CHALMERS

Face identification in near IR-videos using Smart Eye tracking profiles

Master's Thesis in Signals & Systems

FREDRIK WALTERSON
& KENAN BABIC

Department of Signals & Systems
Signal processing
Chalmers University of Technology
Gothenburg, Sweden 2014

Master's Thesis 2014:SSYX04

Abstract

In this thesis we have evaluated the possibility of using data extracted from Smart Eye tracking profiles for the task of driver identification in near-IR videos using face recognition techniques. Two texture-based methods are presented to solve this task. The first method utilizes local binary patterns (LBP) combined with local discriminant analysis (LDA) to identify subjects whilst the second method investigates brute-force template matching for identification purposes.

The LBP-method was evaluated for both closed-set and open-set identification and achieved good results. In the closed-set scenario, a recognition rate of 98% was reached on our own data set of 47 videos containing 24 different subjects, despite the fact that the subjects had modified their appearances in half of the videos. The LBP-method is fast and our results demonstrates the LBP-method's ability to handle partial facial occulusions and appearance variations. The template matching method showed decent results with a peak recognition rate of 81% in the closed-set scenario. However, the method proved very time-consuming due to large number of templates in the profiles, limiting the method's viability in real-time applications. Future refinements and extensions are proposed for both methods.

Acknowledgements

We would like to thank our supervisors at Chalmers and Smarteye for their support and also thank all the people (co-workers, friends and family) who volunteered as subjects for our data base. This project has been a challenge, but we have learned a lot and hopefully the readers will find something of interest in this thesis.

Fredrik Walterson & Kenan Babic, Gothenburg, January 11, 2015

Contents

1 Introduction **1**

2 Project description **5**
2.1 Purpose and goal statement 5
2.2 Restrictions 5

3 Theory **7**
3.1 Face detection 7
3.1.1 Viola-Jones algorithm for face detection 8
3.2 Template matching 11
3.3 Feature description 11
3.3.1 Local binary patterns 13
3.4 Linear subspaces 14
3.4.1 Nearest neighbourhood classifier 16

4 Method **17**
4.1 Test environment 17
4.1.1 System description 17
4.1.2 Video databases 18
4.1.3 Video classification 19
4.1.4 Closed-set identification 23
4.1.5 Open-set identification 23
4.2 Face detection 25
4.3 Feature extraction 27
4.4 Facial alignment 28
4.5 Face recognition 30
4.5.1 Using local binary patterns 30
4.5.2 Using template matching 33

5 Results & Discussion **41**
5.1 Face detection 41
5.2 Feature extraction 41
5.3 Facial alignment 42
5.4 Face recognition 43
5.4.1 LBP-method 43
5.4.2 Template matching 53
5.4.3 Alternative identification methods 59

6 Conclusions **60**

7 Future work **61**

Bibliography **67**

Appendix A Additional tables **68**
A.1 List of OpenCV Haar-cascades 68
A.2 Run times for LBP-method 68
A.3 Video database information 69
A.4 Pilot video database information 70

Chapter 1: Introduction

Face recognition is a subject that has developed rapidly during the last decade. Its progress can be largely attributed to its highly sought-after applications; Entertainment, information security, surveillance and law enforcement are just a few areas where face recognition technology is coveted [1]. Fingerprints and iris recognition are two highly accurate biometrics (measures that can be used to describe human characteristics or traits) but both require the user to be cooperative [2]. Face recognition on the other hand is a biometric that can be used in non-cooperative, unconstrained settings where the subjects do not know that they are being identified.

The demands between areas of usage might vary, but the rewards for a definite solution to the problem of unconstrained face recognition and the possibilities that such technology could offer are seemingly endless. Replacing computer passwords, PIN codes or passport controls with images or video sequences of faces would have drastic changes on everyday life. Face recognition has in fact been used for passport control at the China-Hong Kong border for self-service immigration clearance since 2005, with more than 400,000 people crossing every day [3]. This is a sign that the technology is getting closer to such applications, but that it has not fully matured yet. New discoveries in overlapping disciplines such as image processing, pattern recognition, machine learning, computer vision, computer graphics and psychology are often applied and united in face recognition [1], making it a diverse subject. It is exciting, difficult and can be approached in many different ways.

Although the first automatic face recognition system was proposed in 1973, it was not until the early 1990s that the research gained momentum. The seminal work on Eigenfaces and face representations in low dimensional spaces played an important role in popularizing the subject [3], since it was a new way of approaching the problem of face description and many people found that interesting [4]. Other factors such as increasing computational power and the creation of more diverse publicly available face databases has also helped stimulate the development.

Providing more annotated data to train and test algorithms on and defining fixed evaluation protocols has simplified the process of comparing methods and results immensely. Most of the research on face recognition has historically been done on gray-scale image databases with limited or controlled variation. Typical examples include the FERET database [5] (1199 subjects captured in various controlled setups of pose and illumination), the Yale database [6] (15 subjects in 11 different configurations) and the BioID database [7] (1521 images of 23 subjects with no restrictions except full frontal pose). More recently, the research has been focused on face recognition in unconstrained environments, with barely any restrictions on variations in facial expressions, clothing, lighting, pose and background. The "Labeled faces in the wild" (LFW) data set [8] was created using images found on the web and has been used extensively to evaluate performance in unconstrained environments due to its natural or random variability. Research on face recognition in other mediums such as high resolution images and 3D scans [9], spectral images [2] and videos [10] has also become more prevalent. All mediums may have their own strengths, such as higher quality informaton, but they may also possess their own set of challenging problems.

The task of face recognition can be seen as a non-linear, high-dimensional, pattern matching problem [2]. It can be divided further into face verification, where a face image is matched against another face image to determine if they display the same person or not, and face identification, where a match instead is searched for among a set of N previously stored face images [3]. With no prior knowledge of what can be considered relevant information when perfoming face recognition, this is a very challenging problem, especially due to all the inherent variations in face images. Yang et al. [11] lists varying scale, head orientations and poses, facial expressions, imaging conditions (lighting and camera characteristics), facial features (beards, mustaches, glasses) and partial facial occulusions as difficulties in the general case of face detection. These unfortunately also apply to face recognition. Abate et al. [2] also considers time as a factor that might affect performance, since aging could drastically alter the apperance of an individual.

The complexity of the problem increases when considering fully automated systems. The face recognition problem will then extend into several sub-steps, which might vary depending on the method of choice. The common denominator of all methods is that they require one or many face images, so naturally the first step is to detect any present face. The face is then cropped from the image and preprocessed into a more suitable representation, which usually involves normalization with respect to illumination or geometric transformations, for example translations and rotations. This is to improve performance in unconstrained settings and reduce the pose and lighting variations that might occur [3]. Once normalized, the final step of face recognition can be divided additionally into feature representation and classification [12]. Features are extraced from the face in order to simplify the task of finding a match in the face database. They are usually created by encoding the face information to a compact and discriminative representation using a descriptor, that optimally should be invariant to face image variations. Examples of feature descriptors include local binary patterns, histogram of oriented gradients, Gabor wavelets and SIFT descriptors [13] [14]. A classifier is then applied to the feature to find the best match amongst the stored faces, that also has been processed in this way. Researchers have continuously improved the performance of their systems by adding new concepts and changing the methods within this type of pipeline, where the constructed methodology is very intertwined with the used data medium.

Smart Eye AB is a company that provides specially designed camera systems and state-of-the-art gaze tracking software using near-IR video streams [15]. Their products are used in a wide array of applications, for example research relating to behavioural patterns of drivers and detection of behind-the-wheel drowsiness. Their system tracks the gaze and facial features and builds a tracking profile which contains information of the user, such as feature templates, full camera images and 3D-models. The profiles are augmented while the program is running, so the tracking accuracy increases the longer a person is tracked, and it takes time for the system to reach a point of high tracking accuracy. It is therefore of interest to use existing tracking profiles on previous users, to avoid creating new profiles each time. The goal of this project is to develop a face recognition algorithm which can recognize the current user by the information present in the tracking profiles, so that the correct user profile can be loaded automatically.

Near IR-images has been used before in face recognition by Li et al. [16], but their work was done in a cooperative setting. The intended application of this project is driver identification during the short period of time after which the user has entered the vehicle. The users will be non-cooperative and in motion with varying day-to-day appearances, making it a very difficult task. A successful identification at this stage of the trip could however be used to load the stored profile and quickly increase tracking accuracy, and also in future customer applications, such as automatic configuration of user-specific seat, steering wheel and mirror settings. Since the scenario is very specific and the image representation is rather uncommon, the project includes the creation of a database that can be used for evaluation of the face recognition system. Other gray-scale databases captured in the visible spectrum such as FERET cannot be used [5] because of the difference in image representation. The task of identification is simplified by the fact that there are a limited amount of people using the same vehicle, our database is therefore not required to be as large as ordinary face recognition databases containing many thousand subjects.

In this thesis we present two methods that use textural information from the tracking profiles to identify the subjects in pre-recorded near-IR videos. The first method utilizes a local binary patterns combined with linear discriminant analysis (LDA) to encode face images in a compact and discriminative form, simplifying the process of matching faces between users. Local binary patterns is a local texture operator that has previously shown great results when applied to face recognition [17], both by itself and together with LDA [13]. This suggests that a combination of both could also work well for videos. The second method uses brute-force template matching, using normalized cross-correlation, to identify the users. Both methods treat the videos as a sequence of gray-scale images.

The thesis is structured as follows: The purpose and goal of the project is summarized in Section 2, together with the restrictions of the project. In Section 3, a detailed description of the subject of face recognition and the theory behind our methods are presented. Section 4 describes our methods, how they were implemented and what parameters that were used. Our results are shown and discussed in Section 5, and the thesis is summarized and our conclusions are presented in Section 6.

Chapter 2: Project description

2.1 Purpose and goal statement

The purpose of this project is to investigate how the content in the Smart Eye tracking profiles, both image data (templates and full-size images) and 3D models, can be used to identify users in a generic real-world scenario where the Smart Eye camera systems are currently being used. This would be beneficial for both Smart Eye and their customers, since an identification feature could be used by Smart Eye to prevent the creation of new tracking profiles for previously seen users, decreasing the time until high tracking accuracy can be accomplished. In future customer applications, the knowledge of user identity could be utilized to load stored user-specific comfort settings. The goal is to propose one or many methods that can be shown to successfully identity previous users and to distinguish them from new users, for whom a tracking profile needs to be created. The option of adding additional information to the tracking profiles in order to enhance the proposed methods should also be evaluated.

2.2 Restrictions

The project will evaluate face recognition on a pre-recorded database of users instead of on real-time video streams. The goal is to find a method that works well and not to create a fully functional application. This allows us to evaluate the trade-off between identification accuracy and computational time. Optimally, the proposed method allows for real-time applications with high recognition rate. The database will be recorded in a controlled artificial environment with a camera setup used regularly by Smart Eye. The environment and user behaviour will simulate a real-world scenario that is a representative application of the Smart Eye system, more specifically a driver entering a motor-vehicle and acting naturally, performing the tasks ordinarly done in the first thirty seconds of the drive. Performance on this database will therefore hopefully generalize to both the real scenario and similar ones.

Smart Eye Pro 6.0 will be used to create both videos and tracking profiles. Our program will only be compatible with the format associated with that specific verision of Smart Eye Pro. The tracking profiles are created using real-time videos streams in Smart Eye Pro, running the tracking feature continuosly while the user is in camera.

The profiles are assumed to be complete and free from noise, in the sense that they only contain correct information and that they are representative of the user with respect to templates and 3D models. This is certainly not true, but if this is assumption is not made the errors made by our methods cannot be distinguised from the errors in the profiles, unless controlled errors are inserted and the outcomes are measured. That is not in the scope of this project and will be left to others to investigate.

The database will consist of one tracking profile and two video sequences per person, where each video is assumed to contain only one person and that the face is almost always visible during the duration of the video. In the first video, the user will have the same appearance as when the tracking profile was created. Differences in image content between the profile and the video is then only due to scaling and rotations of the face, a simplified scenario to test that the methods are working as intended. For the second video, the user will have a modified appearance, such as wearing hats, sun-glasses et.c. These videos will contain occlusions of the face, testing the methods ability to generalize and handle local face variations, something that will occur in real world settings where the user might change their apperance daily. The size of database is restricted to 20-25 different people. It is small compared to other face recognition databases and cannot be said to represent performance over a whole population, but not many privately owned vehicles are used by more than a handful of people and this should be enough to ensure performance for that specific scenario.

Finally, even though Smart Eye Pro is used to create profiles and videos, our program will be developed independently as a stand-alone program in MATLAB without any previously created Smart Eye software or code. It will only use the profiles and the videos to accomplish its identification task. All code is strictly developed by us or contained as built-in functions in MATLAB unless stated otherwise.

Chapter 3: Theory

3.1 Face detection

Face detection is a specific case of the more general task of object detection, where the goal is to find all available instances of an object in an image, defined by bounding boxes indicating the position and size of the object. This is usually done by moving a window, containing the current region-of-interest (from now on called ROI), across the image in a horizontal raster scan manner, row-wise starting from top to bottom, and using a classifier to determine if the ROI contains the chosen object or not. This must be done using ROI's of varying size in order to find objects of captured in the image at different scales. If the classifier gives a positive response for a specific ROI, the position and size of the ROI is defined as the bounding box of the object and the algorithm moves on until all of the image has been processed.

However, the task is complicated by the fact that objects do not only vary in sizes, but in pose (orientation of the object with respect to the camera), illumination (there could be shadows or reflections on the object for example), appearance (texture variations within the object class), occlusions (parts of the object is obscured) and deformations (facial expressions in face detection). Restricting the task to finding faces offers some case specific cues that be used such as skin colour, facial shape and structure (arrangements of facial features) and motion (if more than one image available). These cues may be useful for small scale models with designed for specific conditions, but not for the more general case of face detection.

The most successful methods for face detection are called appearance-based methods, where a model is trained to find the characteristics of a face using a set of face images. The performance of the method thus depends on the models ability to extract and learn discriminative information about faces and on the face database itself, since it must contain enough variation for to model to be able to generalize.

The algorithm proposed by Viola and Jones [18] has since its publication in 2001 been the de facto method for face detection. Its novel approach increased accuracy but most importantly speed, allowing for real-time applications that at the time could process 15 images per second [19]. It uses a training method called boosting, where a cascade of weak classifiers (multiple classifiers placed after each other) is trained using a large set of simple but computationally efficient features, so that only the best candidates manage to pass through the whole cascade of classifiers. Progress have been made since, especially in the case of unconstrained face detection, thanks to more challenging databases such as the ”Face detection data set and benchmark” [20], their webpage providing summarized results of recent publications evaluated on the data set together with baseline methods such as V-J.

Zhu et al. [21] proposed a unified method for face detection, pose estimation and fiducial point localization using a mixture of trees with a shared pool of parts that performed very well on all tasks. Their model could successfully handle rotations and deformations of the face, because they merge multiple parts located at fiducial points into a global mixture model, that assumes certain tree structures depending on the pose of the face and what parts that are located. Boosting is however still the most applied method for face detection because of its easily available implementations [19]. The remainder of this section will describe the original Viola-Jones algorithm in more detail. For a historic overview of face detection prior to the V-J method, see [11].

3.1.1 Viola-Jones algorithm for face detection

A vital contribution of the method is the use of the integral image representation together with Haar-like features, shown in Figure 3.1. There are four types of features, whose output at a specific coordinate position is a scalar value calculated from the difference between grey and white areas of the feature. The values of the gray and white areas are determined by the sums of pixels values within. The calculation of all features at a certain position is thus a succession of double sums (in x and y direction) and subtractions.

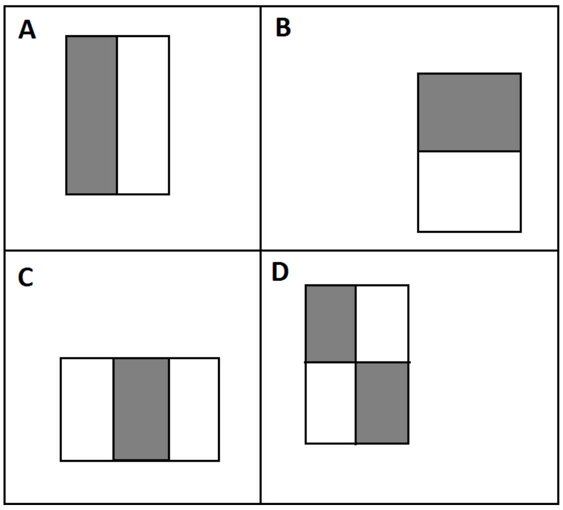


Figure 3.1: The four types of Haar-like features used in the Viola-Jones method.

The integral image, also known as a summed area table, is a technique that enables fast and simple computations of sums over rectangular areas in an image. Each pixel (x,y) in the integral image is equal to the sum of pixels above and to the left of (x,y) in the original image, as shown in Equation 3.1.

$$\mathrm{II}\,(x,y) = \sum_{x' \leq x\ ,\ y' \leq y} \mathrm{I}\,(x',y') \tag{3.1}$$

Any rectangular sum can then be calculated using very few operations. In practical terms, an integral image is the double cumulative sum of an image, along the row dimension and the column dimension. Each point in the integral image corresponds to the sum of the original image up to that point, see Figure 3.2.

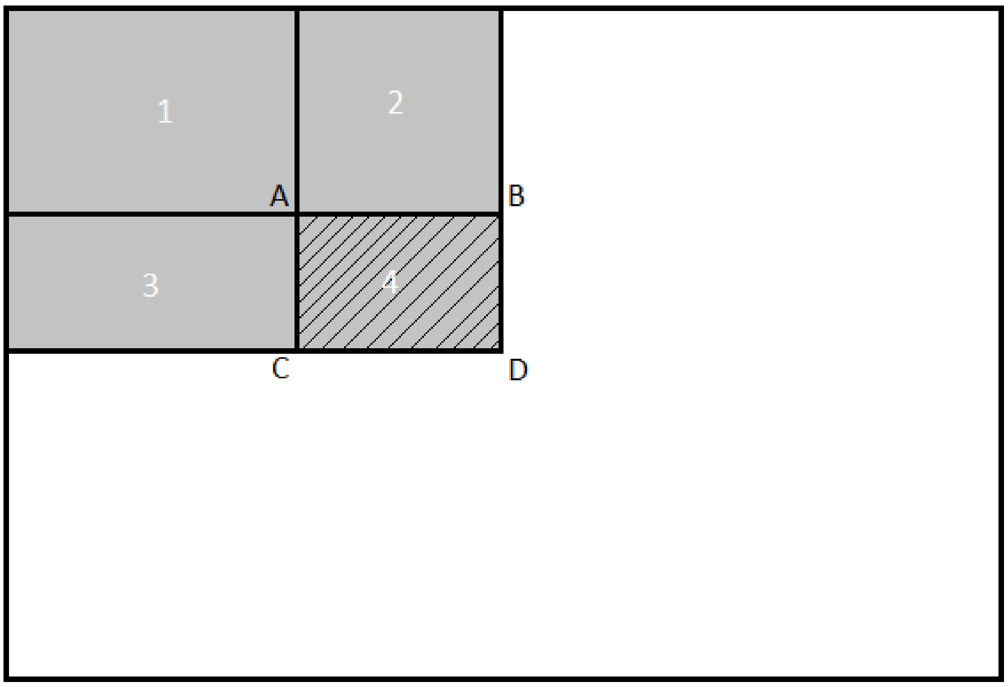


Figure 3.2: The sum of the gray area in the image corresponds to point D in the integral image. In order to obtain the sum of the striped rectangular area, simple arithmetic operations are required.

Any arbitrary rectangle can be defined in the image as four points, these points are the four corners of the rectangle, A, B, C and D as shown in Figure 3.2. There are also four areas numbered in the Figure, and we can now define the value each of these four corners as a sum of these areas. Point A in the integral image corresponds to area 1, B is the union of 1 and 2, C is the union of 1 and 3 and D is the union of all gray areas. This gives us

$$\text{Area } 4 = \mathrm{II}(D) + \mathrm{II}(A) - \mathrm{II}(C) - \mathrm{II}(B) \tag{3.2}$$

and that any arbitrary rectangular sum can be calculated with only four references.

The features are calculated within a specific ROI in an image in all ways possible by varying their size and orientation, forming an over-complete set. The minimum ROI size of 24x24 pixels used in the original paper yields a set of over 160,000 features. The ROI itself is then moved across the image at multiple scales as well, making it a very exhaustive search unless the feature set is reduced. The reduction of features is performed with a method called boosting.

The idea of boosting is that instead of training a single strong classifier from training data, multiple weak classifiers (also called weak learners) are trained and combined into a single strong classifier. This is done by training each weak learner in sequence, optimizing the current learner with respect to classification accuracy on the data set, and then increasing the weights on misclassified examples so that the following weak learners are more prone to correctly classify those examples. The strong classifier is created by a weighted majority vote of the weak learners, which will then have learned to correctly classify different subsets of the data set. The original AdaBoost algorithm can be found in [22, p. 439] and has been shown to achieve classification rates close to 100%, under the assumption that there is enough available training data and each individual learner has an accuracy above 50%.

Viola and Jones use the AdaBoost algorithm to select the features that are most appropriate for classification purposes. They restrict the algorithm such that each weak learner constitute one feature (out of the over 160,000 available in the over-complete set of Haar-like features), the one selected being the one that has the highest classification accuracy. They then update the weights on the examples, so that the next selected feature will be better at classifying the examples the previous feature misclassified. This is repeated until a desired classification accuracy is reached.

However, to speed up the process they restrict the strong classifier to only use as many features as required to met a certain detection rate and instead use many layers, or cascades, of strong classifiers to improve performance. Each layer can be trained to achieve 100% detection rate but with the price of a high false positive rate. By moving down the layers, the detection rate decreases slower than the false positive rate due to them being calculated as multiplicative sums. If D is the final detection rate after 10 layers and each layer has a detection rate of 99%, the detection rate is $D = \prod_{i=1}^{N} d_i = 0.99^{10} \approx 0.9$. If the false positive rate for each layers is 30%, the total false positive rate is $F = \prod_{i=1}^{N} f_i = 0.3^{10} \approx 6 * 10^{-6}$.

This shows that high detection rates can be achieved with very low false positive rates. The approach is faster because of the fact that many larger sub-windows will be rejected already on the first layer of the cascade and very few examples will reach the deepest layers of the cascade. For a pseudo-code algorithm of their boosted cascade classifier, see the original paper [18]. Note that the algorithm is appearance-based and can be used to find any type of object.

3.2 Template matching

There are many ways of finding known patterns in an image, one is to use a technique called template matching. The idea is to use pre-existing models of the desired pattern to be found and try to match the models within the image. This is usually done by sweeping a template, a pattern similar to the one we are trying to find, across the image in a horizontal raster-scan and at each pixel calculate the distance (or similarity) between the template and the current region-of-interest. If the distance is small, then the two patterns are similar, and if it is big, it would tell that the pattern is not similar to the template. The technique can be used for a large variety of problems including pattern recognition, tracking and computer vision [23]. There are several different approaches in template matching (feature-based, area-based etc.) and the choice depends on the application.

One method of finding similarities between templates and images is called normalized cross-correlation (NCC). The strength of NCC is that it is not as sensitive to linear changes in amplitude in comparison to the ordinary cross-correlation [24]. This is achieved by subtracting the mean values of both the template and the ROI from the correlation similarity and dividing by their standard deviations. The NCC similarity measure is defined between the values of -1 and 1, where 1 is returned if template and ROI are exactly the same and -1 is returned if they have 180 degrees phase shift. The NCC can thus be seen as a cross-product between two normalized vectors (the template and the ROI). On the other hand, unlike cross-correlation, NCC does not have a straight forward simple frequency domain expression, which makes it more computationally demanding. Simpler similarity measures include the sum of absolute differences and the geometric distance.

3.3 Feature description

Comparing objects in ordinary gray-scale images is a difficult task. Images can be thought of as a very high dimensional discrete function, spanning a vast image space $\mathbb{R}^n$, where n is the number of pixels. Due to the vastness of image space, object images are very sparsely located, leading to large distances even between similar objects. This is also called the curse of dimensionality. Template matching in ordinary image space is very sensitive to changes in lighting, geometric transformations, occlusions and noise. Images also inherently contains large amounts of redudant information due to the fact that intensity values amongst neighbouring pixels are often highly correlated. Image compression techniques are used to encode images into more compact representations, where redudancy has been removed. A perfect object description would remove distortions together with redundancy and only retain the primary core of what constitutes the object in the image. The idea is that a suitable feature description improves the chances of template matching by transforming the object from image space into a feature space, where the distance between object descriptors are smaller and have better defined decision boundaries.

There are many different types of feature descriptors, some of the most frequently used are histogram of oriented gradients (HoG) [25], local binary patterns (LBP) [26], scale invariant feature transform (SIFT) [14] and Gabor filters [27]. Edges in images represent sudden changes in intensity and are often used for object recognition because of their robustness to lighting variations. They are generally calculated by convolution of the images with a gradient filter. By discretizing the gradient directions into a number of bins, histograms of the gradient direction can be built by counting the number of times each direction occurs. Such histograms are what constitutes the HoG-operator. Gabor filters, also called Gabor wavelets, are Gaussian kernels that have been shown to be able to capture information both with respect to spatial locality and orientation. By convolving a set of Gabor filters of varying scale and orientation, a suitable feature description can be obtained.

In face recognition, the descriptor has to be able to capture discriminative information for faces as a whole while simultaneously also capturing the information that separates faces from each other. Better results have been obtained by combining feature descriptors, examples include HoG+LBP [28] and Gabor+LBP [13],[29]. Hand-crafted features as the ones previously mentioned has recently been substituted by learning algorithms that themselves select features appropriate for discrimination between faces. Lei et al. [30] proposed a way to learn a discriminant face descriptor (DFD) that minimizes the difference between features of the same person and maximizes the difference between features of different persons. Deep convolutional neural networks have also been used to obtain face representations to great success [31]. However, most of the feature descriptors used in face recognition are derivatives of the local binary pattern operator, which will now be explained in detail. For a survey of LBP derivatives refer to the article by Nanni et al. [32].

3.3.1 Local binary patterns

Local binary patterns are feature vectors extracted from a gray-scale image by applying a local texture operator at all pixels and then using the result of the operators to form histograms that are the feature vectors. The original LBP operator is constructed as follows: Given a 3x3 neighbourhood of pixels as shown in Figure 3.3, a binary operator is created for the neighbourhood by comparing the center-pixel to its neighbours in a fixed order, from the left-center pixel in counter-clockwise manner. If a neighbour has a lower intensity than the center pixel it is assigned a zero, otherwise a one. This will yield an 8-bit binary number, whose decimal valued entry in a 256 bin histogram is increased by one. The complete LBP-histogram of an image will then depict the frequency of each individual binary pattern in the image. Due to its design, the feature vectors are robust to monotonic intensity variations since the LBP-operator is not affected by the size of the intensity difference. The feature vectors are not affected by small translations of the face either since the same patterns will be accumulated in the histogram regardless of their positions.

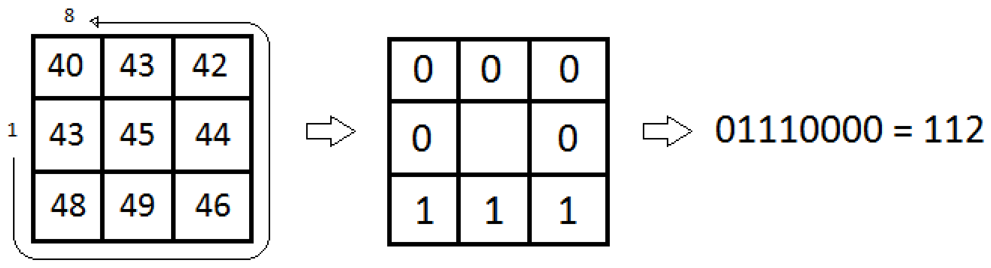


Figure 3.3: For a 3x3 neighbourhood in the image, the local binary pattern is determined by comparing the center pixel to its nearest neighbours in the fixed order assigned by the arrow. The binary pattern yields a binary number which is accumulated by one in the LBP histogram.

The operator can be extended from its nearest neighbours by instead defining a radius R where a chosen number of P points are sampled. The intensity values of the points are then calculated using bilinear interpolation, as explained in [22, p. 122]. The number of points will then determine the number of possible binary patterns and also the length of the feature vector. To reduce the length of the feature vectors, Ojala et al. [26] found that patterns with at most 2 binary transitions (0 to 1 or 1 to 0) provides over 90% of all spatial texture patterns, because they represent structures such as edges, spots and flat areas. These are called uniform patterns. For the LBP operator with R = 1 and P = 8 there are 58 uniforms patterns, all remaining patterns are accumulated into an additional

59:th bin. To simplify notation, this specific local binary pattern operator can be written as $\text{LBP}_{8,1}^{u2}$.

3.4 Linear subspaces

In the previous section, various methods for feature description of gray-scale images were introduced. Even though individual elements in the feature vectors may give more information about the image than for example a single pixel element would, there is still features more important than others. If one were to select a subset of features, which ones should be selected to retain the most amount of information? This process is called feature selection or dimensionality reduction.

Intuitively, a good way of selecting features would be by their variance. Features with low variance may not be very discriminative with respect to the overall structure of the data, compared to features with large variance that may have greater separability between classes. However, features are often correlated, prohibiting their removal. Assume that there is a data set of N observations $\{\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_N\}$, where each observation $\mathbf{x}_j$ has n dimensions (features). All observations are located in a linear space $\mathbb{R}^n$, that is spanned by a set of orthogonal basis vectors $\hat{\mathbf{x}_i} \in \mathbb{R}^n$ also called eigen-vectors, where $i = 1,...,n$, that allows each observation to be expressed as a linear combination of eigen-vectors $\mathbf{x}_j = \sum_{i=1}^{n} a_i \hat{\mathbf{x}}_i$ [22]. The goal is to find a transformation matrix $\mathbf{T}$, that maps the data points onto the eigen-vectors whose directions are defined such that the decrease of variance captured by each eigen-vector is maximized [33]. This means that in the transformed subspace, the first dimension is spanned by the eigen-vector in the direction of the maximum variation in the data, the second dimension represents the second most significant direction with respect to variation, and so on in decreasing order. This can also be defined as maximizing the determinant of the total scatter matrix, such that

$$\mathbf{T} = \arg\max_{W} \left|W^T S_T W\right| = [\hat{\mathbf{x}_1},...,\hat{\mathbf{x}_n}] \tag{3.3}$$

where $S_T = \sum_{j=1}^{N} (\mathbf{x}_j - \mu)(\mathbf{x}_j - \mu)^T$ is the total scatter matrix and μ is the mean of all observations [6]. A transformation that fulfills this condition is called the Karhunen-Loève transformation, whereas the method is generally referred to as Principal component analysis (PCA). A feature vector $\mathbf{x}_j$ can be transformed to the linear subspace by a simple matrix multiplication $\mathbf{y}_j = \mathbf{T}^T \mathbf{x}_j$. The variance of each individual eigen-vector is given by its eigenvalue, since $\lambda_i = \sigma_i^2$. The orthogonality ensures that the vectors will be mutally independent of each other and that variables in the transformed feature space is decorrelated. Features with small eigenvalues can thus be removed without a significant loss of variance by projecting the data onto the linear subspace $\mathbb{R}^m$ spanned by the eigen-vectors $\{\hat{\mathbf{x}}_1,\hat{\mathbf{x}}_2,...,\hat{\mathbf{x}}_m\}$ with the largest eigenvalues. Note that the data retains its dimensionality as long as its mapped onto all eigen-vectors, i.e. $m = n$. Finding the eigen-decomposition is usually done by singular value decomposition of the covariance matrix of the data, this is possible because the matrix is symmetric positive semi-definite [34].

PCA is in a sense a unsupervised learning algorithms, since it requires no prior knowledge of the data and only utilizes the data itself to find a suitable representation. However, if class labels are known for each data point then more optimal transformations with respect to data structuring can be found. Linear discriminant analysis (LDA) is a supervised learning algorithm that finds a transformation that maximizes the ratio of the determinant of the between-class scatter matrix and the determinant of the within-class scatter matrix [6]. As in PCA, a set of eigen-vectors and eigenvalues are used to transform the data onto a linear subspace, where the eigen-vectors now are both separating data points of different classes while simultaneously clustering data points of similar class. A property of LDA is that the dimensionality is reduced to $m = c - 1$ where c is the number of classes, which for images is a large reduction in dimensionality. Note that LDA is almost always preceded by PCA [35][36], since removing low variance dimensions is a good way of reducing noise, but also because the dimensionality needs to be reduced in order for the within-scatter matrix to be non-singular [6].

Whitening is a technique that is often performed in subspace methods and has shown to improve performance [6]. A whitened space is defined as subspace where each feature dimension has sample variance one. This is realized by dividing each dimension x_i in the transformed space (after PCA) by its standard deviation, which is the square-root of the corresponding eigenvalue. The feature vector in the whitened subspace is therefore

$$\bar{x}_i = \frac{x_i}{\sigma_i} = \frac{x_i}{\sqrt{\lambda_i}} \tag{3.4}$$

This can be imagined as a sphering of the data, where the axes of the subspace (the eigen-vectors) will have same length. The reason to use whitening is that when comparing data points in the subspace, with for example euclidean distance, each feature dimension should be treated as equally discriminative since they are uncorrelated and thus should have equal sample variance, otherwise the distance contribution by some features will be larger than others. Even though eigen-directions with large eigenvalues contains contains more of the variation in the data, the discriminative power of less significant eigen-directions should not be suppressed, which is why whitened subspaces are preferred in pattern recognition.

3.4.1 Nearest neighbourhood classifier

The simplest method for matching feature vectors is using the nearest neighbourhood classifier. It calculates the distance between the probe vector to be classified and the gallery vectors, and then assigns the probe the class label of its nearest neighbour in the gallery. If the distance is zero, then the images matched are exactly the same. The distance measure can be converted to a similarity measure simply by negating it, such that the chosen match is the one with the maximum similarity value. The choice of distance metric depends on the type of task, only euclidean distance, cosine distance and chi-square similarity were evaluated in this project. These are defined as

$$\text{Euclidean distance: } d^{e}_{xy} = \sqrt{(\mathbf{x}-\mathbf{y})(\mathbf{x}-\mathbf{y})^{T}} \tag{3.5}$$

$$\text{Cosine distance: } d^{c}_{xy} = 1 - \frac{\mathbf{x}\cdot\mathbf{y}}{\|\mathbf{x}\|\,\|\mathbf{y}\|} \tag{3.6}$$

$$\text{Chi-square distance: } d^{\chi^2}_{xy} = \sum_{i=1}^{N} \frac{(x_i - y_i)^2}{x_i + y_i} \tag{3.7}$$

where $\mathbf{x}$ and $\mathbf{y}$ are row vectors of length N. For an analysis of nearest neighbour pattern classification, see the article by Cover et al. [37]. The advantage of the NN-classifier is that it does not require any training stage and that it naturally extends to multi-class classification. Training other classifiers such as support vector machines (SVM's) and neural networks is often computationally demanding for high-dimensional data with many examples. Even though they may increase matching performance by accounting for non-linearity in the data, training would have to be done every time the gallery is altered.

Chapter 4: Method

4.1 Test environment

Since the Smart Eye image acquisition system is not operating in the wavelengths of visible light, the image content will differ from colour or gray scale images. This means that training algorithms on public databases which is not in the near-IR spectrum could be harmful to algorithm performance and testing on such databases could also prove to be non-representative of the actual performance on Smart Eye images. There is no publicly available database of near-IR videos to our knowledge and thus we have chosen to create our own database for evaluation of the methods developed in this project.

4.1.1 System description

An artificial test environment was used in this project to simulate a real-world scenario where a driver enters a motor-vehicle. The setup included an adjustable driver's seat, a steering wheel and a table as the top of the dashboard. Three Basler acA640-120GM monochrome cameras using 8 mm focal length lenses with aperture F1.4 was used for image acquisition purposes. These were placed so that one was on the dashboard in-front of the driver looking through the steering wheel, one was on the right-side of the driver looking up to mimic a camera mounted below the dashboard, and the last camera was placed to the left of the steering wheel at the corner of the dashboard (beside the imagined left rear-view mirror). This camera setup is regularly used in Smarteye applications, Figure 4.1 shows a set of images taken using the setup.

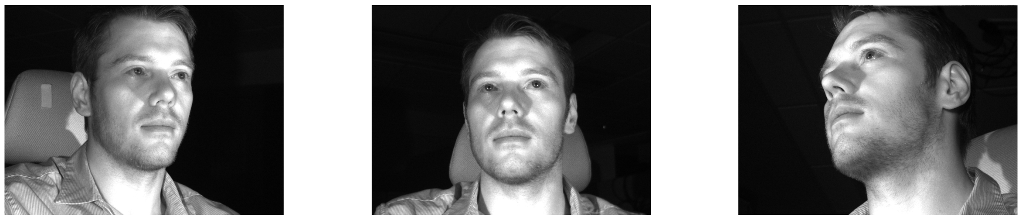

(a) Camera 1 **(b)** Camera 2 **(c)** Camera 3

Figure 4.1: Example images taken with the camera setup.

These cameras respond to both visible light (400-700 nm) and near-infrared light (700-1100 nm). To control the illumination, 40 V flashes sending out light at 850 nm was placed beside each camera pointing towards the user. The cameras and flashes were then connected to a 12 V Smart Eye Exponator connected to a computer. The image data retrieved from the cameras were 8-bit monochromatic with resolution 640 x 480 pixels at a frame rate of 60 Hz. With 3 cameras this generates around 3.1 GB image data per minute. All videos were stored uncompressed in the format *.smb, a binary format with each image separated by a description header. With knowledge of the contents of the header, the videos can easily be read in MATLAB.

The Smart Eye tracking profiles also contains image data stored in *.smb files that can be read in a similar manner as the videos, where the data consists of feature templates followed by a selection of full-view images in different poses. Accompanying the image data is also a ASCII file that describes the content of the *.smb file. This file is parsed using our own MATLAB routine to associate each template with a facial feature.

4.1.2 Video databases

Two databases were created for this project, a small preliminary one that was used to test the algorithms and a larger database which was collected for the real evaluation. The databases themselves consist of two parts: a profile data set containing user data in the form of models and templates, and a data set of video sequences. The profiles were used to create a stored data set of subject information that could be matched against the video sequences. All recordings and profile tracking were done using the Smart Eye Pro 6.0 software.

The first data collection task was to create a tracking profile of the user, who was asked to look around in order for the software to be able to build a profile with templates from all angles of the face. There was no way to control the quality of the profiles until after their creation, which leads to a variance in quality of the tracking profiles. For example, some profiles were more extensive, containing up to 900 templates, whereas others only had about 250 templates. This must be considered in the analysis of the results.

The user was then asked to act in a manner similar to how he or she would in a real car while the cameras were recording. The length of the sequences were limited to 2000 image sets, which is equivalent to a video sequence slightly longer than 30 seconds with a 60 Hz setting, a time frame during which a recognition is desirable. If all the video sequences were longer it would yield large amounts of data to analyze, which would be very time consuming. On the other hand, if they are too short there is a risk that the person would not expose him/herself enough for the methods to be properly evaluated.

The pilot database did not have any specific requirements regarding the user behaviour or demographic. It was recorded in a small room with two cameras, using a fixed aperture setting during the recording session. It consists of seven people, all of which were males in the age range of 24-51 years with an average age of 30 years distributed over 11 videos. Further information can be found in Appendix A.4. All persons had one video with ordinary apperance, out of which four also had a video with a modified appearance. The modifications are specified in Appendix A.4.

The real database was requested to have a wider representation in age, gender and ethnicity. The setup was as described in Section 4.1.1 and during the recording session a fixed step-by-step methodology was followed: the user was first asked to sit down in the car seat and adjust the seat to his/her liking, then the cameras were adjusted so that the face was centered in all of them and the apertures were adjusted so that all faces were equally exposed. Finally, to ensure correct tracking performance, the positions of the cameras in the world coordinate system was recalibrated.

This database consisted of 24 people with a male to female ratio of 15:9, over an age interval from 24 to 51 years with an average age of 30 years. The goal was to have an as varied demography as possible in the database to be able to fully evaluate the face recognition algorithms. Almost every person in the database (23 out of 24) were recorded two times, once with ordinary apperance and a second time with a modified appearance. If they wore glasses, they took them off and put on a hat for example, see Appendix A.3 for specifics.

4.1.3 Video classification

Our task is to match the subject in each of the N videos to a known database of M subject-specific profiles, where the subject in each video n is represented by a personal profile m. This is the ordinary setup, additional setups can be created to test algorithm performance in cases where the user is previously unseen (i.e. new to the system) by simply removing the subject's profile.

For a generic test run of the face recognition system, each video is processed and returns a subject ID, represented by an integer. The ID is determined using the video data and the data extracted from the previously stored profiles. When all videos have been processed, the identities selected by the algorithms are compared to the real identities of the subjects in the videos and if the response is a match then the video has been correctly classified. Mathematically, this can be written as

$$r_n = \begin{cases} 1 \text{ , if } \mathrm{ID}_n = \mathrm{ID}_n^* \\ 0 \text{ , if } \mathrm{ID}_n \neq \mathrm{ID}_n^* \end{cases} \tag{4.1}$$

where r_n is the classification response for the n:th video, ID_n is the identity chosen by the system and ID_n^* is the correct identity of the subject in the video. The recognition rate of the system is defined as

$$R = \frac{\text{\# of correctly classified videos}}{\text{\# of videos in the database}} = \frac{\sum_{n=1}^{N} r_n}{N} \tag{4.2}$$

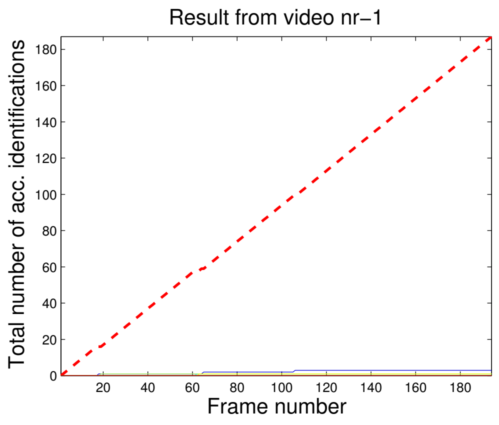
This measures the global performance of the system, but does not detail the performance of the algorithm on each individual video or the strength of each decision made by the system on specific videos.

Each video is treated as a sequence of images, where each image is processed independently by the system. If the current image meets the requirements of the system, the identity of the person in the image is determined by selecting the closest match in the gallery defined by some similarity measure. Just as in Equation 4.1, the identification response for each processed image i is defined as

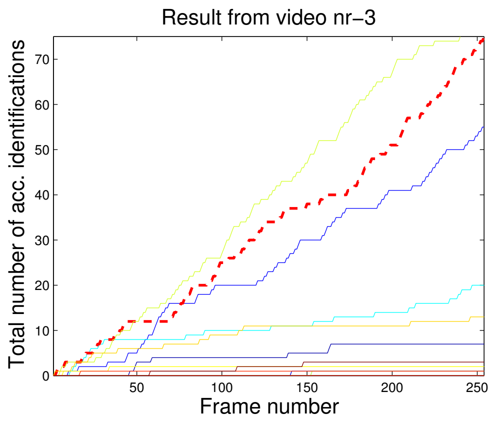
$$f_{n,i} = \begin{cases} 1 \text{ , if } \mathrm{id}_{n,i} = \mathrm{ID}_n^* \\ 0 \text{ , if } \mathrm{id}_{n,i} \neq \mathrm{ID}_n^* \end{cases} \tag{4.3}$$

where $\mathrm{id}_{n,i}$ is the chosen image identity. This means that for the n:th video containing z processed face images, we have a vector $\mathbf{v}_n = [\mathrm{id}_1,..,\mathrm{id}_z]$ containing the identification responses of all frames in that video. The final decision of a subjects identity in a video is chosen as the identity whose profile has been the best match the most amount of times i.e. $\mathrm{ID}_n = \arg\max(\mathbf{f})$ where $f_j = \sum_{i=1}^{z} f_{j,i}$ is the amount of times profile j was selected for video n. Additionally, in cases where multiple identities have the same amount of identity classifications (it is common that the first two images are classified as different persons), the identity whose average distance metric is lowest over all classified images is selected.

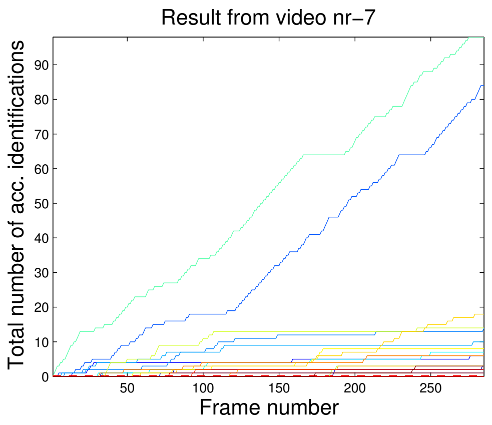
The recognition rate will vary depending on how many images in each video that were processed, it is therefore of interest to investigate the recognition rate as a function of the number of images processed. Images are not randomly picked from the video, but in the correct order corresponding to the video. Since the goal is to correctly identity the subjects as early as possible, face images are evaluated as they are obtained.



(a) Good results.



(b) Decent results.

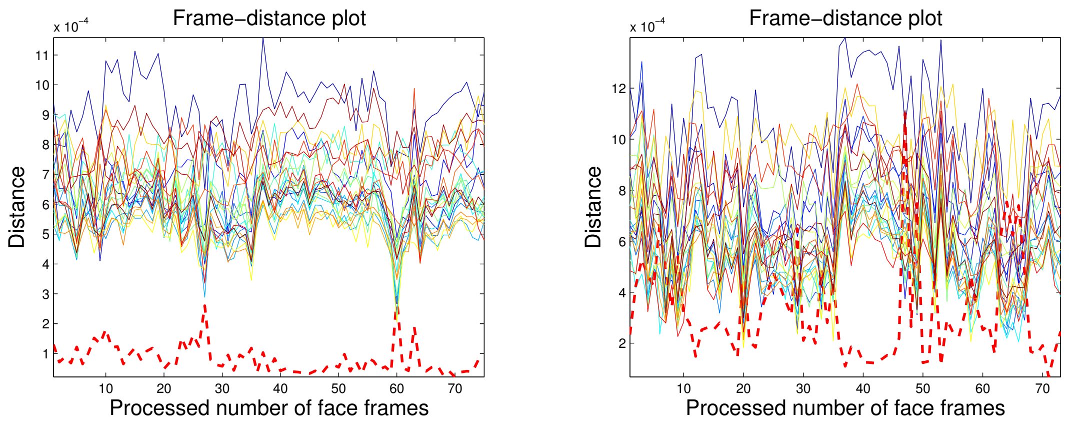


(c) Bad results.

Figure 4.2: Plots showing the accumulated number of idenfications as a function of the number of processed face images, the correct identity is marked by the red dashed line.

Typical results from a few processed videos is shown in Figure 4.2. On videos where the algorithm has high performance, as in Figure 4.2a, the correct identity (marked by the red dashed line) is almost exclusively chosen as the output for every image. This ensures that the final decision on the identity of the subject in the video is correct, no matter the number of images processed. This is not the case in Figure 4.2b, where the final decision will vary depending on how many images the algorithm has processed. Figure 4.2c illustrates a video where the correct identity is barely selected at all, an indication that the person has either changed their appearance drastically or that the gallery for some reason does not contain face descriptors representative of the subjects actual appearance. The latter case is far more concerning, since this indicates that the algorithm cannot properly process the information in the tracking profiles. One must therefore evaluate modified and unmodified videos seperately to ensure that algorithm performance is mostly lowered by modifications rather than improper matching procedures.

As the time between images processed by the system might vary between videos, plotting the recognition rate as a function of the number of processed images might provide an estimate of how much information that is needed for a correct classification, but not of the average time to make that classification. The recognition rate is thus also plotted against time.



(a) Video of unmodified subject. **(b)** Video of modified subject.

Figure 4.3: The mean distance between the current image face descriptor and the identities in the gallery for the aligned cosine setting evaluated for two different videos. The red dashed line is the correct match. The difference in distance between correct and false matches in much smaller when the subject is modified.

Subjects modifying their appearance is problematic for identification tasks, almost always resulting in reduced recognition rates. The effects of modifications can be investigated by calculating the probe to gallery distance as a function of the number of processed face images. Since every subject has multiple entries in the gallery, a mean distance between the probe and each identity must be used. This is shown in Figure 4.3 for one modified and one unmodified video, with the correct identity marked by the red dashed line. It can be seen that the distance between the probe and the correct identity is much smaller in the unmodified video than in the modified. The difference in distance between the correct and the false matches is also very small in the modified video, leading to misclassifications of images where the distance is larger, due to perhaps head rotations or facial occulusions.

4.1.4 Closed-set identification

When a threshold has been fixed on either the amount of images or the time required to make the final decision classification, a cumulative match characteristic (CMC) curve can be created. This is done by plotting the probability that the correct identity is among the top n matches as a function of rank-n. The probe, in this case the subject being identified, has rank-n if it is the n:th closest neighbour to the correct identity in the gallery [38]. Plotting CMC-curves is the common way of visualizing results in closed-set identification, where the probe is assigned a class label from the gallery no matter the distance to its nearest neighbour. Because of the fact that our system classifies images sequences, the definition has to be changed such that rank-n instead is determined by the number of correct image identifications instead of a single image similarity.

A probe video n with corresponding profile identity m has rank-n if f_m is part of the n:th largest elements of $\mathbf{f}_n$, the vector that contains the number of times each profile has been selected as the image identity. Mathematically, we define the identification (and detection) rate at rank-n as

$$P_{DI}\left(\tau,n\right) = \frac{|\{f_m : m \in \mathscr{P}_\mathscr{G}, \text{rank}\left(f_n\right) \leq n\}|}{|\mathscr{P}_\mathscr{G}|} \tag{4.4}$$

$$= \frac{|\{f_m : m \in \mathscr{P}_\mathscr{G}, \text{rank}\left(f_m\right) \leq n\}|}{N} \tag{4.5}$$

where $\mathscr{P}_\mathscr{G}$ is the gallery. This definition thus gives us the probability that the probe of a subject that belongs to the gallery has the n:th most amount of identifications to the correct sample in the gallery. The ordinary definition of identification rate can be found in in [38].

4.1.5 Open-set identification

The problem can be changed to open-set identification by imposing a distance threshold on the distance metric, such that if the distance to the nearest neighbour is larger than the threshold, then the probe is assumed to not be present in the gallery and the subjects identity is set to unknown. Receiver operating characteristic curves, shortened ROC-curves can then be plotted. ROC-curves plots the recognition rate as a function of the false positive rate. To do this one must first choose which rank-n to plot, rank-1 being the most common choice, and then vary the threshold. The open-set recognition rate is then

$$P_{DI}\left(\tau,n\right) = \frac{\left|\left\{f_m : m \in \mathscr{P}_\mathscr{G}, \text{rank}\left(f_m\right) = 1, \text{and } d_{*m} \leq \tau\right\}\right|}{\left|\mathscr{P}_\mathscr{G}\right|} \tag{4.6}$$

where f_m is the amount of times profile m has been selected, $\mathscr{P}_\mathscr{G}$ is the gallery , τ is a threshold on the distance measure, and d_{*m} is the smallest distance between the probe and the closest match in the gallery. This definition thus gives us the probability that the probe of a subject that belongs to the gallery has the smallest distance measure to the correct sample in the gallery and its distance is smaller than the distance threshold. The distance threshold is set to $\tau = \infty$ in closed-set identification. The recognition rate is now only concerned with the best match.

Just as for the recognition rate, the false positive rate has to be redefined to accommodate for classification of image sequences. Even if one image was classified as not part of the gallery, other images could be classified as part of the gallery and thus cause problems. If a probe is classified as not part of the gallery, it is denoted as class unknown. A false positive is thus defined as the case when the class unknown has less identifications than one of the profiles part of the gallery $\mathscr{P}_\mathscr{G}$ for a probe with profile $m \in \mathscr{P}_\mathscr{N}$ not in the gallery. Note that $|\mathscr{P}_\mathscr{G} \cup \mathscr{P}_\mathscr{N}| = N$ is the size of the database. The false positive rate is therefore

$$P_{FA}\left(\tau\right) = \frac{\left|\left\{f_m : m \in \mathscr{P}_\mathscr{N}, f_0 < \max\left(\mathbf{f}\right)\right\}\right|}{\left|\mathscr{P}_\mathscr{N}\right|} \tag{4.7}$$

where $\mathscr{P}_\mathscr{N}$ is the subset of the data base that is not part of the gallery, f_0 is the number of times that the class unknown was selected for subject with profile m and $\mathbf{f} = [f_1,...,f_M]$ contains the number of times each profile was selected for image identification. Note that in comparison to the closed-set identification, the data set must in open-set identification also contain subjects not part of the gallery and thus our data set must be divided into two equally sized parts. The ordinary definition of the false positive rate for images can be found in [38].

The performance of the system in open-set identification is usually measured by the identification rate at a specific false positive rate or by the area-under-curve (AUC), which can be calculated using the trapezoidal method. The AUC measures the performance of the system over the whole range of false positive rates.

4.2 Face detection

The Viola-Jones face detection algorithm with boosted Haar-cascades, as explained in Section 3.1.1, was used to locate faces in the videos and profiles. It is fast, accurate and allows easy implementation in MATLAB with the Computer Vision toolbox, which contains functions that can be used to create cascade objects and applying them to images. To reduce the information that needs to be processed, a decreased number of image sets per second were searched for faces, the original frame rate of 60 Hz (image sets per second) is rather redudant and the difference between closely acquired images is very small. A frame rate of 6 Hz was used for the LBP-method which is enough to capture movements of the subject while keeping the images unique. The template matching method used a lower frame rate of 2 Hz because of its long computational time.

Instead of training new Haar-cascades, the publicly available ones in the open-source computer vision toolbox OpenCV was used [39]. It contains multiple cascades, four of them trained on full frontal faces. More information on these Haar-cascades can be found in the article by Castrillón et al [40]. See Appendix A.1 for a full list of the Haar-cascades used in this project and a summary of their corresponding settings.

Images of full frontal faces are generally more descriptive and discriminative than images of rotated faces, which is why full frontal cascades are desirable in face recognition tasks. However, the OpenCV cascades are trained on different image sets, so one cascade of classifiers might consider a face full frontal and another might not. In order to make the face detector more robust, all four frontal face OpenCV Haar-cascades are applied on the image. Only when all cascades declares a successful face detection is the face classified as full frontal and thus cropped from the image. The face box used for cropping the image is set too the mean of the face boxes produced by each cascade. This method increases robustness both to face box size and head rotations, with most face detections close to full frontal.

A typical detection made by the face detector is shown in Figure 4.4. The blue boxes mark the detections by the four individual OpenCV Haar-cascades and the mean face box marked in red is the final output. Because of the fact this is a clear full frontal face image, the blue boxes are tightly packed and hard to discern from eachother. The mean face box is returned since all of the Haar-cascades had a positive detection. If one of them had not found a face, an empty face box had been returned. This detection scheme could thus remove correct detections, where only two or three Haar-cascade has returned a response, at the cost of detection

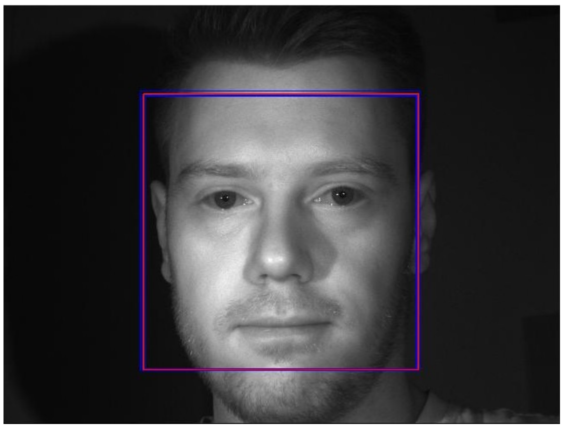


Figure 4.4: A face detected using the Viola-Jones algorithm and OpenCV Haar-cascades. The blue boxes represent individual detections made by different Haar-cascades, whereas the red box is the mean box of all detections and the final output of the detector.

robustness. Using less than four Haar-cascades has not been investigated.

The mean face box is in itself not the optimal choice, since it is exposed to outliers. However, it gets more stable with an increasing number of detections, so four cascades should yield decently stable detections. The Viola-Jones algorithm might not produce the best croppings in terms of facial accuracy but it is very stable with respect to false detections.

Each cascade can be tuned by the following parameters: min size, max size, scale factor and merge threshold. These can be used to tailor the performance of the detector for each specific task.

Min size and max size is used to regulate the bounding box search interval. Since the goal is to find faces in a relatively constrained environment, the bounding box size can be used to decrease run-time and homogenize results. Face images will often be scaled to a fixed size at a later stage, and in these scaled images there will be variations in information content between faces that initially were small and others that initially were large. Scaling is an approximative operation and fixing the face size range ensures that results in later operations are not compromised. Min size [200,200] and max size [400,400] were selected, which seemed to include most full frontal face images in the profiles.

The scale factor determines in which increments the bounding box size is increased. Small increments increases detection accuracy but also increases run-time. The merge threshold regulates how many detections that needs to be merged in order to be classified as a correct detection. Increasing this parameter might increase accuracy up to a certain value, after which detections are rejected too frequently. With no particular preference between accuracy and speed, the standard values of these parameters were used. See Appendix A.1 for specifics.

4.3 Feature extraction

The Viola-Jones algorithm was also used with the OpenCV Haar-cascades to extract facial feature points from the face images. The face images are mostly full frontal (some rotated faces are still included), so the centers of the eyes and the mouth are extracted from the face images due to their robustness and invariance over different individuals. This is essential to the facial alignment step, since it cannot be performed without feature points. The following cascade settings were used: a scale factor set to 1.05 and a merge threshold set to 4. The min size was set to [20,20] for the eyes and [15,25] for the mouth, which were the smallest values allowed. Max sizes were set to [100,100] and [100,150] for the bounding boxes. The merge threshold was set to 8 for the eyes and 12 for the mouth to decrease the amount of spurious responses. An explaination of these settings is given in Section 4.2 and a summary of the settings is available in Appendix A.1.

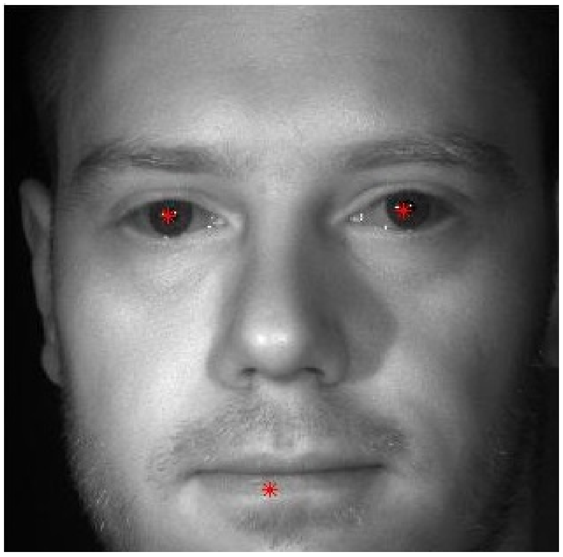


Figure 4.5: Eye and mouth center points detected using the Viola-Jones algorithm and OpenCV Haar-cascades.

Figure 4.5 shows the detected features after the image in Figure 4.4 has been cropped using the detection marked by the red box. The figure shows that all three features has been successfully found with good accuracy on the eye centers and decent on the mouth position. It is reasonable to assume that the detections for the mouth position will have a larger variance because of a larger difference in mouth appearance between individuals than the eye centers.

To increase accuracy and speed, the knowledge of facial geometry for full frontal face images was used. It was implemented by only searching for features in the parts of the image where the features intuitively should be located. The left eye center was only searched for in the top-left quadrant, the right eye center in the top-right quadrant, and the mouth center in the bottom half of the face image. This could be done more accurately by tiling the face image into even smaller pieces, but this is sufficient to remove obviously inaccurate detection results.

In the cases where multiple boxes for the same feature are found, the box with the smallest area was selected for the eyes and the box with the largest area was selected for the mouth.

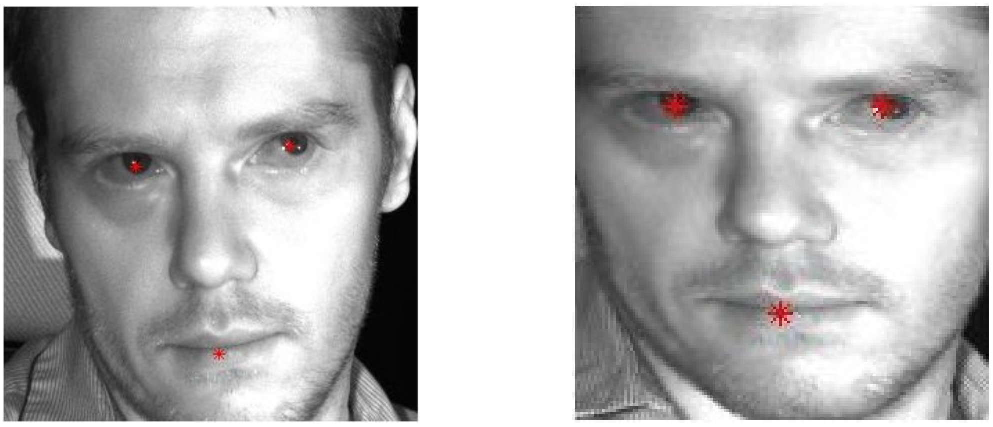
4.4 Facial alignment

Aligning faces with respect to scale, orientation and cropping has been shown to improve performance in face recognition and is often a crucial step in such systems [41]. Using the previously extracted eye and mouth center points and a simplistic 2D face model, an affine transformation can be found by solving the linear least-squares problem defined by Equation (11.17) in [42]. This transformation can be applied to the facial image to normalize the face with respect to scale and rotation.

The face model was defined as the positions of the eye centers and the mouth seen from a full frontal pose. The interocular distance between the eye centers was used as the only model parameter and set to D = 112 pixel units. This is motivated later in the identification step. Eye positions were chosen to have the same height, in such a way that a connecting line segment between has a 0 degree angle in reference to the image coordinate system. The mouth center position was defined as D pixel units below the eyes in the middle of the connecting line.

The affine transformation, once calculated, is then applied to the full image containing the face. The facial feature points were also transformed, since the transformation affects their coordinate positions. Note that all three points are necessary to calculate the affine transformation, which utilizes sheering in x and y directions and translation to find the optimal transformation, whereas only two points are required to normalize for scaling and rotation. However, the translation elements in the affine transformation are set to zero since the position of the face model is arbitrary, only its shape is of interest. For consistency, only images where all three facial features were found are aligned. The rest are regarded as not suitable and an empty matrix is returned instead of the aligned facial image. This provides assurance that the aligned face images usually are good full frontal views of the face.

When the full image has been aligned using the transformation, the aligned face can finally be cropped. The face center is defined as the center point between the eyes and mouth center, this is roughly the position of the nose. Using the face center as a point of reference, the face box is set to size [224,224] pixels i.e. two times the interocular eye distance defined earlier for the face model and the face is cropped. The face box must however be within the image for the cropping to be successful, a try catch statement is thus used to avoid errors and an empty matrix is returned if the box is out of bounds.



(a) Unaligned face. **(b)** Aligned and re-cropped face.

Figure 4.6: A face before and after the alignment step, detected feature points are marked in red. The eyes have been horizontally aligned and the image has been cropped tighter around the face. Note that the images are not shown in their original sizes, the aligned image has been enlarged for viewing purposes.

The results of a typical facial alignment is shown in Figure 4.6, where the affine transformation has successfully aligned the feature points according to the face model. The eyes are horizontally aligned to eachother with the interocular distance normalized to the fixed distance and cropping is centered directly on the face. In this example, the face has only been rotated around the cameras z-axis which points straight out of the camera. The affine transformation have managed to reduce the difference in facial symmetry due to rotations, even though the output is slightly skewed.

4.5 Face recognition

Two methods were evaluated seperataly in the face recognition step, these are described in detail below.

4.5.1 Using local binary patterns

Two stages are involved when using local binary patterns, from now on called LBP, in face recognition. First, a database of biometric signatures, in this case LBP-vectors, must be created so that new previously unseen signatures can be classified using a matching process. This is the training stage, where face descriptors (or feature vectors) are extracted using the image content in the Smart Eye tracking profiles. Once the training has been completed, images or videos containing a person of unknown identity can then be processed and compared to the database to find the best match and hopefully the correct identity of the person.

Each profile contains several full scale, non-cropped images of the subject in different poses. The Viola-Jones face detector explained in Section 4.2 is applied to all images and face boxes are extracted. The face is then either cropped from the image and nothing else or the face image gets processed by the feature extractor (Section 4.3) so that the facial alignment step (Section 4.4) can be performed. Both options are available to allow for performance comparisons and evaluate the benefit of the alignment step.

Once a face image is retrieved, aligned or not, it is passed to the identity detection step for encoding. To increase the number of face images in the database, the left-right flipped verision of the face image is also processed and stored. The face image is first resized to the fixed size of [224,224] pixels. This is necessary for non-aligned images due to the varying face box output of the face detector. The size is already correct for aligned images thanks to the choice of interocular distance and cropping.

The methodology for LBP face recognition proposed by Ahonen et al. [17] has been adopted with some simplifications. The face image is first tiled, meaning that the image is divided into non-overlapping blocks. Each tile is encoded using the LBP-operator which yields a normalized histogram containing the image information for that specific tile. Its length depends on the settings of the LBP-operator. The $\text{LBP}^{u2}_{8,2}$ settings suggested in [17] were used, see Section 3.3.1 for an explaination of the operator. To encode face tiles into LBP-histograms, the code provided by Heikkilä and Ahonen [43] was used. All tile-specific histograms are finally concatenated into a single feature vector, a face descriptor, which now is an encoded representation of the face image. An example of the tiling process is shown in Figure 4.7, dividing the image into four tiles. No further normalization is performed on the concatenated feature vector, because it did not seem to increase performance.

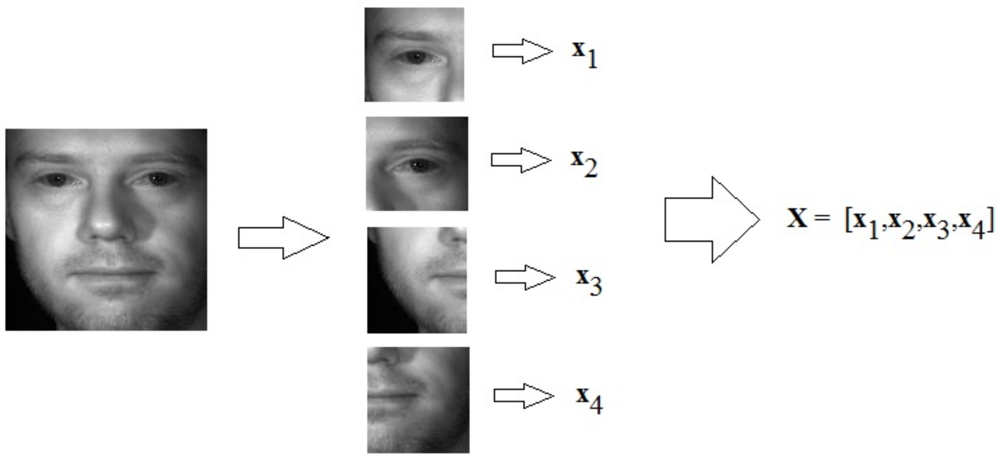

$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\mathbf{x}_4$
$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4]$

Figure 4.7: Illustration of the encoding process. The face image is divided into tiles, that each are encoded using the LBP-operator into normalized histograms, which then are concatenated into a single face descriptor.

The size of the face descriptor depends on the length of the tile-specific feature vectors and the number of tiles, which in turn is determined by the size of the face image and the size of the tiles. One must therefore make a choice on the trade-off between description accuracy and locality of the operator and the length of the face descriptor, which will have dimension $1x(N * M)$ where N is the length of the LBP operator for each tile and M is the number of tiles. The face size was set to [224,224] pixels and the tile size to [32,32] pixels, this generates a tile grid of size [7,7]. The uniform $\text{LBP}^{u2}_{8,2}$ has 59 bins so the final concatenated feature vector has $59 * 7 * 7 = 2891$ elements. This is small compared to the dimensionality of the face image.

Once a face descriptor has been created it is added to the gallery of all previously stored face desciptors together with a number indicating its class label (identity).

The database may however not be defined in a way optimal for descriptor matching. Principal component analysis is applied to the database to transform it to the linear subspace that is defined by the inherit variance of the data. To remove noisy and statistically irrelevant dimensions, only the eigenvectors whose eigenvalues constitutes 99.999% of the total sample variance in the data is kept. The code used for perfoming both PCA and LDA was created by Deng Cai and can be found at [44]. It is worthy to note that the eigenvalues returned by the code are all normalized to approximately 1, suggesting that the eigenvectors have been whitened, and that distances that are calculated in the LDA subspace are therefore Mahalanobis distances. For a detailed description of PCA, LDA and whitening; see Section 3.4.

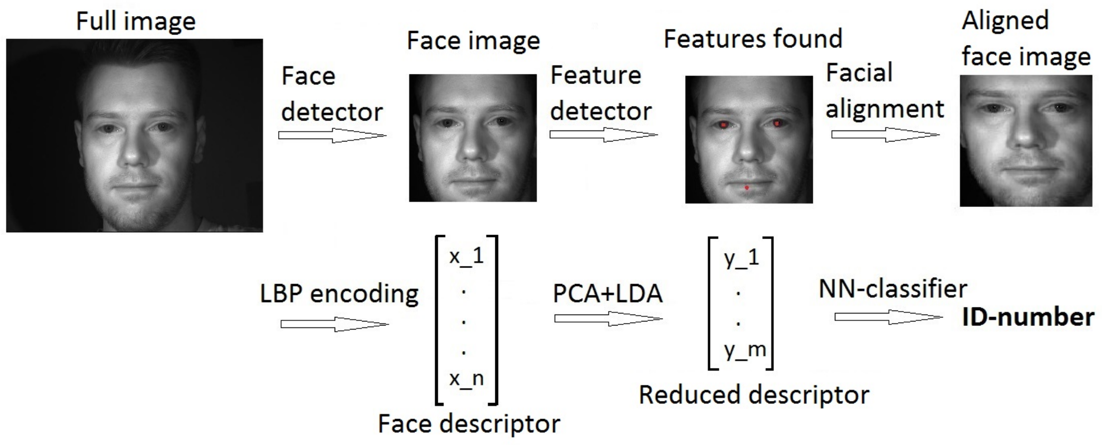


Figure 4.8: Illustration of the face recognition process. The face is detected, searched for features, aligned, encoded using the LBP operator, processed using the linear subspace methods and then finally classified using the nearest neighbour method.

The database now contains information suitable for matching. When an unknown face image is found in a video or a sequence of images, it is processed as shown in Figure 4.8. Instead of storing the reduced descriptor in the gallery, the probe is matched to all descriptors in the gallery using the nearest neighbour classifier. Restricting the largest allowed distance to the nearest neighbour with a threshold changes the the task from closed-set identification to open-set identification, as explained in Section 4.1.3. Neighbours with distances above the threshold are then deemed too far away from the subject to actually be the same person and the identity is assigned as unknown.

The choice of a proper threshold depends on the length of the feature vector and the distance metric itself. The former dependency can be removed by dividing the distance by the length of the feature vector, whereas the latter cannot and distance metrics has to be evaluated separately. Three different distance metrics were evaluated for the 1-NN classifier: euclidean, cosine and chi-square distance. The chi-square distance requires the feature vectors to be positive normalized histograms and was therefore omitted for the PCA+LDA method. Since testing various distance metrics generates a lot of data to process, open-set identification was only evaluated on the distance metric and method that performed the best in the closed-set identification.

For every image that is processed, an identity is assigned and the result is stored in an accumulator matrix that counts the number of times each identity has been selected, including the unknown identity. The number of processed images for each video will vary since the subjects do not perform the exact same movements, which affects the output of the face detector. Only the result up to the minimum amount of images included in all videos are shown. It would not be wise to include result after that, since the recognition rate would then be calculated for a subset of the database.

4.5.2 Using template matching

One of the downsides of the LBP-method is that it requires full frontal images from a video source of which there is no control of the content, therefore it is not guaranteed, although plausible, that a full frontal image will be present in the image stream. The worst case running scenario would then be one camera placed in an non-frontal position where the chance of the driver facing that camera is low, in this case the LBP-method would falter since very few, or non at all, full frontal face images will be captured.

The way to tackle this issue is to implement a method which is not pose dependant, and since the Smarteye tracking profiles contain templates of all features in different poses, they have been used in implementing a template matching algorithm.

The face recognition problem is then reduced to "How to recognize a person from a set of templates?", and it is not straight forward to answer this question. Since each profile contains a high number of templates, and it is assumed that since we all are humans with human facial features, all templates will correlate positively with every face, since the NCC scores are between negative one and positive one, but they will only score very high in the image of the same person. This is however not always true and sometimes person A's template will score close to 1 on an image of person B, see fig 4.9, therefore the recognition cannot be based on any single template, but has to use one of the two classification methods developed in this project.

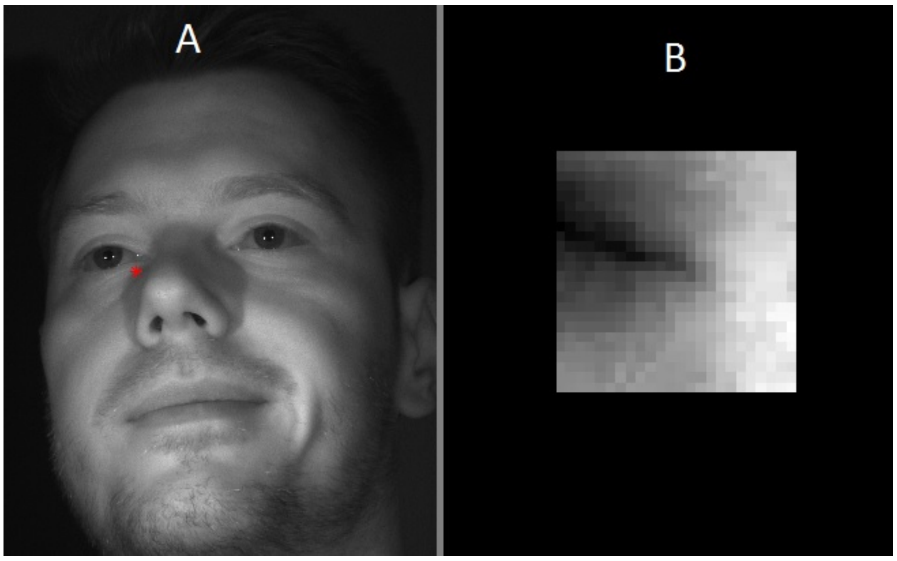


Figure 4.9: Left image shows a face image of a person A, and the image to the right shows a template of a mouth corner of a person B. The red star in the left image indicates where the template from B scored 0.964 in A, which shows that even features may score high in other parts of the face, and therefore no conclusions can be made from a single template score

Frame score classification (FSC) is a classification method which consists of a two step classifier. First it classifies separate frames independently, a frame score vector is formed to count how many frames each profile has been classified as. Then the whole video is classified as the profile with the highest frame score.

A frame classification is done by matching every template, using NCC, to the frame, and if the strength of the cross-correlation is higher than a threshold T, then a template counter, C_{temp}, is incremented, see equation 4.8. A template score vector (TSV) of size $n_{profiles}$, see equation 4.9, keeps track on how many templates in each profile were stronger than the threshold for said frame.

$$\begin{Bmatrix} \text{if } Temp_i > T & \text{then } C^p_{temp} = C^p_{temp} + 1 \\ else & nothing \end{Bmatrix} \tag{4.8}$$

Video \ ID	1	2	3	4	5	6	7
1	**22**	1	0	0	0	0	1
2	**13**	0	4	0	0	0	8
3	3	**11**	5	1	1	0	1
4	0	**14**	4	0	0	0	0
5	0	0	**36**	0	0	0	2
6	3	0	1	**16**	0	0	1
7	0	2	0	**22**	0	0	5
8	2	0	0	0	**21**	1	14
9	1	2	0	0	**22**	3	8
10	4	2	1	0	0	**10**	2
11	0	0	1	0	0	0	**33**

Table 4.1: The FSV is shown here for each video in the pilot database, the highlighted numbers are the correct IDs for each video, and as it can also be seen is that all the highlighted IDs also have the highest score, which would give a 100% recognition rate. The threshold for this classification was 0.945.

$$\text{TSV} = \begin{bmatrix} C^{1}_{temp} \\ C^{2}_{temp} \\ . \\ . \\ C^{n}_{temp} \end{bmatrix} \tag{4.9}$$

The frame classification is then done by choosing the profile which had the highest score in the TSV, this score is itself counted for each profile and saved in a frame score vector (FSV) similar to the TSV. When all frames have been classified , a final classification is done by counting all the separate frame classifications and choosing the profile with the highest frame score, see figure 4.10. An example of how the TSV looks when the pilot database is analyzed for each frame can be seen in figure 4.11. It can be seen that the correct profile have in general a higher score than the other profiles, the FSV:s for all videos are shown in table 4.1

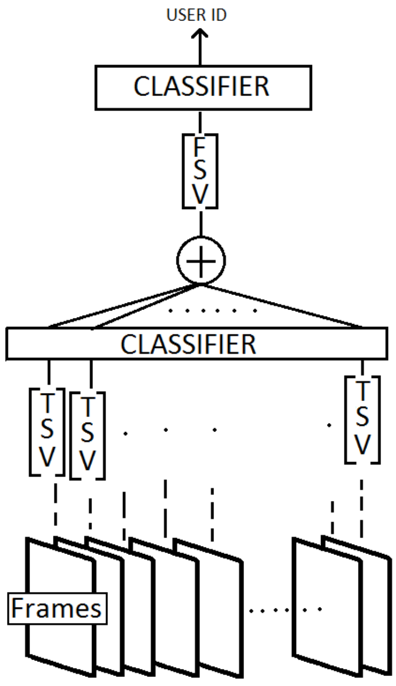


Figure 4.10: FSC is a classification method with a two-step classifier. In the first step, each frame is classified by its TSV to a profile ID, and in the second step, all the classifications are summed up into a FSV and the ID which got the highest score is finally classified as the user ID.

Accumulative score classification (ASC) which is similar to the FSC, but instead of classifying every frame, the TSV is accumulated over all frames. The final classification is then based on the maximum value found in the accumulated TSV. The accumulated TSV for the analyzed videos in the pilot database can be shown in figure 4.12

The template matching shows good results on the pilot database, but the issue is that the time consumption is proportional to the number of templates. It shows that, on an Intel Quad Core CPU @ 2.40 GHz, it takes on average 0.08s to cross-correlate one template (24x24) to an image (640x480). It does not sound like much, but if there are 10 profiles, each with 100 templates, the time consumed to template match would be 0.08*10*100 = 80s, which is not practical, especially since the number of templates is normally higher. There are two ways to try to reduce the time consumption, either to make the algorithm faster, which is a challange, or to reduce the amount of data processed. The following settings are tunable:
- Size of image. *Either to resize the image, or to crop the image to only include*

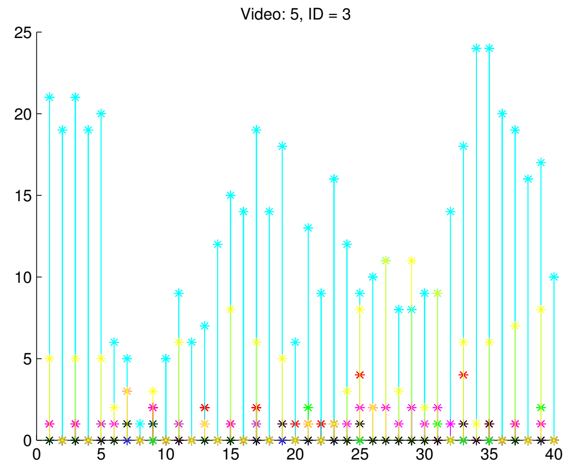


Figure 4.11: In the analysis of this video, only every 30th set of images were analyzed by the algorithm, and only for 40 frames. The video shows person ID 3 in an simulated car ride, and ID=3 was cyan in this image, it is easy to see that the highest TSV for almost all frames was the correct profile. The threshold was chosen to 0.945

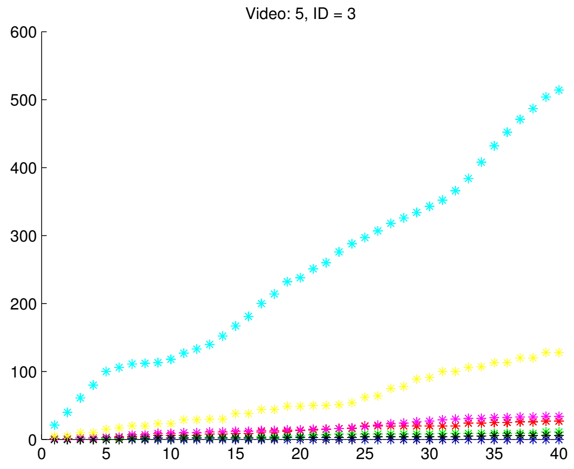


Figure 4.12: The figures show the same analyzed video as in figure 4.11, but shows the accumulated TSV, the color coding is the same as before, it can be seen that the correct profile accumulates a high template score.

the face, or both.
- Number of templates. *We can reduce the number of templates which are corralated*

- Which frames to analyze. *Instead of analyzing each and every frame, we can pick and choose some frames in order to reduce the data volume processed.*

Regarding the size of the image, it is of no interest to cross correlate against the part of the image that is not the face of the user, therefore we can use, for example, the Viola-Jones face detector to reduce the size of the image, see figure 4.15, in this example the amount of pixels were reduced by roughly 70%, and the time to do NCC went from 0.0707s -> 0.0118s, which is a reduction with 83%, it could be thought that the time consumption also should reduced with 70%, but the implemented NCC-function makes a simple calculation in order to choose between time-domain convolution and fourier-domain filtering to do the cross-correlation, therefore it can optimize the method depending on the size of the image, that is why the speed might improve to more than expected when cropped. The image can also be resized, which also is proportional to the pixels in the image, if the image is resized to 50% of the original size, 320x240, the amount of pixels is a quarter of the original, and the time is reduced to 0.018s which is 25% of the original time, just as expected.

- Number of templates. As mentioned, the time consumption is proportional to the number of templates, therefore it is of interest to try and keep it as low as possible. The problem is that there is no way of knowing what information is stored in the templates before it is looked at, what is known is which feature it represents and from which camera it was captured. Therefore there is no way to control the quality of the templates or from which pose they are taken. Optimally we would like to have a number of templates evenly scattered over a range of poses for each feature.
The problem of many templates, is that there is a risk that the Smarteye software has several templates that are very similar to eachother, and that the template matching would give a biased result, just because one profile has many templates that look one way and therefore has a high score.
The path that was chosen to deal with this is to do the same normalized cross-correlation as before, but NCC between two templates, all-to-all, and disregard the ones that have a cross-correlation value above a certain threshold, see equation 4.10, see figure 4.13. It is assumed that if two templates correlate high with eachother, they will therefore also correlate high in the image, giving a biased result.

$$\begin{Bmatrix} \forall T_i, i \neq j \text{ if } NCC(T_i,T_j) > T, & \text{then throw away } T_j \\ else & nothing \end{Bmatrix} \tag{4.10}$$

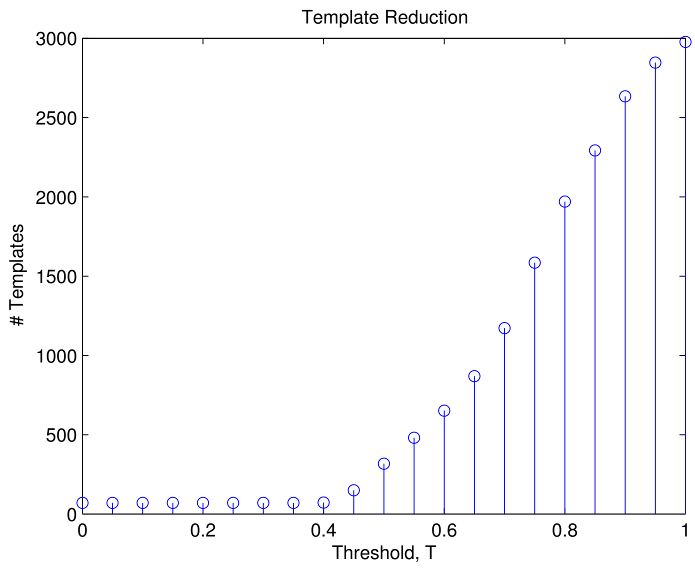


Figure 4.13: Here, the total amount of templates is seen how it varies with different template reduction thresholds.

Another issue that arises when the number of templates is reduced, is that good templates might be deleted so that there is no templates above the threshold, but sometimes the results might not change much, see figure 4.14.

Possibly the easiest way to reduce the amount of data that needs to be processed is simply by skipping frames completely that are not seen as important. Since the videos are captured in 60Hz, that is 60 frames per second, the consecutive frames will differ only slightly, making the analysis of them uninteresting. The best would be to build up a collection of frames from different poses, but since the pose is unknown in the video, only every x:th frame is processed. X used for template matching was chosen to 30, which would lead to 2 processed image sets per second. The amount of data processed is then reduced by 96.7%, but it is possible to even further reduce the frames by only analyzing the ones which have a face in them as detected by the Viola-Jones algorithm, but this would take away the advantage that the template matching has over the LBP-method.

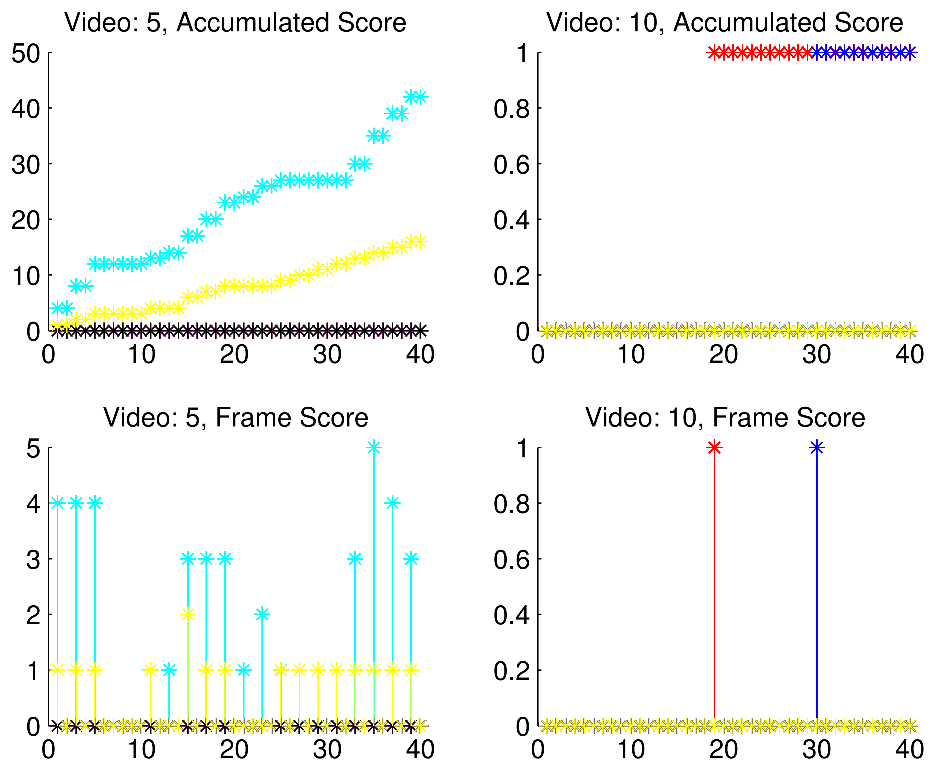


Figure 4.14: (Left) Looks like a scaled version of the unreduced classification, even though the template reduction threshold is 0.1, (Right) Video 10 did not manage to get a correct classification, only two templates managed to match above the threshold (0.945).

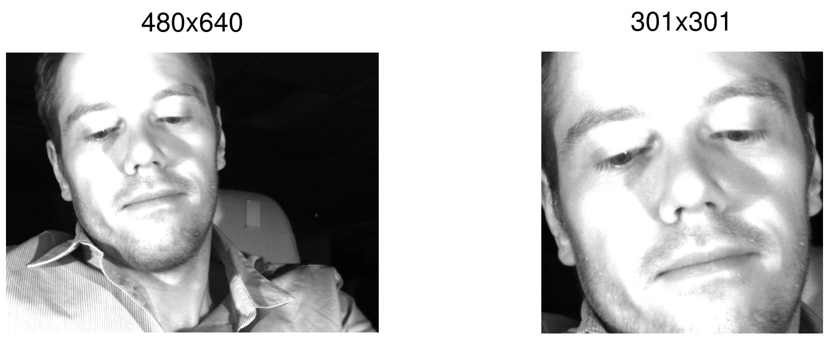


Figure 4.15: The picture to the left shows the original image of the size 480x640, and to the right, the face-image, as detected by the Adaboost algorithm.

Chapter 5: Results & Discussion

5.1 Face detection

Applying the face detector described in Section 4.2 to all the images in the profiles reduces the number of images from 1696 to 1614. The reduction in gallery size is not very large since the images stored in the profiles contains faces with multiple facial features visible. The images that were rejected are most likely of faces rotated too much from full frontal pose. Searching each image for a face takes approximately 0.029 seconds, which is decently fast but not enough to handle the system in real-time. With a frame rate of 60 Hz and 3 cameras, the time to process one image would need to be below $\frac{1}{60*3} = 0.0056$ seconds, more than 5 times faster. This is a clear trade-off, the additional accuracy provided by four cascades may not outweigh the loss in speed in other contexts since real-time was not a goal of this particular project, as explained in Section 2.2.

5.2 Feature extraction

All three facial features were found in only 1098 out of the 1614 face images, further reducing the size of the gallery. Even though detection of features is decently fast, the feature detector cannot handle partial occulusions such as glasses very well. The run time is around 0.1 seconds per face image, despite limiting the search areas for each feature. It could have been investigated if Haar-cascades specifically trained to find the left eye and right eye seperately would have increased accuracy.

A solution to the problem of outlier detections would have been to use a prior distribution model to limit the search areas for each feature [45] or to select the most probable configuration of features [46]. By analyzing a database of facial images with annotated feature points, the mean and variance of each feature point position can be calculated and thus provide an area where the feature is most likely to be found. This can be done if the face images are assumed to be in a full frontal pose and could have increase both speed and accuracy of the feature detectors.

While three and even two features (both eye centers) are enough to perform a global facial alignment as explained in Section 4.4, a detector that found more facial feature could have increased robustness in the alignment step, as discussed in Section 5.4.1.

5.3 Facial alignment

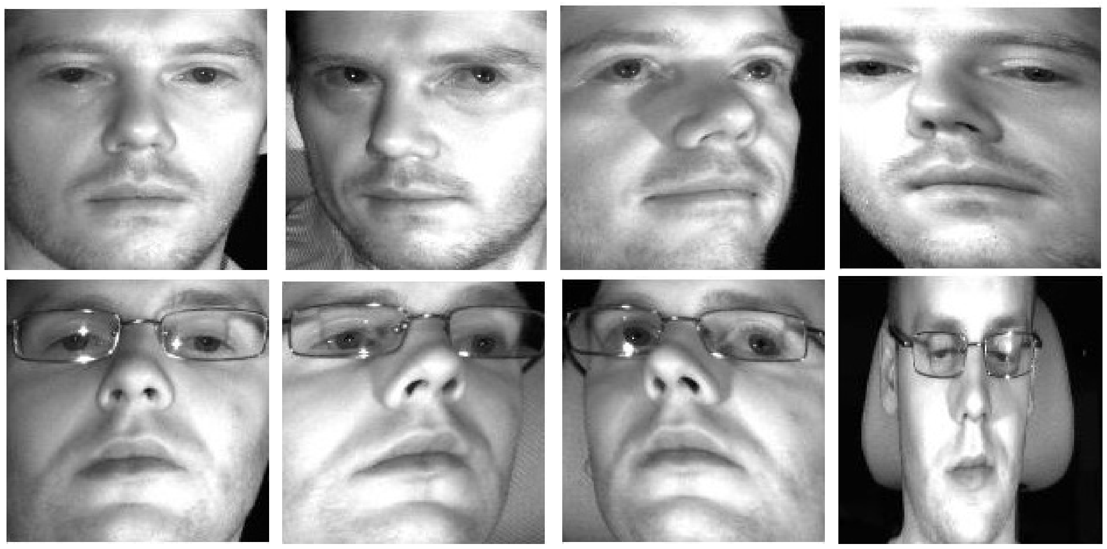


Figure 5.1: A set of aligned faces. The cropping is worse for faces rotated around the cameras y-axis, because the affine transformation assumes that the face is full frontal. The right-most column contains two cases where the feature detector has provided faulty feature points, yielding a badly aligned or cropped image.

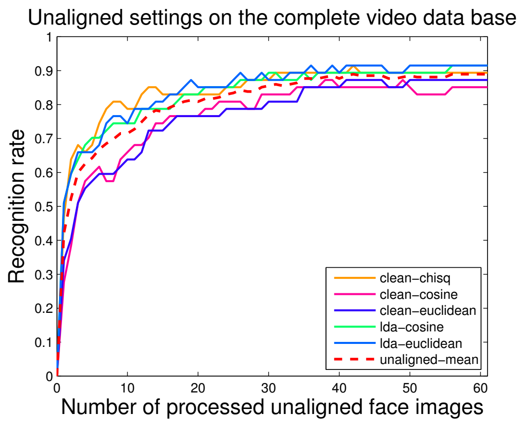
A set of eight aligned images is shown in Figure 5.1. For completely full frontal images or images rotated around the cameras z-axis, the alignment and cropped works very well, but performs worse for rotations around the y-axis. The face model is defined for full frontal, with the mouth being centered between the eyes. As the head is rotated around the y-axis, the eye positions move closer together from the cameras perspective and thus the cropping does not get a tight fit around the face anymore. The right-most column shows two cases where the feature detector has provided the alignment step with faulty feature positions. Such cases can lead to misalignment and incorrect normalization with respect to scale, which both yields bad cropping. Note that rotations around the x-axis will not affect the reuslts as much because the interocular distance will remain almost constant from the cameras perspective, only scaling in the vertical direction will vary.

This facial alignment method is very simple and can reduce transformations in some cases, but then a rotated face will differ from a face that was in an initially correct position because of skewing. Our method cannot reduce deformations of the face due to expressions either, the facial alignment step can unfortunately only be as good as the feature detection step allows it to be. If more feature points were found, an alignment method that more effectively reduces distortions due to head rotations and facial expression, such as piecewise affine warping [31][47] could be used.

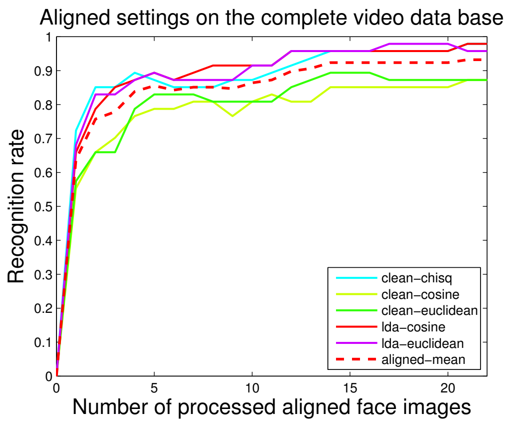
The current alignment method would also benefit from access to more than three feature points. The affine transformation could then have been calculated more robustly and not been affected as much by outlier detections, where either of the feature points are misplaced, which usually leads to severe deformations of the facial image. The number of face detections discarded due to missing eye or mouth detections could have been decreased, since additional feature points would have provided redundancy when calculating the affine transformation. They would also have provided more information about the boundaries of the face and thus ensured tighter cropping.

5.4 Face recognition

5.4.1 LBP-method



(a) Unaligned.

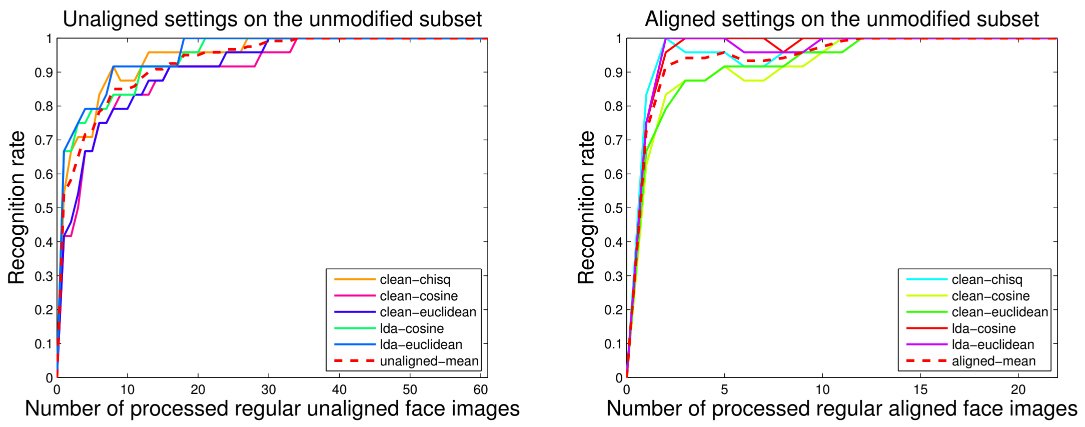


(b) Aligned.

Figure 5.2: The recognition rate of the LBP-methods on the complete database as a function of the number of processed face images for different distance metrics. Unaligned reaches a mean recognition rate of 89%, whereas aligned achieves 93%. Note the difference in processed face images between the methods.

Figure 5.2 shows the results of the aligned and the unaligned methods seperately for varying settings on the complete video database. PCA in combination with LDA was applied to the face descriptors for all settings except the ones marked ”clean”.

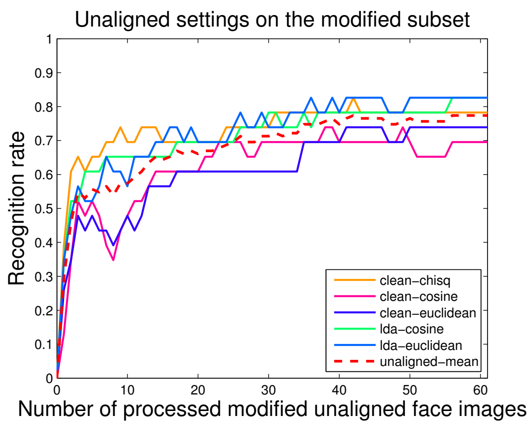
The aligned settings outperforms the unaligned ones, despite the fact that it has processed a fewer amount of face images. The unaligned settings scores a mean recognition rate of 89.0% after 61 processed face images, with the LDA-euclidean and LDA-cosine acquiring the highest recognition rate of 91.5%. The aligned settings has a mean recognition rate of 93.2% after 22 processed face images, with the LDA-cosine performing the best and acquiring a recognition rate of 97.9%. The alignment procedure thus increases overall performance, leading to better decisions despite using less data. The clean settings performs worse in both cases, indicating that dimensionality reduction and class clustering using the PCA+LDA method is beneficial to the performance. However, the chi-square similarity is almost equal in performance despite using the full feature vector, motivating the choice made by Ahonen et al. [17] to use the chi-square similarity for LBP face recognition. Our performance could have perhaps been improved further by using their weighing scheme, such that more discriminative tiles contributes more to the similarity score.



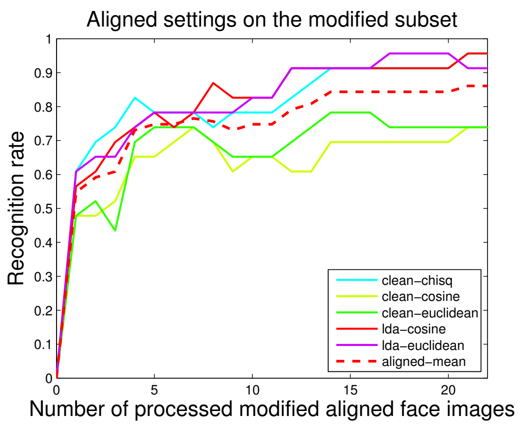
(a) Unaligned. **(b)** Aligned.

Figure 5.3: The recognition rate of the LBP-methods on the unmodified database as a function of the number of processed face images for different distance metrics. Both methods achieve 100% recognition rate. Note the difference in processed face images between the methods.

Figure 5.3 shows the result on the unmodified subset of the database. Both the unaligned and aligned settings acquire a mean recongition rate of 100%, with the unaligned method requiring 35 images and the aligned method requiring only 12 images. This shows that the algorithm eventually correctly classifies all unmodified videos for both methods. The LDA-cosine performs slightly better overall than LDA-euclidean and clean chi-square, only needing 9 face images per video for the aligned method to stabilize at 100% recognition rate.



(a) Unaligned.

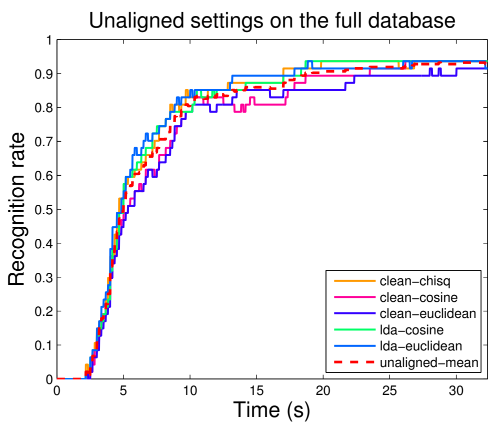


(b) Aligned.

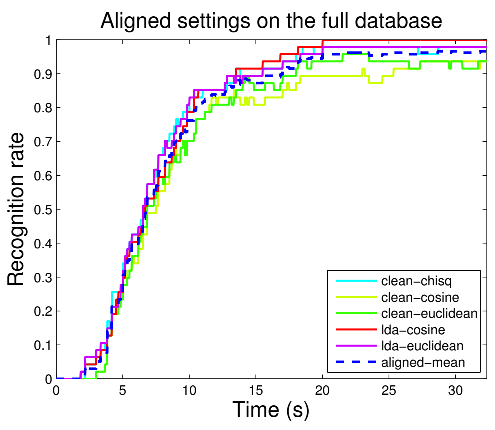
Figure 5.4: The recognition rate of the LBP-methods on the modified database as a function of the number of processed face images for different distance metrics. Unaligned reaches a mean recognition rate of 77%, whereas aligned achieves 86%. Note the difference in processed face images between the methods.

The superiority of the aligned method is even clearer in the modified subset of the database shown in Figure 5.4. The unaligned method only reaches a mean recognition rate of 77.0%, the LDA settings once again performing the best with a score of 82.5%. The aligned method achieves a 86.0% mean recognition rate, with LDA-cosine reaching 95.7% recognition rate, outperfoming euclidean distance slightly. We can conclude that recognition rate is overall lower for modified videos as intuition suggests, since that subset of the database contains facial occlusions and different appearances of the subjects.

The recognition rate can also be investigated as a function of time. This is interesting because of the fact that the aligned method requires better face images where the facial features have been found. This leads to a difference between the methods with respect to the time it takes for them to achieve their results. The results for the full database are shown in Figure 5.5. It is apparent once again that the aligned method achieves a higher mean recognition rate of 96% compared to 93% after 30 seconds. The unaligned method performs better in the beginning because it rejects less face images. Its performance also converges to a lower recognition rate because of the lower quality face images. Note that the subjects generally do



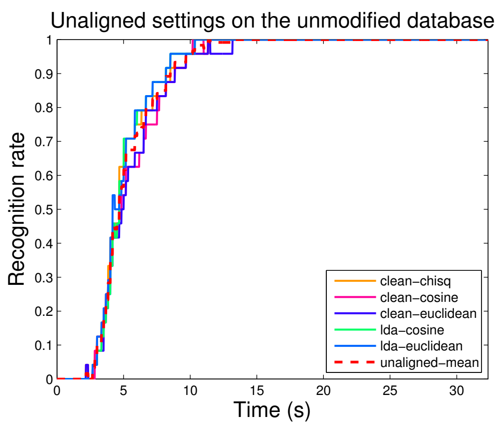
(a) Unaligned.



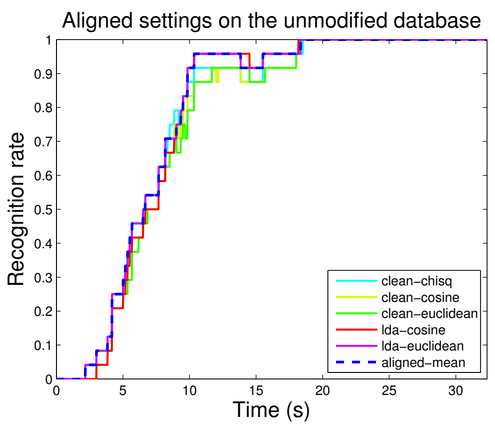
(b) Aligned.

Figure 5.5: The recognition rate of the LBP-methods on the full database as a function of time. Unaligned reaches a mean recognition rate of 93%, whereas aligned achieves 96% after 30 seconds has passed.

not sit down until after about 4 seconds, explaining the delay in the graphs. The LDA-cosine still performs the best, acquiring 100% recognition rate on the aligned method after 20 seconds.



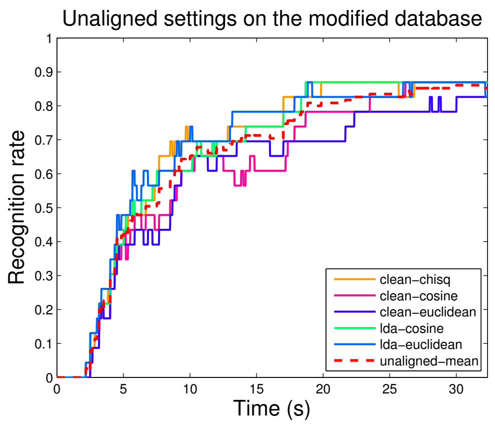
(a) Unaligned.



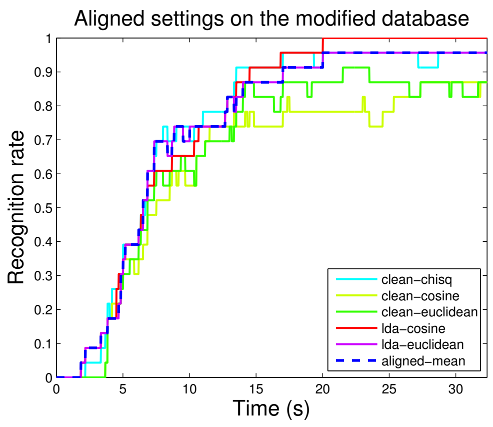
(b) Aligned.

Figure 5.6: The recognition rate of the LBP-methods on the unmodified subset as a function of time. Both methods achieve 100% recognition rate, the unaligned method converging faster.

The results of aligned and unaligned settings for the unmodified and modified subsets are shown in Figure 5.6 and Figure 5.7 respectively. These figures reveal that the unaligned method actually outperforms the aligned method on the unmodified subset with respect to time, reaching a mean recognition rate of 100% after 13 seconds compared to 18 seconds for the aligned method, because it does not throw away as many face detections as the aligned method. But once the subjects alter their appearance, the unaligned method underperforms and cannot reach the same mean recognition rate as the aligned method, achieving 86% compared to 95%. Only the LDA-cosine setting reaches 100% recognition rate on the modified subset.



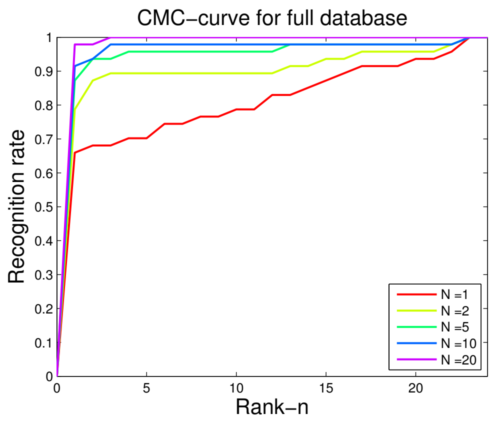
(a) Unaligned.



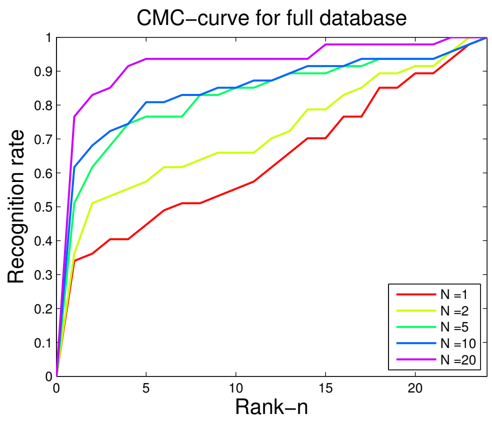
(b) Aligned.

Figure 5.7: The recognition rate of the LBP-methods on the modified subset as a function of time. Unaligned reaches a mean recognition rate of 86%, whereas aligned achieves 95% after 30 seconds has passed.

From these results we can conclude that the aligned method requires less images to acheive a higher recognition rate, outperforming the unaligned method especially on the modified subset, even though it takes slightly longer time to obtain the needed amount of face images to surpass the results. An alignment procedure that works also on rotated faces or faces with occuluded features together with improved feature detectors would cut the gap in time for the aligned method and hopefully also allow for faster increase in recognition rate. The LDA-cosine setting is the overall top-performer in almost all cases, with LDA-euclidean and clean chi-square slightly behind. This is reasonable because of the fact that the LDA method has a supervised learning step that is designed to increase separation between classes and decrease separation within classes. Since cosine often outperforms euclidean, it seems to be a better measure of similarity between vectors. The chi-square distance does however seems like a good alternative in cases where a supervised learning step is not desirable, such as online setups where the database will increase in size and require re-training of the LDA transformation every time the database is updated.



(a) Aligned LDA-cosine setting.

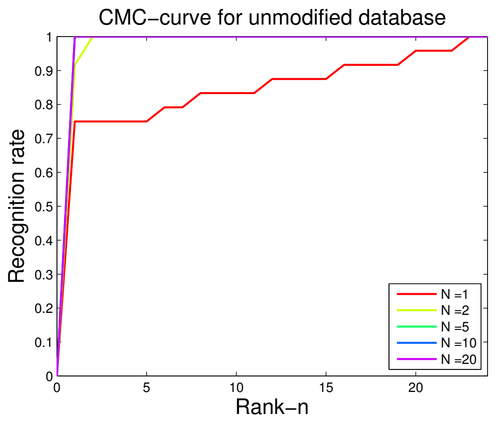


(b) Unaligned clean euclidean setting.

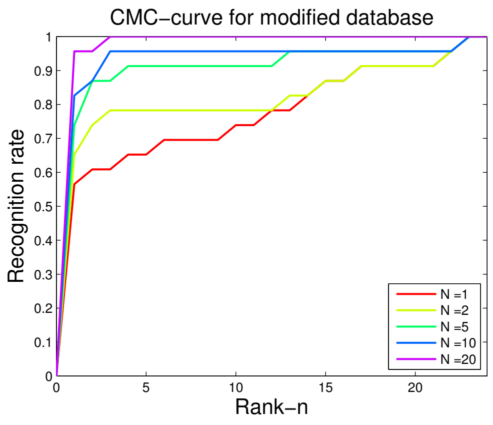
Figure 5.8: CMC-curves for two settings on the full database with a false positive rate of 1 for varying number of processed face images.

The cumulative match characteristic (CMC) was evaluated for the aligned LDA-cosine setting by plotting CMC-curves, with the recognition rate shown as a function of rank. A false positive rate of 1.0 was used to keep the problem a closed-set identification with the distance threshold set to $\tau = \infty$. The full database was evaluated for varying $N =$ of frames until identification. The result is shown together with the base line unaligned clean euclidean setting in Figure 5.8, chosen for simplicity where a face has been LBP-encoded straight from the face detector.

The rank-1 recognition rate is increased drastically for the aligned LDA-cosine setting compared to the base line for all values of the number of processed face images. The case where one image has been processed can be compared to turning the video database into an image database since only the first face image is processed. This curve in itself is interesting, since high performance on the first image would make processing of more images redudant. It is clear that the curves are lifted with increasing number of processed images for both settings, confirming the intuitive notion that processing more face images increases the probability of the correct match being found, or that it is at least ranked as a top candidate. The



(a) Unmodified subset.



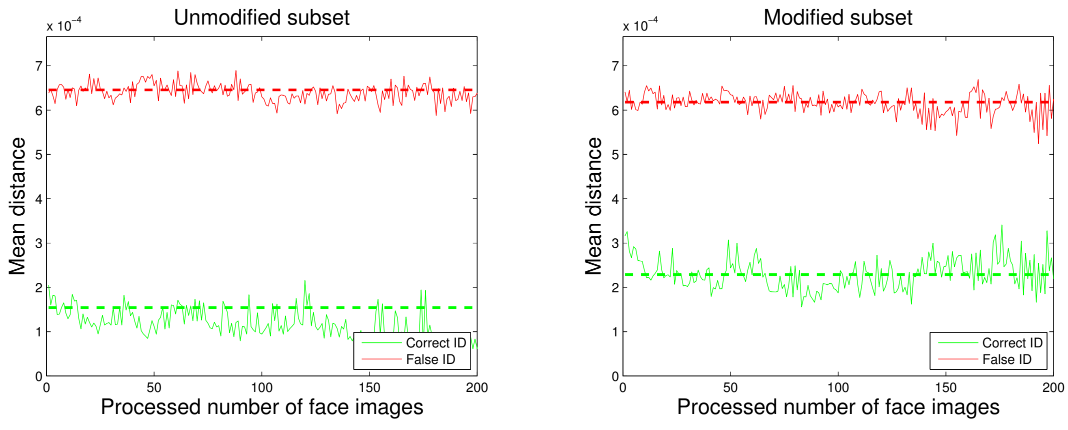
(b) Modified subset.

Figure 5.9: CMC-curves for the LDA-cosine setting over each subset of the database with a false positive rate of 1 for varying number of processed face images.

CMC-curves for the unmodified and modified subset for the LDA-cosine method is displayed in Figure 5.9. The performance is decreased for all values of the number of processed face images for the modified subset, whereas only a small number of images is required to achieve good performance on the unmodified subset.

The mean distance between the correct and false matches for both the modified and the unmodified subset of the database was also investigated to compare modified and unmodified videos. Figure 5.10 shows that there is a clear separation in mean distance, which is slightly increased for the modified subset, $\bar{d}_{mean} = 2.2 * 10^{-4}$ compared to $\bar{d}_{mean} = 1.6 * 10^{-4}$. A distance threshold should therefore be set above the modified correct mean distance, such as $\tau = 3 * 10^{-4}$, to remove weak classifications and imposters.

ROC-curves can be plotted for the open-set identification by varying τ and removing randomly selected profiles from the database so that half of the videos are part of the gallery and the rest are of unknown persons not present in the gallery. The result strongly depends on which profiles that are removed due to the small database size, but only two runs were performed for each setting due to time constraints and an average result of those runs was calculated. Optimally a high number of runs should be used to calculate the average to get a better estimation of the actual performance on the data set. The results are shown in Figure 5.11 with different values of N for the aligned LDA-cosine and aligned clean chi-square

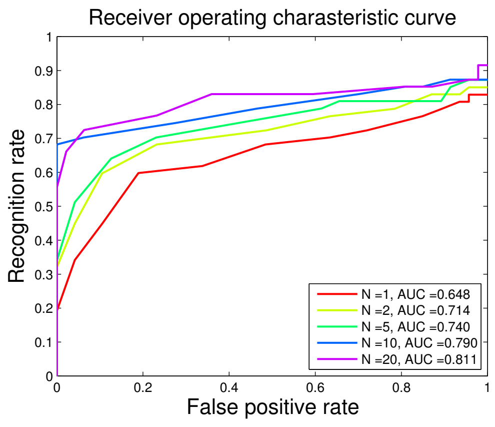


(a) Unmodified subset. **(b)** Modified subset.

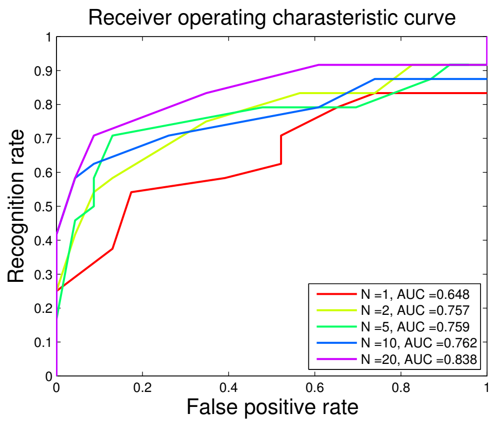
Figure 5.10: The mean distance between the current image face descriptor and the gallery for the aligned cosine setting, divided into the correct match and false match, for both the unmodified and the modified subset of the video database as a function of the number of images processed. The dashed lines represent the mean values.

methods. These were selected to allow for a comparison between the best supervised (LDA-cosine) and best unsupervised (clean chi-square) settings. The AUC is also shown in the figures. Note that the range of the threshold τ must be changed manually for each method since the range of distances varies.

The figure shows that the clean chi-square setting outperforms the LDA-cosine setting: The recognition rate for N = 20 converges faster and reaches a higher value. The AUC is also higher with 83.8% compared to 81.1% (where 1 is the maximum value). The performance of the LDA-cosine setting is not as good for open-set identification as for closed-set identification, a result of the fact that it only maximizes separation with respect to the subjects in the gallery. Once subjects are removed from the gallery, the LDA-cosine can no longer retain its superiority over the unsupervised chi-square setting, since it may have removed dimensions necessary to efficiently discriminate between subjects that are part of the gallery and imposters that are not. This shows the subtlety of face recognition and that algorithms also must be chosen with respect to the desired type of identification task (face verification, closed-set identification or open-set identification). It would be interesting to investigate how the LDA settings are affected by the size of the database as well, if more training data benefits the method in the open-set identification by allowing the LDA-algorithm to learn a transformation that better separates the



(a) Aligned LDA-cosine setting.



(b) Aligned clean chi-square setting.

Figure 5.11: ROC-curves for two settings on the database for varying a number of processed face images. The graphs show the average of two runs where the database for each run was randomly drawn so that half of the profiles were part of the gallery.

overall population in face space instead of just a small subset.

One must remember that these results are dependent on the size of the database and that performance is also tighly connected to the content of the database. A larger variance in performance between different runs is to be expected for small data sets because of the fact that they do not cover as much of the possible variation as large data sets. This means that one cannot create a database of the same size and with the same settings but with different people and expect similar results. Our database was rather homogenous with respect to age and ethnicity, mostly consisting of middle-aged northern europeans. However, the task of differentiating people from eachother is harder in a homogenous data set than in a varied data set, so this shows that the method can work well despite similar appearances among subjects. This does not guarantee performance in other homogenous data sets though, since we cannot predict if discrimination amongst subjects is harder for the algorithm in other subsets of the total population.

The effects of adding the left-right flipped images to the database were not evaluated. Even though it increases the amount of images in the database to compare with, it may not be suitable and should be taken into consideration since it affects the decision boundaries in the classification step, especially for the LDA method. Adding flipped images to each class may lead to a transformed LDA-space suboptimal for comparing ordinary non-flipped images, since the algorithm clusters flipped and non-flipped images of the same class together and may therefore impede its ability to generalize on ordinary images where the subject has modified its appearance. Optimally, the alignment procedure should produce a facial image completely devoid of facial asymmetry, so that adding left-right flipping becomes redudant.

Due to the fact that the LDA method has whitened eigenvectors as output, it transforms the data not only according to the class seperation of LDA but also according to the variance of the transformed space, such that distances are calculated in Mahalanobis space [35], or a dimensionless feature space. This may give an unfair advantage to the runs that uses the LDA method compared to the clean runs, since any metric comparison is done in a transformed space designed to separate the data with respect to its variance. This is generally more discriminative and Mahalanobis cosine have been shown to perform better in PCA and LDA methods than metrics in their original data space [48] [35]. To calculate the Mahalanobis distance in the ordinary data space instead of in a linear subspace (PCA or LDA), one requires the inverse of the co-variance matrix of the feature database to be positive definite in order for the built-in MATLAB distance functions to work. This was not fulfilled and thus the attempt to calculate the Mahalanobis distance was abandoned.

Instead of using only PCA+LDA to find an optimal subspace, the unified framework for subspace analysis proposed by Wang et al. [36] that also utilizes Bayesian analysis besides PCA+LDA to reduce the intrapersonal variance of faces due to transformations such as facial expressions and illuminations could have been investigated. Their results suggest that all the three subspace methods complement eachother and thus should be used together.

The choice of using the ordinary NN-classifier could also be questioned. The effects of classification with more than the 1-NN would have been interesting to investigate, especially how it would affect the clean settings, because of the fact that the high dimensionality leads to very sparse data points and badly defined decision boundaries. Multi-class support vector machines and neural networks are options for the classification of face descriptors, but they were dismissed for the simplicity of the NN-classifier, both in conceptualization and implementation.

It would be interesting to investigate a classification method that takes advantage of the previously processed face images when classifying new images, instead of treating each image as an indepedent decision. This would most likely both increase the identification accuracy and decrease the amount of information needed to make a correct final decision on the identity of the user. An easy approach would be to calculate the class centers of all individuals in the gallery and then assigning the probe class center the identity of its nearest neighbour class center. A more refined method would be to use the manifold-manifold distance [49], designed to match sets of images with eachother. This better corresponds to the actual task posed when using the tracking profiles to identify subjects in video sequences, since both the profiles and videos contain multiple images of the same face. The complications with this approach would be that if the implmentation was done for video streams instead of video sequences, the probe set of face images collected from the stream would continuosly increase thus changing the manifold of the probe set with each processed face image. This could be circumvented by investigating the recognition rate as a function of the number of processed images used for the calculation of the probe manifold and choose an appropriate image or time threshold that meets a set average recognition rate.

No intensity normalization scheme was evaluated because of the fact that the camera system itself reduces the variations in illumination. The LBP-operator is also designed in such a way that is it invariant to monotonic gray-scale changes [17], a property of histogram equalization which is normally used for intensity normalization. It would have been interesting to investigate if normalization would have improved or impaired performance, since it is a necessary step on ordinary grayscale images. However, Li et al. [16] also used near IR-cameras and flashes to reduce illumination issues and did not use any additional normalization either and therefore it may not be suitable for IR setups.

Even though the LBP-method has been evaluated for full frontal images, the method itself could be used for faces in any pose, given that the faces retrieved and stored from the profiles are in that pose. Profile photos for example may not be as discriminative between individuals as full frontal photos, leading to lower recognition rates in other poses, but the theory is not exclusive to full frontal. The alignment procedure would have to use corresponding face models when aligning and other features instead of mouth and eyes may have to be extracted, but nothing limits the method it from working in other poses. Optimally, many different ”identificators” should be created, working in varying pose ranges and thus the algorithm could handle pose variations much better and not have to throw away as much video data as when only using full frontal images.

5.4.2 Template matching

Since the template matching is quite a time consuming task as of now, there had to be done some restrictions in the number of simulations, and in trying to reduce

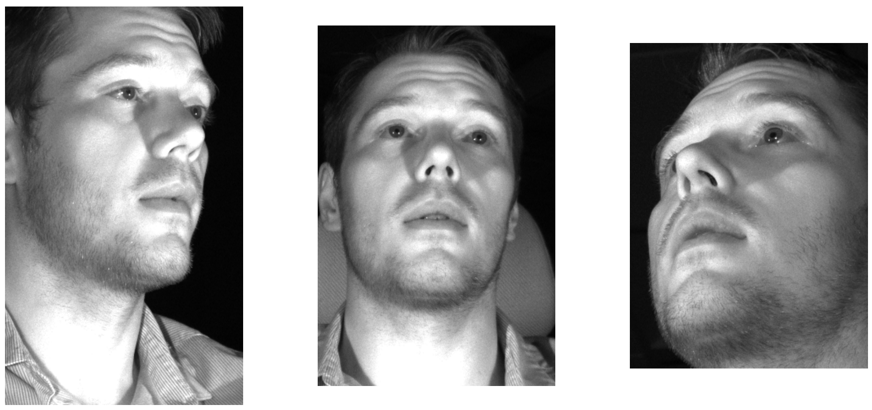


Figure 5.12: Face detection using the ABD, it can be seen that the whole face is captured, however because of the reflected light from the throat, also a big portion of the neck is included.

the amount of data to process in order to make it runable. As will be shown, the database was analyzed using the viola-jones face detector, because it reduced the amount of images that had to be processed, but also a pararell simuation was done using the adaptive blob detector (ABD), see fig 5.12 and 5.13, for a number of videos to show that the results may be extrapolated.

As mentioned in chapter 4.5.2, each simulation can be done with different parameters, such as how many frames shall be processed and template reduction. Since the database contains more than 13000 templates for 24 profiles, the time it will take to match is unpractically long, and since there were no computers available to be dedicated to the task it had to be done on a personal computer, which was not available at all times either.

Information about the first run can be seen in figure 5.14. As can be seen the first

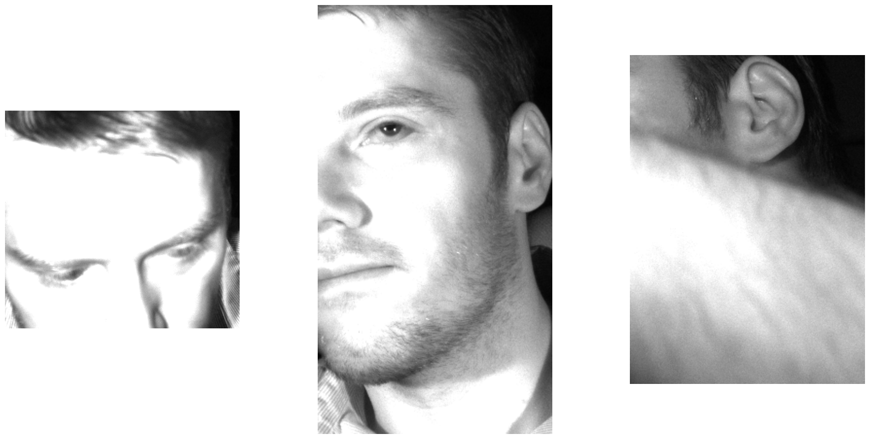


Figure 5.13: Because the ABD does not consider anything other than "blob-structures" in the image, it cannot distinguish between face or non-face images, therefore some faces might be cropped (left and middle) or occluded (right).

run reduced the templates by 20%, which was mainly done because the method is so time consuming, and a template reduction parameter of 0.8 seemed to be reasonable to still remove an amount of templates while retaining a sound set of templates, however there is no known way to actually investigate this claim.

In the classification stage we first look at the FSC, and one of the difficulties is to decide which threshold is good to use, therefore a bypass has been done by sweeping over a range of plausable thresholds and evaluate the method based on maximum performance.
As can be expected the results in figure 5.15, which are summarized in figure 5.16, show that the method FSC has better performance for non-modified videos. We can read out in the figure 5.16 that the best reached performance for all videos is 83%, which is good results compared to a random classifier, however, the large number of templates makes the whole process generally unpractical, as can be seen

Run nr:	**1**		
Templates:	13788	Process every x frames:	30
Template Reduction	0,8	Face detector?	Viola-Jones
Templates after reductio	10919	Resize Image?	No
% Retained	79,19 %		
		Frames Processed:	9822
Videos Processed:	47	Frames Analyzed:	4538
Run Time:	7d 13h 3m 12s	Ratio	46,20 %

Figure 5.14: Table shows the settings which were used for the first template matching run and the metainformation.

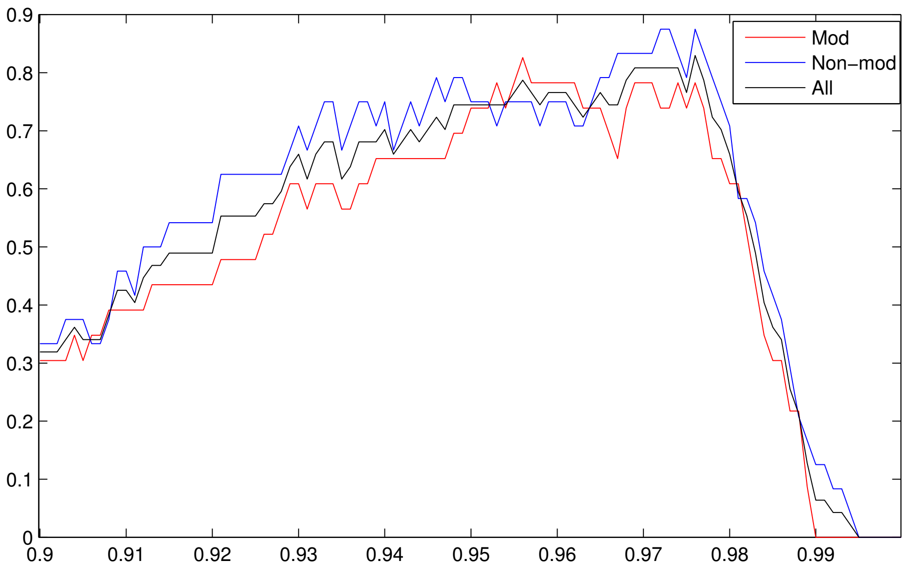
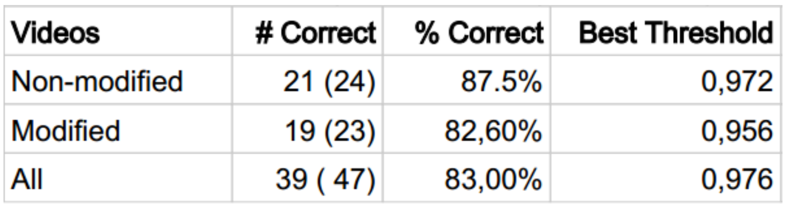


Figure 5.15: Here the number of correct classifications is shown as a function of threshold for Modified, non-modified and all videos for closed set identification.

in the table in figure 5.14.

We could also look at how the recognition rate (FSC) increases over frames analyzed, then we could see how fast a good recognition can be extracted. The number of analyzed frames each video has in the first run, see figure 5.17, is totally dependant on the pose of the user because the Viola-Jones face detector can only detect full frontal faces. We would like to now plot the recognition rate based on



Videos	# Correct	% Correct	Best Threshold
Non-modified	21 (24)	87.5%	0,972
Modified	19 (23)	82,60%	0,956
All	39 (47)	83,00%	0,976

Figure 5.16: Here the best performance from the first run are shown for the FSC.

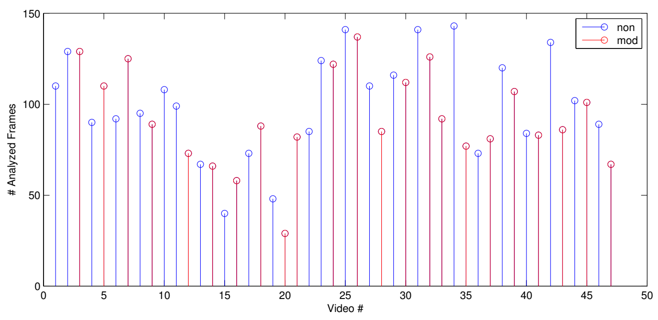


Figure 5.17: The number of analyzed frames in each video. For modified videos, the minimum found was 29, and maximum was 137, while for non-modified the minimum was 40 and maximum 143.

the frames, the thresholds will be as detected in the analysis above, see fig 5.16. In order to include all videos, we will restrict the modified videos to 29 firstly detected face frames and the 40 first for non-modified videos.

As can be seen in figure 5.18, even after a quite small number of frames we could get an okay recognition rate. We can also see that the identification did not reach the maximum as seen in figure 5.16, so we know that the more frames (data) we have the better classification can be achieved. It is interesting to see that the modified videos achieved higher recognition rate during fewer frames than the unmodified set, although this is entirely dependant on the quality of those frames.

Another run was done using the ABD, in order to see how the identification was with all frames included, since the ABD always give an output in the image whether or not an user is actually present, see figure 5.19. As we can see the run time for the second run is approximately the same, but with half of the number of analyzed frames, this is a result of the ABD, as you can see in figure 5.12, the output box is usually much bigger that it needs to be, which makes it more time consuming, and only 11 videos were analyzed, this also restricted by the time available. What we can do is to compare the analysis of the first 11 videos in both runs, and see

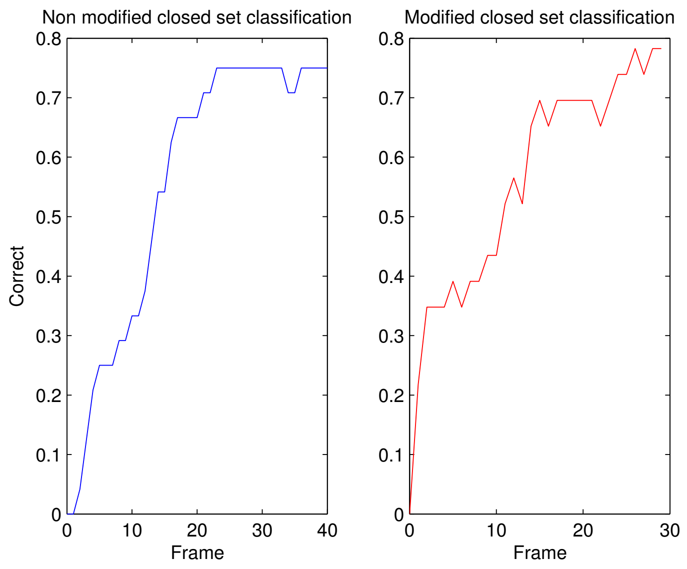


Figure 5.18: recognition rate of 75% was reached for the non modified and 78% for the modified set.

Run nr:	2			
Templates:	13788		Process every x	30
Template Reduction	0,8		Face detector?	ABD
Templates after reducti	10919		Resize Image?	No
% Retained	79,19 %			
			Frames Proces	2235
Videos Processed:	11		Frames Analyze	2235
Run Time:	7d 6h 57m 7s		Ratio	100 %

Figure 5.19: The table shows data and settings for the second run of the template matching algorithm.

how they compare, if they are similar enough, we can extrapolate the results to all videos, arguing that logically the rest of the videos will also show similar results. In figure 5.20, the recognition rates are shown for both run 1 and run 2 for the first 11 videos only. We can see that for modified videos the method managed to get 100% recognition rate for some threshold, but all videos the maximum recognition rate was the same 81%, for for slightly different thresholds. But we can assume that if we were to analyze all videos we would get somewhat similar peak performance as in run 1.

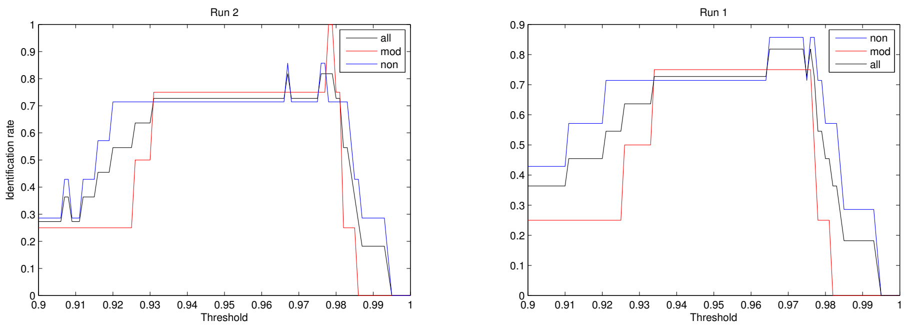


Figure 5.20: (Left) recognition rate for only the first 11 videos with run 2. (Right) Same analysis as previously but on only the first 11 videos.

5.4.3 Alternative identification methods

The profiles as mentioned earlier also contains a face model of each person with the coordinates of around a dozen facial feature points defined in three dimensions. This face model could be used to classify subjects, under the assumption that individuals have unique facial structures and that the differences in distances and angles between features of different persons are large enough for discriminative tasks and is not completely overwhelmed by noise.

This could be done in two ways: Either by projecting 2D feature points into 3D space such that a comparison can be made between the stored models and the model of the unidentified subject, or by projecting the stored 3D models to 2D and then comparing the models. The first method required a minimum of 2 cameras with different views of the subject in order to be able to create a 3D model, whereas the second method could work with only one camera.

The bottle-neck of both methods is that they require accurate facial feature extraction together with proper pose alignment in order for the differences to not consist mostly of noise. Two feature extraction methods were evaluated, a method that used SVM's trained on HoG-descriptors to find facial features and the template matching method, as explained in Section 4.5.2. Neither of these feature extraction methods were precise enough in their detection of facial landmarks to proceed with the projective geometry and thus no proper conclusions could be made regarding the use of the Smart Eye 3D models for identification purposes.

Chapter 6: Conclusions

In this thesis we have evaluated two methods for face recognition in videos captured in the near-IR spectrum. The subject identification was done by utilizing data available in previously stored Smart Eye tracking profiles. The evaluation of the methods was performed on our own database of videos and tracking profiles, consisting of 24 people and 47 videos. The data available in the tracking profiles proved to be sufficient enough to reach good recognition rates for both methods.

The method based on the local binary pattern operator achieved good results on both the modified and unmodified subset of the database. The setting aligned LDA-cosine got the best result in closed-set identification (100% and 95% recognition rate respectively) and the aligned chi-square similarity performed the best in open-set identification (49% recognition rate at 2% false positive rate on the complete data set). This indicates that a supervised learning step increases the performance significantly in closed-set identification, where all subjects were part of the gallery, whereas in open-set identification the supervised LDA-method could not generalize as well and was outperformed by the unsupervised chi-square setting. The decrease in performance may be due to the removal of dimensions necessary for discrimination between imposters and actual subjects in the database. Overall, the LBP-method is robust to both geometric transformations (scaling and rotations) and to facial occulusions due to its ability to capture image information on both a local and global level. The results showed that the facial alignment and the dimensionality reduction procedure both were beneficial to the identification process, significantly increasing the performance of the LBP-method. Despite our rather simplistic implementation of the LBP-method, we have shown that it performs well and that it could succesfully be used for driver identification.

The results for the template matching algorithm were acceptable, 81% recognition rate for all videos, although they showed that the performance can be good in certain circumstances (100% for modified videos in run 2). The problem is to decide a threshold for the template scores and to keep the run time down. The optimal threshold varies quite a bit from run to run, but it was high in most cases (0.97-0.98) indicating that most of the templates were not used. The biggest flaw of the method is the run-time, and this is related to the large number of templates, a factor which cannot be controlled in a simple manner. Some ways to reduce the run-time can be to use a more specific face detector, such as the Viola-Jones detector, which reduces the analyzed number of images. The Adaptive Blob Detector (ABD) further increased the run-time, since it includes a larger output image than the more precise Viola-Jones detector.

Chapter 7: Future work

Refinements and extensions of the LBP-method would include using multiple identificators that works for face detections in different angles, so that not only full frontal face images are used for identification. Detection of more than three facial features would be beneficial as well, so that the alignment procedure could be enhanced by not only finding a transformation for the complete face image but for local parts of the face as suggested by [31], that would to an extent prevent the deformations of the face that are caused by a global face transformation. The LBP-operator could be improved or exchanged for better face encoding, either by using the simple multi-radius approach suggested in [13], or any of the extensions described in [32]. Performance could also possible be increased by treating the task as a set-to-set comparison instead of an accumulated example-to-set comparison with the manifold-manifold distance [49], which would be more accurate problem description with respect to the actual data available during each identification step. These are all ideas to pursue in hopes of achieving higher recognition rates.

For the template matching method, a way to control the quality of templates is required. One option is to compare all the templates to eachother and assume that most of the templates are in fact of good quality, and therefore we could keep the templates which are good by retaining the ones that are similar and throwing away the outliers, and by that means reduce the number greatly. Other ways is to improve the correlation method, by searching for the maximum correlation in the image instead of doing an exhaustive search. This could be done by correlating at different scales and then choosing to further analyze an area around regions of interest. However if the database is large enough, this method is likely not desireable at this moment, since the number of templates will be big anyway.

Bibliography

[1] W. Zhao, R. Chellappa, A guided tour of face processing, in: W. Zhao, R. Chellappa (Eds.), *Face Processing : Advanced Modeling and Methods*, Elsevier / Academic Press, Amsterdam, Boston, 2006.

[2] A. F. Abate, M. Nappi, D. Riccio, G. Sabatino, *2D and 3D face recognition: A survey*, Pattern Recognition Letters (2007) 1885–1906.

[3] S. Z. Li, A. K. Jain, Introduction, in: S. Z. Li, A. K. Jain (Eds.), *Handbook of Face Recognition, 2nd Edition.*, Springer-Verlag London Limited, London, 2011.

[4] M. Turk, Eigenfaces and beyond, in: W. Zhao, R. Chellappa (Eds.), *Face Processing : Advanced Modeling and Methods*, Elsevier / Academic Press, Amsterdam, Boston, 2006.

[5] P. J. Phillips, H. Moon, S. A. Rizvi, P. J. Rauss, *The FERET Evaluation Methodology for Face-Recognition Algorithms*, IEEE Trans. Pattern Anal. Mach. Intell. 22 (10) (2000) 1090–1104.
URL `http://dx.doi.org/10.1109/34.879790`

[6] P. N. Belhumeur, J. a. P. Hespanha, D. J. Kriegman, *Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection*, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 711–720.
URL `http://dx.doi.org/10.1109/34.598228`

[7] O. Jesorsky, K. J. Kirchberg, R. W. Frischholz, *Robust Face Detection Using the Hausdorff Distance*, Springer, 2001, pp. 90–95.

[8] G. B. Huang, M. Ramesh, T. Berg, E. Learned-Miller, *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*, Tech. Rep. 07-49, University of Massachusetts, Amherst (October 2007).

[9] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, W. Worek, *Preliminary Face Recognition Grand Challenge Results*, in: Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition, FGR '06, IEEE Computer Society, Washington, DC, USA, 2006, pp. 15–24.
URL `http://dx.doi.org/10.1109/FGR.2006.87`

[10] L. Wolf, T. Hassner, I. Maoz, *Face recognition in unconstrained videos with matched background similarity*, in: in Proc. IEEE Conf. Comput. Vision Pattern Recognition, 2011.

[11] M.-H. Yang, N. Ahuja, *Face detection and gesture recognition for human-computer interaction*, The Kluwer international series in video computing, Kluwer Academic pub. cop., Boston (Mass.), Dordrecht, London, 2001.

[12] Z. Lei, M. Pietikainen, S. Z. Li, *Learning Discriminant Face Descriptor*, IEEE Transactions on Pattern Analysis and Machine Intelligence 36 (2) (2014) 289–302.

[13] S. Yan, H. Wang, X. Tang, T. S. Huang, *Exploring Feature Descritors for Face Recognition*, in: ICASSP (1), IEEE, 2007, pp. 629–632.
URL `http://dblp.uni-trier.de/db/conf/icassp/icassp2007-1.html#YanWTH07`

[14] D. G. Lowe, *Distinctive Image Features from Scale-Invariant Keypoints*, Int. J. Comput. Vision 60 (2) (2004) 91–110.
URL `http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94`

[15] Smart Eye AB Official Home Page, `www.smarteye.se`, Accessed: 2014-10-20.

[16] S. Z. Li, R. Chu, M. Ao, L. Zhang, R. He, *Highly Accurate and Fast Face Recognition Using Near Infrared Images*, in: D. Zhang, A. K. Jain (Eds.), ICB, Vol. 3832 of Lecture Notes in Computer Science, Springer, 2006, pp. 151–158.
URL `http://dblp.uni-trier.de/db/conf/icb/icb2006.html#LiCAZH06`

[17] T. Ahonen, A. Hadid, M. Pietikainen, *Face Description with Local Binary Patterns: Application to Face Recognition*, IEEE Trans. Pattern Anal. Mach. Intell. 28 (12) (2006) 2037–2041.
URL `http://dx.doi.org/10.1109/TPAMI.2006.244`

[18] P. Viola, M. Jones, *Robust Real-time Object Detection*, in: International Journal of Computer Vision, 2001.

[19] S. Z. Li, J. Wu, Face detection, in: S. Z. Li, A. K. Jain (Eds.), *Handbook of Face Recognition, 2nd Edition.*, Springer-Verlag London Limited, London, 2011.

[20] V. Jain, E. Learned-Miller, *FDDB: A Benchmark for Face Detection in Unconstrained Settings*, Tech. Rep. UM-CS-2010-009, University of Massachusetts, Amherst (2010).

[21] X. Zhu, D. Ramanan, *Face detection, pose estimation, and landmark localization in the wild*, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012, 2012, pp. 2879–2886.
URL `http://dx.doi.org/10.1109/CVPR.2012.6248014`

[22] M. Sonka, V. Hlavic, R. Boyle, *Image Processing, Analysis, and Machine Vision, International Student Edition*, 3rd Edition, Cengage Learning, Stamford, CT., 2008.

[23] T.Mahalakshmi, R.Muthaiah, P.Swaminathan, Review article: An overview of template matching technique in image processing, Research Journal of Applied Sciences, Engineering and Technology 4 (2012) 5469–5473.
URL `http://maxwellsci.com/print/rjaset/v4-5469-5473.pdf`

[24] D.-M. Tsai, C.-T. Lin, Fast normalized cross correlation for defect detection, Pattern Recognition Letters 24 (15) (2003) 2625 – 2631.
URL `http://www.sciencedirect.com/science/article/pii/S0167865503001065`

[25] N. Dalal, B. Triggs, *Histograms of Oriented Gradients for Human Detection*, in: C. Schmid, S. Soatto, C. Tomasi (Eds.), International Conference on Computer Vision & Pattern Recognition, Vol. 2, INRIA Rhône-Alpes, ZIRST-655, av. de l'Europe, Montbonnot-38334, 2005, pp. 886–893.
URL `http://lear.inrialpes.fr/pubs/2005/DT05`

[26] T. Ojala, M. Pietikäinen, T. Mäenpää, *Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns*, IEEE Trans. Pattern Anal. Mach. Intell. 24 (7) (2002) 971–987.
URL `http://dx.doi.org/10.1109/TPAMI.2002.1017623`

[27] C. Liu, H. Wechsler, *Gabor Feature Based Classification Using the Enhanced Fisher Linear Discriminant Model for Face Recognition*, IEEE Trans. Image Processing 11 (2002) 467–476.

[28] N.-S. Vu, A. Caplier, *Enhanced Patterns of Oriented Edge Magnitudes for Face Recognition and Image Matching*, Trans. Img. Proc. 21 (3) (2012) 1352–1365.
URL `http://dx.doi.org/10.1109/TIP.2011.2166974`

[29] X. Tan, B. Triggs, *Fusing Gabor and LBP Feature Sets for Kernel-based Face Recognition*, in: Proceedings of the 3rd International Conference on Analysis and Modeling of Faces and Gestures, AMFG'07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 235–249.
URL `http://dl.acm.org/citation.cfm?id=1775256.1775278`

[30] Z. Lei, M. Pietikainen, S. Z. Li, *Learning Discriminant Face Descriptor*, IEEE Transactions on Pattern Analysis and Machine Intelligence 36 (2) (2014) 289–302.

[31] *DeepFace: Closing the Gap to Human-Level Performance in Face Verification*, in: Conference on Computer Vision and Pattern Recognition (CVPR).

[32] L. Nanni, A. Lumini, S. Brahnam, *Survey on LBP Based Texture Descriptors for Image Classification*, Expert Syst. Appl. 39 (3) (2012) 3634–3641.
URL `http://dx.doi.org/10.1016/j.eswa.2011.09.054`

[33] S. Haykin, Neural Networks and Learning Machines, 3rd Edition, Person Education, Inc., Upper Saddle River, NJ, USA, 2009.

[34] Stanford's Tutorial for PCA and Whitening, `http://ufldl.stanford.edu/tutorial/unsupervised/PCAWhitening/`, Accessed: 2014-10-21.

[35] J. R. Beveridge, D. S. Bolme, B. A. Draper, M. Teixeira, *The CSU Face Identification Evaluation System*, Mach. Vis. Appl. 16 (2) (2005) 128–138.
URL `http://dblp.uni-trier.de/db/journals/mva/mva16.html#BeveridgeBDT05`

[36] X. Wang, X. Tang, *A Unified Framework for Subspace Face Recognition*, IEEE Trans. Pattern Anal. Mach. Intell. 26 (9) (2004) 1222–1228.
URL `http://dx.doi.org/10.1109/TPAMI.2004.57`

[37] T. Cover, P. Hart, *Nearest Neighbor Pattern Classification*, IEEE Trans. Inf. Theor. 13 (1) (2006) 21–27.
URL `http://dx.doi.org/10.1109/TIT.1967.1053964`

[38] P. J. Phillips, P. Grother, R. J. Micheals, *Evaluation Methods in Face Recognition*, in: Handbook of Face Recognition, 2nd Edition., 2011, pp. 551–574.
URL `http://dx.doi.org/10.1007/978-0-85729-932-1_21`

[39] G. Bradski, *The OpenCV Library*, Dr. Dobb's Journal of Software Tools.
URL `http://opencv.org/`

[40] M. Castrillón, O. Déniz, D. Hernández, J. Lorenzo, *A Comparison of Face and Facial Feature Detectors Based on the Viola-Jones General Object Detection Framework*, Mach. Vision Appl. 22 (3) (2011) 481–494.
URL `http://dx.doi.org/10.1007/s00138-010-0250-7`

[41] G. B. Huang, M. A. Mattar, H. Lee, E. G. Learned-Miller, *Learning to Align from Scratch.*, in: P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger (Eds.), NIPS, 2012, pp. 773–781.
URL `http://dblp.uni-trier.de/db/conf/nips/nips2012.html#HuangMLL12`

[42] G. Stockman, L. G. Shapiro, Matching in 2d, in: *Computer Vision*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[43] M. Heikkilä, T. Ahonen, MATLAB implementation of the local binary pattern operator, `http://www.cse.oulu.fi/CMV/Downloads/LBPMatlab`, Accessed: 2014-08-04.

[44] D. Cai, MATLAB implementation of PCA and LDA, `http://www.cad.zju.edu.cn/home/dengcai/Data/DimensionReduction.html`, Accessed: 2014-08-04.

[45] M. R. Quiñones, D. Masip, J. Vitrià, *Automatic Detection of Facial Feature Points via HOGs and Geometric Prior Models*, in: IbPRIA, Lecture Notes in Computer Science, Springer, 2011, pp. 371–378.

[46] M. K. Hasan, C. J. Pal, *Improving alignment of faces for recognition.*, in: ROSE, IEEE, 2011, pp. 249–254.
URL `http://dblp.uni-trier.de/db/conf/rose/rose2011.html#HasanP11`

[47] T. Berg, P. N. Belhumeur, *Tom-vs-Pete Classifiers and Identity-Preserving Alignment for Face Verification*, in: British Machine Vision Conference, BMVC 2012, Surrey, UK, September 3-7, 2012, 2012, pp. 1–11.
URL `http://dx.doi.org/10.5244/C.26.129`

[48] P. Miller, J. Lyle, *The Effect of Distance Measures on the Recognition Rates of PCA and LDA Based Facial Recognition*, Tech. rep., Digitial Image Processing, Clemson University, Accessed: 2014-10-10.
URL `http://www.ces.clemson.edu/~stb/ece847/projects/distance_measures.pdf`

[49] R. Wang, S. Shan, X. Chen, W. Gao, *Manifold-Manifold Distance with application to face recognition based on image set.*, in: CVPR, IEEE Computer Society.

[50] R. I. Hartley, A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd Edition, Cambridge University Press, ISBN: 0521540518, 2004.

[51] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, M. Pantic, *300 Faces in-the-Wild Challenge: The First Facial Landmark Localization Challenge*, in: The IEEE International Conference on Computer Vision (ICCV) Workshops, 2013.

[52] A. Patil, S. Kolhe, P. Patil, *2D Face Recognition Techniques: A Survey*, International Journal of Machine Intelligence vol. 2 (2010) pp 74–83.

[53] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach. Learn. Res. 3 (2003) 1157–1182.
URL `http://dl.acm.org/citation.cfm?id=944919.944968`

[54] D. DeMenthon, Software and articles regarding the POSIT Algoritm, `http://www.cfar.umd.edu/~daniel/Site_2/Code.html`, Accessed: 2014-10-07.

Appendix A: Additional tables

A.1 List of OpenCV Haar-cascades

Type of Haar-cascade	Name	Min size	Max size	Scale factor	Merge threshold
Face	haarcascade_frontalface_default.xml	[200,200]	[400,400]	1.1	4
Face	haarcascade_frontalface_alt.xml	[200,200]	[400,400]	1.1	4
Face	haarcascade_frontalface_alt2.xml	[200,200]	[400,400]	1.1	4
Face	haarcascade_frontalface_alt_tree.xml	[200,200]	[400,400]	1.1	4
Eye	haarcascade_eye.xml	[20,20]	[100,100]	1.05	8
Mouth	haarcascade_mcs_mouth.xml	[15,25]	[100,150]	1.05	12

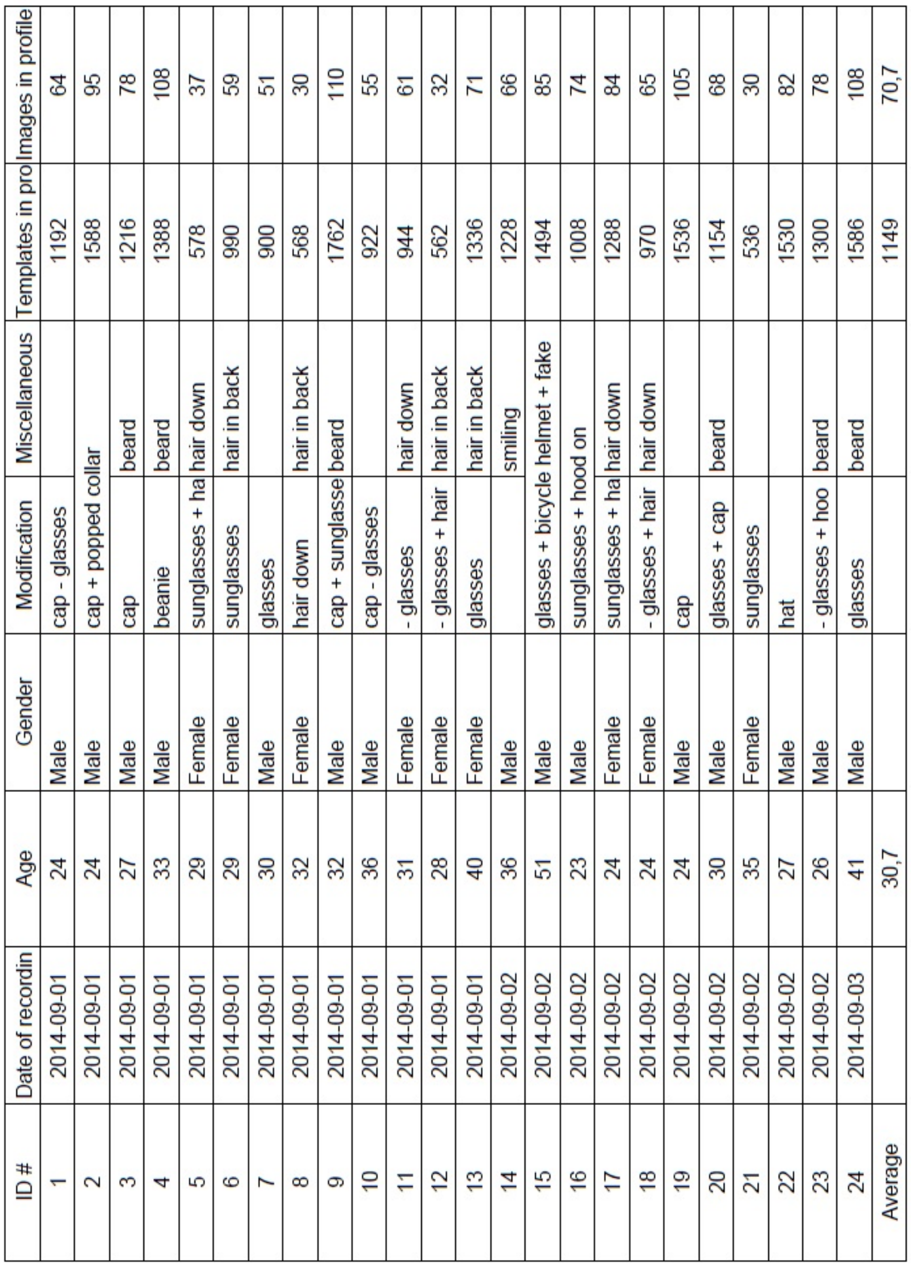
Table A.1: Table containing the types, names and settings of the Haar-cascades used in the face and feature detection steps. These are freely available in the OpenCV package that can be downloaded at [39]. The settings are modified in MATLAB. More detailed descriptions of the cascades can be found in [40].

A.2 Run times for LBP-method

Step	Run-time per image (seconds)
Face detection	0.029
Feature extraction	0.10
Facial alignment	0.013
LBP-face recognition	0.047

Table A.2: Table displaying the run times of each step in the system for the LBP-method. Performance evaluated on a 2.4 GHz Intel i7 processor.

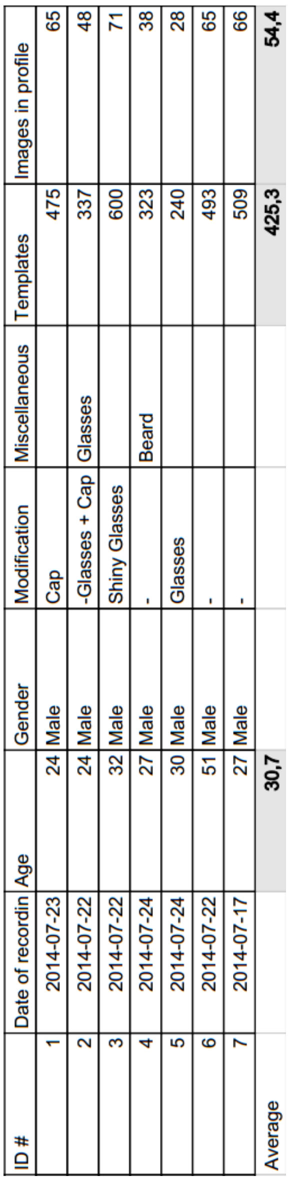
A.3 Video database information



ID #	Date of recordin	Age	Gender	Modification	Miscellaneous	Templates in pro	Images in profile
1	2014-09-01	24	Male	cap - glasses		1192	64
2	2014-09-01	24	Male	cap + popped collar		1588	95
3	2014-09-01	27	Male	cap	beard	1216	78
4	2014-09-01	33	Male	beanie	beard	1388	108
5	2014-09-01	29	Female	sunglasses + ha	hair down	578	37
6	2014-09-01	29	Female	sunglasses	hair in back	990	59
7	2014-09-01	30	Male	glasses		900	51
8	2014-09-01	32	Female	hair down	hair in back	568	30
9	2014-09-01	32	Male	cap + sunglasse	beard	1762	110
10	2014-09-01	36	Male	cap - glasses		922	55
11	2014-09-01	31	Female	- glasses	hair down	944	61
12	2014-09-01	28	Female	- glasses + hair	hair in back	562	32
13	2014-09-01	40	Female	glasses	hair in back	1336	71
14	2014-09-02	36	Male		smiling	1228	66
15	2014-09-02	51	Male	glasses + bicycle helmet + fake		1494	85
16	2014-09-02	23	Male	sunglasses + hood on		1008	74
17	2014-09-02	24	Female	sunglasses + ha	hair down	1288	84
18	2014-09-02	24	Female	- glasses + hair	hair down	970	65
19	2014-09-02	24	Male	cap		1536	105
20	2014-09-02	30	Male	glasses + cap	beard	1154	68
21	2014-09-02	35	Female	sunglasses		536	30
22	2014-09-02	27	Male	hat		1530	82
23	2014-09-02	26	Male	- glasses + hoo	beard	1300	78
24	2014-09-03	41	Male	glasses	beard	1586	108
Average		30,7				1149	70,7

Figure A.1: Table containing detailed information about the complete video and profile database.

A.4 Pilot video database information



ID #	Date of recordin	Age	Gender	Modification	Miscellaneous	Templates	Images in profile
1	2014-07-23	24	Male	Cap		475	65
2	2014-07-22	24	Male	-Glasses + Cap	Glasses	337	48
3	2014-07-22	32	Male	Shiny Glasses		600	71
4	2014-07-24	27	Male	-	Beard	323	38
5	2014-07-24	30	Male	Glasses		240	28
6	2014-07-22	51	Male	-		493	65
7	2014-07-17	27	Male	-		509	66
Average		**30,7**				**425,3**	**54,4**

Figure A.2: Table containing detailed information about the pilot video and profile database.