Interpolation Between Eigenspaces Using Rotation in Multiple Dimensions

Tomokazu Takahashi[1,2], Lina[1], Ichiro Ide[1],
Yoshito Mekada[3], and Hiroshi Murase[1]

[1] Graduate School of Information Science, Nagoya University, Japan
ttakahashi@murase.m.is.nagoya-u.ac.jp
[2] No Japan Society for the Promotion of Science
[3] Department of Life System Science and Technology Chukyo University, Japan

Abstract. We propose a method for interpolation between eigenspaces. Techniques that represent observed patterns as multivariate normal distribution have actively been developed to make it robust over observation noises. In the recognition of images that vary based on continuous parameters such as camera angles, one cause that degrades performance is training images that are observed discretely while the parameters are varied continuously. The proposed method interpolates between eigenspaces by analogy from rotation of a hyper-ellipsoid in high dimensional space. Experiments using face images captured in various illumination conditions demonstrate the validity and effectiveness of the proposed interpolation method.

1 Introduction

Appearance-based pattern recognition techniques that represent observed patterns as multivariate normal distribution have actively been developed to make them robust over observation noises. The subspace method [1] and related techniques [2,3] enable us to achieve accurate recognition under conditions where such observation noises as pose and illumination variations exist. Performance, however, degrades when the variations are far larger than expected. On the other hand, the parametric eigenspace method [4] deals with variations using manifolds that are parametric curved lines or surfaces. The manifolds are parameterized by parameters corresponding to controlled pose and illumination conditions in the training phase. This enables object recognition and at the same time parameter estimation that estimates pose and illumination parameters when an input image is given. However, this method is not very tolerant of uncontrolled noises that are not parameterized, e.g., translation, rotation, or motion blurring of input images.

Accordingly, Lina et al. have developed a method that embeds multivariate normal density information in each point on the manifolds [5]. This method generates density information as a mean vector and a covariance matrix from training images that are degraded by artificial noises such as translation, rotation, or motion blurring. Each noise is controlled by a noise model and its

Y. Yagi et al. (Eds.): ACCV 2007, Part II, LNCS 4844, pp. 774–783, 2007.

parameter. To obtain density information between consecutive poses and generate smooth manifolds, the method interpolates training images degraded by the identical noise model and the parameter between consecutive poses. By considering various other observation noises, however, controlling noises by model and parameter is difficult; therefore, making correspondence between training images is not realistic. Increasing computational cost with a growing number of training images is also a problem.

In light of the above background, we propose a method to smoothly interpolate between eigenspaces by analogy from rotation of a hyper-ellipsoid in a high dimensional space. Section 2 introduces the mathematical foundation, the interpolation of a rotation matrix using diagonalization and its geometrical significance, followed by Section 3, where the proposed interpolation method is described. Section 4 demonstrates the validity and effectiveness of interpolation by the proposed method from experiment results using face images captured in various illumination conditions. Section 5 summarizes the paper.

2 Interpolation of Rotation Matrices in an n-Dimensional Space

2.1 Diagonalization of a Rotation Matrix

An $n \times n$ real number matrix ${}_n\boldsymbol{R}$ is a rotation matrix when it satisfies the following conditions:

$$ {}_n\boldsymbol{R}\,{}_n\boldsymbol{R}^T = {}_n\boldsymbol{R}^T\,{}_n\boldsymbol{R} = {}_n\boldsymbol{I}, \qquad \det({}_n\boldsymbol{R}) = 1, \tag{1} $$

where $\boldsymbol{A}^T$ represents a transpose matrix of $\boldsymbol{A}$ and ${}_n\boldsymbol{I}$ represents an $n \times n$ identity matrix. ${}_n\boldsymbol{R}$ can be diagonalized with an $n \times n$ unitary matrix and a diagonal matrix ${}_n\boldsymbol{D}$ including complex elements as

$$ {}_n\boldsymbol{R} = {}_n\boldsymbol{U}\,{}_n\boldsymbol{D}\,{}_n\boldsymbol{U}^\dagger. \tag{2} $$

Here, $\boldsymbol{A}^\dagger$ represents a complex conjugate transpose matrix of $\boldsymbol{A}$. The following equation is obtained for a real number x:

$$ {}_n\boldsymbol{R}^x = {}_n\boldsymbol{U}\,{}_n\boldsymbol{D}^x\,{}_n\boldsymbol{U}^\dagger. \tag{3} $$

${}_n\boldsymbol{R}^x$ represents an interpolated rotation when $0 \le x \le 1$ and an extrapolated rotation in other cases. This means that once $\boldsymbol{U}_n$ is calculated, the interpolation and extrapolation of ${}_n\boldsymbol{R}$ can be easily obtained.

2.2 Geometrical Significance of Diagonalization

A two-dimensional rotation matrix ${}_2\boldsymbol{R}(\theta)$ whose $\theta(-\pi < \theta \le \pi)$ is its rotation angle can be diagonalized as

$$ {}_2\boldsymbol{R}(\theta) = {}_2\boldsymbol{U}\,{}_2\boldsymbol{D}(\theta)\,{}_2\boldsymbol{U}^\dagger, \tag{4} $$

where

$$ {}_2\boldsymbol{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \tag{5} $$

$$ {}_2\boldsymbol{D}(\theta) = \begin{bmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{bmatrix}. \tag{6} $$

Here, since $e^{i\theta} = \cos\theta + i\sin\theta$ $(Euler's\ formula \mathrm{C} |e^{i\theta}| = |e^{-i\theta}| = 1)$, ${}_2\boldsymbol{R}(\theta)^x = {}_2\boldsymbol{R}(x\theta)$ as well as ${}_2\boldsymbol{D}(\theta)^x = {}_2\boldsymbol{D}(x\theta)$ for a real number x.

The Eigen-equation of ${}_n\boldsymbol{R}$ has m sets of complex conjugate roots whose absolute value is 1 when $n = 2m$. Meanwhile, when $n = 2m+1$, ${}_n\boldsymbol{R}$ has the same m sets of complex conjugate roots and 1 as roots. Therefore, ${}_n\boldsymbol{D}$ in Equation 2 can be described as

$$ {}_n\boldsymbol{D}(\boldsymbol{\theta}) = \begin{cases} \begin{bmatrix} {}_2\boldsymbol{D}(\theta_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & {}_2\boldsymbol{D}(\theta_m) \end{bmatrix} & (n = 2m) \\ \begin{bmatrix} 1 & & \cdots & 0 \\ & {}_2\boldsymbol{D}(\theta_1) & & \vdots \\ \vdots & & \ddots & \\ 0 & \cdots & & {}_2\boldsymbol{D}(\theta_m) \end{bmatrix} & (n = 2m+1) \end{cases} \tag{7} $$

by an m dimensional vector $\boldsymbol{\theta} = (\theta_j | -\pi < \theta_j \leq \pi, j = 1, 2, \cdots, m)$ composed of m rotation angles. Thus Equation 2 can be described as

$$ {}_n\boldsymbol{R}(\boldsymbol{\theta}) = {}_n\boldsymbol{U}\,{}_n\boldsymbol{D}(\boldsymbol{\theta})\,{}_n\boldsymbol{U}^\dagger. \tag{8} $$

This means that ${}_n\boldsymbol{R}^x$ in Equation 3 is obtained as ${}_n\boldsymbol{R}(x\boldsymbol{\theta})$ by simply linearly interpolating the vector.

Additionally,

$$ {}_n\boldsymbol{R}(\boldsymbol{\theta}) = {}_n\boldsymbol{U}\,{}_n\boldsymbol{U'}^\dagger\,{}_n\boldsymbol{R'}(\theta)\,{}_n\boldsymbol{U'}\,{}_n\boldsymbol{U}^\dagger. \tag{9} $$

Here, when $n = 2m+1$,

$$ {}_n\boldsymbol{R'}(\boldsymbol{\theta}) = \begin{bmatrix} 1 & & \cdots & \mathbf{0} \\ & {}_2\boldsymbol{R}(\theta_1) & & \vdots \\ \vdots & & \ddots & \\ \mathbf{0} & \cdots & & {}_2\boldsymbol{R}(\theta_m) \end{bmatrix}, \tag{10} $$

$$ {}_n\boldsymbol{U'} = \begin{bmatrix} 1 & & \cdots & \mathbf{0} \\ & {}_2\boldsymbol{U} & & \vdots \\ \vdots & & \ddots & \\ \mathbf{0} & \cdots & & {}_2\boldsymbol{U} \end{bmatrix}. \tag{11} $$

Meanwhile, when $n = 2m$, ${}_n\boldsymbol{R'}(\boldsymbol{\theta})$ and ${}_n\boldsymbol{U'}$ are obtained by removing the first column and the first row from the matrix in the same way as Equation 7. Because

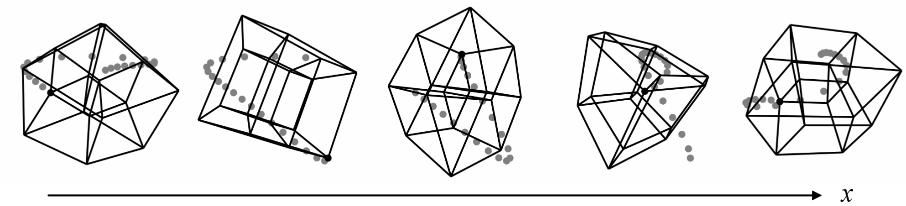

Fig. 1. Pose interpolation for a four-dimensional cube

the set of all n-dimensional rotation matrixes forms a group with multiplication called the $SO(n)$ (*Special Orthogonal group*), ${}_n\boldsymbol{U}_n\boldsymbol{U'}^{\dagger}$ can be transformed into a rotation matrix ${}_n\boldsymbol{R''}$; therefore, using only real number rotation matrices, ${}_n\boldsymbol{R}(\boldsymbol{\theta})$ can be described as

$$ {}_n\boldsymbol{R}(\boldsymbol{\theta}) = {}_n\boldsymbol{R''}\,{}_n\boldsymbol{R'}(\boldsymbol{\theta})\,{}_n\boldsymbol{R''}^T. \tag{12}$$

Using this expression, computational cost and memory are expected to be reduced in the sense of using real number matrices instead of complex number matrices. Note that the interpolated results are identical to those obtained from the simple diagonalization shown in Equation 2. ${}_n\boldsymbol{R'}(\boldsymbol{\theta})$ represents rotations on m independent rotational planes where no rotation affects other rotational planes. This means that Equation 12 expresses ${}_n\boldsymbol{R}(\boldsymbol{\theta})$ as a sequence of rotation matrices that are a unique rotation ${}_n\boldsymbol{R''}$ for ${}_n\boldsymbol{R}(\boldsymbol{\theta})$, a rotation on independent planes ${}_n\boldsymbol{R'}(\boldsymbol{\theta})$, and the inverse of the unique rotation matrix.

2.3 Rotation of a Four-Dimensional Cube

We interpolated poses of a four-dimensional cube using the rotation matrix interpolation method described above. First, two rotation matrices, $\boldsymbol{R}_0, \boldsymbol{R}_1$, were randomly chosen as key poses of the cube, and then poses between the key poses were interpolated by Equation 13. For every interpolated pose, we visualized the cube by a wireframe model using perspective projection:

$$ \boldsymbol{R}_x = \boldsymbol{R}_{0\to 1}(x\boldsymbol{\theta})\boldsymbol{R}_0. \tag{13}$$

Here, $\boldsymbol{R}_{a\to b}(\boldsymbol{\theta}) = \boldsymbol{R}_b\boldsymbol{R}_a{}^T$. This equation corresponds to the linear interpolation of rotation matrices. Figure 1 shows the interpolated results. To simplify seeing how the four-dimensional cube rotates, the vertex trajectory is plotted by dots.

3 Interpolation of Eigenspaces Using Rotation of a Hyper-Ellipsoid

3.1 Approach

The proposed method interpolates eigenspaces considering an eigenspace as a multivariate normal density. The iso-density points of a multivariate normal density are known to form a hyper-ellipsoid surface. Eigenvectors and eigenvalues

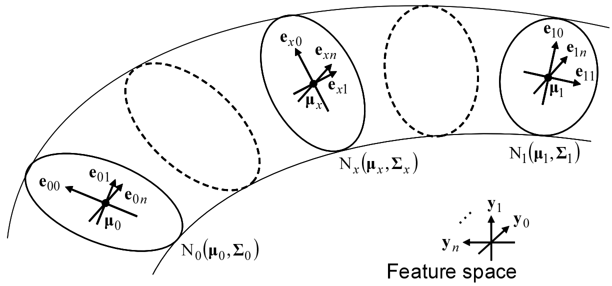

Fig. 2. Interpolation of hyper-ellipsoids

can be considered the directions of the hyper-ellipsoid's axes and their lengths, respectively. We consider that the eigenspaces between two eigenspaces could be interpolated by rotation of a hyper-ellipsoid with the expansion and contraction of the length of each axis of the ellipsoid (Figure 2).

The interpolation of ellipsoids has the following two problems. First, the correspondence of one ellipsoid's axes to another ellipsoid's axes cannot be determined uniquely. Secondly, the rotation angle cannot be determined uniquely because ellipsoids are symmetrical. From these problems, in general, ellipsoids cannot be interpolated uniquely from two ellipsoids. The following two conditions are imposed in the proposed method to obtain a unique interpolation.

[**Condition 1**] Minimize the interpolated ellipsoid's volume variations.
[**Condition 2**] Minimize the interpolated ellipsoid's rotation angle variations.

3.2 Algorithm

When two multivariate normal densities $\mathrm{N}_0(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ and $\mathrm{N}_1(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ are given, an interpolated or extrapolated density $\mathrm{N}_x(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$ for a real number x is calculated by the following procedure. Here, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ represent an n-dimensional mean vector and an $n \times n$ covariance matrix, respectively.

Interpolation of mean vectors: $\boldsymbol{\mu}_x$ is obtained by a simple linear interpolation by the following equation. This corresponds to interpolation of the ellipsoids' centers.

$$\boldsymbol{\mu}_x = (1 - x)\boldsymbol{\mu}_0 + x\boldsymbol{\mu}_1. \tag{14}$$

Interpolation of covariance matrices: Eigenvectors and eigenvalues of each covariance matrix have information about the pose of the ellipsoid and the lengths of its axes, respectively. First, $n \times n$ matrices $\boldsymbol{E}_0$ and $\boldsymbol{E}_1$ are formed by aligning each eigenvector $\boldsymbol{e}_{0j}$ and $\boldsymbol{e}_{1j} (j = 1, 2, \cdots, n)$ of $\boldsymbol{\Sigma}_0$ and $\boldsymbol{\Sigma}_1$. At the same time, n-dimensional vectors $\boldsymbol{\lambda}_0 \mathrm{C} \boldsymbol{\lambda}_1$ are formed by aligning eigenvalues $\lambda_{0j}, \lambda_{1j} (j = 1, 2, \cdots, n)$.

[Step 1] To obtain the correspondences of axes between ellipsoids based on [Condition 1], $\boldsymbol{E}'_0$ and $\boldsymbol{E}'_1$ are formed by sorting eigenvectors in $\boldsymbol{E}_0$ and $\boldsymbol{E}_1$ in the order of their eigenvalues. $\boldsymbol{\lambda}'_0$ and $\boldsymbol{\lambda}'_1$ are formed from $\boldsymbol{\lambda}_0$C$\boldsymbol{\lambda}$, as well.
[Step 2] Based on [Condition 2], $\boldsymbol{e}'_{1j}(j = 1, 2, \cdots, n)$ is inverted if ${\boldsymbol{e}'_{0j}}^T \boldsymbol{e}'_{1j} < 0$ so that the angle between corresponded axes is less than or equal to $\frac{\pi}{2}$.
[Step 3] $\boldsymbol{e}'_{0n}$ is inverted if $\det(\boldsymbol{E}'_0) = -1$, and $\boldsymbol{e}'_{1n}$ is inverted if $\det(\boldsymbol{E}'_1) = -1$, as well, so that $\boldsymbol{E}'_0$ and $\boldsymbol{E}'_1$ should meet Equation 1.

The eigenvalue of $\boldsymbol{\Sigma}_x$, λ_{xj} is calculated by

$$\lambda_{xj} = \left((1-x)\sqrt{\lambda'_{0j}} + x\sqrt{\lambda'_{1j}}\right)^2, \tag{15}$$

and its eigenvectors $\boldsymbol{E}_x$ is calculated by

$$\boldsymbol{E}_x = \boldsymbol{R}_{0\to1}(x\boldsymbol{\theta})\boldsymbol{E}'_0. \tag{16}$$

Here, $\boldsymbol{R}_{0\to1}(\boldsymbol{\theta}) = \boldsymbol{E}'_1 {\boldsymbol{E}'_0}^T$D Therefore, $\boldsymbol{\Sigma}_x$is calculated by

$$\boldsymbol{\Sigma}_x = \boldsymbol{E}_x \boldsymbol{\Lambda}_x {\boldsymbol{E}_x}^T. \tag{17}$$

HereC$\boldsymbol{\Lambda}_x$ represents a diagonal matrix that has $\lambda_{xj}(j = 1, 2, \cdots, n)$ as its diagonal elements.

4 Experiments Using Actual Images

To demonstrate the effectiveness and validity of the proposed interpolation method, we conducted face recognition experiments based on a subspace method. Training images were captured from two different angles in various illumination conditions, whereas input images were captured only from intermediate angles. In the training phase, a subspace for each camera angle was constructed from images captured in different illumination conditions. We compared the performance between recognition by the two subspaces and the interpolated subspaces.

4.1 Conditions

In the experiments, we used the face images of ten persons captured from three different angles (two for training and one for testing) in 51 different illumination conditions. Figures 3 and 4 show examples of the persons' images and images captured in various conditions. In Figure 5, images from camera angles c_0 and c_1 were used for training and $c_{0.5}$ for testing. The images were chosen from the face image dataset, "Yale Face Database B"[6].

We represented each image as a low dimensional vector in a 30-dimensional feature space using a dimension reduction technique based on PCA. In the training phase, for each person p, autocorrelation matrices $\boldsymbol{\Sigma}_0^{(p)}$ and $\boldsymbol{\Sigma}_1^{(p)}$ were calculated from images obtained from angles c_0 and c_1, and then matrices $\boldsymbol{E}_0^{(p)}$ and $\boldsymbol{E}_1^{(p)}$ were obtained that consist of eigenvectors of the autocorrelation matrices.

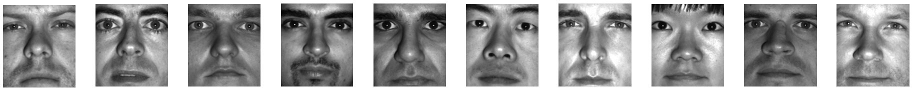

Fig. 3. Sample images of ten persons' faces used in experiment

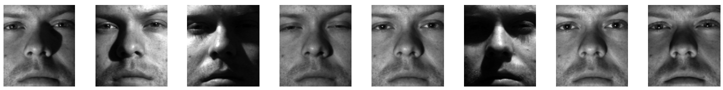

Fig. 4. Sample images captured in various illumination conditions used in experiment

In the recognition phase, the similarity between the subspaces and a test image captured from $c_{0.5}$ were measured, and recognition result $\hat{p}$ was obtained that gives maximum similarity. The similarity between an input vector $\boldsymbol{z}$ and the $K(\leq 30)$-dimensional subspace of $\boldsymbol{E}_x^{(p)}$ is calculated by

$$S_x^{(p)}(\boldsymbol{z}) = \sum_{k=1}^{K} < \boldsymbol{e}_{x,k}^{(p)}, \boldsymbol{z} >^2, \tag{18}$$

where $\boldsymbol{E}_x^{(p)}(0 \leq x \leq 1)$ is the interpolated eigenspace between $\boldsymbol{E}_0^{(p)}$ and $\boldsymbol{E}_1^{(p)}$ and $< \cdot, \cdot >$ represents the inner product of the two vectors. $\boldsymbol{E}_x^{(p)}$ is calculated by Equation 16.

The proposed method that uses the subspaces of the interpolated eigenspaces obtains $\hat{p}$ by

$$\hat{p} = \arg\max_{p} \max_{0 \leq x \leq 1} \left(S_x^{(p)}(\boldsymbol{z}) \right). \tag{19}$$

On the other hand, as a comparison method, the recognition method with subspaces of $\boldsymbol{E}_0^{(p)}$ and $\boldsymbol{E}_1^{(p)}$ obtains $\hat{p}$ by

$$\hat{p} = \arg\max_{p} \max \left(S_0^{(p)}(\boldsymbol{z}), \;\; S_1^{(p)}(\boldsymbol{z}) \right). \tag{20}$$

We defined $K = 5$ in Equation 18 empirically through preliminary experiments.

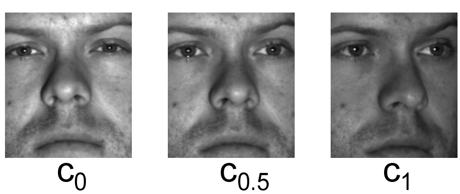

Fig. 5. Sample images captured from three camera angles used in experiment

Table 1. Comparison of recognition rates

Recognition Method	Recognition Rate [%]
Interpolated subspaces by proposed method (Eq. 19)	**71.8**
Two subspaces (Eq. 20)	62.6

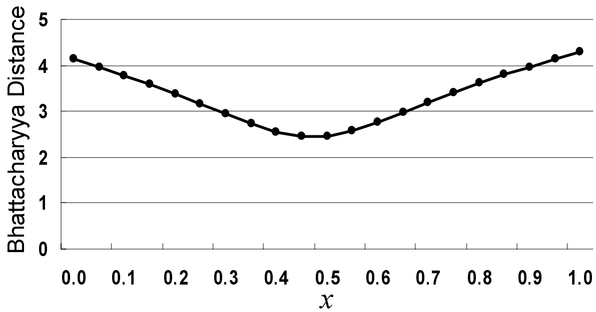

Fig. 6. Bhattacharyya distances between actual normal density and interpolated densities

4.2 Results and Discussion

Table 1 compares the recognition rates of the two methods described in 4.1. From this result, we confirmed the effectiveness of the proposed method for face recognition.

For verification of the validity of interpolation by the proposed method, Figure 6 shows the *Bhattacharyya* distances between normal density obtained from $c_{0.5}$ and the interpolated normal densities from $x = 0$ to $x = 1$ for a person. Since the distance becomes smaller around $x = 0.5$, the validity of interpolation by the proposed method can be observed. In addition, Figure 7 visualizes the interpolated eigenvectors from $x = 0$ to $x = 1$ of the person. We can see that the direction of each eigenvector was changed smoothly by high dimensional rotation.

5 Summary

In this paper, we proposed a method for interpolation between eigenspaces. The experiments on face recognition based on the subspace method demonstrated the effectiveness and validity of the proposed method.

Future works include expansion of the method into higher order interpolation such as a cubic spline and recognition experiments using larger datasets.

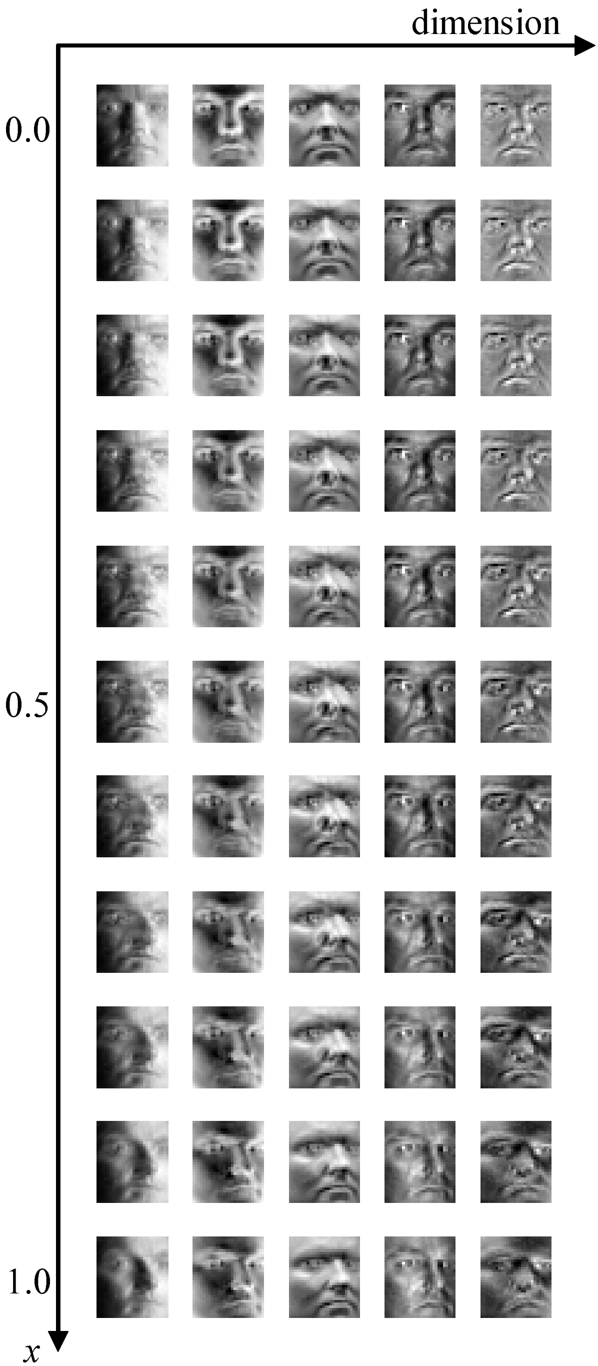

Fig. 7. Interpolated eigenvectors

References

1. Watanabe, S., Pakvasa, N.: Subspace Method of Pattern Recognition. In: Proc. 1st Int. J. Conf. on Pattern Recognition, pp. 25–32 (1971)
2. Turk, M., Pentland, A.: Face Recognition Using Eigenfaces. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, pp. 568–591. IEEE Computer Society Press, Los Alamitos (1991)
3. Moghaddam, B.: Principal Manifolds and Bayesian Subspaces for Visual Recognition. In: Proc. Int. Conf. on Computer Vision, pp. 1131–1136 (1999)
4. Murase, H., Nayar, S.K.: Illumination Planning for Object Recognition Using Parametric Eigenspaces. IEEE Trans. Pattern Analysis and Machine Intelligence 16(12), 1218–1227 (1994)
5. Takahashi, L.T., Ide, I., Murase, H.: Appearance Manifold with Embedded Covariance Matrix for Robust 3-D Object Recognition. In: Proc. 10th IAPR Conf. on Machine Vision Applications, pp. 504–507 (2007)
6. Georghiades, A.S., Belhumeur, P.N., Kriegman, C.D.J.: From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose. IEEE Trans. Pattern Analysis and Machine Intelligence 23(6), 643–660 (2001)