Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
() : 1 – 28
© TÜBİTAK
doi:10.3906/elk-1301-65

A comprehensive comparison of features and embedding methods for face recognition

Hasan Serhan YAVUZ[*], **Hakan ÇEVİKALP**, **Rifat EDİZKAN**
Department of Electrical and Electronics Engineering, Eskişehir Osmangazi University, Eskişehir, Turkey

Received: 11.01.2013 • **Accepted/Published Online:** 09.11.2013 • **Printed:** ..201

Abstract: Face recognition is an essential issue in modern-day applications since it can be used in many areas for several purposes. Many methods have been proposed for face recognition. It is a difficult task since variations in lighting, instantaneous mimic varieties, posing angles, and scaling differences can drastically change the appearance of the face. To suppress these complications, effective feature extraction and proper alignment of face images gain as much importance as the recognition method choice. In this paper, we provide an extensive comparison of the state-of-the-art face recognition methods with the most well-known techniques used in feature representation. In order to test the performances of these various methods, we created a new face database, named the ESOGU face database, which includes frontal images of 100 people taken under different lighting and posing conditions. In addition to our new database, we also present experiments on the well-known AR face database to obtain more general and reliable results. Moreover, we investigate the automatic face detection and automatic normalization of the face images in the databases. Based on this, we discuss the use of such automatic methods for face recognition applications.

Key words: Face recognition, feature extraction, face database, face detection, classification

1. Introduction

Recognizing faces in still and video images has enjoyed significant attention, especially during the past few years [1–4]. One of the reasons for this trend is the wide range of applications of face recognition, including military, commercial, and law enforcement applications. Another reason for the popularity of face recognition is its nonintrusive and user-friendly nature. Recognizing people by using face images can be easily accomplished without people's cooperation or knowledge. On the other hand, recognizing people based on other biometrics such as the iris or fingerprints is not suitable for noncollaborative individuals. That is why facial features scored the highest compatibility among the 6 biometrics studied in [5]. Lastly, advances in related tasks such as face detection and image representation increased the attention on face recognition.

Face recognition can be defined as the identification of individuals from still or video images of a scene by using a stored database of faces labeled with people's identities. It generally includes steps of localization of face regions in a cluttered background, normalization of images, extraction of features from the face regions, and, finally, recognition and verification. It is a difficult task as there are numerous factors such as 3D pose, facial expression, hair style, beard, moustache, and makeup that affect the appearance of an individual's facial features [4]. In addition to these factors, lighting, background, scale changes, noise, and occlusion make the problem even harder.

[*]Correspondence: hsyavuz@ogu.edu.tr

Face detection is the first step in any face recognition application. Face detection is also a challenging task. However, highly successful techniques that have been recently introduced make this task a very advanced field of research [6–9]. Among these methods, the AdaBoost cascade face detector of Viola and Jones [7] is considered state-of-the-art. It uses a computationally efficient cascade of weak classifiers that rapidly rejects background regions while keeping face image regions. Each classifier in the cascade is an AdaBoost ensemble of rectangular Haar-like features.

Many methods have been proposed for the classification of faces returned by a face detector algorithm. Among these methods, appearance-based methods are the most successful and practical ones. In these methods, 2D face images are first properly normalized based on facial feature locations and then they are resized to a fixed size. This is followed by the extraction of features from the images. One of the most common feature types is the gray-level values of pixels arranged in a fixed (typically raster) order. In addition to the gray levels, more elaborate features that yield better results than gray levels, such as wavelets [10], Gabor filter-based features [11,12], and discrete cosine transform (DCT) features [13,14], are used as well. Recently, local histogram-based features including local binary patterns (LBPs) [15] and local ternary patterns (LTPs) [16] have become very popular owing to their resistance to normalization errors and small lighting changes.

After the feature extraction step, each face image is usually represented with a feature vector, and thus a face image can be thought of as a point in this vector space. The dimensionality of the vector space is typically high, even for images of modest size. As a result, classification methods that work on this representation encounter the so-called curse of dimensionality problem. However, since face images have similar structures, face feature vectors are correlated and they lie on a manifold of intrinsically low dimension. Thus, any image in the high-dimensional vector space can be represented in a lower-dimensional space without losing a significant amount of important information. The eigenface method [17] was proposed for finding such a lower-dimensional space. The key idea behind the eigenface method, which uses principal component analysis (PCA), is to find the best set of embedding directions that will maximize the total scatter across all image vectors. In choosing a criterion that maximizes the total scatter, the eigenface method tends to model unwanted within-class variations such as those resulting from differences in lighting, facial expression, and other factors [18]. The linear discriminant analysis (LDA) method overcomes the limitations of the eigenface method by applying Fisher's linear discriminant criterion, which maximizes the ratio of the determinant of between-class scatter to the determinant of within-class scatter in the embedded space. In face recognition tasks, LDA cannot be applied directly since the dimensionality of the face feature vectors is typically very high compared to the number of samples in the training set. As a consequence, the within-class scatter matrix is rank-deficient, a problem known as the small sample size problem. The most common way to overcome this problem is to apply a two-stage PCA + LDA method (also known as the Fisherface method), in which PCA is first used for dimension reduction so as to make the within-class scatter matrix nonsingular before the application of LDA [19,20]. However, in the PCA + LDA method, applying PCA for dimensionality reduction has the potential to remove directions that contain discriminative information. The discriminative common vector (DCV) method [18] has been proposed to overcome the limitations of PCA + LDA. In DCV, the FLDA criterion is modified such that the embedding directions are forced to come from the null space of the within-class scatter. As a result, the singularity problem is avoided and it is proved that the resulting embedding directions span the optimal discriminant subspace [21].

When the face image vectors lie on a nonlinear manifold, the linear subspace-based methods discussed above may not work well. The Laplacianface method [22] has been introduced for estimating the nonlinear face manifold based on the fact that manifold structure is modeled by a nearest-neighbor graph, which preserves the local structure of the image feature vectors. In contrast to Laplacianfaces, kernel methods [21,23–26] approximate the nonlinear manifold by nonlinearly mapping the image feature vectors into a higher-dimensional space and then applying the linear subspace method in the mapped space. These methods are formulated in terms of the dot products of the mapped samples, and kernel functions are used to compute these dot products. Therefore, the nonlinear mapping function and the mapped samples are not used explicitly, which makes the method computationally feasible.

In this paper, we first provide an extensive comparison of the most well-known embedding methods by using different feature representations. In the literature, the researchers focused on either comparing different embedding methods using one type of representation (typically gray-level values) [17–26] or different types of representations tested with a limited number of embedding methods (typically Fisherfaces) [15,16]. In contrast, we test almost all popular image representations including gray levels, LBP, LTP, DCT, and Gabor features using the leading embedding methods of eigenfaces, Fisherfaces, DCV, their kernel counterparts, and Laplacianfaces.

It is well known that the success of appearance-based face recognition systems heavily relies on accurate face and facial feature localization. Thus, most researchers skip the face detection stage by selecting face regions and facial features manually. As a second contribution, we investigate the effect of the performance of face detection and facial feature localization on face recognition accuracy, which is largely ignored in the literature (there is a attempt in this direction in [27], but the author only investigates small localization errors by adding artificial noise). In this paper, in the first scenario, we manually select the face region and normalize the face images accurately based on manually located facial features (the way commonly preferred in the literature), and then we compute recognition accuracies based on this precise normalization. In the second scenario, we use the state-of-the-art algorithms to automatically detect face regions and facial features, and normalization is done based on these imprecise detector outputs. Finally, we compute classification accuracies and compare the results to the ones obtained by manual normalization. These results are more realistic and give an idea about what to expect from an intelligent face recognition system that is designed for real-world face recognition applications.

Lastly, we introduce a new face recognition database called the Eskişehir Osmangazi University (ESOGU) face database. Most researchers report their results on different face databases by using their own setup since a standard protocol is not defined. Thus, the results for the same face databases are not directly comparable most of the time. To avoid this problem, in addition to the face images in this database, we provide manually normalized images and all image descriptions obtained using the feature representations given above. We also provide randomized indices to create training, validation, and test sets. Therefore, researchers can apply their algorithms using the same experimental setup as in this paper and can compare their results directly to the ones reported here. More detailed information can be found in Section 3.4.

2. Methods

We use appearance-based face recognition methods in this study. Alignment of the face region, feature choice, and embedding method choice significantly affect the overall recognition performance in these methods. Now we summarize the full face recognition methodology that we studied in this paper.

2.1. Face alignment

The performance of the recognition algorithms generally depends on careful cropping of the face region from the whole image and proper positioning of the face image into a template so that the position of the features relative to a fixed coordinate system can be examined in a standard way. In general, the face alignment procedure is usually done by manually selecting some fiducial points on the face (such as eyes, mouth, nose, etc.) and applying a transformation to match an initially defined face template. In this study, we have studied both manual and automatic face alignments, which are described in detail in Section 3. A manual alignment procedure is necessary to compare the methods under controlled conditions, whereas an automatic alignment procedure gives some idea about the real-world performance of the methods when adopted into intelligent face recognition systems.

2.2. Features

In appearance-based face recognition, the pixel intensity values or gray-level values in the image are generally represented in the form of a high-dimensional vector, and they constitute the raw representation of a digital image. However, the dimensionality of the gray-level vector representation can sometimes be very high and can include redundant information. In this case, one can transform the gray-level data into the set of features using any feature extraction technique. Features are expected to be more compact and representative than the gray-level representation. In this study, in addition to the gray levels, we experimented with the following representations to seek more elaborate features yielding better results in face recognition: 2D Gabor filter, DCT, LBP, and LTP-based features. Brief explanations of these methods are given in the following sections.

2.2.1. 2D Gabor filter-based features

Gabor filter-based feature extraction has been emerging as one of the most promising ways to represent an image in a way that is similar to the representation of an image in the human brain. Marcelja [28] stated that the representation of an image in the visual cortex must involve both location and spatial frequency variables and the visual cortex representation corresponds closely to the Gabor scheme. Daugman [29] developed the 2D form of the Gabor functions and Lee [30] showed that 2D Gabor functions can be used in image representation as a series of local spatial bandpass filters.

A 2D Gabor function is simply the product of an elliptical Gaussian with the complex exponential. Let (x, y) denote the spatial coordinate variables. Setting the sharpness of the Gaussian in the x-axis as α and in the y-axis as β, and defining the constant ratios $\gamma = \frac{f}{\alpha}$, $\eta = \frac{f}{\beta}$ such that the functions on different frequencies behave as a scaled version of each other, the 2D Gabor function can be written as given in Eq. (1) [11].

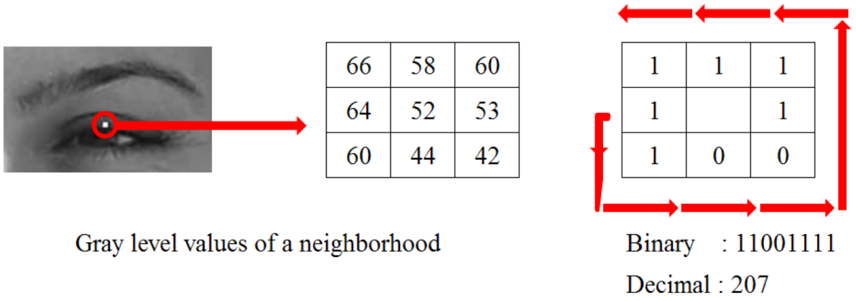
$$\mathrm{GABOR}\,(x,y) = \frac{f^2}{\pi\gamma\eta} \exp\left(-\left[\frac{f^2}{\gamma^2}x_r^2 + \frac{f^2}{\eta^2}y_r^2\right]\right) \exp\left(j2\pi f x_r\right) \tag{1}$$

Here, $x_r = x\cos(\theta) + y\sin(\theta)$, $y_r = -x\sin(\theta) + y\cos(\theta)$, f is the frequency of the modulating sinusoidal plane wave, and θ is the orientation of the major axis of the elliptical Gaussian.

The 2D Gabor function is complex, but we can use its magnitude to construct a set of Gabor filters, called the filterbank, with different scales and orientations. Since the local frequency and orientation are unknown, N_s number of different scales and N_o number of different orientations are usually performed in the filter bank. This results in a total of $(N_s \times N_o)$ different filters. The outputs of each filter are concatenated and the resulting vector finally forms the Gabor feature of the image. The dimensionality of the resulting feature is usually very high and the resulting feature could be downsampled so that the size of the feature is diminished to a moderate level for memory considerations.

2.2.2. LBP features

The local binary patterns were first described in [31] and have been found to be a powerful feature for texture classification. The method has further been used as a feature extraction method in face recognition [15]. The LBP method makes a pairwise comparison of a pixel value with its neighboring pixels in order to assign a label to every pixel of the image, resulting in a binary number [15]. The neighborhood size may differ to deal with textures at different scales, but the 3 × 3 neighborhood is the most common. Basically, the LBP pattern is obtained as follows: the center pixel is chosen as the reference pixel. The intensity value of the reference pixel is compared to the other pixels in its neighborhood. If the reference pixel value is greater than or equal to the neighboring pixel, the value of the neighboring pixel is set to 1; otherwise, it is set to 0. The binary pattern for the reference pixel is then obtained by collecting binary digits from its neighboring pixels. The final resulting binary value is converted into its decimal representation. This procedure is visualized in Figure 1.



66	58	60
64	52	53
60	44	42

1	1	1
1		1
1	0	0

Figure 1. An LBP extraction using 3 × 3 neighborhood.

The LBP operator can be formulated as given in Eq. (2). Here, x_c and y_c denote the central pixel locations, i_c is the intensity value of the central pixel, i_n with $n = 0, \cdots, 7$ denotes the intensity value of the nth neighboring pixel, and $s[k]$ is the unit step function, which returns 1 if $k \geq 0$ and 0, or otherwise:

$$\mathrm{LBP}\left(x_c, y_c\right) = \sum_{n=0}^{7} 2^n s\left[i_n - i_c\right] \quad . \tag{2}$$

The LBP method is capable of representing faces in a very simple but effective way. Additionally, using this method, particularly the lighting intensity changes in the image can be reduced.

2.2.3. LTP features

The LTP method is an extension of the LBP method. LTPs are supposed to be less sensitive to noise in uniform regions and have more discriminative power [16]. Basically, to be more robust to noise, the LTP extends the LBP to 3-valued codes: $(-1, 0, 1)$. The gray-level values in the neighborhood of a central pixel are quantized to -1 if they are less than $(i_c - t)$ value, they are quantized to 0 if they are in the closed interval of $[i_c - t,\ i_c + t]$, and they are quantized to 1 if they are greater than $(i_c + t)$, where t is a user-specified threshold. The LTP extraction procedure is illustrated in Figure 2.

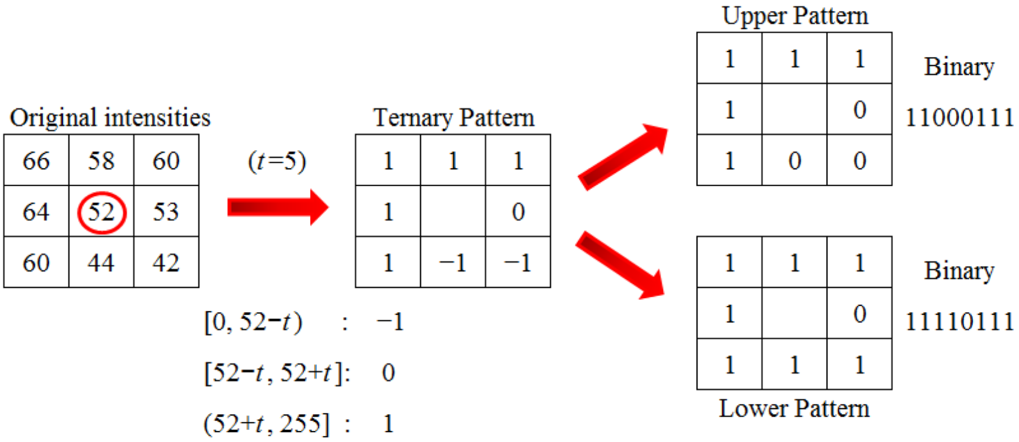


Figure 2. An LTP extraction using 3 × 3 neighborhood.

The ternary pattern in this method has three codes, but instead of applying 3^n-valued codes, the final codes are converted to binary representation by splitting them into their positive and negative halves, as shown in Figure 2. In other words, the LTP method subsequently treats the ternary pattern as two separate channels of LBP descriptors for which separate histograms and similarity metrics are computed and the results are combined at the end of the computation [16].

2.2.4. DCT features

The DCT is a Fourier-related transform that attempts to decorrelate the image data and finds compact representation of the data in the form of the DCT coefficients. The DCT provides a good compromise between information-packing ability and computational complexity. As a result, the DCT has become an international standard for transform coding systems [32]. The DCT has also been used as a feature extraction method in face recognition studies [13,14]. The formulation of the DCT for an $N \times N$ image having intensity $i(x,y)$ can be given as in Eq. (3).

$$\mathrm{DCT}(u,v) = \alpha(u)\,\alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} i(x,y) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \cos\left[\frac{\pi(2y+1)v}{2N}\right] \qquad (3)$$

Here, $u, v = 0, 1, \cdots, N-1$ and $\alpha(k) = \begin{cases} \sqrt{1/N}, & k = 0 \\ \sqrt{2/N}, & k = 1, 2, \cdots, N-1 \end{cases}$.

The DCT coefficients form an $N \times N$ matrix. The first DCT coefficient located at the top left corner represents the average intensity value of the block. The uniform brightness change in the image block is eliminated by removing this coefficient from the representation. Other coefficients represent the spatial frequency components of the image and they are located in the matrix from low- to high-frequency order. The DCTs obtained from the low-frequency band are important for representing significant facial features such as hair, face, nose, mouth, and eyes. These coefficients are located in the upper left corner of the DCT matrix.

The DCT feature extraction is applied to the face image after detection and normalization. In this method, the face image is divided into $M \times M$ blocks and each block is represented by its DCT coefficient. In each block, a number of coefficients are collected from the upper left corner of the DCT matrix in a zigzag manner. The face feature representation is obtained by concatenating these coefficients [13,14,33].

2.3. Embedding methods

Face images are represented with feature vectors rather than gray-level values after the feature extraction step. Thus, at this point, a face image can be thought of as an element of another vector space whose dimensionality is still high even for images of modest size. As a result, classification methods that work on this representation encounter the so-called curse of dimensionality problem. However, using embedding techniques, any image in the high-dimensional vector space can be represented in a lower-dimensional space without losing a significant amount of important information. In this study, we consider the most well-known linear and nonlinear embedding methods: eigenface [17], LDA [19], DCV [18], Laplacianfaces [22], kernel PCA [25,34], kernel LDA [23,24], and kernel DCV [21].

In order to fix the notations, let us assume that the training set is composed of C classes, where the ith class is denoted by $\omega^{(i)}$ containing n_i samples, and let $\mathbf{x}_{im}$ be a d-dimensional $(d = d_1 \times d_2)$ column vector representation of a $d_1 \times d_2$ face image sample coming from the ith class. There will be a total of $n = \sum\limits_i n_i$ samples in the training set. The matrix $\mathbf{X}$ corresponds to the data sample matrix where each column shows a sample in the training set $(\mathbf{X} = [\mathbf{x}_{11}\ \mathbf{x}_{12}\ \cdots\ \mathbf{x}_{im}\ \cdots\ \mathbf{x}_{Cn_C}])$. We summarize the embedding methods used in this study in the following subsections.

2.3.1. The eigenface method

The aim of the eigenface [17] method is to determine the optimum projection matrix that projects the sample vectors onto a lower dimensional space. The optimum projection matrix is a matrix in which each column is formed with the first few principal components, also called the basis vectors of the subspace. The basis vectors of the subspace are obtained by using the scatter matrix $\mathbf{S}_T$, defined as:

$$\mathbf{S}_T = \sum_{i=1}^{C} \sum_{m=1}^{n_i} (\mathbf{x}_{im} - \mu)(\mathbf{x}_{im} - \mu)^T, \tag{4}$$

where μ is the mean of all images in the training set. The singular value decomposition of the scatter matrix yields eigenvalues and the corresponding eigenvectors of $\mathbf{S}_T$. Since the scatter matrix is a positive semidefinite matrix, all eigenvalues are greater than or equal to 0. When the eigenvalues are sorted in descending order, the first few eigenvectors corresponding to the largest eigenvalues are taken as the basis vectors of the subspace. Assuming that first l_d principal components are used to construct the subspace, then the optimum projection matrix is formed by putting the first l_d eigenvectors in the column of a matrix. Usually, $l_d << d$. If the eigenvectors are represented with $\mathbf{v}_k$, $k = 1, \cdots, l_d$ and the optimum projection matrix is represented by $\mathbf{W} = [\mathbf{v}_1\ \mathbf{v}_2\ \cdots\ \mathbf{v}_{l_d}]$, then the embedding is realized with the following projection that yields $\mathbf{y}_{im} \in \Re^{l_d}$ as the output vector.

$$\mathbf{y}_{im} = \mathbf{W}^T \mathbf{x}_{im} \tag{5}$$

2.3.2. The LDA method

The LDA method overcomes the limitations of the eigenface method by applying Fisher's linear discriminant criterion [19,20]. This criterion aims to maximize the legibility of different classes while at the same time minimizing the similarity of the samples appertaining to the same class. Let the within-class scatter matrix, $\mathbf{S}_W$, and the between-class scatter matrix, $\mathbf{S}_B$, be defined as follows, where μ_i is the mean of the samples belonging to the ith class and μ is the mean of the training set including all samples:

$$\mathbf{S}_W = \sum_{i=1}^{C} \sum_{m=1}^{n_i} \left(\mathbf{x}_{im} - \mu_i\right)\left(\mathbf{x}_{im} - \mu_i\right)^T, \tag{6}$$

$$\mathbf{S}_B = \sum_{i=1}^{C} n_i \left(\mu_i - \mu\right)\left(\mu_i - \mu\right)^T. \tag{7}$$

The LDA method then tries to maximize the ratio $J\left(\mathbf{W}_{opt}\right) = \arg\max_{\mathbf{W}} \frac{\left|\mathbf{W}^T \mathbf{S}_B \mathbf{W}\right|}{\left|\mathbf{W}^T \mathbf{S}_W \mathbf{W}\right|}$. This ratio is maximized when the column vectors of projection matrix $\mathbf{W}$ are the eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$. In face recognition tasks, this method cannot be applied directly since the dimension of the sample space is typically larger than the number of samples in the training set, known as the small sample size problem. As a consequence, $\mathbf{S}_W$ is singular in this case and the inverse of it cannot be calculated. The most common way to overcome this problem is to apply a 2-stage PCA + LDA method (also known as the Fisherface method), in which PCA is first used for dimension reduction so as to make the within-class scatter matrix nonsingular before the application of LDA [19]. In this study, we applied this method and in some references it may be called FLDA.

At the end, the projection matrix is formed by using a similar procedure as in the eigenface method. The singular value decomposition is applied to the matrix $\mathbf{S}_W^{-1}\mathbf{S}_B$ in this case, and the subspace basis vectors are chosen as the eigenvectors corresponding to the nonzero eigenvalues. As a result, the embedding is carried out as given in Eq. (5).

2.3.3. The DCV method

The DCV method is another powerful embedding method for face recognition [18]. In DCV, the FLDA criterion is modified such that the embedding directions are forced to come from the null space of the within-class scatter. As a result, the singularity problem is avoided and the resulting embedding directions span the optimal discriminative subspace. The DCV method can be only used when the dimension of the sample space is greater than the rank of $\mathbf{S}_W$ [18].

The DCV method starts with the common vector determination of the within-class scatter matrix $\mathbf{S}_W$ given in Eq. (6). As mentioned in the previous section, the dimension of the sample space is typically larger than the number of samples in the training set, which yields a singular scatter matrix. If the rank of $\mathbf{S}_W$ is r, this guarantees the first r eigenvalues to be positive and the rest are zero when the singular value decomposition is applied to it. Eigenvectors corresponding to positive eigenvalues are the basis of the range space and eigenvectors corresponding to zero eigenvalues are the basis of the null space of $\mathbf{S}_W$. Let the eigenvectors of $\mathbf{S}_W$ corresponding to positive eigenvalues be denoted by $\mathbf{v}_j$, $j = 1, 2, \cdots, r$, and then the common vectors of the classes can be determined by using these vectors as given in Eq. (8).

$$\mathbf{x}_i^{com} = \mathbf{x}_{im} - \sum_{j=1}^{r} \left\langle \mathbf{x}_{im}, \mathbf{v}_j \right\rangle \mathbf{v}_j, \quad \forall m \tag{8}$$

Here, $\mathbf{x}_i^{com} \in \Re^d$ denotes the common vector of the ith class and the $\langle \cdot, \cdot \rangle$ operator represents the vector inner product of its entries. This equation is satisfied for any choice of m. After that, the scatter matrix of the common vectors, $\mathbf{S}_{com}$, is defined as given in Eq. (9).

$$\mathbf{S}_{com} = \sum_{i=1}^{C} \left(\mathbf{x}_i^{com} - \mu_{com}\right) \left(\mathbf{x}_i^{com} - \mu_{com}\right)^T \tag{9}$$

Here $\mu_{com} = \frac{1}{C} \sum\limits_{i=1}^{C} \mathbf{x}_i^{com}$ denotes the mean of the common vectors.

In the end, the projection matrix is formed by applying the singular value decomposition to $\mathbf{S}_{com}$. There are $(C-1)$ positive eigenvalues and the eigenvectors corresponding to these eigenvalues are chosen as the basis vectors forming the projection matrix. If these eigenvectors are denoted by $\mathbf{w}_k$, then the projection matrix is formed as $\mathbf{W} = [\mathbf{w}_1 \ \mathbf{w}_2 \ \cdots \ \mathbf{w}_{l_d}]$ and the embedding is realized as given in Eq. (5). Note that $l_d = C - 1$ in this case.

2.3.4. The Laplacianfaces method

The Laplacianfaces method [22] searches for an embedding space in which the similarity among the local neighborhoods is preserved. First, a similarity (also called adjacency) graph with n nodes is constructed. An edge between nodes i and j is created based on neighborhoods (e.g., using the k-nearest neighbors). After that, each edge is weighted according to a similarity function, e.g., the heat kernel function. The weights S_{ij} lie in the range [0,1] and take higher values for closer samples. The goal of Laplacianfaces is to minimize the loss function

$$J\left(\mathbf{w}\right) = \frac{1}{2} \sum_{k,l} \left(y_k - y_l\right)^2 S_{ij}, \tag{10}$$

where $\mathbf{w}$ is the embedding direction, and $y_k = \mathbf{w}^T \mathbf{x}_k$ is a one-dimensional representation of $\mathbf{x}_k$, which denotes any sample in the training set. This loss function assigns a high penalty to mapping neighboring points $\mathbf{x}_k$ and $\mathbf{x}_l$ far apart. The loss function can be written in a more compact form as

$$J\left(\mathbf{w}\right) = \mathbf{w}^T \mathbf{X} \left(\mathbf{D} - \mathbf{S}\right) \mathbf{X}^T \mathbf{w} = \mathbf{w}^T \mathbf{X} \mathbf{L} \mathbf{X}^T \mathbf{w}, \tag{11}$$

where $\mathbf{S}$ is the matrix of weights, and $\mathbf{D}$ is the diagonal matrix whose entries are the column (or row) sums of $\mathbf{S}$. The matrix $\mathbf{L} = (\mathbf{D} - \mathbf{S})$ is called the Laplacian matrix. An additional constraint, $\mathbf{w}^T \mathbf{X} \mathbf{D} \mathbf{X}^T \mathbf{w} = 1$, is included to normalize the projected data. The final embedding matrix $\mathbf{W}$ is constructed by the eigenvectors, which are the minimum eigenvalue solutions to the generalized eigenvalue problem:

$$\mathbf{X} \mathbf{L} \mathbf{X}^T \mathbf{w} = \lambda \mathbf{X} \mathbf{D} \mathbf{X}^T \mathbf{w}. \tag{12}$$

Note that the two matrices $\mathbf{XLX}^T$ and $\mathbf{XDX}^T$ are both symmetric and positive semidefinite since the Laplacian matrix $\mathbf{L}$ and the diagonal matrix $\mathbf{D}$ are both symmetric and positive semidefinite. After the projection matrix $\mathbf{W}$ is formed by using $\mathbf{w}$ vectors in the columns, the embedding can be realized as given in Eq. (5).

2.3.5. The kernel PCA method

The kernel methods have been proposed to overcome the separability of data that have an intrinsically nonlinear structure. For these cases, the linear methods may not provide sufficient discriminative power. Thus, the kernel trick, which is a way of mapping the data to a higher-dimensional inner product space, is applied to the linear methods with the expectation that the data gain meaningful linear structure in the new space and become linearly separable there. The basic idea of the kernel methods can be summarized as first transforming the data samples into a higher-dimensional space $\Im$ through a nonlinear mapping function $\varphi(\cdot)$ such that $\varphi(\cdot) : \Re^d \to \Im$, and then applying the linear methods in this space.

The principle of the kernel PCA method has a similar idea [34]. Let $\varphi(\mathbf{x}_{im})$ denote the mapped samples; the total scatter matrix $\mathbf{S}_T^{\Phi}$ of this method is then as given in Eq. (13).

$$\mathbf{S}_T^{\Phi} = \sum_{i=1}^{C} \sum_{m=1}^{n_i} \left(\varphi(\mathbf{x}_{im}) - \mu^{\Phi}\right) \left(\varphi(\mathbf{x}_{im}) - \mu^{\Phi}\right)^T \tag{13}$$

Here, μ^{Φ} denotes the mean of all mapped samples. The principal components are computed by solving the eigenvalue problem:

$$\lambda \mathbf{w} = \mathbf{S}_T^{\Phi} \mathbf{w}. \tag{14}$$

All eigenvectors $\mathbf{w}$ corresponding to the nonzero eigenvalues lie in the span of $\left\{\varphi(\mathbf{x}_{im}) - \mu^{\Phi},\ \forall i, m\right\}$, resulting in:

$$\mathbf{w} = \sum_{i=1}^{C} \sum_{m=1}^{n_i} \alpha_{im} \left(\varphi(\mathbf{x}_{im}) - \mu^{\Phi}\right). \tag{15}$$

The solution of the eigenvalue problem in Eq. (14) is equivalent to the solution of the eigenvalue problem of $n\lambda\alpha = \mathbf{K}\alpha$, where the $\mathbf{K} \in \Re^{n \times n}$ matrix is the kernel matrix, n is the total number of samples, and $\alpha = [\alpha_{11}, \cdots, \alpha_{im}, \cdots, \alpha_{Cn_C}]^T$. Since $\{\mathbf{w}_j, j = 1, \cdots, n-1\}$ must be an orthonormal set, the vectors α_j must be normalized such that $\langle \alpha_j, \alpha_j \rangle = 1$ [35].

Finally, the most significant l_d vectors can be chosen to construct the projection matrix. If these eigenvectors are denoted by $\{\mathbf{w}_k, k = 1, \cdots, l_d\}$, then the projection matrix is formed as $\mathbf{W} = [\mathbf{w}_1\ \mathbf{w}_2\ \cdots\ \mathbf{w}_{l_d}]$ and the embedding is realized as given in Eq. (16).

$$\mathbf{y}_{im} = \mathbf{W}^T \left(\varphi(\mathbf{x}_{im}) - \mu^{\Phi}\right) \tag{16}$$

Note that this computation can easily be evaluated through inner products with respect to the sample vectors in the training set.

2.3.6. The kernel LDA method

The kernel LDA method is an extension of Fisher's linear discriminant analysis using the kernel trick in a similar way as described the previous subsection [23,35,36]. In this method, the within- and between-class scatter matrices are determined using the following equations.

$$\mathbf{S}_W^{\Phi} = \sum_{i=1}^{C} \sum_{m=1}^{n_i} \left(\varphi(\mathbf{x}_{im}) - \mu_i^{\Phi}\right) \left(\varphi(\mathbf{x}_{im}) - \mu_i^{\Phi}\right)^T \tag{17}$$

$$\mathbf{S}_B^{\Phi} = \sum_{i=1}^{C} n_i \left(\mu_i^{\Phi} - \mu^{\Phi}\right) \left(\mu_i^{\Phi} - \mu^{\Phi}\right)^T \tag{18}$$

Here, μ_i^{Φ} represents the mean of the mapped samples belonging to the ith class and μ^{Φ} is the mean of the all samples. This method aims to maximize the FLDA criterion $J_{FLDA}^{\Phi}(\mathbf{W}_{opt}) = \max \frac{|\mathbf{W}^T \mathbf{S}_B^{\Phi} \mathbf{W}|}{|\mathbf{W}^T \mathbf{S}_W^{\Phi} \mathbf{W}|}$ in the mapped space $\Im$. The projection vectors that maximize this criterion are obtained by solving the following eigenvalue problem:

$$\lambda \mathbf{S}_W^{\Phi}\mathbf{w} = \mathbf{S}_B^{\Phi}\mathbf{w}. \tag{19}$$

This problem can be reformulated in terms of dot products as

$$\lambda \mathbf{K}_W\mathbf{K}_W^T\alpha = \mathbf{K}_B\mathbf{K}_B^T\alpha, \tag{20}$$

where $\mathbf{K}_W, \mathbf{K}_B \in \Re^{n\times n}$ are the kernel matrices. However, the matrix $\mathbf{K}_W\mathbf{K}_W^T$ is rank-deficient since its rank cannot be larger than $n-C$. Therefore, a small diagonal perturbation matrix Δ is added to it in order to make it nonsingular. The singular value decomposition is then applied to the final matrix $\left(\mathbf{K}_W\mathbf{K}_W^T + \Delta\right)^{-1}\mathbf{K}_B\mathbf{K}_B^T$ to determine the most significant eigenvectors. The eigenvectors are first normalized as in the kernel PCA case and they are used to construct the projection matrix $\mathbf{W}$ in the same way, and the final embedding is realized as given in Eq. (16).

2.3.6. The kernel DCV method

The kernel trick can also be applied to the discriminative common vectors method, which maps the original sample space to a higher dimensional mapped space [21]. The mapped space could be arbitrarily large and possibly have an infinite dimensionality. This is why the null space separation of the within-class scatter matrix is always accomplished in the mapped space, which makes a perfect environment for the application of the DCV method. Therefore, by applying the kernel DCV method, a 100% recognition accuracy rate with respect to the training data can be obtained for linearly nonseparable classes [35].

In the kernel DCV method, the optimal projection vectors can be found by first projecting the training set samples onto a total scatter matrix through the kernel PCA. The vectors that span the null space of the within-class scatter matrix of the transformed samples can then be determined by applying the DCV method, given in Section 2.3.3, in the transformed space. The kernel DCV procedure can be summarized as follows.

Let $\mathbf{S}_T^{\Phi}$ denote the total scatter matrix given in Eq. (13), $\mathcal{R}\left(\mathbf{S}_T^{\Phi}\right)$ denote the range space of $\mathbf{S}_T^{\Phi}$, and $\mathbf{\Phi} = [\varphi(\mathbf{x}_{11})\ \ \varphi(\mathbf{x}_{12})\ \ \cdots\ \ \varphi(\mathbf{x}_{Cn_C})]$ represent the matrix whose columns are the transformed training samples in the transformed space $\Im$. First, the training set samples are projected onto $\mathcal{R}\left(\mathbf{S}_T^{\Phi}\right)$ through the kernel PCA method given in Section 2.3.5. The kernel matrix is given as

$$\mathbf{K} = \mathbf{\Phi}^T\mathbf{\Phi} = \left.\langle\varphi\left(\mathbf{x}_{im}\right), \varphi\left(\mathbf{x}_{jn}\right)\rangle\right|_{\substack{i,j=1,\cdots,C \\ m=1,\cdots,n_i;\, n=1,\cdots,n_j}} = \left.k\left(\mathbf{x}_{im}, \mathbf{x}_{jn}\right)\right|_{\substack{i,j=1,\cdots,C \\ m=1,\cdots,n_i;\, n=1,\cdots,n_j}}, \tag{21}$$

where $k(\cdot)$ represents the kernel function. The kernel matrix $\mathbf{K} \in \Re^{n\times n}$ satisfies the equality

$$\tilde{\mathbf{K}} = \mathbf{K} - (1/n)\mathbf{1K} - (1/n)\mathbf{K1} + (1/n^2)\mathbf{1K1} = \mathbf{U\Lambda U}^T \quad , \tag{22}$$

where $\tilde{\mathbf{K}} \in \Re^{n\times n}$ and $\mathbf{1} \in \Re^{n\times n}$ is a matrix whose entries are all ones. $\mathbf{\Lambda}$ is the diagonal matrix of nonzero eigenvalues and $\mathbf{U}$ is the matrix of normalized eigenvectors associated with $\mathbf{\Lambda}$. The matrix that transforms the training set samples onto $\mathcal{R}\left(\mathbf{S}_T^{\Phi}\right)$ is $\left(\mathbf{\Phi} - (1/n)\mathbf{\Phi 1}\right)\mathbf{U\Lambda}^{-1/2}$ [21]. The corresponding total class scatter and within-class and between-class scatter matrices are given in Eqs. (23)–(25).

$$\tilde{\mathbf{S}}_T^{\Phi} = \left(\left(\mathbf{\Phi} - (1/n)\mathbf{\Phi}\mathbf{1}\right)\mathbf{U}\mathbf{\Lambda}^{-1/2}\right)^T \mathbf{S}_T^{\Phi} \left(\left(\mathbf{\Phi} - (1/n)\mathbf{\Phi}\mathbf{1}\right)\mathbf{U}\mathbf{\Lambda}^{-1/2}\right) \tag{23}$$

$$\tilde{\mathbf{S}}_W^{\Phi} = \left(\left(\mathbf{\Phi} - (1/n)\mathbf{\Phi}\mathbf{1}\right)\mathbf{U}\mathbf{\Lambda}^{-1/2}\right)^T \mathbf{S}_W^{\Phi} \left(\left(\mathbf{\Phi} - (1/n)\mathbf{\Phi}\mathbf{1}\right)\mathbf{U}\mathbf{\Lambda}^{-1/2}\right) \tag{24}$$

$$\tilde{\mathbf{S}}_B^{\Phi} = \left(\left(\mathbf{\Phi} - (1/n)\mathbf{\Phi}\mathbf{1}\right)\mathbf{U}\mathbf{\Lambda}^{-1/2}\right)^T \mathbf{S}_B^{\Phi} \left(\left(\mathbf{\Phi} - (1/n)\mathbf{\Phi}\mathbf{1}\right)\mathbf{U}\mathbf{\Lambda}^{-1/2}\right) \tag{25}$$

Here, $\mathbf{S}_W^{\Phi}$ and $\mathbf{S}_B^{\Phi}$ are the traditional within-class and between-class scatter matrices given in Eqs. (17) and (18). Next, the vectors that span the null space of $\tilde{\mathbf{S}}_W^{\Phi}$ are found by the singular value decomposition procedure. The normalized eigenvectors corresponding to zero eigenvalues of $\tilde{\mathbf{S}}_W^{\Phi}$ form an orthonormal basis for the null space of $\tilde{\mathbf{S}}_W^{\Phi}$. Let $\mathbf{V}$ be the matrix whose columns are the computed eigenvectors corresponding to zero eigenvalues such that $\mathbf{V}^T\tilde{\mathbf{S}}_W^{\Phi}\mathbf{V} = 0$. After that, the null space of $\mathbf{V}^T\tilde{\mathbf{S}}_B^{\Phi}\mathbf{V}$, if it exists, is removed and the projection directions are rotated so that the new total and between-class scatter matrices are diagonal to imply that the scatter matrices of the feature vectors of the training set samples are uncorrelated, as given in Eq. (26).

$$\mathbf{V}^T\tilde{\mathbf{S}}_B^{\Phi}\mathbf{V} = \mathbf{V}^T\tilde{\mathbf{S}}_T^{\Phi}\mathbf{V} = \mathbf{V}^T\mathbf{\Lambda}\mathbf{V} = \mathbf{Q}\tilde{\mathbf{\Lambda}}\mathbf{Q}^T \tag{26}$$

The final projection matrix of the kernel DCV method can be found as:

$$\mathbf{W} = \left(\mathbf{\Phi} - (1/n)\mathbf{\Phi}\mathbf{1}\right)\mathbf{U}\mathbf{\Lambda}^{-1/2}\mathbf{V}\mathbf{Q}. \tag{27}$$

There are, at most, $C - 1$ projection vectors. The final feature vector of a sample is realized as given in Eq. (16).

3. Experimental work

In this study, we first arranged a new database including the frontal images of 100 people taken under different lighting and posing conditions. We then performed face recognition experiments on our database to test the results of the feature extraction and embedding methods. For comparison, we also tested the recognition results of the methods on another database, namely the well-known AR face database [37]. We applied two different experimental designs in the preparation of the face images from the original frontal images for both databases. The first design was prepared to see the performances of the features and embedding methods under the assiduously preprocessed face images, a process that includes manual alignment of the face regions, whereas the second design was prepared to get an idea about their performances in real-world intelligent recognition applications, a process that includes automatic alignment of face regions. This section describes our experimental study in detail with the following organization: we introduce our database in the first subsection, define our face alignment designs in the second and third subsections, give the recognition experiments' setup with recognition results in the fourth subsection, and, finally, discuss our results in the fourth subsection.

3.1. ESOGU face database

The ESOGU face database includes 100 frontal images of 78 males and 22 females. The images were captured using a Sony DSC-H20 digital camera in two sessions with at least 10 days of time lag between the sessions.

In each session, a total of 9 poses were taken (6 of them with different facial expressions and 3 of them with different lighting conditions). As a result, the database includes 18 images per person, which makes a total of 1800 images.

The images, including facial expressions, were taken with a moderate amount of daylight illumination. To strain different illumination conditions, 3 extra Multiblitz 1000 W tungsten lights were located at the left, right, and front of the face and in turns they were turned on to generate different illuminations. The layout plan of the studio is given in Figure 3.

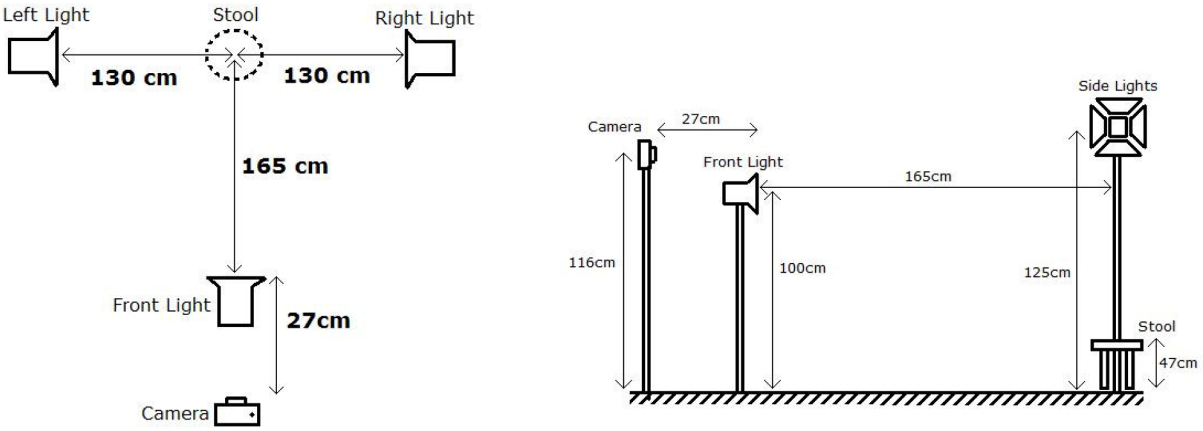
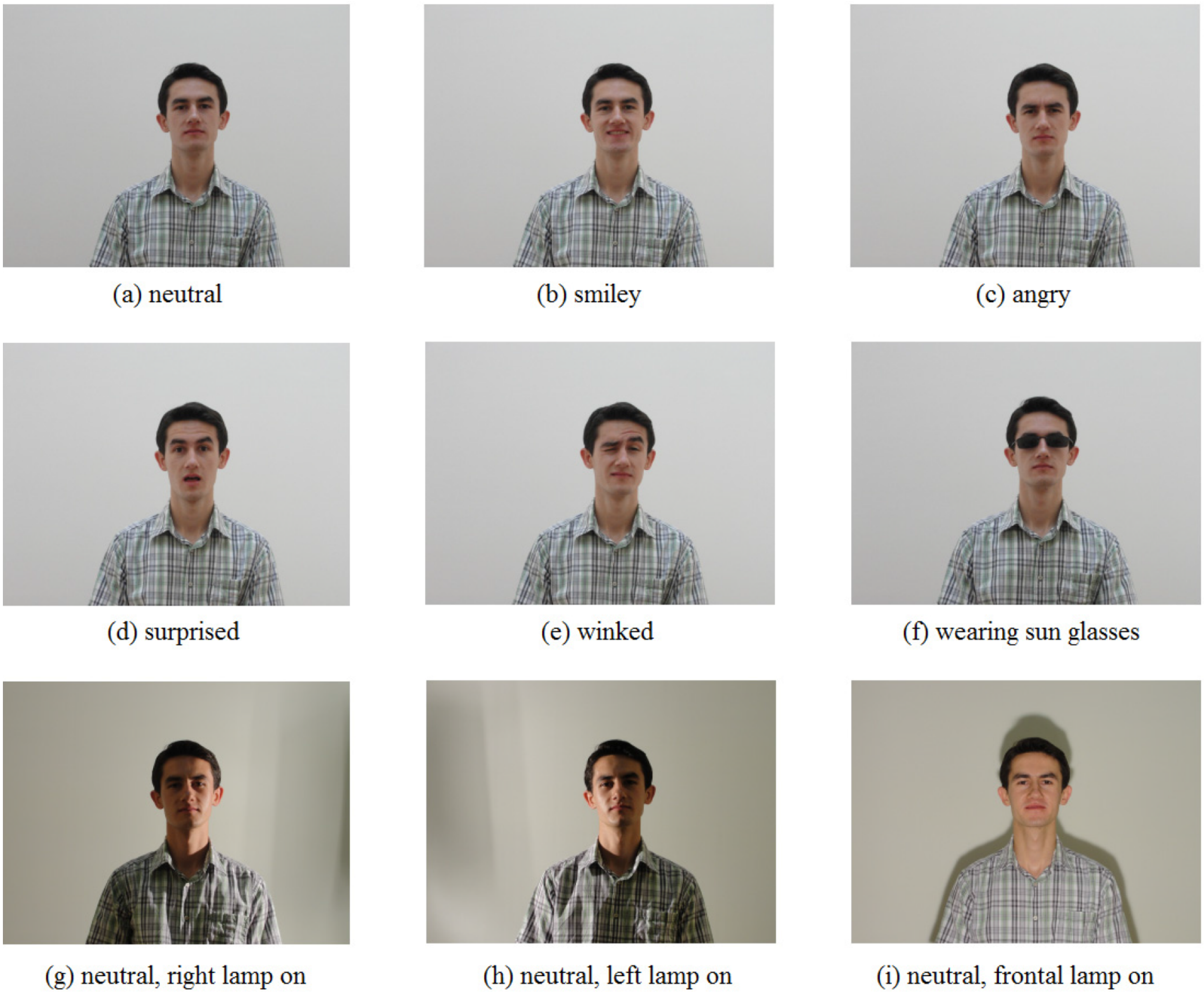


Figure 3. The layout plan of the studio.

A sample image set summarizing facial expressions for one person in a session is illustrated in Figure 4. Figure 4 shows (a) neutral, (b) smiling, (c) angry, (d) surprised, (e) winking, (f) wearing sunglasses, (g) neutral with the right lamp on, (h) neutral with the left lamp on, and (i) neutral with the frontal lamp on poses. The other session includes identical poses but with a time gap between them. We did not restrict the hair, beard, makeup, or clothing styles of the contributors. The resulting images were taken with a resolution of 72 × 72 dpi and a size of 2592 × 1944 pixels each.

3.2. Face alignment procedure 1

The recognition performance of the face recognition system is highly dependent on the proper alignment of the face images. As the first alignment procedure, we chose to use the most common method for the discrimination of the face image. In this method, the face region is carefully cropped from the frontal image in a standard way, which generally needs some human effort. In most of the studies in this area, the face region is cropped with a fixed rectangular window size such that some fiducial points, such as the eyes or mouth, are located at some special location in the window and the cropped region is used as the face image [1–4,18–26]. In this study, we prepared a rectangular window template in which the center of the left and right eyes of the people are positioned at some fixed coordinates, the input image is rotated and scaled, and the face region is cropped so that it exactly matches the face template. To this end, we manually annotated the eye coordinates of the original frontal image and made a standard linear conformal transformation to map the input image to the template. The transformation basically includes the rotation and scaling procedure. The rotation angle is determined by using the inclination of the line joining the two eyes and the scaling coefficient is determined by using the distance between the two eyes. Final cropping is realized by using the new coordinates of the eyes in the rotated and scaled image. The operation is summarized in Figure 5. First eye coordinates are annotated manually (shown in Figure 5a), and then the image is rotated and scaled (shown in Figure 5b) such that the final cropped image exactly matches the face template (shown in Figure 5c).

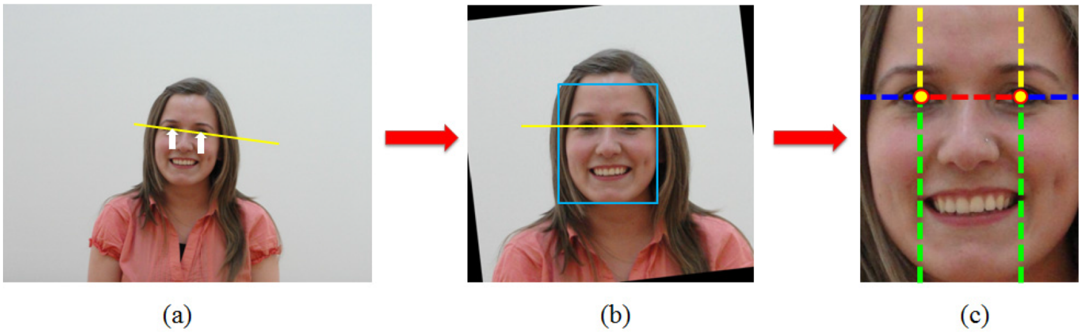


(a) neutral (b) smiley (c) angry

(d) surprised (e) winked (f) wearing sun glasses

(g) neutral, right lamp on (h) neutral, left lamp on (i) neutral, frontal lamp on

Figure 4. Posing variations of a person in the database.



(a) (b) (c)

Figure 5. Face alignment procedure based on the locations of the eyes.

3.3. Face alignment procedure 2

The previous procedure needs human effort to result in a very negligible normalization error. However, we can determine the face region and follow similar steps using automatic algorithms, which require no human effort. We call this procedure the automatic face alignment procedure. It uses the detection of the face region first, and after that, it searches the coordinates of the center of the eyes within the face region by using some special algorithms. We adopted this procedure to the study with the intention of using it in an automatic face recognition system.

In this study, we used two state-of-the-art face detection algorithms: the AdaBoost cascade face detector of Viola and Jones [7], and the cascade face detector of nearest convex model classifiers [9]. The AdaBoost face detector is a very popular publicly available algorithm. It consists of a computationally efficient cascade of weak classifiers that rapidly reject the background regions while keeping face image regions. Each classifier in the cascade is an AdaBoost ensemble of rectangular Haar-like features. The other face detector [9] was recently developed in our lab and we will briefly explain it below.

In contrast to other face detection algorithms that formulate the problem as a binary classification, the cascade face detector of the nearest convex model classifiers approach [9] focuses on approximating the regions spanned by the instances of a face class using a cascade of efficient convex class models. Once the regions are estimated as belonging to the face class, a feature vector extracted from an image region is classified (whether it is a face or background) based on its distances to the convex class models, which are arranged in a cascade structure. The method uses the sliding window approach during detection of faces. In this approach, each image is scanned from the top left to the bottom right at different scales by sliding a fixed size window, and each local window is classified as a face or background.

The cascade face detector of nearest convex model classifiers consists of three stages: the first stage of the cascade uses a combination of linear hyperplanes for approximation of face class, and the goal of this stage is to accept almost all of the face class samples and at the same time to reject a majority of the background samples rapidly. The second stage of the cascade includes the support vector data description classifier [38], which uses a linear hypersphere model for approximating face class. This stage is also fast and it works with the first stage in a complementary way by rejecting the majority of false positives accepted by that stage. The last stage of the cascade makes the final decision, and it uses a nonlinear hypersphere model for approximation of face class. This is the slowest stage, but it works only with a few promising samples accepted by the first two stages. In order to represent images, we have used a combined representation by concatenating the feature vectors obtained by using LBP and histograms of oriented gradients descriptors [39]. The proposed detector outperformed the AdaBoost cascade detector and some other face detectors on different face detection databases. A keen reader is referred to [9] for more details on this detector.

After the detection of the face region, we need to determine the coordinates of the center of the eyes of the face. This process is another challenging task [40]. For this purpose, we used one of the most efficient algorithms given in [41]. The automatic detection of the face region and the automatic determination of the eye coordinates are summarized in Figure 6. The first step of the procedure is the face detection as given in Figure 6a, and next the eyes are searched within the face region and the center coordinates of the eyes are found automatically as denoted by the red circles in Figure 6b. The same mapping of the previous section is then applied to match the face image to the face template given in Figure 6c. The difference between this procedure and the previous procedure is that the alignment of the face is realized without any human effort in this case. Note that any wrong determination of the coordinates of the eyes results in some error for matching the template exactly. By comparing the results given in Figures 5c and 6c, one can observe that the resultant face coordinates exactly match the template in Figure 5c, but there is a matching error in Figure 6c. This is because of the wrong determination of the left eye coordinate in the automatic detection of the eyes.

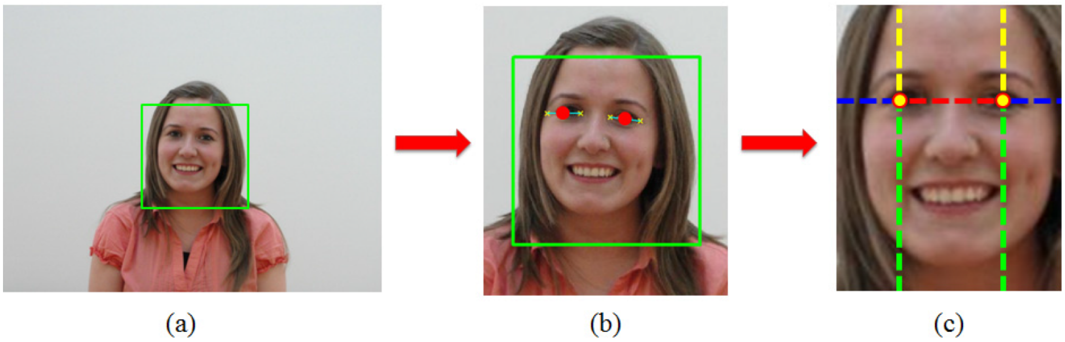


Figure 6. Automatic face alignment procedure.

3.4. Face recognition experiments

We performed face recognition experiments on the ESOGU face database and AR face database using both procedures given in the preceding two subsections. After the face image is aligned, in addition to the gray-level values of the face images, the features mentioned in Section 2.2 are extracted and they are used with the embedding methods mentioned in Section 2.3. As a whole, we have tested five different features, the gray-level values, the 2D Gabor filter features, LBP, LTP, and DCT features, with the 7 embedding methods: PCA (eigenfaces), LDA, Laplacianfaces, DCV, kernel PCA, kernel LDA, and kernel DCV. Experiments were simulated in MATLAB (ver. 7.9.0.529) and source codes are available at http://www2.ogu.edu.tr/∼mlcv/FaceRecog/esogumlcv_FaceRecogProgs.rar.

In the experiments, we need to divide the images in the database such that some randomly selected ones are used in the training and the rest are used in the test stages. To end up with a diffuse study, we tried a group of experiments including different numbers of samples in the training sets. For instance, the ESOGU face database includes 18 images per person and we tried the training set groups including $n_i = 3, 6, 9, 12, 15$ numbers of samples for each class with the rest of the $18 - n_i$ samples used in the test set. The AR face database includes 26 images per person and we tried the training set groups for $n_i = 5, 9, 13, 17, 21$ samples.

Random selection of samples has been fixed for all methods by constructing random index lists. In this procedure, the images are arranged in a way such that every image has an index number. We then constructed 20 different lists in which these index numbers were randomly mixed in order. For example, images for the ESOGU face database have index numbers from 1 to 18, but in the random list, the ordering is made randomly so that when the $n_i = 5$ case is examined, the first 5 samples are used in the training and the remaining 13 samples are used in the test sets. This standardization of random selection lists make the experiment comparable with numerous methods under the same conditions. Random selection lists are saved and any keen researcher can easily experiment with his/her methods using our experimental setup and database.

Some embedding methods used in the study have some parameters that needed to be previously defined. Therefore, we created a total of 20 random index lists in which we intended to use the first 10 runs for the parameter evaluation experiments and the next 10 runs to assess the performance. The final correct recognition rates are calculated as the average of the recognition rates of 10 runs. For the parameter evaluation case, the parameter that gives the maximum recognition rate is chosen in the experiments.

3.4.1. Experiments on the ESOGU face database with manually aligned face images

The ESOGU face database includes 1800 images of 100 people and we used all poses in the experiment. For memory considerations, we have chosen a face template of size 70 × 55 and have done the face alignment procedures according to this principle. Image samples of a person from the ESOGU database are shown in Figure 7. Images in row 1 are the poses of session 1 and images in row 2 are the poses of session 2.

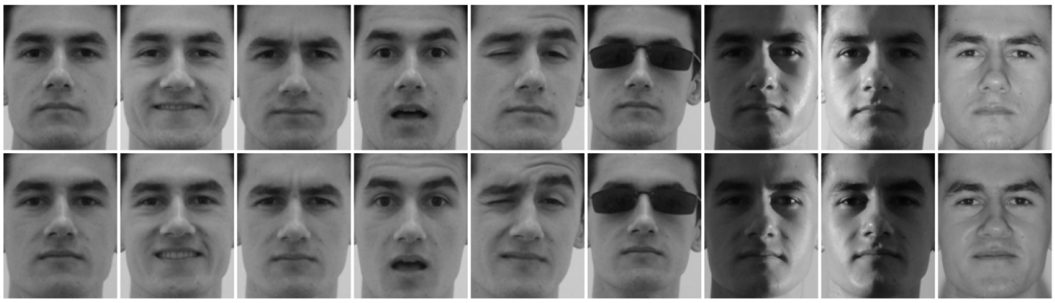


Figure 7. Resultant face images of a person from the ESOGU database where the face region is aligned with procedure 1.

In the experiments, the features are extracted as given in Section 2.2. Gray-level values are concatenated to form the gray-level feature vector of a sample $\mathbf{x}_{graylevel}^{ESOGU} \in \Re^{3850}$. LBP features are extracted using the 3 × 3 neighborhood of a pixel. The whole image is subdivided into 7 × 5 subwindows and the LBP code of each subwindow is extracted, and then they are concatenated, resulting in a final feature vector of $\mathbf{x}_{LBP}^{ESOGU} \in \Re^{2065}$. LTP features are composed of 2 channel LBPs and they are extracted in the same way as the LBP, resulting in a feature vector of $\mathbf{x}_{LTP}^{ESOGU} \in \Re^{4130}$. The DCT features result in a feature vector $\mathbf{x}_{DCT}^{ESOGU} \in \Re^{3150}$. The Gabor features are extracted using the parameters $\gamma = \eta = \frac{\sigma}{\sqrt{2\pi}}$ with the standard deviation of the Gaussian $\sigma = 2\pi$, orientation angles $\theta_k = \frac{k\pi}{8}$, $k = 0, 1, \cdots, 7$, and scalings with $f_a = \frac{f_{\max}}{2^{a/2}}$, $a = 0, 1, 2, 3, 4$, where $f_{\max} = \sqrt{2}$. The filter results of 5 different scales and 8 different orientations are concatenated, giving a huge vector. Finally, due to memory considerations, the resulting vector is downsampled in a feature vector of $\mathbf{x}_{Gabor}^{ESOGU} \in \Re^{9600}$.

The recognition results for manually aligned face images of the ESOGU database are given in Table 1, where n_i denotes the number of images used in the training set. The recognition rates are given in percentages, which denote the average rates of the 10 experiments with the standard deviation information near them. Values given in bold denote the embedding method that gives the maximum recognition rate, and the underlined values denote the feature that gives the maximum rate in each round of the experiments.

We have 5 features and 7 embeddings, and for each feature-embedding pair, we did recognition experiments for 5 different numbers of training samples that made a huge table. To compare the embedding methods, we concentrated on the maximum recognition rate in each row of Table 1. In the table, there are 26 maximum values. In 12 of 26, the kernel DCV; in 10 of 26, the kernel LDA; in 3 of 26, the DCV; and in 1 of 26, the Laplacianfaces method performed better than the others. The PCA, LDA, and kernel PCA did not give a best recognition rate in any round. In terms of the embedding method types, we can say that kernel methods give better results than linear methods in general. However, when we compare the results of linear and nonlinear methods separately, we can say that the DCV gives the best recognition rates among the linear methods, and kernel DCV and kernel LDA perform better than the Laplacianfaces and kernel PCA among the kernel methods.

We can compare the results of the different features to determine the best, similar to the previous case. When we consider the features giving the maximum recognition rates for each n_i, we see that the LBP features

Table 1. ESOGU face database recognition results (manually aligned images).

Feature	n_i	PCA		LDA		DCV		Laplacianface		Kernel PCA		Kernel LDA		Kernel DCV	
		(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ
Gray level	$n_i = 3$	49.31	10.93	69.03	7.09	76.45	7.13	68.02	7.15	49.68	10.79	75.10	6.40	**77.84**	6.55
	$n_i = 6$	65.29	7.00	85.69	4.82	88.03	5.31	85.93	5.76	65.55	7.12	88.35	5.14	**89.86**	5.17
	$n_i = 9$	72.64	7.29	88.31	5.06	90.28	4.94	90.03	5.28	72.93	7.35	91.02	5.16	**91.79**	4.62
	$n_i = 12$	75.17	10.22	91.50	6.38	93.80	5.60	94.32	5.76	75.37	10.14	95.13	5.96	**95.18**	5.25
	$n_i = 15$	76.63	16.38	93.07	4.49	95.47	3.11	97.07	1.98	76.83	16.18	**97.57**	1.89	97.10	2.50
LBP	$n_i = 3$	74.72	6.84	86.41	8.56	89.80	7.18	87.25	8.98	74.96	6.73	89.67	6.94	**89.93**	7.35
	$n_i = 6$	88.93	4.66	98.08	1.74	98.29	1.34	98.08	1.59	88.93	4.64	98.60	1.07	**98.63**	1.34
	$n_i = 9$	92.66	4.34	98.56	1.17	98.91	0.52	98.97	0.71	92.74	4.29	**99.29**	0.37	99.22	0.40
	$n_i = 12$	92.83	6.40	99.07	0.81	99.42	0.50	99.53	0.56	92.80	6.43	99.60	0.36	**99.62**	0.31
	$n_i = 15$	93.60	10.08	98.40	2.19	99.53	0.50	99.77	0.42	93.60	9.97	**99.80**	0.23	99.73	0.41
LTP	$n_i = 3$	62.04	12.07	85.69	7.24	**88.67**	6.69	86.89	7.30	62.43	12.03	88.13	6.58	88.61	6.59
	$n_i = 6$	80.73	6.95	97.16	2.09	97.58	1.79	97.04	2.27	80.97	7.00	97.57	1.67	**97.64**	1.58
	$n_i = 9$	86.89	6.33	98.14	1.80	98.41	1.42	98.18	1.50	87.14	6.35	**98.52**	1.63	98.47	1.45
	$n_i = 12$	87.92	9.39	98.55	2.14	99.02	1.40	**99.05**	1.46	88.23	9.51	98.88	2.00	98.98	1.74
	$n_i = 15$	88.70	15.01	98.77	3.78	**99.20**	2.18	99.17	1.95	88.93	15.08	**99.20**	2.30	99	2.53
DCT	$n_i = 3$	54.62	9.54	67.84	7.00	76.84	5.91	69.03	7.22	54.65	9.60	75.93	5.89	**77.45**	6.41
	$n_i = 6$	68.55	6.06	85.47	4.49	88.27	4.71	85.62	5.11	68.57	6.03	89.47	4.19	**89.86**	4.23
	$n_i = 9$	75.62	4.11	88.10	4.57	90.26	4.35	90.14	4.32	75.52	4.18	92.28	4.03	**92.53**	3.56
	$n_i = 12$	78.73	5.24	91.03	6.10	92.95	6.33	93.88	5.32	78.62	5.36	**95.65**	4.57	95.25	4.13
	$n_i = 15$	81.37	7.67	92.37	4.61	95.17	3.39	95.83	2.82	81.50	7.74	**97.37**	1.77	96.63	2.35
Gabor	$n_i = 3$	57.78	8.09	80.23	7.29	85.25	6.70	79.73	7.70	58.19	8.11	83.86	6.84	**85.33**	6.72
	$n_i = 6$	72.57	7.14	92.82	3.79	**95.02**	3.00	93.70	2.68	72.85	7.13	94.92	2.09	95.01	2.73
	$n_i = 9$	79.74	6.28	94.63	2.67	95.89	2.40	96.43	2.17	79.98	6.31	**96.70**	2.12	96.57	2.14
	$n_i = 12$	80.93	8.17	95.70	3.61	97.70	2.13	98.42	1.74	81.08	8.14	**98.47**	1.67	98.07	1.69
	$n_i = 15$	82.23	13.46	96.03	5.87	98.33	1.63	99.10	0.90	82.33	13.38	**99.13**	0.86	98.70	1.48

are superior to the others in all cases. LTP, Gabor, DCT, and gray-level features follow them, respectively. In fact, all features give higher than 90% recognition results when used with the best resulting embeddings and equal number of training and test sets, which can be accepted as evidence that they can perform well. However, the recognition rates of the LBP features are no doubt superior to the others, especially when used with the kernel LDA or kernel DCV methods.

Finally, when we consider the recognition results depending on just n_i, we can say that they have an increasing behavior. This implies that using more samples in the training results in better separable class models yielding high recognition rates, as expected.

3.4.2. Experiments on the AR face database with manually aligned face images

The AR face database [37] includes 26 poses per person taken under different lighting conditions and occlusions. Poses are taken in two different sessions. We randomly selected 50 people from the AR face database and used all poses in the experiment. We preprocessed the initial frontal images in a similar way to the previous case, resulting in a face image template size of 114 × 84. Face images of one person from the database are shown in Figure 8. Session 1 poses are given in the first row and session 2 poses are given in the second row of the figure.

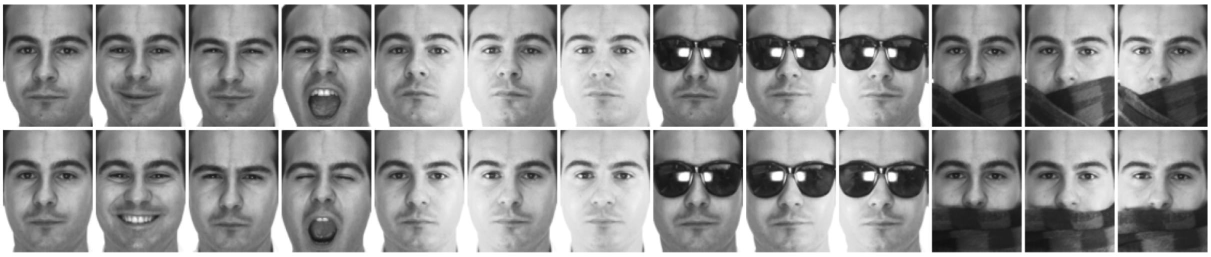


Figure 8. Resultant face images of a person from the AR database where the face region is aligned with procedure 1.

We used the same experimental setup mentioned at the beginning of the section. The features of the images in the AR face database are extracted in the same way as with the ESOGU face database. The resulting feature sizes can be summarized as: $\mathbf{x}^{AR}_{graylevel} \in \Re^{9576}$, $\mathbf{x}^{AR}_{LBP} \in \Re^{2065}$, $\mathbf{x}^{AR}_{LTP} \in \Re^{4130}$, $\mathbf{x}^{AR}_{DCT} \in \Re^{8250}$, $\mathbf{x}^{AR}_{Gabor} \in \Re^{12920}$. Note that because of subdividing an equal amount of windows, the final sizes of LBP and LTP feature vectors are the same as those in the ESOGU database, and because of having different template image sizes, the sizes of the rest of the features are different. The experimental results of the AR face database for manually aligned images are given in Table 2.

There are 25 rounds and 30 maximum values in Table 2. When we look at the table to compare the embedding methods, we see that the kernel DCV gives maximum recognition rates in 15, kernel LDA in 7, LDA in 5, DCV in 2, and Laplacianfaces in 1 of them. PCA and kernel PCA do not give a best result in any round. In linear methods, LDA performs better than DCV in this case, but their results are close to each other in general. In the kernel methods, kernel DCV performs better than kernel LDA, but again these two methods result in approximate values.

The AR face database is a widely used database in face recognition. In this study, we used a subset of the AR face database, but we can make comparisons with the results of some other studies. In order to make a fair comparison, the experimental setup and the data processing styles should be the same in all papers, but this is not possible. Therefore, an approximate comparison can be done with [42], which provides the results of many studies in a combined way. The results of similar embedding methods used in that study have been reported as averages, which look similar to the results found in this study.

Table 2. AR face database recognition results (manually aligned images).

Feature	n_i	PCA		LDA		DCV		Laplacianface		Kernel PCA		Kernel LDA		Kernel DCV	
		(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ
Gray level	$n_i = 5$	54.22	4.07	81.71	4.07	81.79	6.80	79.56	6.09	54.81	4.17	81.89	5.99	**83.48**	6.48
	$n_i = 9$	67.55	7.15	91.79	7.15	90.75	9.68	90.28	11.31	68.08	7.27	91.06	12.10	**92.25**	10.69
	$n_i = 13$	72.97	5.09	97.74	5.09	98.57	1.50	97.83	1.62	73.75	4.84	98.85	0.87	**99.14**	0.74
	$n_i = 17$	76.02	5.53	98.16	5.53	98.93	2.20	99.07	1.13	76.60	5.37	99.36	0.74	**99.47**	0.72
	$n_i = 21$	81.64	9.92	98.64	9.92	99.36	2.66	99.56	0.74	82.04	9.53	99.64	0.51	**99.72**	0.63
LBP	$n_i = 5$	89.12	3.39	**95.36**	2.76	93.20	3.75	94.24	2.97	89.37	3.35	93.47	3.63	93.81	3.63
	$n_i = 9$	94.02	4.65	**97.98**	3.05	97.21	4.34	97.56	3.10	94.01	4.61	97.52	4.10	97.79	3.76
	$n_i = 13$	96.88	2.27	99.60	0.44	99.65	0.38	99.57	0.43	97.00	2.19	**99.75**	0.41	99.69	0.34
	$n_i = 17$	98.18	1.18	99.80	0.34	99.84	0.21	99.82	0.33	98.27	1.10	**99.93**	0.15	99.91	0.16
	$n_i = 21$	98.04	1.77	99.92	0.17	99.88	0.19	99.88	0.19	98.16	1.65	**99.96**	0.13	**99.96**	0.13
LTP	$n_i = 5$	74.23	5.77	94.24	3.11	94.72	2.86	92.80	2.99	74.17	5.79	94.01	2.57	**94.83**	2.93
	$n_i = 9$	83.34	5.87	**98.65**	1.76	98.25	2.45	97.85	3.09	83.39	5.97	97.96	2.81	98.22	2.42
	$n_i = 13$	87.98	4.89	99.78	0.23	99.82	0.19	99.72	0.25	88.55	4.83	**99.83**	0.22	**99.83**	0.17
	$n_i = 17$	90.33	5.09	99.87	0.24	99.89	0.22	99.91	0.16	90.56	4.98	99.87	0.21	**99.96**	0.09
	$n_i = 21$	89.96	9.83	99.96	0.13	99.96	0.13	99.96	0.13	90.32	9.30	99.92	0.25	**100**	0.00
DCT	$n_i = 5$	73.75	7.53	83.12	5.52	**84.76**	6.19	80.60	5.36	74.07	7.42	83.72	5.68	84.50	6.17
	$n_i = 9$	81.35	9.37	91.81	7.81	92.05	9.34	90.67	8.54	81.65	9.20	91.56	8.79	**92.22**	8.93
	$n_i = 13$	87.17	4.10	97.09	1.96	97.65	1.66	97.02	2.00	87.38	4.05	97.77	1.61	**97.86**	1.33
	$n_i = 17$	90.07	4.46	97.49	2.90	98.07	2.24	97.29	3.10	90.04	4.50	98.02	2.17	**98.27**	1.96
	$n_i = 21$	92.36	4.35	98.28	3.40	98.92	2.35	98.56	3.04	92.44	4.50	98.76	2.43	**99.00**	2.20
Gabor	$n_i = 5$	72.67	4.85	90.99	4.75	85.03	8.53	**91.38**	3.74	73.38	4.93	84.02	8.16	85.79	8.20
	$n_i = 9$	81.22	8.27	**97.32**	3.60	94.39	8.78	95.79	6.43	81.67	8.24	93.49	10.32	94.95	7.93
	$n_i = 13$	86.89	4.15	99.77	0.20	99.71	0.23	99.58	0.30	87.42	4.00	**99.78**	0.22	99.71	0.21
	$n_i = 17$	88.93	4.36	99.73	0.39	99.84	0.24	99.80	0.22	89.33	4.24	**99.91**	0.16	99.82	0.23
	$n_i = 21$	90.84	5.60	**99.96**	0.13	**99.96**	0.13	99.88	0.27	91.20	5.76	**99.96**	0.13	**99.96**	0.13

In terms of the feature types, LTP in this case generally results in highly recognizable features. Only when $n_i = 5$ do LBP features give the maximum recognition result. In other cases $(n_i = 9, 13, 17, 21)$, LTP achieves better recognition rates. In one case, when $n_i = 21$ and the kernel DCV is used as the embedding method, complete recognition with no error is achieved with LTP features.

3.4.3. Experiments on the ESOGU face database with automatically aligned face images

We conducted recognition experiments by using the face images obtained with the automatic alignment procedure of the frontal images given in Section 3.3. In this procedure, the face is mapped to the template automatically using the aforementioned detection algorithms. Faces that are not detected are left blank and labeled with an extra class to discriminate them from the proper detections. After that, the same experimental procedure is applied with the important exception that the face images that were not detected are accepted as wrong classifications as a default. To this end, we use the first n_i correctly detected face images in the training, and the rest of the images are used in the test phases. If there is any misdetected image in the test, it is directly assigned as a wrong recognition at the end.

For the ESOGU face database, 68 images of 1800 images are not detected. The initial correct detection rate is thus 96.22%. This is not a bad result at all. Misdetection generally occurs in poses where we expose extra illumination, especially from the side. Rather than the face detection, the determination of the eye coordinates is another crucial issue in the automatic alignment procedure. An improper determination of the eye coordinates results in a wrong mapping of the face to the template, which has a great influence in recognition performance. Because the operation is made by automatic algorithms, we notice some mistakes in the final face images that are automatically aligned. Some automatically aligned face samples from the ESOGU database are shown in Figure 9. The black images in the figure represent the faces that are not detected. Scaling and shifting differences of the images in the figure are because of the wrong determination of the exact coordinates of the center of eyes. By way of illustration, we include some good and some bad examples in the figure.

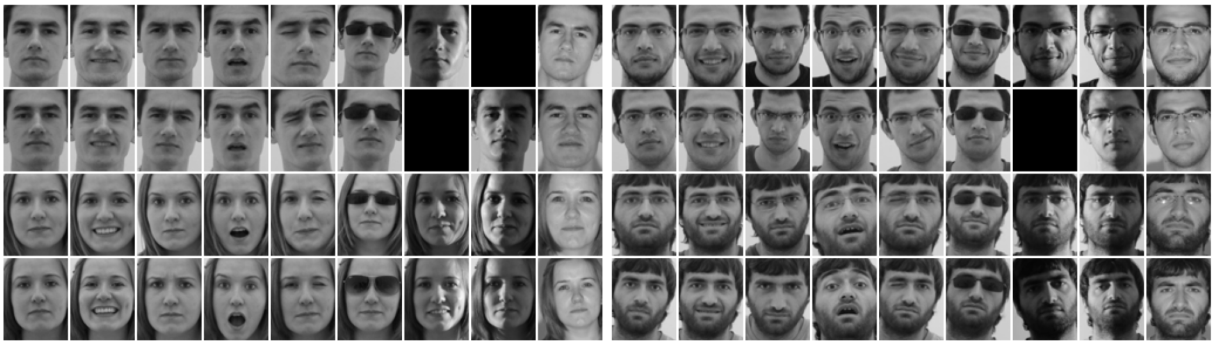


Figure 9. Some face images from the ESOGU database in which the face region is aligned automatically using procedure 2.

Recognition experiments on the automatically aligned faces of the ESOGU database were performed and the results are given in Table 3. In the setup, misdetected sample indices are used at the end of the index with an ordering such that the faces that were not detected are used at the end of the test set list. There are at most 4 misdetections for a person in the database, so we used $n_i = 3, 6, 9, 12$ samples in the training and the other $15, 12, 9, 6$ respective samples in the testing to ensure that the faces not detected are used in the test phase such that we can label them as a wrong recognition.

Table 3. ESOGU face database recognition results (automatically aligned images).

Feature	n_i	PCA		LDA		DCV		Laplacianface		Kernel PCA		Kernel LDA		Kernel DCV	
		(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ
Gray level	$n_i = 3$	34.17	9.79	37.07	5.53	49.87	4.92	37.30	5.81	33.90	9.71	50.96	5.25	**52.07**	5.55
	$n_i = 6$	47.14	4.75	53.68	2.65	63.20	2.92	59.85	3.35	46.96	4.89	**70.00**	3.72	69.63	3.80
	$n_i = 9$	53.03	5.04	57.83	4.10	66.62	5.08	67.38	5.04	52.87	5.02	**75.04**	5.42	74.26	5.18
	$n_i = 12$	54.25	5.76	58.18	5.90	67.77	6.55	70.92	6.08	54.15	5.67	**77.23**	4.45	75.78	4.96
LBP	$n_i = 3$	56.97	10.94	71.07	7.66	77.05	6.47	73.55	7.73	56.97	10.97	**77.09**	6.35	77.07	6.44
	$n_i = 6$	72.22	4.72	86.90	2.12	88.78	1.71	86.70	2.11	72.19	4.77	89.39	1.63	**89.63**	1.66
	$n_i = 9$	76.28	2.88	86.94	1.94	88.26	1.73	87.70	1.57	76.31	2.88	**89.41**	1.20	89.21	1.37
	$n_i = 12$	75.18	2.67	83.73	1.96	85.83	1.18	85.80	1.28	75.20	2.82	**87.08**	0.67	86.83	0.95
LTP	$n_i = 3$	49.57	11.98	73.80	6.21	**79.29**	5.99	76.23	6.76	49.10	11.96	78.20	6.26	79.25	5.94
	$n_i = 6$	65.75	5.30	89.13	1.89	90.18	1.80	88.96	1.95	65.42	5.46	89.98	1.66	**90.28**	1.76
	$n_i = 9$	70.66	4.37	88.94	2.11	89.69	1.63	89.36	1.84	70.46	4.29	**89.86**	1.55	89.68	1.89
	$n_i = 12$	69.75	5.49	86.05	2.07	86.73	1.39	86.47	1.51	69.73	5.15	**87.07**	1.19	86.77	1.51
DCT	$n_i = 3$	30.32	8.04	32.08	6.09	42.82	5.19	33.63	5.49	30.31	8.07	42.34	6.12	**44.00**	6.22
	$n_i = 6$	42.13	5.29	50.31	3.14	58.13	2.53	51.72	3.66	42.14	5.30	61.18	2.99	**62.53**	3.49
	$n_i = 9$	49.07	4.30	56.30	3.93	63.97	4.57	61.84	4.59	49.03	4.27	69.19	5.10	**69.78**	4.78
	$n_i = 12$	51.75	4.10	56.13	6.00	64.95	5.78	66.05	5.37	51.80	4.15	**72.37**	5.68	72.20	5.30
Gabor	$n_i = 3$	45.35	9.65	65.99	7.06	73.11	6.06	65.89	7.26	45.09	9.80	71.97	6.23	**73.17**	6.19
	$n_i = 6$	58.99	5.51	83.33	3.41	85.91	3.43	83.96	3.31	58.77	5.92	86.45	2.95	**86.74**	3.16
	$n_i = 9$	64.07	5.17	84.06	2.86	86.48	2.52	85.94	2.52	64.22	5.23	87.74	2.13	**87.82**	1.88
	$n_i = 12$	64.47	5.40	81.28	3.83	84.58	2.27	84.88	1.88	64.82	5.04	**85.65**	1.42	85.63	1.57

In this case, there are 20 rounds of recognition experiments. By looking at the results in terms of the embedding methods, kernel LDA gives 10, kernel DCV gives 9, and DCV gives 1 maximum recognition rates over 20 rounds. LDA and Laplacianfaces did not result in a maximum recognition rate in any turn, but they generally resulted in approximate values to the leading methods. However, PCA among the linear methods and kernel PCA among the kernel methods generally give the worst recognition rates.

From the feature aspect, considering the maximum recognition rate giving embeddings, LTP serves better features for $n_i = 3, 6, 9$ cases and LBP, with just a slightly higher recognition rate than LTP, serves better features for the $n_i = 12$ case. Gabor features follow the leading features, and the gray-level and DCT features give worse values than the others.

3.4.4. Experiments on the AR face database with automatically aligned face images

Similar recognition experiments were performed on the AR face database using the automatically aligned face images. For this database, the correct face detection rate is 91.38%, which is lower than that of the ESOGU database. The AR face database includes more poses with lighting differences and extra occlusions than the ESOGU database, and misdetections generally occur in the occluded images under differently lit conditions. Some examples of the automatically aligned faces of the AR database are given in Figure 10.

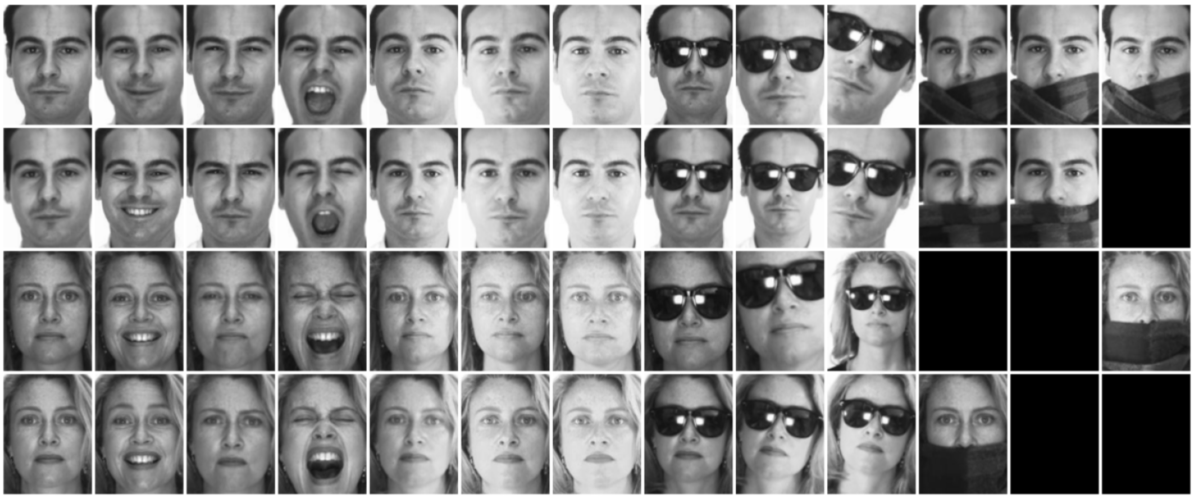


Figure 10. Some face images from the AR database in which the face region is aligned automatically using procedure 2.

Recognition results on the automatically aligned faces of the AR database are given in Table 4. Similar to the previous case, the face images that were not detected are used in the test phase and accepted as the wrong recognition as a result. In this database, there are at most 6 misdetections for a person at the end, so we used $n_i = 5, 9, 13, 17$ samples in the training and the remaining $9, 13, 17, 21$ respective samples in the test.

There are 20 rounds of recognition experiments in this case. By looking at the recognition results in terms of the embedding methods, we see that the kernel DCV method outperforms the others in 18 rounds. LDA in one round and LDA and DCV in another give the maximum recognition rate. When we consider the feature extraction methods, the LTP features reach the maximum recognition rates in all turns. The LBP, Gabor, gray-level, and DCT results follow those of the LTP.

We can easily notice one more fact, which is that the recognition rates in the AR face database drastically decrease when the automatic alignment procedure is used. The first reason for this conclusion is the high

Table 4. AR face database recognition results (automatically aligned images).

Feature	n_i	PCA		LDA		DCV		Laplacianface		Kernel PCA		Kernel LDA		Kernel DCV	
		(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ	(%)	σ
Gray level	$n_i = 5$	36.19	6.26	54.30	3.03	58.66	2.38	50.69	3.41	35.70	6.21	58.56	2.91	**59.53**	2.90
	$n_i = 9$	42.54	3.49	58.06	4.28	60.98	5.22	57.76	4.30	42.05	3.57	63.88	5.68	**64.29**	5.84
	$n_i = 13$	43.48	5.34	56.08	6.47	59.46	6.41	58.05	5.60	42.95	5.21	63.85	5.48	**64.32**	5.64
	$n_i = 17$	38.87	7.22	53.36	9.64	57.11	8.97	56.29	8.51	38.53	7.20	61.07	7.07	**61.16**	6.88
LBP	$n_i = 5$	51.63	4.79	72.72	1.46	73.24	2.17	70.15	1.63	51.55	4.73	73.38	1.98	**73.63**	2.15
	$n_i = 9$	56.88	1.72	75.04	2.82	76.09	2.86	74.64	2.84	56.85	1.78	76.59	2.65	**76.65**	2.71
	$n_i = 13$	57.63	3.82	73.82	3.36	74.55	3.47	73.49	3.78	57.57	3.79	75.18	3.24	**75.23**	3.27
	$n_i = 17$	55.02	5.45	69.11	3.24	70.18	3.04	69.36	3.56	55.00	5.40	70.44	2.77	**70.53**	2.75
LTP	$n_i = 5$	51.93	6.83	74.70	2.37	**76.20**	2.51	71.81	2.03	51.93	6.83	75.74	2.04	76.19	2.14
	$n_i = 9$	57.78	3.16	77.16	2.89	78.36	2.55	76.46	2.42	57.79	3.18	78.25	2.33	**78.38**	2.48
	$n_i = 13$	58.72	4.51	75.55	3.35	76.43	2.91	75.17	2.82	58.74	4.53	76.38	2.57	**76.58**	2.58
	$n_i = 17$	54.84	6.18	70.91	2.85	71.07	2.81	70.18	2.90	54.82	6.15	71.16	2.79	**71.40**	2.43
DCT	$n_i = 5$	40.93	4.94	47.64	4.37	54.06	2.71	44.85	3.93	40.50	5.18	53.42	3.19	**54.57**	2.94
	$n_i = 9$	45.60	4.69	54.29	3.96	57.96	4.07	53.24	4.43	44.99	4.43	58.29	4.74	**59.26**	4.03
	$n_i = 13$	47.38	5.52	53.49	6.30	56.32	6.46	52.95	5.90	46.94	5.39	58.58	6.46	**59.17**	6.85
	$n_i = 17$	45.89	8.06	51.58	10.45	54.78	9.81	51.73	9.77	45.18	8.05	56.51	8.85	**56.91**	8.87
Gabor	$n_i = 5$	50.10	4.97	**70.04**	2.97	69.51	6.12	68.55	1.96	50.08	4.83	68.45	5.03	69.34	5.94
	$n_i = 9$	54.40	4.07	74.51	4.17	74.95	4.16	73.41	3.24	54.53	4.04	73.71	4.05	**74.95**	4.20
	$n_i = 13$	54.74	5.29	73.38	4.04	74.00	3.76	72.52	3.81	54.74	5.34	73.32	3.58	**74.05**	3.72
	$n_i = 17$	52.31	7.96	68.80	4.09	69.04	3.84	68.02	4.24	52.27	7.95	68.36	4.39	**69.62**	3.54

detection error due to lighting differences. The second reason is the automatic annotation error, i.e., an improper determination of the center of the eye coordinates in automatic algorithms. One more reason may be the high number of occluded images in the database. When combined with the detection and annotation errors, occluded images of people may be hard to distinguish. The best result was accomplished as 78% using the LTP features and kernel DCV embedding with $n_i = 9$ for the automatically aligned faces. However, when used with the automatically aligned faces, it was around 98%, which clearly points to automatic annotation errors.

3.5. Discussion

In this study, we conducted face recognition experiments to compare the recognition performances of the different types of features and embedding methods for face recognition. We used two databases, and for both of them we used two types of face images: manually aligned faces and automatically aligned faces.

In the experiments with the ESOGU face database with manually aligned images, the kernel DCV and the kernel FLDA are the leading embedding methods and LBP features are the leading feature types. For the AR face database with manually aligned images, the kernel DCV is better among the embeddings and the LTP features are better among the feature types. For the automatically aligned faces, we get similar conclusions in that the LBP and LTP features are better than the other features, whereas kernel DCV and kernel LDA are better than the other embedding methods in general.

The ordering of the features can be given as follows: LTP and LBP features generally perform with maximum recognition rates, Gabor features follow them with similar results, and the DCT and gray-level features are generally at the end of the list. Although LBP features are mostly used in the area of texture classification, we have seen that they also perform well in face recognition. The computational cost of LBP and LTP features is not high, and they also result in a lower-dimensional feature vector in contrast to the gray-level values. Gabor features give good recognition results as well, but they require a costly computation that could slow down the recognition procedure, and the resultant Gabor feature has a generally higher dimension than the gray-level features. Usage of DCT can improve the recognition performance compared to the gray-level values in some cases, but they generally give similar results. From this spot, we can separate LBP and LTP features as better feature types from the others by having a lower dimensionality and higher recognition results.

From the embedding methods side, the kernel DCV and the kernel LDA generally give higher recognition rates than the Laplacianfaces and the kernel PCA among the kernel methods. For the properly aligned images, they can give approximately fully correct recognition when a large number of images are used in the training. Similarly, the linear versions of these two methods, LDA and DCV, give better recognition rates among the linear methods. We should draw attention to the fact that the PCA and the kernel PCA give the worst recognition rates in all cases, which is not a surprising conclusion at all. To clarify, even though the LDA- and DCV-based methods give approximate recognition results, the DCV is one step further from the LDA in giving higher recognition rates. The linear DCV is also a computationally efficient embedding method in which there is no additional parameter that needs to be tuned. It guarantees full recognition accuracy for the training samples, whereas the other embeddings may or may not result in full recognition accuracy in the training samples.

Recognition rates due to automatically aligned faces are generally lower than the recognition rates of the manually aligned face images. There are 2 basic reasons for this situation: the detection and the annotation errors. When a human annotates the face and eye coordinates, he generally makes a negligible amount of error, but when the face and the eye coordinates are determined automatically with the computer algorithms, they may result in large errors that affect the consequence. Even though we used up-to-date detection techniques in this study, we have seen that the recognition rates decrease considerably when we try the automatic alignment procedure.

The automatic alignment procedure of the faces can be improved by adopting extra image processing techniques and retraining the detectors with some samples from the databases. We used the detectors with their intrinsically trained parameters to be objective in this study. However, we encountered misdetections, especially in the images including illumination differences. This problem can be solved by adding samples including such face images and retraining the detector system. Proper localization of the eyes can also be achieved by introducing an extra image processing procedure to minimize the error, but this would consume extra time.

The experiments also demonstrate that the proper alignment of the face improves the performance of the system in general. For future considerations, to be used in a real-time face recognition system, the alignment procedure should be taken into account to achieve good recognition performance. When the alignment is realized with zero or a very small number of errors, the recognition results would be at high levels.

4. Conclusion

The face recognition problem is a difficult one because it includes many complications, such as the variety of the head or mimic poses, illumination differences, shape, makeup, and occlusion effects. Face recognition methods have two steps in general: the feature extraction and the classification. In this paper, we provided an extensive comparison of the state-of-the-art embedding methods used in classification with some extra feature extraction techniques in face representation. In order to test the performances, we created a new face database and conducted face recognition experiments using our database and another well-known AR face database. The experimental results demonstrated that the adaptation of some features other than the gray-level values can result in better recognition rates. In general, LBP and LTP features are better among the feature types. The LBP and LTP features are first used in the texture classification, but in this study we have seen that they also performed well in the face recognition area. From perspective of embedding methods, kernel methods generally improve recognition performance. Among them, the kernel DCV is the leading embedding method, followed by the kernel LDA with close results. The Laplacianfaces also give approximate results, but the kernel PCA does not give comparative results to those of the others. Similar ordering is experienced in linear embedding methods as well; the DCV gives better results than the LDA in general, but they both give close recognition rates. However, the PCA gives the worst recognition rates of all.

In addition to the feature types and the embedding methods, we have also demonstrated in this study that the proper alignment of the faces has an important influence on the performance of the recognition system. Manually aligned faces result in better recognition rates than the automatically aligned faces do. This highlights the importance of an error-free alignment of faces, implying that researchers should put an emphasis on the face alignment procedure, which is generally neglected in a majority of the studies. As a future study, the improvement of automatic face alignment procedures can be considered to be used with a full automatic recognition system.

Acknowledgment

This work was supported by Eskişehir Osmangazi University under the scientific research project with project number 200915015, Turkey.

References

[1] Zhao W, Chellappa R, Phillips PJ, Rosenfeld A. Face recognition: a literature survey. ACM Comput Surv 2003; 35: 399–458.

[2] Jafri R, Arabnia HR. A survey of face recognition techniques. Journal of Information Processing Systems 2009; 5: 41–68.

[3] Zhang X, Gao Y. Face recognition across pose: a review. Pattern Recogn 2009; 42: 2876–2896.

[4] Zhao W, Chellappa R. Image-based face recognition: Issues and methods. In: Javidi B, editor. Image Recognition and Classification: Algorithms, Systems, and Classifications. New York, NY, USA: Marcel Dekker Inc., 2002. pp. 375–402.

[5] Hietmeyer R. Biometric identification promises fast and secure processing of airline passengers. International Civil Aviation Organization Journal 2000; 55: 10–11.

[6] Yang MH, Kriegman DJ, Ahuja N. Detecting faces in images: a survey. IEEE T Pattern Anal 2002; 24: 34–58.

[7] Viola P, Jones MJ. Robust real-time face detection. Int J Comput Vision 2004; 57: 137–154.

[8] Levi K, Weiss Y. Learning object detection from a small number of examples. In: International Conference on Computer Vision and Pattern Recognition; 27 June–2 July 2004. New York, NY, USA: IEEE. pp. 53–60.

[9] Cevikalp H, Triggs B. Efficient object detection using cascades of nearest convex model classifiers. In: International Conference on Computer Vision and Pattern Recognition; 16–21 June 2012. New York, NY, USA: IEEE. pp. 3138–3145.

[10] Zhang BL, Zhang H, Ge SS. Face recognition by applying wavelet subband representation and kernel associative memory. IEEE T Neural Networ 2004; 15: 166–177.

[11] Shen L, Bai L. A review on Gabor wavelets for face recognition. Pattern Anal Appl 2006; 9: 273–292.

[12] Tao D, Li X, Maybank SJ. General tensor discriminant analysis and Gabor features for gait recognition. IEEE T Pattern Anal 2007; 29: 1700–1715.

[13] Hafed ZM, Levine MD. Face recognition using the discrete cosine transform. Int J Comput Vision 2001; 43: 167–188.

[14] Ekenel HK, Stiefelhagen R. Local appearance based face recognition using discrete cosine transform. In: European Signal Processing Conference; September 2005.

[15] Ahonen T, Hadid A, Pietikainen M. Face description with local binary patterns: application to face recognition. IEEE T Pattern Anal 2006; 28: 2037–2041.

[16] Tan X, Triggs B. Enhanced local texture feature sets for face recognition under difficult lighting conditions. IEEE T Image Process 2010; 19: 1635–1650.

[17] Turk M, Pentland AP. Eigenfaces for recognition. J Cognitive Neurosci 1991; 3: 71–86.

[18] Cevikalp H, Neamtu M, Wilkes M, Barkana A. Discriminative common vectors for face recognition. IEEE T Pattern Anal 2005; 27: 4–13.

[19] Belhumeur PN, Hespanha JP, Kriegman DJ. Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. IEEE T Pattern Anal 1997; 19: 711–720.

[20] Swets DL, Weng J. Using discriminant eigenfeatures for image retrieval. IEEE T Pattern Anal 1996; 18: 831–836.

[21] Cevikalp H, Neamtu M, Wilkes M. Discriminative common vector method with kernels. IEEE T Neural Networ 2006; 17: 1550–1565.

[22] He X, Yan S, Hu Y, Niyogi P, Zhang HJ. Face recognition using Laplacianfaces. IEEE T Pattern Anal 2005; 27: 328–340.

[23] Mika S, Ratsch G, Weston J, Scholkopf B, Muller KR. Fisher discriminant analysis with kernels. In: Proceedings of the 1999 IEEE Signal Processing Society Workshop; 23–25 August 1999. New York, NY, USA: IEEE. pp. 41–48.

[24] Baudat G, Anouar F. Generalized discriminant analysis using kernel approach. Neural Comput 2000; 12: 2385–2404.

[25] Kim KI, Jung K, Kim HJ. Face recognition using kernel principal component analysis. IEEE Signal Proc Let 2002; 9: 40–42.

[26] Yang MH. Kernel eigenfaces vs. kernel Fisherfaces: face recognition using kernel methods. In: International Conference on Automatic Face and Gesture Recognition; 21 May 2002. New York, NY, USA: IEEE.

[27] Martinez AM. Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class. IEEE T Pattern Anal 2002; 24: 748–763.

[28] Marcelja S. Mathematical description of the responses of simple cortical cells. J Opt Soc Am 1980; 70: 1297–1300.

[29] Daugman JG. Two-dimensional spectral analysis of cortical receptive field profile. Vision Res 1980; 20: 847–856.

[30] Lee TS. Image representation using 2D Gabor wavelets. IEEE T Pattern Anal 1996; 18: 1–13.

[31] Ojala T, Pietikainen M, Harwood D. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. In: Proceedings of the 12th IAPR International Conference on Pattern Recognition; 9–13 October 1994. New York, NY, USA: IEEE. pp. 582–585.

[32] Gonzalez RC, Woods RE. Digital Image Processing. 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2002.

[33] Chen Y, Jiang S, Abraham A. Face recognition using DCT and hybrid flexible neural tree. In: International Conference on Neural Networks and Brain; 13–15 October 2005. New York, NY, USA: IEEE. pp. 1459–1463.

[34] Scholkopf B, Smola A, Muller K. Kernel principal component analysis. Lect Notes Comput Sc 1997; 1327: 583–588.

[35] Çevikalp H. Feature extraction techniques in high-dimensional spaces: Linear and nonlinear approaches. PhD, Vanderbilt University, Nashville, TN, USA, 2005.

[36] Yang J, Frangi AF, Jin Z, Yang J. Essence of kernel Fisher discriminant: KPCA plus LDA. Pattern Recogn 2004; 37: 2097–2100.

[37] Martinez AM, Benavente R. The AR Face Database. CVC Technical Report #24. Barcelona, Spain: Computer Vision Center, 1998.

[38] Tax DMJ, Duin RPW. Support vector data description. Mach Learn 2004; 54: 45–66.

[39] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: International Conference on Computer Vision and Pattern Recognition; 25 June 2005. New York, NY, USA: IEEE. pp. 886–893.

[40] Cevikalp H, Yavuz HS, Edizkan R, Gunduz H, Kandemir CM. Comparisons of features for automatic eye and mouth localization. In: 2011 International Symposium on Innovations in Intelligent Systems and Applications; 15–18 June 2011. New York, NY, USA: IEEE. pp. 576–580.

[41] Everingham M, Sivic J, Zisserman A. "Hello! My name is... Buffy" – Automatic naming of characters in TV video. In: Proceedings of the British Machine Vision Conference; 2006.

[42] Abate AF, Nappi M, Riccio D, Sabatino G. 2D and 3D face recognition: a survey. Pattern Recogn Lett 2007; 28: 1885–1906.