Eigenfaces and Deformations

Oleg Makhnin
Dept. of Mathematics
New Mexico Tech

February 27, 2009

Part I. Eigenfaces
Part II. Deformations
Part III. Eigenfaces and deformations

Eigenfaces

L. Sirovich and M. Kirby (1987)

A popular technique to identify main features of a face; also serves as an illustration to Principal Components Analysis (PCA)/ Karhunen-Loeve decomposition/ Empirical Orthogonal functions etc.

Data: n observations of a d-dim vector $\mathbf{x}$. Represent as a matrix

$$\mathbf{X} = [\mathbf{x}_1 .... \mathbf{x}_n]$$

In our context: d = dimension of the image ($d = n_{row} \times n_{col}$), taken as a vector. (Ignore spatial information!)

Data

Data: from FERET database

Facial REcognition Technology (see http://www.itl.nist.gov/iad/humanid/feret/feret_master.html) (about 1000 subjects and 4000 frontal images)

Frontal images, 96*64, so that d = 6144, n = 200 so far. Two images per person (100 persons) to give an idea about variability.

Eigenvalue decomposition

Eigenvalue decomposition of Covariance matrix:

$$\mathbf{C} = \frac{1}{n-1}(\mathbf{X} - mean)'(\mathbf{X} - mean)$$

- symmetric, positive-definite
- mean = columnwise mean (mean of $\mathbf{x}_1$ in the 1st column, mean of $\mathbf{x}_2$ in the 2nd etc.)

thus

$$\mathbf{C} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}'$$

$\boldsymbol{\Lambda}$ is diagonal with eigenvalues, $\mathbf{V}$ is unitary, columns are eigenvectors

However, $\mathbf{C}$ is a $d \times d$ matrix, of rank only n:
problems when $d > n$.

SVD

Another way: SVD (singular value decomposition)

$$\mathbf{X} - mean = \mathbf{X}_o = \mathbf{UDV}'$$

$\mathbf{U}, \mathbf{V}$ are unitary, $\mathbf{D}$ is diagonal

$\mathbf{X}$ is not necessarily symmetric now

note

$$\mathbf{C} = \mathbf{X}_o'\mathbf{X}_o = \mathbf{VDU}'\mathbf{UDV}' = \mathbf{VDDV}'$$

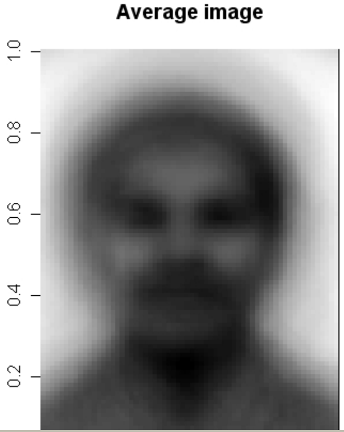
so that

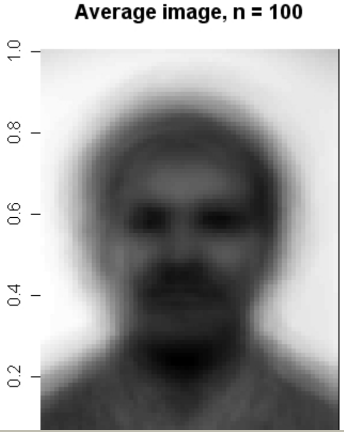
$$\mathbf{\Lambda} = \frac{1}{n-1}\mathbf{DD}$$

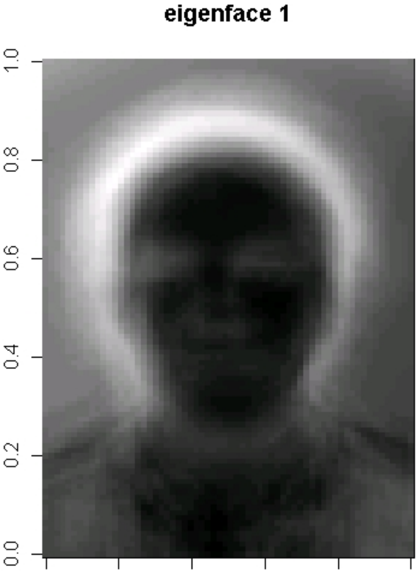
Columns of $\mathbf{V}$ form an orthonormal *adaptive* basis.

Illustration

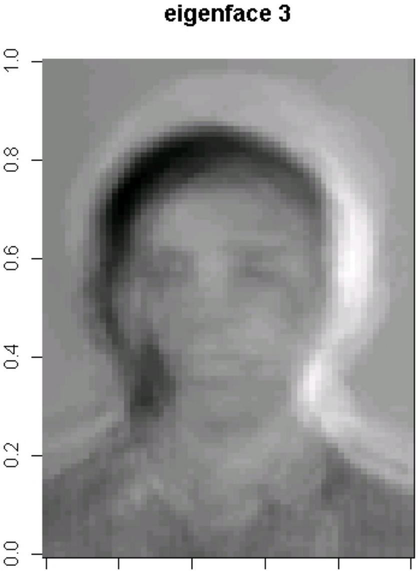
The columns of $\mathbf{V}$, or "principal components" are known as *eigenfaces*.

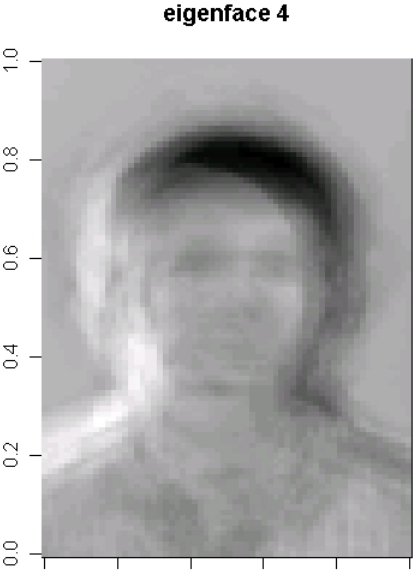


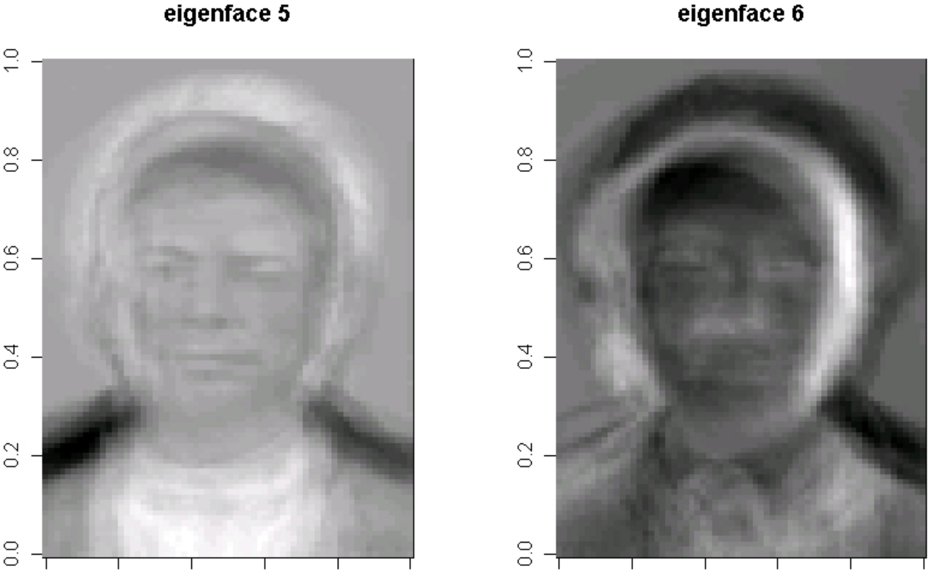


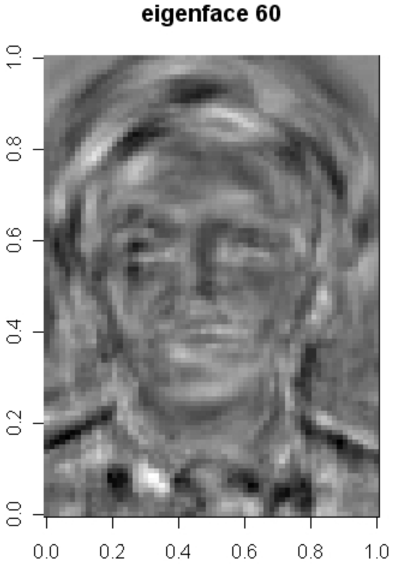

eigenface 1
1.0
0.8
0.6
0.4
0.2
0.0

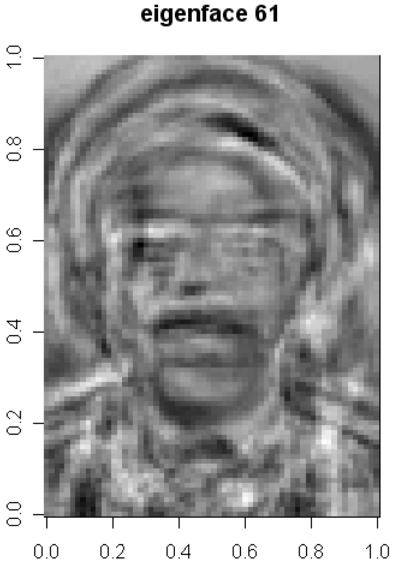

eigenface 2
1.0
0.8
0.6
0.4
0.2
0.0


eigenface 3
0.0
0.2
0.4
0.6
0.8
1.0


eigenface 4
0.0
0.2
0.4
0.6
0.8
1.0


eigenface 5
1.0
0.8
0.6
0.4
0.2
0.0
eigenface 6
1.0
0.8
0.6
0.4
0.2
0.0


eigenface 60
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0


eigenface 61
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0

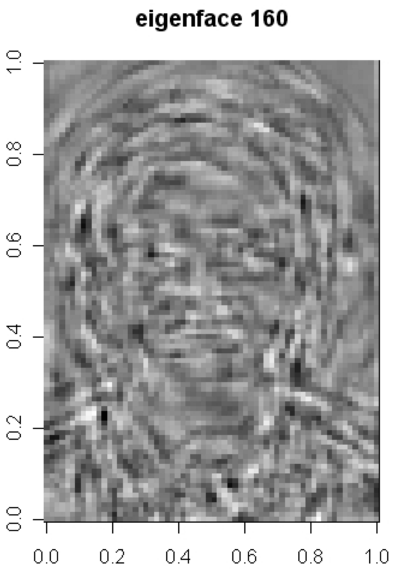
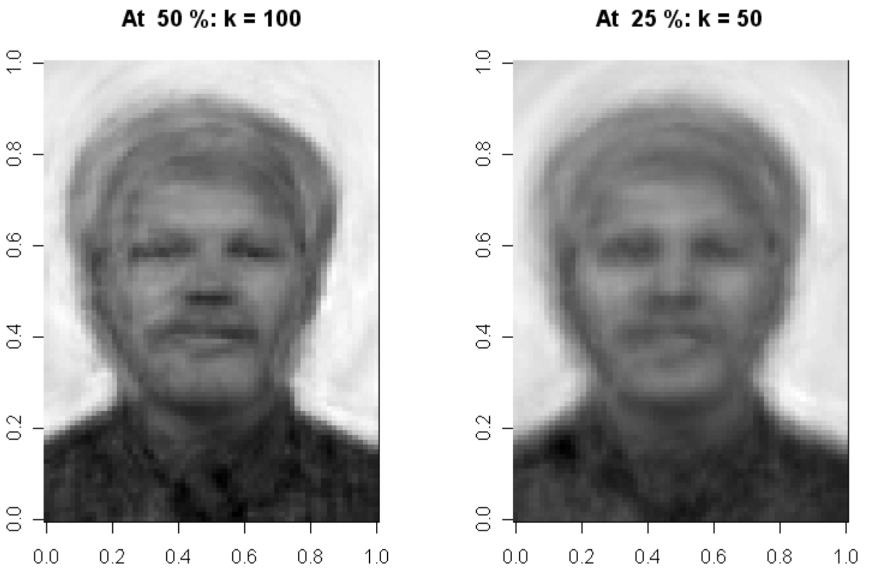

eigenface 160
eigenface 161
0.0
0.2
0.4
0.6
0.8
1.0

Image “Compression”

If we only keep k most important eigenfaces, we can reduce the image dimensionality/ database size


Lossless (n= 200)
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0


At 75 %: k = 150
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0


At 50 %: k = 100
At 25 %: k = 50
0.0
0.2
0.4
0.6
0.8
1.0

Difficulties

- The vector approach ignores the spatial structure of the face image.

Difficulties

- The vector approach ignores the spatial structure of the face image.
- Different centering/scale and lighting conditions. Different expressions.

Difficulties

- The vector approach ignores the spatial structure of the face image.
- Different centering/scale and lighting conditions. Different expressions.
- Eigenfaces are mostly blurry features (i.e. “low-frequency” information)

Difficulties

- The vector approach ignores the spatial structure of the face image.
- Different centering/scale and lighting conditions. Different expressions.
- Eigenfaces are mostly blurry features (i.e. "low-frequency" information)

Variations of the technique: Fisherfaces (uses discriminant analysis to separate images of one person from images of another; e.g. answers the question which regions are important)

Difficulties

- The vector approach ignores the spatial structure of the face image.
- Different centering/scale and lighting conditions. Different expressions.
- Eigenfaces are mostly blurry features (i.e. "low-frequency" information)

Variations of the technique: Fisherfaces (uses discriminant analysis to separate images of one person from images of another; e.g. answers the question which regions are important)

LaplacianFaces (uses manifold techniques)

Deformations

- Deformation is defined on a coarse grid (e.g. iris paper)

J. Thornton, M. Savvides and V. Kumar (2007), *A Bayesian Approach to Deformed Pattern Matching of Iris Images.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 29(4), 596-606

(however they only applied deformation to matching a given image to the database, not in the database construction itself)

- then is smoothly extended pixel-wise

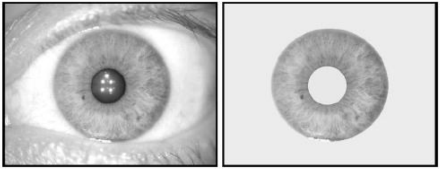


Fig. 1. Sample eye image and its isolated iris pattern.

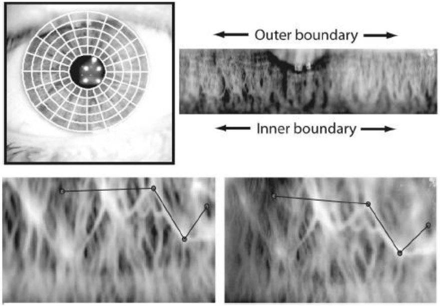
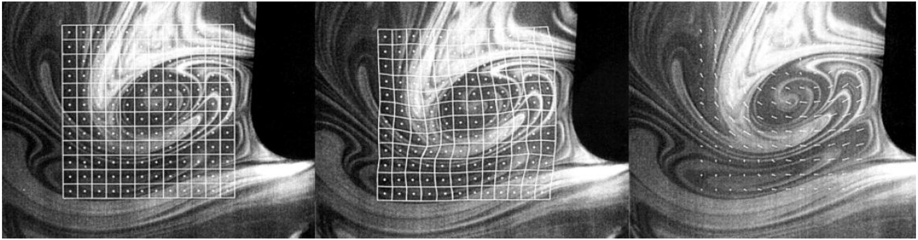


Fig. 2. Top left: Location of iris region by boundary detection. Top right: Unwrapped iris pattern in normalized polar coordinates. Bottom: Close-ups of segmented patterns captured from same eye (landmark points illustrate relative deformation).

Deformations

Unexpected link: particle image velocimetry (PIV)
F Scarano (2002) Iterative image deformation methods in PIV
Meas. Sci. Technol. 13 R1-R19 PII: S0957-0233(02)20239-8



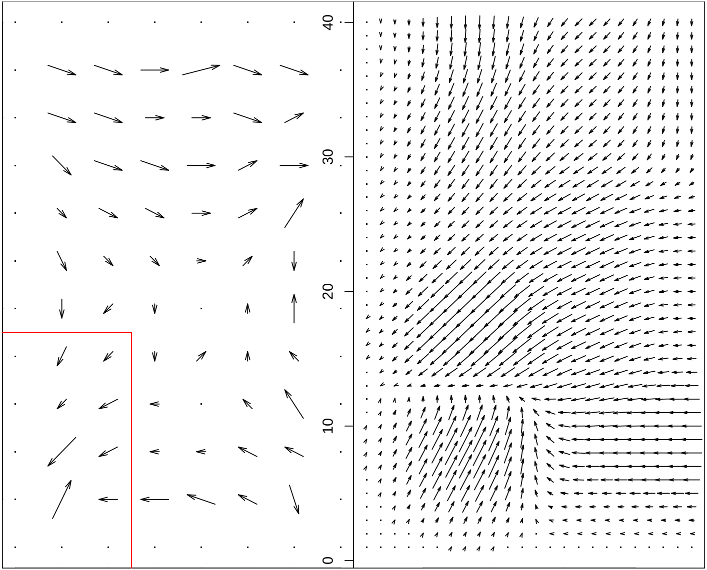
Primitive idea

1) estimate optimal shifts by maximizing correlation between square fragments of the coarse grid

2) smooth out to define the deformation vector field for every pixel

(primitive: extrapolate from the corners of a square formed by points of coarse grid)
(more sophisticated: splines)

3) move the pixels according to the deformation field


0
10
20
30
40

Primitive idea (cont'd): Moving the pixels

A problem: non-integer shifts

Primitive solution: weigh the intensity according to fractional value of the shift

`< pic here >`

More sophisticated: apply Fourier transform (FT) to the image; then shifting the image will mean applying a phase shift to the FT, then inverse FT. However, deformation is not constant.

$\Longrightarrow$ Digital filtering

Fitting a coarse deformation

Primitive idea: move blocks of pixels around to reach highest correlation between deformed image and reference image

More advanced approach:

- minimize square discrepancy between the deformed image and the reference image. (likely Monte-Carlo methods)

Fitting a coarse deformation

Primitive idea: move blocks of pixels around to reach highest correlation between deformed image and reference image

More advanced approach:

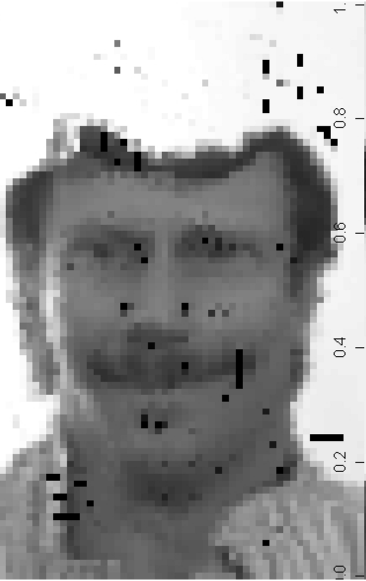
- minimize square discrepancy between the deformed image and the reference image. (likely Monte-Carlo methods)
- Bayesian twist: add a prior that regularizes the deformation field. Then we can find the MAP (Maximum a Posteriori Estimate) which is generalization of the Maximum Likelihood/ Least square approaches

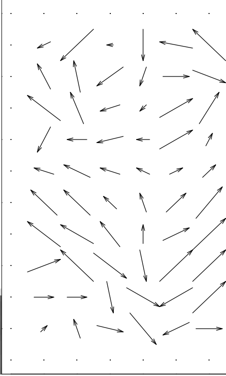
Experiments


0.2
0.4
0.6
0.8


0.
0.2
0.4
0.6
0.8
1.


0.8
0.6
0.4
0.2


0.8
0.6
0.4
0.2



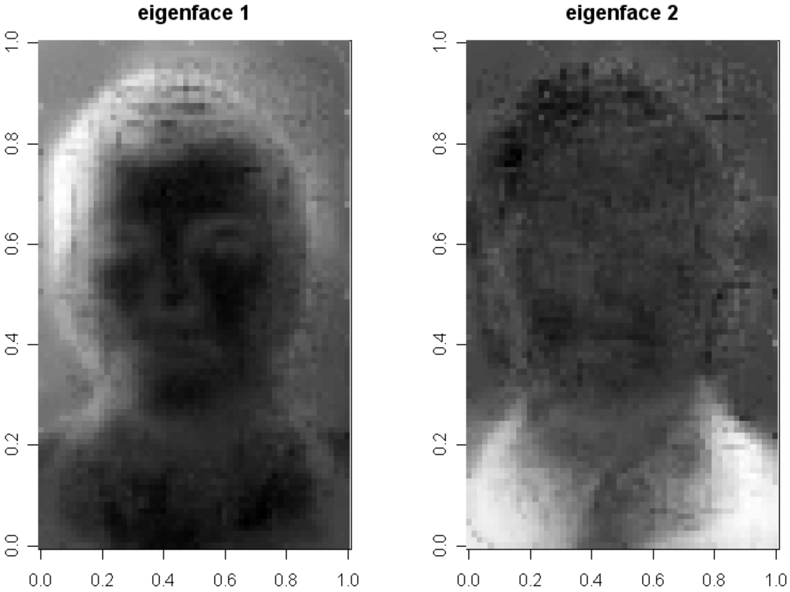
Much work to be done!!!

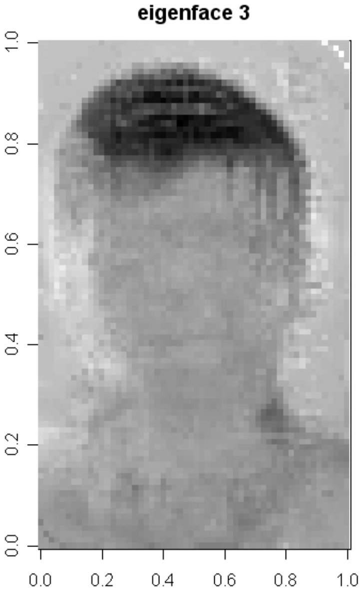
Deformations and eigenfaces

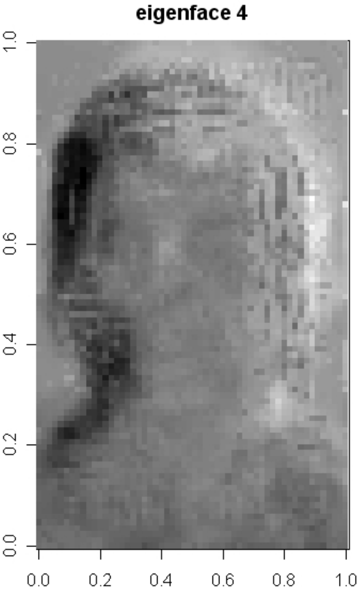
Deform all other images to one arbitrarily chosen reference.
What happens to eigenfaces?

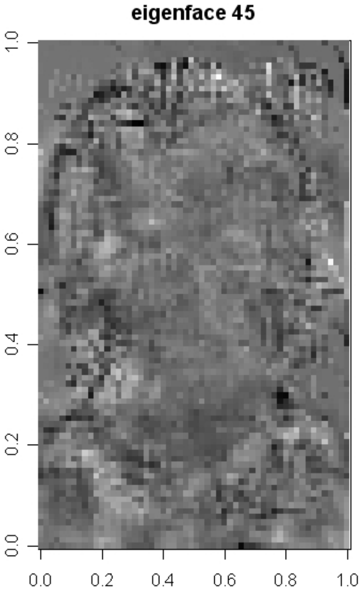
New mean image

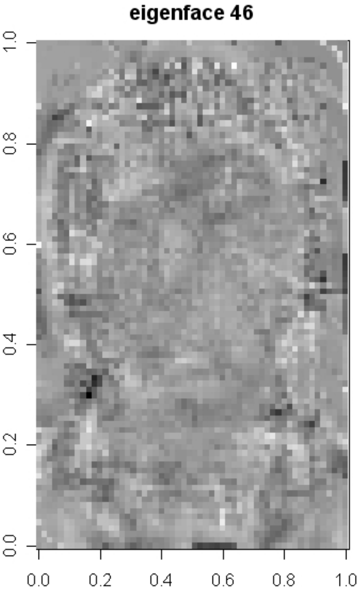
Reference image


eigenface 1
eigenface 2
0.0
0.2
0.4
0.6
0.8
1.0


eigenface 3
0.0
0.2
0.4
0.6
0.8
1.0


eigenface 4
0.0
0.2
0.4
0.6
0.8
1.0


eigenface 45
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0


eigenface 46
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0

Eigenfaces and deformations

So far, just deformed all images to an arbitrarily chosen reference, see how the eigenfaces change.

Potentially: should be an iterative process, for example

1) define a single "canonical image" for each person
2) define eigenfaces based on the canonical images
3) recalculate canonical images (deformation) based on the few important eigenfaces

If the task is e.g. pattern-matching, then match the (suitably deformed) query image to a canonical image.

QUESTIONS?

THANK YOU!

www.nmt.edu/~olegm/talks/DEF.pdf

for pdf file