR denotes how many of the labeled sequences have been successfully detected by the system. FDR represents the share of falsely detected sequences among the detected ones. A correct detection is given if a detected sequence overlaps at least 50 % of a labeled one. The total CDR for different overlap values can be read off Figure 4.1. Since the results are computed on sequence level, they cannot be directly compared to results achieved by, e. g., Viola and Jones (2001) who measure the performance on the total number of processed positive and negative sub-windows in all images.

## 4.2 Face recognizer

In this section, following the analysis of the available data, the closed-set and open-set identification tasks are introduced, together with the corresponding performance measures. The remaining subsections extensively evaluate the performance of the different classification models introduced in Chapter 3.

| Sequence | CDR (%) | FDR (%) |
|---|---|---|
| A | 92.9 | 9.3 |
| B | 83.3 | 0.0 |
| C | 100.0 | 0.0 |
| D | 95.2 | 9.1 |
| **Total** | **93.8** | **7.6** |

*Table 4.2*

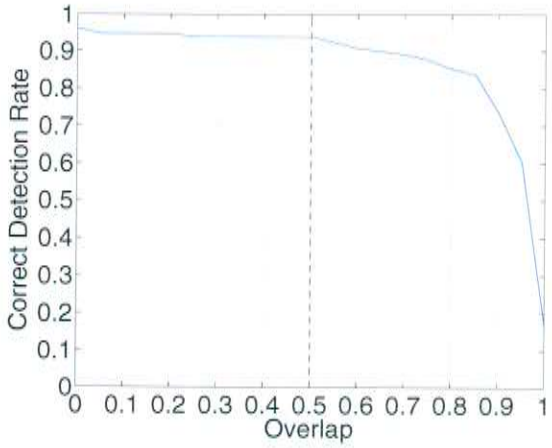



*Figure 4.1*

*Detection performance of face recorder. Results in the table are given as correct detection rate (CDR) and false detection rate (FDR). These measures are based on sequences rather than frames with an overlap of at least 50 percent. The diagram shows the CDR dependent on the overlap.*

### 4.2.1 Experimental setup

#### Data set II

Data set II is made up of videos of 41 persons. The number of sequences per person varies between 5 and 250. This large span is a consequence of real-world data collection since some people happened to enter the room more often than others. Data has been collected over a time period of seven months, resulting in 2,292 sequences altogether. The data is divided into three sets for training, parameter estimation and evaluation according to the recording date. This reflects a realistic usage scenario, in which the recognition system is trained once and afterwards confronted with new data. Since the sequences of each person are not equally distributed over the data collection time span, the split date is determined for each person individually.

For each person, the training set contains two thirds of the available data, but not more than 35 sequences. This allows for reasonable amount of variation within the training data while not making it unnecessarily large. In addition, while still favoring common people with a more detailed model, it avoids the generation of redundant model components which can lead to less common people being dominated. In fact, the number of sequences is only a rough estimate for the training set size, as it does not take into account the length and the quality of these sequences. The actual training set consists of the face images extracted from them. The automatic extraction process produces approximately five percent of misaligned and therefore unusable images, which are sorted out manually. In the cleaned set, the number of samples per person ranges approximately from 80 to 1,500. The complete basic training set consists of 21,875 im-

| | Number of sequences |
|---|---|
| Training set | 905 |
| Parameter set | 386 |
| Test set | 1,001 |
| Total | 2,292 |

***Table 4.3:*** *Data set II: Sizes of the three subsets. Splits are made person-wise. A listing by individual can be found in Appendix A.*

ages. Taking into account the augmentation with virtual samples, the final training set counts around 1.77 million elements. While this seems a lot on first sight, the feature vector size of 320 dimensions requires a large amount of training data in order to be able to build meaningful models.

Approximately one third of a person's remaining data is used to create a parameter estimation set in order to compute the weights for the Weighted-KNN scheme. Due to the small amount of available data for some persons, this set does not contain sequences of everybody. Nevertheless, it is general enough to model the parameters appropriately. The remaining data forms the test set for evaluation purposes. The set sizes are listed in Table 4.3. A more detailed analysis of data set II can be found in Appendix A, specifying the set sizes and number of registered faces per individual as well as the distribution of individual sequence lengths. Appendix B lists the model sizes for each person.

### Recognition tasks

In order to evaluate the recognition performance, two tasks are presented to the system.

**Closed-set identification** This task shows the baseline performance of the system. Given that a person is registered in the database, the system needs to classify him or her. The performance is measured as *correct classification rate* (CCR). For the frame-based approaches, it is computed over all frames in which the eyes could be successfully either detected or tracked and therefore the face could be registered. The CCR of the video-based approaches is computed as percentage of correctly recognized sequences in the test set.

**Open-set identification** This task extends the previous one by the difficulty that unknown people, i. e., persons which are not registered in the database, can be encountered. Therefore, prior to classification as one of the possible identities, the system has to decide whether a person is known or unknown. Impostors are to be rejected, while genuine members of the database need to be accepted and classified correctly. To model this task with the existing data set, the system is trained

in a leave-one-out manner. One person at a time is removed from the database and is presented to the system as impostor during the subsequent evaluation on all sequences. This process is repeated $N$ times, so that each person takes the impostor role once. The acceptance-rejection criterion is a threshold on the confidence of the classification, which is a value between 0 and 1. If the confidence value is not high enough, the person is rejected.

For the video-based KNN models, a measure of confidence of the classification is derived by min-max normalization (see Equation (3.6)) of the accumulated scores at the end of the sequence. The Frame-KNN and Frame-GMM scores are already normalized and can serve as confidence measure without further processing. This applies for the video-based GMM approaches as well since they compute probability scores.

Compared to closed-set identification, two more error types can occur. Additional to false classifications, the system can erroneously either reject genuine identities or accept impostors. All three errors have to be traded-off against each other as it is not possible to minimize them at the same time. For this reason, a different performance measure is necessary. The employed *equal error rate (EER)* denotes the minimum combined error rate. It is reached when

$$\mathrm{FAR} = \mathrm{FRR} + \mathrm{FCR} \tag{4.1}$$

i. e., when the *false acceptance rate (FAR)* among the impostors is equal to the sum of the *false rejection rate (FRR)* and the *false classification rate (FCR)* among the registered persons. The rates are defined as

$$\mathrm{FAR} = \frac{n_{i,\mathrm{accepted}}}{n_i} \tag{4.2}$$

$$\mathrm{FRR} = \frac{n_{g,\mathrm{rejected}}}{n_g} \tag{4.3}$$

$$\mathrm{FCR} = \frac{n_{g,\mathrm{misclassified}}}{n_g} = 1 - \mathrm{CCR} \tag{4.4}$$

where $n$ denotes number of frames or sequences and the subscripts $g$ and $i$ denote genuine or impostor samples, respectively.

### 4.2.2 Comparison of frame- and video-based recognition

To evaluate the performance improvement gained by using video sequences instead of single frames for recognition, the system is tested on closed-set identification. Results for the open-set case follow, starting with Section 4.2.6. Table 4.4 clearly indicates a major performance increase independent of the underlying model. It is to be remarked, that the approximately 38,000 frames used in the frame-based recognition are exactly

| | Frame-based | Video-based | Weighted | Smooth |
|---|---|---|---|---|
| KNN | 68.4 % | 90.9 % | 92.5 % | - |
| GMM | 62.7 % | 86.7 % | - | 87.8 % |

***Table 4.4:*** *Comparison of frame- and video-based recognition. Smooth-GMM uses* $\varepsilon = 10^{-5}$.

the same as in the video-based one. So, the *registration* of the faces is still video-based because eyes are tracked over time, but the *classification* is done on individual frames.

Among the 1,001 test sequences, there were 58 in which no eye pair could be detected and which could therefore not be recognized. This represents a share of 5.8 % of the test data. This is a good result as it indicates that eye detection is successful in more than 94 % of the cases across the full bandwidth of variations.

As far as the different models are concerned, the discriminative approaches perform better than the generative ones. There are two possible reasons for this. First, a generative model is a parametric model and the required parameters need to be learned from training data. The problem is, that the higher the dimensionality of the feature vectors, the more training data is necessary to allow meaningful model generation. While there is a lot of training data for some persons, there is little for others. As will be further explained in Section 4.2.4, a lack of training data affects the individual recognition performance. The second possible reason for the lower performance of the GMM-based models compared to the KNN-ones is the possibility, that the model did not generalize well because the training data might not be representative after all. Again, this is very likely to be the case for persons with little training data. As a consequence, the model adapted to peculiarities of the training set. The discriminative models, in contrast, try to classify new data only based on the existing data, without making any assumptions about its distribution. It is less affected by little training data, given that the available data sufficiently spreads out. If this is not the case, i. e., all examples cluster closely together, this approach will fail as well. However, the results show that the available amount of data is more appropriately modeled using a discriminative approach rather than a generative one.

### 4.2.3 Recognition rate by rank

To investigate the robustness of the results, it is worth looking at the results including rank-2 and rank-3 classifications, i. e., cases in which the correct identity is among the best two or three hypotheses. As clearly depicted in Figure 4.2 and Table 4.5, the frame-based approach often gets close to the correct decision. However, it has to decide on the identity even in the case that the single feature vector is of questionable quality. The approach lacks an opportunity to support or discard the hypothesis using additional data as done by the sequence-based methods. These are able to exploit the temporal dependency of consecutive frames and to promote the rank-2 and rank-3 classifications of

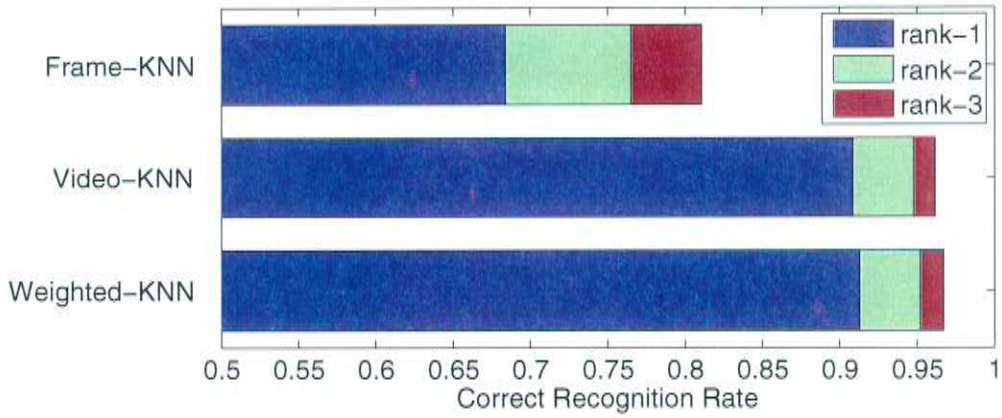



***Figure 4.2:*** *Correct recognition rate by rank for the KNN models.*

| | Frame-KNN | Video-KNN | Weighted-KNN |
|---|---|---|---|
| rank-1 | 68.4 | 90.9 | 92.5 |
| rank-2 | 76.5 | 94.8 | 95.6 |
| rank-3 | 81.1 | 96.2 | 96.7 |

***Table 4.5:*** *Correct recognition rate by rank for the KNN models.*

the frame-based models to first place. Since many frames contribute to the decision, the overall performance improvement is larger than the difference between the correct and rank-3 classifications in the frame-based approach. The more frames can be evaluated, the more likely it is to obtain a correct result. This gets confirmed by the observation that the average length of correctly classified sequences is larger — 39 frames — than that of misclassified ones with 28 frames as depicted in Figure 4.3. If the frames of a short sequence are additionally of bad quality, e. g., caused by low resolution or misalignment, or the input generally differs largely from the modeled data, the resulting scores for each identity are not expressive. In these cases, their ranking allows, at most, vague assumptions, if at all, about the identity of the person in question. This explains the smaller performance gain for video-based data if rank-2 and rank-3 classifications are included, compared to frame-based approaches.

### 4.2.4 Recognition rate by subject

The training set contains considerably differing amounts of training data for each individual. To answer the question how this affects the recognition performance, Figure 4.4 shows the correct classification rate per person. Unfortunately, the system fails to recognize five people in the test set completely. However, as can be seen from Table 4.6, highlighted in dark gray, there is very little test data available for these people.

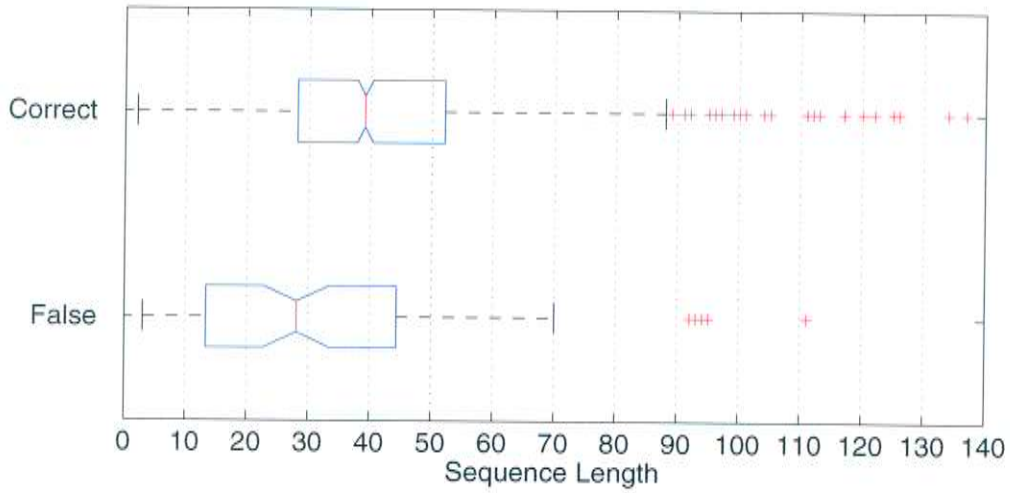



***Figure 4.3:*** *Box plot showing the distribution of sequence lengths for correct and false classifications. It is based on results achieved with the Weighted-KNN approach. The diagram depicts median, upper and lower quartile, spread of the data and outliers.*

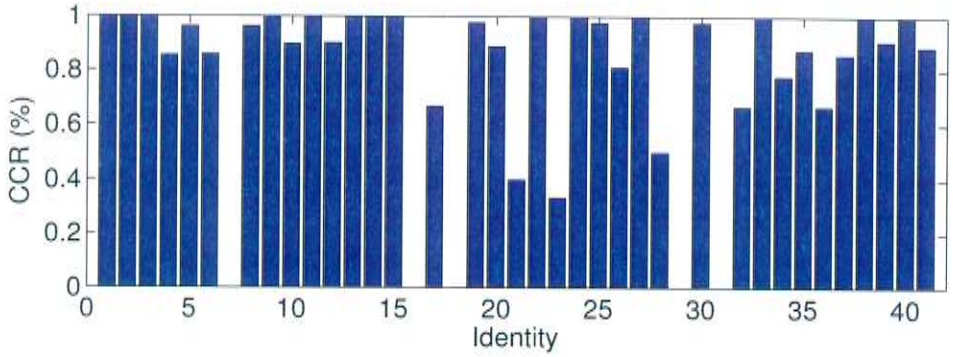



***Figure 4.4:*** *Correct recognition rate by subject based on Weighted-KNN*

| Identity | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| No. of sequences | 3 | 2 | 2 | 90 | 53 | 7 | 3 | 28 | 3 | 40 | 5 | 32 | ... |

| ... | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ... | 5 | 33 | 5 | 2 | 3 | 2 | 179 | 96 | 11 | 50 | 3 | 6 | 49 | 16 | 31 | ... |

| ... | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ... | 7 | 2 | 45 | 2 | 3 | 51 | 13 | 9 | 22 | 7 | 20 | 11 | 41 | 9 |

***Table 4.6:*** *Overview of test set size by person. Dark gray denotes persons which are not recognized at all, light gray highlights individuals with CCR <80 % (cf. Figure 4.4). These number include the missed sequences (see Appendix A for detailed listing).*

Apart from implying that there is only little training data available as well, this means that results can break down easily since a single sequence contributes a third or even half of a person's classification rate. Besides, if test data happens to be from the same day, it is likely that recognition performance is similar among the sequences due to similar recording conditions in terms of illumination and appearance. Therefore, recognition can fail for a complete subset.

Nevertheless, small amounts of data do not necessarily entail bad recognition performance as can be seen from many examples in Figure 4.4 and Table 4.6. Furthermore, looking at the person-wise results, one can see that more than 70 percent of the people are correctly classified with a CCR above 80 percent. Persons with lower results are highlighted in light gray in the table.

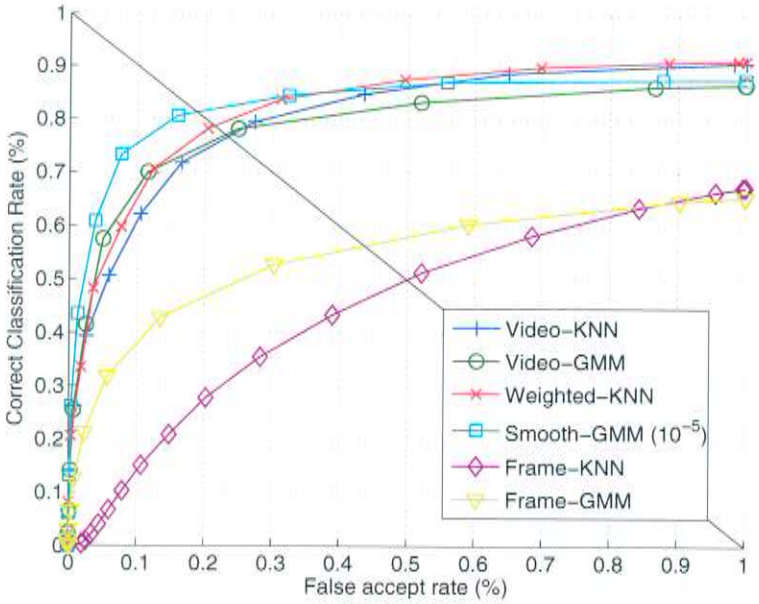
Persons 21, 34 and 36 deserve some special attention. On first sight, they seem to have quite low recognition rates compared to the available amount of data. But looking at the distribution of missed sequences in the test set (see Appendix A), it gets clear that these individuals are missed most compared to the other registered persons. The recognition results are only based on 5, 9 and 3 sequences, respectively. The number of extracted training images suggest, that the miss rate within the training set is similar. Especially individuals 21 and 36 have a very small number of training images, taking into account the number of available sequences. As a consequence, the models do not capture the appearance of these individuals well. The problems are caused, in all three cases, by the persons' haircut which often partly occludes one or both eyes. Actually, this affects the results of "good" sequences as well, as it implies that, most likely, the eyes could only be detected in and tracked over a small number of frames. So there is less evidence available for classification and the result will be less stable and more likely to be wrong.

### 4.2.5 Influence of data set augmentation

To justify the increased training efforts caused by the larger training set size, an experiment was conducted to compare the recognition performance using augmented and unaugmented training data. The comparison can only cover the KNN models as it is not possible to train an appropriate GMM. This is due to the fact that many individuals have fewer images in the training set than the feature vector's dimensionality. As listed in Table 4.7, recognition performance increases significantly in all three KNN cases. This shows that the data augmentation is well worth the increased memory and time resources. Adding noise to detected eye locations leads to samples of different scale and rotation which increases the variation bandwidth and reduces the influence of possible registration errors. Since the data set size is increased by factor 81, even persons with few genuine training images can be modeled appropriately.

| | Frame-KNN | Video-KNN | Weighted-KNN |
|---|---|---|---|
| genuine training set | 56.6 % | 87.6 % | 88.2 % |
| augmented training set | 68.4 % | 90.9 % | 92.5 % |
| *significantly better* | ✓ | ✓ | ✓ |

***Table 4.7:*** *Influence of data set augmentation with virtual samples. All results improve significantly with a significance level of 0 %, 2 % and 0 %, respectively. Significance was computed with crosstabulation.*




| | EER (%) |
|---|---|
| Video-KNN | 23.4 |
| Video-GMM | 23.0 |
| Weighted-KNN | 21.0 |
| Smooth-GMM | **18.7** |
| Frame-KNN | 50.0 |
| Frame-GMM | 43.7 |

***Figure 4.5:*** *Recognition performance on the open-set identification task. The black line denotes equal error.*

### 4.2.6 Open-set identification

This and the following subsections will examine the open-set identification performance of the proposed system. Figure 4.5 gives an overview of the results of the models discussed in Section 3.4. A thorough investigation of the different frame-weighting methods and of differently parametrized Smooth-GMM models will follow in the subsequent sections. The ranking of the results presented here is similar to the closed-set results. As expected, the frame-based approaches deliver the worst results. Weighted-KNN slightly outperforms Video-KNN. Video-GMM performs worst of the video-based approaches. The exception is Smooth-GMM, which performs worse than both video-based KNN models for FARs of more than 60 %, but clearly provides the best performance in terms of EER.

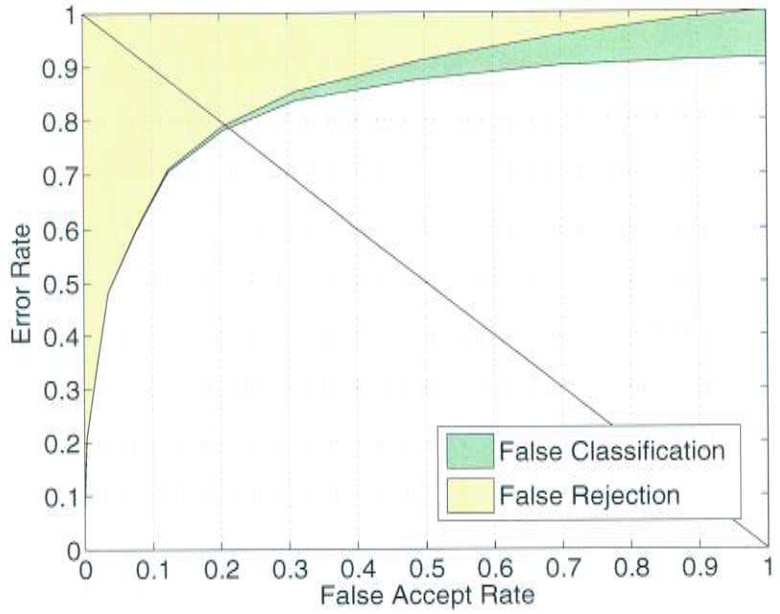



***Figure 4.6:*** *Analysis of the contribution of FRR and FCR to the overall error rate of Weighted-KNN. As earlier, the black line denotes equal error.*

The *receiver operating characteristic (ROC)* curve in this figure shows the CCR depending on the percentage of impostors accepted to the system. Hence, the question arises to which extent each of the other two types of error, false rejection and false classification, impairs the classification performance. To investigate this, Figure 4.6 plots FRR and FCR separately for Weighted-KNN. The lower bound corresponds to the CCR as depicted in Figure 4.5. At the point of equal error, there remains only a minimal FCR of about 1 percent while the major part of about 20 percent is caused by the false rejection of genuine identities. This is mostly caused, as above, both by low quality input data and unmodeled variations, caused by the little training set sizes for some persons.

To understand how confidence values represent a sensible measure to decide whether a person is known to the system, it is insightful to investigate how scores develop over time for genuine and impostor identities. In Figure 4.7, scores are plotted for the same test sequence. In the first case, the person is a genuine member of the database. The final score is very distinct from the rank-2 and rank-3 scores which results in a high confidence. In the second case, the person is removed from the database and takes the role of an impostor. As a result, the "rivals" from the first case, score higher but there is no clear winner. In fact, the best three hypotheses lie very close together, even after 45 frames. This leads to a low confidence value and increases the likelihood of rejection, depending on the threshold.

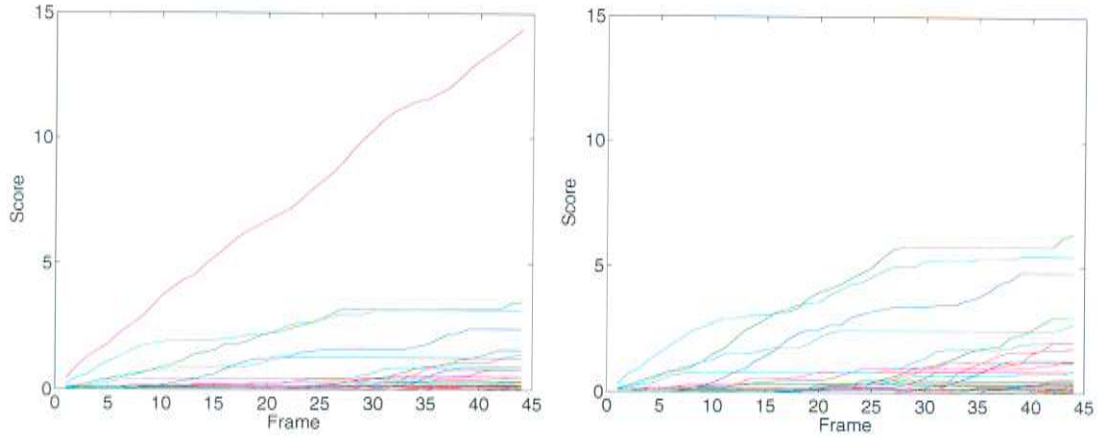



***Figure 4.7:*** *Comparison of score development for (a) genuine and (b) impostor identities derived from the same test sequence with Video-KNN.*

### 4.2.7 Influence of frame weighting

Two different methods have been proposed in Section 3.4.1 to judge a frame's quality and accordingly weight its influence in the classification process. Weighted-KNN uses the combination of the computed weights, for which the results have been already reported above. This section will investigate each of the two weighting schemes on its own and relate it to the unweighted and double weighted cases. Actually, the latter two form a lower and upper bound, as can be seen from Figure 4.8, for the former two. The two weighting schemes affect different parts of the ROC curve. The DTM scheme improves the recognition rate for high false acceptance rates. A FAR of 100 percent is equivalent to closed-set identification in terms of the ROC curve because the CCR can only be computed over genuine samples. In that case, DTM helps to reduce the influence of input that does not fit the model as caused by, e. g., failed registration. Thus, it reduces the false classification rate, but it is not able to discriminate between known and unknown persons as the feature vector of an impostor can be indeed very similar to the model.

Genuine identities, however, usually have smaller distances to one single class representative than to all other classes in the model, while impostors are similarly close to multiple classes (cf. Figure 3.8). This ambiguity is exploited by the DT2ND weighting scheme to identify impostors. Their scores are reduced, leading to smaller confidence values, which in turn result in better rejection. The same threshold causes rejection of more impostors than in the unweighted case or, to put it the other way round, the threshold can be reduced causing fewer false rejections. As a consequence, the EER is reduced in open-set identification.

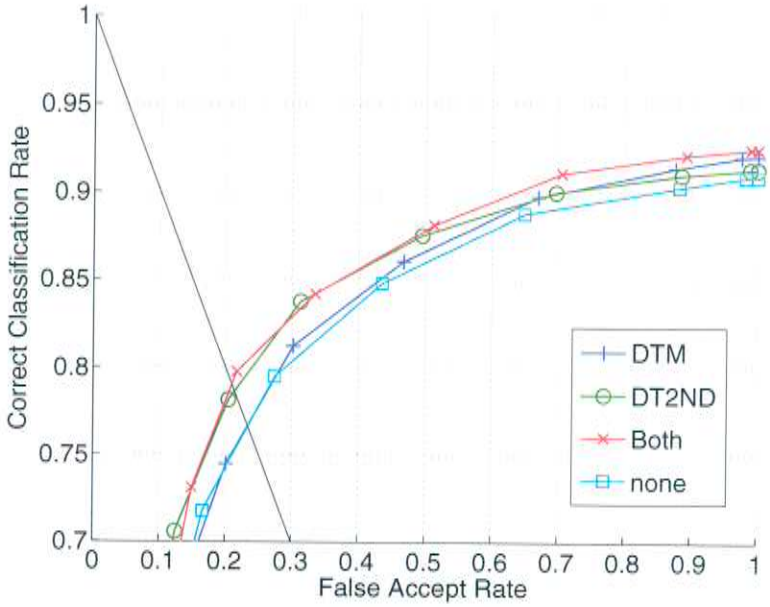



***Figure 4.8:*** *Influence of frame weighting. The black line denotes equal error. Please note the different scaling compared to the previous figures. "Both" corresponds to Weighted-KNN, "none" to Video-KNN.*

### 4.2.8 Influence of smoothing

To examine the effects of the constraint that a person's identity does not change within a sequence, as formulated in Section 3.4.2, the Smooth-GMM model is evaluated at different levels of smoothing. The smaller the $\varepsilon$-value, the stronger is the constraint. As Figure 4.9 shows, a moderate amount of smoothing improves the open-set identification performance. Small numbers of ambiguous and inconsistent frames do not derogate the currently best score while many consistent frames increase the confidence of the decision. As a consequence, a smoothed classification result is more distinct than an unsmoothed one. Since smoothing generally favors sequences with consistent frame hypotheses over ones with inconsistent classifications, it does not necessarily reduce the number of false classifications but the augmented confidence leads to a reduction of the false rejection rate. In contrast to genuine identities, impostors often cause inconsistent frame scores, so that the resulting low confidence leads to a proper rejection. This can readily be seen by comparing Smooth-GMM with Video-GMM and Weighted-KNN in Figure 4.5. As mentioned earlier, Smooth-GMM performs slightly better than Video-GMM, but, nevertheless, it does not reach the CCR of Weighted-KNN. In terms of EER, however, it outperforms all other approaches due to the increased clarity of classification results.

However, if the smoothing factor is chosen too small, the system gets stuck on the decision of the first frames. Even if all subsequent frames are classified as a single

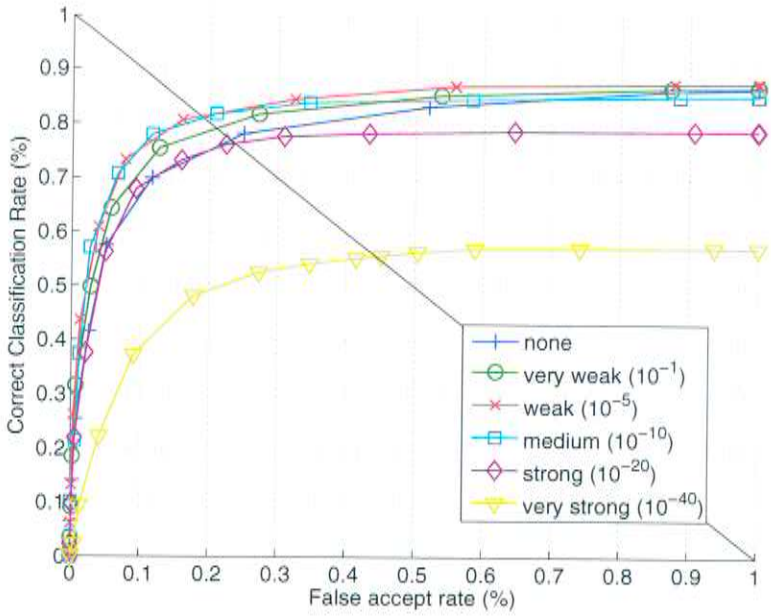



***Figure 4.9:*** *Influence of identity smoothing. The values in parentheses are the $\varepsilon$-values used for smoothing in Equation* (3.14)*. The unsmoothed curve corresponds to the Video-GMM model.*

different identity, this person's video-based score will grow only marginally because the frame-based scores are practically reduced to zero.

Based on the observation that the first frames are generally of low quality, especially due to low resolution, it would be possible to omit them in the classification process. This assumption is not included into the current system as this would restrict the system to this specific door scenario.

## 4.3 System speed

This section will present the processing times per frame in order to substantiate the claim for real-time capability. The times reported for the face recorder are determined on sequence A of data set I, the times for the face recognizer are based on measurements taken of seven randomly selected sequences from data set II. Each sequence was processed five times. The recognition was performed with a KNN model, namely Video-KNN, since frame-weighting is currently done only during score post-processing, not in the system itself. The times are very similar if GMMs are used and not listed separately. Speed evaluation is performed on a Pentium 4 at 3 GHz with 1 GB RAM.

The average processing time per frame is 37 ms. Since the computational effort varies between different processing steps, it is necessary to examine some of them on their own. These are listed in Table 4.8. The distinction between "face recorder" and

| | |
|---|---|
| **Face recorder** | |
| Empty image | 28 ms |
| While recording | 18 ms |
| **Face recognizer** | |
| Pre-init | 46 ms |
| Initialization of Kalman filter | 65 ms |
| During recognition | 30 ms |
| Overall average | 37 ms |

***Table 4.8:*** *Processing times per frame during different stages.*

"face recognizer" simply reflects the different data in evaluation, and does not denote a separation of functionality.

*Empty image* refers to plain background images, i. e., no face is visible. Its processing comprises the continuously running skin segmentation which is fast but, due to a lacking ROI, is performed on the full resolution image. While recording, the amount of processed input data is reduced, but the face detection and tracking consumes additional processing time. In total, however, speed increases by one third, so that proper recording of 25 fps is ensured.

During recognition, the processing is slowed down due to the feature extraction and classification stages. The most time consuming part is the initialization of the Kalman filter. It takes especially long if several valid eye pair hypotheses exist as each of these can trigger a rotation of the image patch with subsequent face detection. *Can* because initialization continues as soon as one hypothesis is confirmed. Since this search is not necessarily restricted to the frame in which a face is finally detected, the processing time preceding the initialization is listed as well. Besides including the unsuccessful validation of eye pair hypotheses, it covers the computational effort needed to build the track hypotheses in the first place.

All in all, the system is able to process the camera input at 25 fps in real-time. The initialization of the Kalman filter causes a small lag which does not affect system performance. Even if the system would be on every frame as slow as in the initialization step, it would still be able to process 15 fps and therefore be considered real-time capable.

# 5 Conclusion

Face recognition systems provide a wide variety of application areas. In contrast to other biometric systems which use fingerprints or iris scans, they do not require the explicit cooperation of the person to be identified. While they can be employed in security-related domains like video surveillance on the one hand, they allow as well for "convenience" applications, such as smart homes or cars that automatically adapt to the user or robots that show personalized behavior depending on who they are dealing with.

The key to this unaware recognition is a system that is able to process data under everyday conditions and to accomplish this in real-time. Violations of these requirements necessarily interrupt the persons to be identified or restrict them in their actions. An example of such a restriction is the necessity to stop for a moment and watch straight at the camera while not being allowed to wear any accessories in order to be recognized. As shown in this work, the proposed system is able to fulfill both requirements.

A large set of segmented data was automatically collected under real-life conditions including extreme variations in illumination, expression, pose, appearance or partial occlusion. The local appearance-based approach is shown to handle these variations well. This can be seen from the fact that they do not impede successful recognition in a majority of the cases. The local approach is supported by video-based recognition which greatly improves recognition performance compared to single-frame classification. The exploitation of temporal dynamics between frames boosts the correct recognition rate from less than 70 percent to more than 90 percent. Additionally, it increases the number of evaluable frames in the first place by improving eye detection using tracking methods. It is shown that longer sequences are more likely to be correctly recognized than shorter ones.

Besides capturing many variations, the real-life conditions during data collection entail largely different numbers of samples per person, as people are not restricted to pass the camera at least or at most a certain number of times. While a high number of samples generally improves recognition rates because more variation is captured and modeled, it does not mean that subjects with little training data are not recognizable. In fact, the system fails on very few people and this is not only caused by little training data but by small sets of low-quality test data as well. In total, more than 70 percent of the people are each correctly classified in more than 80 percent of the cases.

Augmentation of the training data with virtual samples increases captured scale and in-plane pose variations and reduces the impact of possible registration errors. This especially allows persons with little training data to be modeled in more detail. Overall, training data augmentation improves the recognition rate significantly.

| | CCR (%) | EER (%) |
|---|---|---|
| Weighted-KNN | **92.5** | 21.0 |
| Smooth-GMM ($\varepsilon = 10^{-5}$) | 87.8 | **18.7** |

***Table 5.1:*** *Summary of the best recognition results.*

Since the mentioned application scenarios are not necessarily restricted to a fixed group of people, the system needs to be able to handle unknown persons. Naturally, the increased difficulty of open-set identification leads to higher error rates. The EER of just under 19 percent reflects the difficulties introduced by the real-life quality of the data. This assumption is backed by the results of the frame-weighting and smoothing experiments which reduce the EER by decreasing the impact of low-quality and badly modeled frames. For KNN-models, two frame-weighting schemes, DTM and DT2ND, were introduced to achieve this by giving individual weights to single frames. For GMM-based models, a smoothing term was used to weight the current frame in relation to the current hypothesis. Table 5.1 summarizes the best results achieved in the recognition experiments.

It is shown that the system is able to perform the task within real-time constraints. On average, a processing rate of 25 fps can be achieved. Taking into account the short but most time-consuming part of the system, the initialization of the Kalman filter, the system still processes 15 fps which is still considered to be real-time.

# 6 Future Work

The following suggestions are made to enhance the current system. First of all, in order to improve the closed-set performance, additional frame weighting methods should be explored. Starting from the currently used DTM weighting scheme, these are needed to reliably identify bad quality frames and reduce their noisy influence on the sequence scores. A detailed analysis of the training data according to the *menagerie* introduced by Doddington et al. (1998) would allow for a system that can adapt certain weights and thresholds depending on the class to be recognized. The menagerie concept divides the data into the four non-disjoint classes *sheep*, *goats*, *lambs* and *wolves*. These classes describe how easy a subject can be recognized, be imitated by or imitate another class. Initially introduced for speaker recognition, Wittman et al. (2006) showed that these categories apply to face recognition problems as well.

In order to improve handling of partial occlusions, Ekenel and Stiefelhagen (2006) suggest to evaluate a reduced number of blocks instead of all 64. They propose to use a selection scheme based on block-wise similarity scores for the mean training image and a test image. This score can be either based on pixel values or DCT coefficients. The blocks with the highest scores are selected for recognition because these are the least likely to be occluded since they are very close to the representation. Since this approach is based on mean values over the training set, it is not applicable to real-life data which especially includes head pose variations. With these, certain blocks can contain different parts of the face in different images and a mean block value over all images will have no expressiveness. However, the approach could be extended by using a block's entropy as selection criterion. As this would concentrate on high detail areas of the image, it could be misleaded by high-detail accessories like finely patterned scarves or caps. Nevertheless, many occlusions caused by sunglasses, hands or even hair are comparatively homogeneous and would therefore be ignored or down-weighted in the classification process.

Instead of selecting blocks in the test image, a different approach would be to select a set of blocks specific for each class. The test vector would then be evaluated against these downsized representatives. Including the outcome of the "zoo analysis" proposed above, this could possibly resolve some of the recognition problems arising from the four categories. Furthermore, since fewer dimensions have to be processed, this will result in a speed-up. Additionally, with a reduce problem dimensionality, generative models like GMMs require less training data to build meaningful models. But, this approach runs the risk to adapt to specific characteristics of the training set because the training data can always only provide a "snapshot" of individuals and variations.

Concerning the open-set identification task, it is crucial to further reduce the EER. The current approach takes into account the frame-based scores by applying a threshold to the final sequence score. While DT2ND-weighting reduces the impact of ambiguous frames in the Weighted-KNN scheme, a smoothing term weakens the influence of inconsistent frames in the Smooth-GMM scheme. Both approaches reduce the EER independently. A classifier trained on *several* ambiguity and consistency measures is likely to achieve a considerable improvement in the separation of known and unknown people over the current final-score-based confidence threshold. Imaginable additional features to train this classifier comprise a general measure of frame-score consistency over the whole sequence in contrast to Smooth-GMM, which compares the current frame classification to the currently best hypothesis, the resolution of the non-normalized face images as frame quality measure and sequence length as well as absolute final score as further score confidence measures. However, it has to be kept in mind that impostor detection is a non-trivial task because it requires the separation of an arbitrary subset of all possible faces from the rest.

If the FAR and FRR have been reduced to a reasonable degree, i. e., if it is possible to distinguish impostors from genuine users reasonably well, it will be possible to extend the system to learn new people automatically. Each time, an unknown person is encountered, the test vectors can be used to train a model for this person in the background and add it to the database as soon as it exceeds a certain size. Similarly, small models capturing little variation can be extended every time that person is recognized with sufficient confidence. While some faces match the model very well and lead to the stable score, the remaining detected faces can be used to refine the model. However, automatic, unsupervised extension of the database always bears the risk that a new model is generated even though the current sequence captures a strong variation of an already registered person.

Nevertheless, this approach could then even be used to automatically train the system from scratch. A house will get to know its visitors and a robot will make new friends.

# A Detailed Overview of Data Set II

## A.1 Set sizes per individual

| Identity | Number of sequences per set: Training | Parameter estimation | Test | Total number of sequences | Number of training images |
|---|---|---|---|---|---|
| 1 | 7 | 1 | 3 | 11 | 260 |
| 2 | 5 | – | 2 | 7 | 82 |
| 3 | 5 | – | 2 | 7 | 93 |
| 4 | 35 | 35 | 90 (−1) | 160 | 1,127 |
| 5 | 35 | 25 | 53 (−1) | 113 | 693 |
| 6 | 20 | 3 | 7 | 30 | 282 |
| 7 | 8 | 1 | 3 | 12 | 113 |
| 8 | 35 | 13 | 28 (−2) | 76 | 1,114 |
| 9 | 6 | – | 3 | 9 | 91 |
| 10 | 35 | 19 | 40 (−1) | 94 | 598 |
| 11 | 15 | 2 | 5 | 22 | 685 |
| 12 | 35 | 15 | 32 (−2) | 82 | 973 |
| 13 | 13 | 2 | 5 | 20 | 352 |
| 14 | 35 | 16 | 33 | 84 | 1,120 |
| 15 | 11 | 1 | 5 | 17 | 279 |
| 16 | 3 | – | 2 | 5 | 87 |
| 17 | 6 | – | 3 | 9 | 116 |
| 18 | 4 | – | 2 | 6 | 129 |
| 19 | 35 | 35 | 179 (−1) | 249 | 1,016 |
| 20 | 35 | 35 | 96 (−7) | 166 | 1,325 |
| 21 | 30 | 5 | 11 (−6) | 46 | 94 |
| 22 | 35 | 24 | 50 | 109 | 1,191 |
| 23 | 8 | 1 | 3 | 12 | 93 |
| 24 | 14 | 2 | 6 | 22 | 752 |
| 25 | 35 | 23 | 49 (−4) | 107 | 505 |
| ... | ... | ... | ... | ... | ... |

| Identity | Number of sequences per set | | | Total number of sequences | Number of training images |
|---|---|---|---|---|---|
| | Training | Parameter estimation | Test | | |
| ... | ... | ... | ... | ... | ... |
| 26 | 31 | 7 | 16 | 54 | 588 |
| 27 | 35 | 15 | 31 (−4) | 81 | 381 |
| 28 | 18 | 3 | 7 (−1) | 28 | 189 |
| 29 | 3 | – | 2 | 5 | 80 |
| 30 | 35 | 21 | 45 | 101 | 873 |
| 31 | 3 | – | 2 | 5 | 84 |
| 32 | 5 | – | 3 | 8 | 177 |
| 33 | 35 | 24 | 51 (−3) | 110 | 900 |
| 34 | 35 | 5 | 13 (−4) | 53 | 503 |
| 35 | 24 | 4 | 9 (−1) | 37 | 489 |
| 36 | 35 | 10 | 22 (−19) | 67 | 259 |
| 37 | 17 | 2 | 7 | 26 | 283 |
| 38 | 35 | 9 | 20 (−1) | 64 | 634 |
| 39 | 28 | 4 | 11 | 43 | 897 |
| 40 | 35 | 20 | 41 | 96 | 1,463 |
| 41 | 26 | 4 | 9 | 39 | 651 |
| Total | 905 | 386 | 1,001 (−58) | 2,292 | 21,875 |

***Table A.1:*** *Dataset II: Detailed listing of how many sequences of each person are part of each set. The negative numbers in parentheses denote how many sequences could not be registered for recognition.*

## A.2 Registered images per person

| Identity | Number of images | | |
|---|---|---|---|
| | Training | Parameter Estimation | Test |
| 1 | 260 | 47 | 234 |
| 2 | 82 | – | 86 |
| 3 | 93 | – | 167 |
| 4 | 1,127 | 1,566 | 4,410 |
| 5 | 693 | 572 | 1,988 |
| 6 | 282 | 97 | 464 |
| 7 | 113 | 3 | 131 |
| 8 | 1,114 | 585 | 1,205 |
| 9 | 91 | – | 108 |
| 10 | 598 | 807 | 1,658 |
| 11 | 685 | 122 | 289 |
| 12 | 973 | 537 | 973 |
| 13 | 352 | 88 | 256 |
| 14 | 1,120 | 695 | 1,766 |
| 15 | 279 | 27 | 212 |
| 16 | 87 | – | 136 |
| 17 | 116 | – | 106 |
| 18 | 129 | – | 40 |
| 19 | 1,016 | 1,175 | 7,811 |
| 20 | 1,325 | 1,167 | 3,971 |
| 21 | 94 | 79 | 124 |
| 22 | 1,191 | 1,446 | 2110 |
| 23 | 93 | 29 | 159 |
| 24 | 752 | 97 | 235 |
| 25 | 505 | 914 | 1,671 |
| 26 | 588 | 24 | 916 |
| 27 | 381 | 414 | 1,063 |
| 28 | 189 | 79 | 255 |
| 29 | 80 | – | 77 |
| 30 | 873 | 621 | 2,259 |
| 31 | 84 | – | 156 |
| 32 | 177 | – | 109 |
| ... | ... | ... | ... |

| Identity | Number of images | | |
|---|---|---|---|
| | Training | Parameter Estimation | Test |
| ... | ... | ... | ... |
| 33 | 900 | 642 | 2,025 |
| 34 | 503 | 373 | 191 |
| 35 | 489 | 80 | 311 |
| 36 | 259 | 57 | 104 |
| 37 | 283 | 41 | 157 |
| 38 | 634 | 281 | 730 |
| 39 | 897 | 149 | 391 |
| 40 | 1,463 | 770 | 1,901 |
| 41 | 651 | 121 | 457 |
| Total | 21,875 | 13,708 | 41,756 |

***Table A.2:*** *Dataset II: Number of registered images per person.*

## A.3 Individual sequence lengths

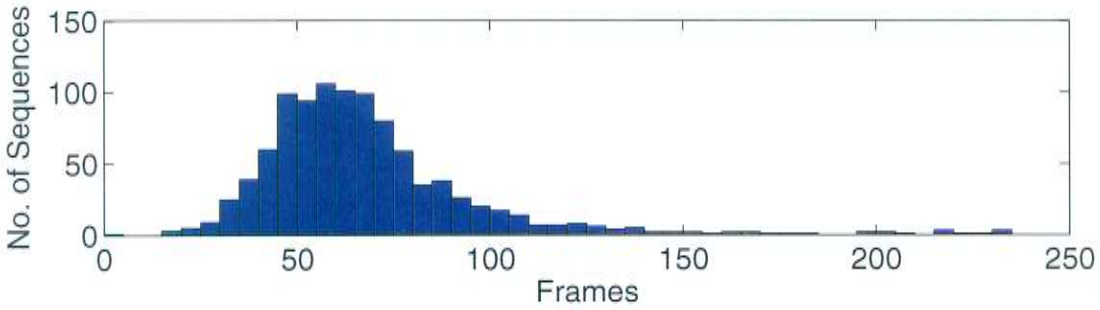



***Figure A.1:*** *Number of frames per sequence. Leading and trailing frames — ten each — that were added to the detected sequence during recording are excluded in this plot. For better comparability with results in Figure 4.3, the sequence sizes are given for the test set, but can be considered representative.*

# B Model Sizes

| Identity | Number of components | Identity | Number of components |
|---|---|---|---|
| 1 | 6 | ... | ... |
| 2 | 2 | 22 | 25 |
| 3 | 4 | 23 | 2 |
| 4 | 23 | 24 | 16 |
| 5 | 15 | 25 | 11 |
| 6 | 6 | 26 | 12 |
| 7 | 3 | 27 | 8 |
| 8 | 23 | 28 | 4 |
| 9 | 2 | 29 | 2 |
| 10 | 13 | 30 | 18 |
| 11 | 14 | 31 | 2 |
| 12 | 20 | 32 | 4 |
| 13 | 8 | 33 | 19 |
| 14 | 23 | 34 | 11 |
| 15 | 6 | 35 | 10 |
| 16 | 2 | 36 | 6 |
| 17 | 3 | 37 | 6 |
| 18 | 3 | 38 | 13 |
| 19 | 21 | 39 | 19 |
| 20 | 27 | 40 | 30 |
| 21 | 2 | 41 | 14 |
| ... | ... | Total | 459 |

***Table B.1:*** *Model sizes: Number of model components for each person. The numbers are the same for both the KNN and GMM approaches.*

# Bibliography


O. Arandjelovic and A. Zisserman. Automatic face recognition for film character retrieval in feature-length films. In *Proc. 2005 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, pages 860–867, Washington, DC, USA, 2005.

Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press, New York, 1988.

P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class-specific linear projection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(7):711–720, July 1997.

J. Bilmes. A gentle tutorial on the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical Report ICSI-TR-97-021, University of Berkeley, 1997.

J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

A. Dempster, N. Laird, and D. Rubin. Maximum-likelihood from incomplete data via EM algorithm. *Journal Royal Statistical Society, Series B*, 39:1–38, 1977.

G. Doddington, W. Liggett, A. Martin, M. Przybocki, and D. Reynolds. Sheep, goats, lambs and wolves: A statistical analysis of speaker performance in the NIST 1998 speaker recognition evaluation. In *Int'l. Conf. on Spoken Language Processing*, Sydney, Australia, 1998.

R. Duda, P. Hart, and D. Stork. *Pattern Classification*. John Wiley and Sons, 2001.

H. K. Ekenel and R. Stiefelhagen. Local appearance-based face recognition using discrete cosine transform. In $13^{th}$ *European Signal Processing Conf. (EUSIPCO 2005)*, 2005.

H. K. Ekenel and R. Stiefelhagen. Block selection in the local appearance-based face recognition scheme. *Conf. on Computer Vision and Pattern Recognition Workshop 2006*, page 43, 2006a.

H. K. Ekenel and R. Stiefelhagen. Analysis of local appearance-based face recognition: Effects of feature selection and feature normalization. *Conf. on Computer Vision and Pattern Recognition Workshop 2006*, page 34, 2006b.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and Sytem Sciences*, 55(1): 119–139, 1997.

A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

V. K. Goyal. Theoretical foundations of transform coding. *IEEE Signal Processing Magazine*, 18:9–21, September 2001.

B. Heisele, P. Ho, and T. Poggio. Face recognition with support vector machines: Global versus component-based approach. In *Proc. IEEE Int'l. Conf. on Computer Vision*, volume 2, pages 688–694, 2001a.

B. Heisele, T. Serre, M. Pontil, and T. Poggio. Component-based face detection. In *Proc. 2001 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 657–662, 2001b.

E. Hjelmas and B. K. Lee. Face detection: A survey. *Computer Vision and Image Understanding*, 83(3):236–274, Sept. 2001.

P. J. Huber. *Robust Statistics*. Wiley, 1981.

M. Hunke and A. Waibel. Face locating and tracking for human-computer interaction. In *Conf. Record of the* $28^{th}$ *Asilomar Conf. on Signals, Systems and Computers*, volume 2, pages 1277–1281, Oct. 1994.

Intel Corporation. Open source computer vision library (OpenCV), 2006. URL `http://www.intel.com/technology/computing/opencv/index.htm`. Last visit: Nov. 2006.

A. H. Jazwinski. *Stochastic processes and filtering theory*. Academic Press, New York, 1970.

M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. *Int'l. Journal of Computer Vision*, 46(1):81–96, 2002.

M. J. Jones and P. Viola. Fast multi-view face detection. Technical Report TR2003-96, Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, July 2003.

R. E. Kalman. A new approach to linear filtering and predictive problems. *Trans. ASME, Journal of basic engineering*, (82):34–45, 1960.

J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. On combining classifiers. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

R. Kjeldsen and J. R. Kender. Finding skin in color images. In $2^{nd}$ *Int'l. Conf. on Face and Gesture Recognition*, pages 312–317. IEEE Computer Society, 1996.

K. C. Lee, J. Ho, M. H. Yang, and D. Kriegman. Video-based face recognition using probabilistic appearance manifolds. In *Proc. 2003 IEEE Computer Society Computer Vision and Pattern Recognition*, pages I: 313–320. IEEE Computer Society, 2003.

F. Li and H. Wechsler. Open set face recognition using transduction. *IEEE Trans. Pattern Anal. Mach. Intell*, 27(11):1686–1697, 2005.

X. Liu and T. Chen. Video-based face recognition using adaptive hidden markov models. In *Proc. 2003 IEEE Computer Society Int'l. Conf. on Computer Vision and Pattern Recognition*, pages I: 340–345. IEEE Computer Society, 2003.

J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc.* $5^{th}$ *Symp. on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

A. M. Martinez and R. Benavente. The AR face database. Technical Report 24, CVC, 1998.

B. Martinkauppi. *Face Colour under Varying Illumination - Analysis and Application*. PhD thesis, University of Oulu, Finland, Aug. 2002.

B. Martinkauppi, M. N. Soriano, and M. H. Laaksonen. Behavior of skin color under varying illumination seen by different cameras at different color spaces. In M. A. Hunt, editor, *SPIE Machine Vision in Industrial Inspection IX*, volume 4301, Jan. 2001.

B. Menser and F. Müller. Face detection in color images using principal components analysis. In *Proc.* $7^{th}$ *IEEE Int'l. Conf. on Image Processing and Its Applications*, volume 2, pages 620–624, July 1999.

B. Moghaddam and A. P. Pentland. Probabilistic visual learning for object detection. In *Proc. Int'l. Conf. on Computer Vision*, pages 786–793, 1995.

C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In $6^{th}$ *Int'l. Conf. on Computer Vision*, pages 555–562, 1998.

A. Pentland and T. Choudhury. Face recognition for smart environments. *Computer*, 33(2):50–55, Feb. 2000.

P. J. Phillips, P. J. Grother, R. J. Micheals, D. M. Blackburn, E. Tabassi, and M. Bone. Face recognition vendor test 2002. In *Int'l. Workshop on Analysis and Modeling of Faces and Gestures*, pages 44–44, 2003.

P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min, and W. Worek. Overview of the Face Recognition Grand Challenge. In *Proc. 2005 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 947–954, Los Alamitos, CA, USA, 2005. IEEE Computer Society.

Y. Raja, S. J. McKenna, and S. Gong. Tracking and segmenting people in varying lighting conditions using colour. In $3^{rd}$ *Int'l. Conf. on Face and Gesture Recognition*, pages 228–233. IEEE Computer Society, 1998.

A. Rosenfeld and J. L. Pfaltz. Sequential operations in digital picture processing. *Journal of the ACM*, 13(4):471–494, 1966.

H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(1):23–38, 1998.

M. C. Shin, K. I. Chang, and L. V. Tsap. Does colorspace transformation make any difference on skin detection? In $6^{th}$ *IEEE Workshop on Applications of Computer Vision*, pages 275–279, Los Alamitos, CA, USA, 2002. IEEE Computer Society.

J. Sivic, M. Everingham, and A. Zisserman. Person spotting: Video shot retrieval for face sets. In W. K. Leow, M. S. Lew, T.-S. Chua, W.-Y. Ma, L. Chaisorn, and E. M. Bakker, editors, *Proc. Int'l. Conf. on Image and Video Retrieval*, volume 3568 of *Lecture Notes in Computer Science*, pages 226–236. Springer, July 2005.

R. Snelick, U. Uludag, A. Mink, M. Indovina, and A. K. Jain. Large-scale evaluation of multimodal biometric authentication using state-of-the-art systems. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(3):450–455, 2005.

M. Soriano, B. Martinkauppi, S. Huovinen, and M. Laaksonen. Skin detection in video under changing illumination conditions. In *Proc.* $15^{th}$ *Int'l. Conf. on Pattern Recognition*, volume 1, pages 839–842, 2000.

M. Störring. *Computer Vision and Human Skin Colour*. PhD thesis, Aalborg University, Denmark, Aug. 2004.

M. J. Swain and D. H. Ballard. Color indexing. *Int'l. Journal of Computer Vision*, 7 (1):11–32, 1991.

P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

J.-C. Terrillon, M. N. Shirazi, H. Fukamachi, and S. Akamatsu. Comparative performance of different skin chrominance models andchrominance spaces for the automatic detection of human faces in colorimages. In $4^{th}$ *IEEE Int'l. Conf. on Automatic Face and Gesture Recognition*, pages 54–61, 2000.

M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

V. Vezhnevets, V. Sazonov, and A. Andreeva. A survey on pixel-based skin color detection techniques. In *Proc. Int'l. Conf. on Computer Graphics and Vision*, pages 85–92, Sept. 2003.

P. Viola and M. J. Jones. Robust Real-Time face detection. In *IEEE Int'l. Conf. On Computer Vision*, pages 747–747, July 9–12 2001.

A. Waibel, H. Steusloff, R. Stiefelhagen, and the CHIL Project Consortium. CHIL - computers in the human interaction loop. In $5^{th}$ *Int'l. Workshop on Image Analysis for Multimedia Interactive Services*, Lisbon, Portugal, Apr. 2004.

G. K. Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34:31–44, Apr. 1991.

H. Wang, S. Z. Li, and Y. Wang. Generalized quotient image. In *Proc. 2004 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, pages II: 498–505, 2004.

G. Welch and G. Bishop. An introduction to the Kalman filter. In *SIGGRAPH, Course 08*, 2001.

M. Wittman, P. Davis, and P. Flynn. Empirical studies of the existence of the biometric menagerie in the FRGC 2.0 color image corpus. *Conf. on Computer Vision and Pattern Recognition Workshop 2006*, 0:33, 2006.

M.-H. Yang, D. J. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

T.-W. Yoo and I.-S. Oh. A fast algorithm for tracking human faces based on chromatic histograms. *Pattern Recognition Letters*, 20:967–978, 1999.

B. Zhang, M. Hsu, and U. Dayal. K-harmonic means: A spatial clustering algorithm with boosting. In J. F. Roddick and K. Hornsby, editors, *Int'l. Workshop on Temporal, Spatial and Spatio-Temporal Data Mining*, volume 2007 of *Lecture Notes in Artificial Intelligence*, pages 31–45, Lyon, France, 2000. Springer.

W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Computing Surveys*, 35(4):399–458, Dec. 2003.

S. Zhou, V. Krueger, and R. Chellappa. Probabilistic recognition of human faces from video. *Computer Vision and Image Understanding*, 91(1-2):214–245, Feb. 2003.