CS 479/679
Pattern Recognition – Spring 2006

Dimensionality Reduction Using PCA/LDA
Chapter 3 (Duda et al.) – Section 3.8

Case Studies:
Face Recognition Using Dimensionality Reduction

M. Turk, A. Pentland, "Eigenfaces for Recognition", *Journal of Cognitive Neuroscience*, 3(1), pp. 71-86, 1991.

D. Swets, J. Weng, "Using Discriminant Eigenfeatures for Image Retrieval", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8), pp. 831-836, 1996.

A. Martinez, A. Kak, "PCA versus LDA", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228-233, 2001.

Dimensionality Reduction

- One approach to deal with high dimensional data is by reducing their dimensionality.

- Project high dimensional data onto a lower dimensional sub-space using <u>linear</u> or <u>non-linear</u> transformations.

$$x = \begin{bmatrix} a_1 \\ a_2 \\ ... \\ a_N \end{bmatrix} --> reduce\ dimensionality --> y = \begin{bmatrix} b_1 \\ b_2 \\ ... \\ b_K \end{bmatrix} \quad (K << N)$$

Dimensionality Reduction

- Linear transformations are simple to compute and tractable.

$$Y = U \ X \qquad (b_i = u_i^t a_i)$$

k x 1   k x d   d x 1   (k<<d)

- Classical –linear- approaches:
  - Principal Component Analysis (PCA)
  - Fisher Discriminant Analysis (FDA)

Principal Component Analysis (PCA)

- Each dimensionality reduction technique finds an appropriate transformation by satisfying certain criteria (e.g., information loss, data discrimination, etc.)

- The goal of PCA is to reduce the dimensionality of the data while **retaining as much as possible of the variation present in the dataset.**

Principal Component Analysis (PCA)

- Find a basis in a low dimensional sub-space:

  - Approximate vectors by projecting them in a low dimensional sub-space:

    (1) Original space representation:

    $$x = a_1 v_1 + a_2 v_2 + \ldots + a_N v_N$$

    $\begin{bmatrix} a_1 \\ a_2 \\ \ldots \\ a_N \end{bmatrix}$

    where $v_1, v_2, \ldots, v_n$ is a base in the original N-dimensional space

    (2) Lower-dimensional sub-space representation:

    $\begin{bmatrix} b_1 \\ b_2 \\ \ldots \\ b_K \end{bmatrix}$

    $$\hat{x} = b_1 u_1 + b_2 u_2 + \ldots + b_K u_K$$

    where $u_1, u_2, \ldots, u_K$ is a base in the K-dimensional sub-space (K<N)

- *Note:* if K=N, then $\hat{x} = x$

Principal Component Analysis (PCA)

- Example (K=N):

$$v_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, v_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, v_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \text{(standard basis)}$$

$$x_v = \begin{bmatrix} 3 \\ 3 \\ 3 \end{bmatrix} = 3v_1 + 3v_2 + 3v_3$$

$$u_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, u_2 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, u_3 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{(some other basis)}$$

$$x_u = \begin{bmatrix} 3 \\ 3 \\ 3 \end{bmatrix} = 0u_1 + 0u_2 + 3u_3$$

thus, $x_v = x_u$

Principal Component Analysis (PCA)

- Information loss

  - Dimensionality reduction implies information loss !!
  - PCA preserves as much information as possible:

$$\min \| x - \hat{x} \| \quad (\text{reconstruction error})$$

- What is the "best" lower dimensional sub-space?

  The "best" low-dimensional space is <u>centered</u> at the sample mean and has <u>directions</u> determined by the "best" eigenvectors of the covariance matrix of the data x.

  - By "best" eigenvectors we mean those corresponding to the <u>*largest eigenvalues*</u> (i.e., **"principal components"**).
  - Since the covariance matrix is real and symmetric, these eigenvectors are orthogonal and form a set of basis vectors.

Principal Component Analysis (PCA)

- Methodology
  - Suppose x_1, x_2, ..., x_M are N x 1 vectors

Step 1: $\bar{x} = \dfrac{1}{M} \sum\limits_{i=1}^{M} x_i$

Step 2: subtract the mean: $\Phi_i = x_i - \bar{x}$

Step 3: form the matrix $A = [\Phi_1 \; \Phi_2 \cdots \Phi_M]$ (NxM matrix), then compute:

$$C = \dfrac{1}{M} \sum\limits_{n=1}^{M} \Phi_n \Phi_n^T = AA^T$$

(sample **covariance** matrix, NxN, characterizes the *scatter* of the data)

Step 4: compute the eigenvalues of C: $\lambda_1 > \lambda_2 > \cdots > \lambda_N$

Step 5: compute the eigenvectors of C: $u_1, u_2, \ldots, u_N$

Principal Component Analysis (PCA)

- Methodology – cont.

- Since C is symmetric, $u_1, u_2, \ldots, u_N$ form a basis, (i.e., any vector x or actually $(x - \bar{x})$, can be written as a linear combination of the eigenvectors):

$$x - \bar{x} = b_1 u_1 + b_2 u_2 + \cdots + b_N u_N = \sum_{i=1}^{N} b_i u_i \qquad \boxed{b_i = u_i^T (x - \bar{x})}$$

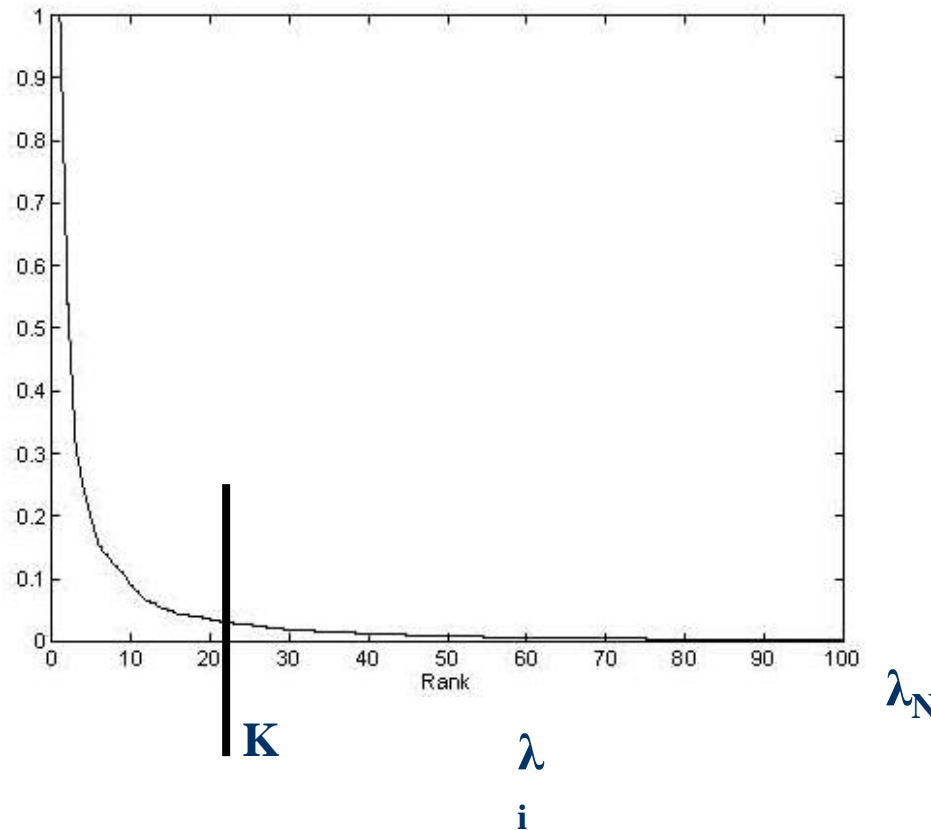Step 6: (**dimensionality reduction step**) keep only the terms corresponding to the K largest eigenvalues:

$$\hat{x} - \bar{x} = \sum_{i=1}^{K} b_i u_i \text{ where } K << N$$

- The representation of $\hat{x} - \bar{x}$ into the basis $u_1, u_2, \ldots, u_K$ is thus

$$\begin{bmatrix} b_1 \\ b_2 \\ \ldots \\ b_K \end{bmatrix}$$

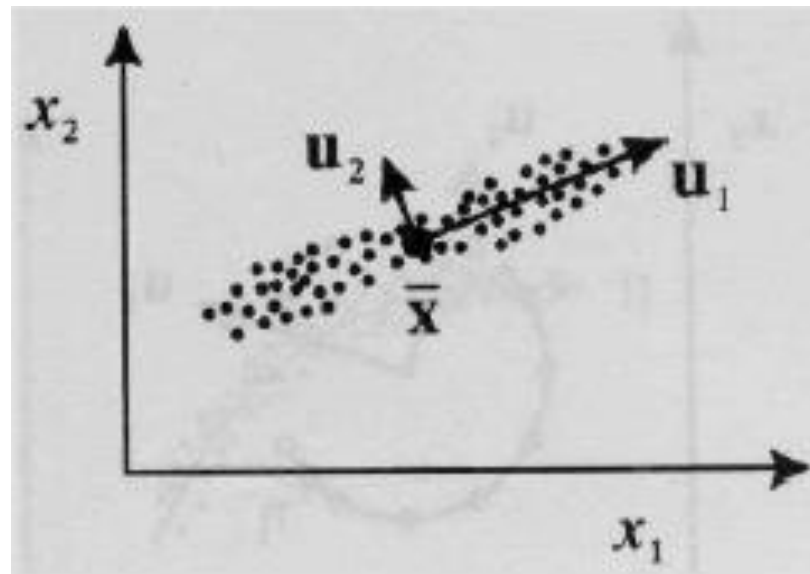# Principal Component Analysis (PCA)

- Eigenvalue spectrum

Principal Component Analysis (PCA)

- Linear transformation implied by PCA

  - The linear transformation $R^N \rightarrow R^K$ that performs the dimensionality reduction is:

$$\begin{bmatrix} b_1 \\ b_2 \\ \ldots \\ b_K \end{bmatrix} = \begin{bmatrix} u_1^T \\ u_2^T \\ \ldots \\ u_K^T \end{bmatrix} (x - \bar{x}) = U^T(x - \bar{x})$$

Principal Component Analysis (PCA)

- Geometric interpretation
  - PCA projects the data along the directions where the data <u>varies the most</u>.
  - These directions are determined by the eigenvectors of the covariance matrix corresponding to the largest eigenvalues.
  - The magnitude of the eigenvalues <u>corresponds to the variance</u> of the data along the eigenvector directions.

Principal Component Analysis (PCA)

- How many principal components to keep?

  - To choose *K*, you can use the following criterion:

$$\frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{N} \lambda_i} > Threshold \quad (e.g., 0.9 \text{ or } 0.95)$$

Principal Component Analysis (PCA)

- What is the error due to dimensionality reduction?

$$e = \left\| x - \hat{x} \right\|$$

$$\hat{x} - \bar{x} = \sum_{i=1}^{K} b_i u_i \text{ or } \hat{x} = \sum_{i=1}^{K} b_i u_i + \bar{x}$$

- It can be shown that the average error due to dimensionality reduction is equal to:

$$\overline{e} = 1/2 \sum_{i=K+1}^{N} \lambda_i$$

Principal Component Analysis (PCA)

- Standardization

  - The principal components are dependent on the _units_ used to measure the original variables as well as on the _range_ of values they assume.

  - We should always standardize the data prior to using PCA.

  - A common standardization method is to transform all the data to have zero mean and unit standard deviation:

$$\frac{x_i - \mu}{\sigma} \quad (\mu \text{ and } \sigma \text{ are the mean and standard deviation of } x_i\text{'s})$$

Principal Component Analysis (PCA)

- Case Study: Eigenfaces for Face Detection/Recognition

  − M. Turk, A. Pentland, "Eigenfaces for Recognition", *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.

- Face Recognition

  − The simplest approach is to think of it as a template matching problem

  − Problems arise when performing recognition in a high-dimensional space.

  − Significant improvements can be achieved by first mapping the data into a *lower dimensionality* space.

  − How to find this lower-dimensional space?

Principal Component Analysis (PCA)

- Main idea behind eigenfaces

average face

- Suppose Γ is an $N^2 \times 1$ vector, corresponding to an $N \times N$ face image I.

- The idea is to represent Γ ($\Phi = \Gamma - \Psi$) into a low-dimensional space:

$$\hat{\Phi} - \Psi = w_1 u_1 + w_2 u_2 + \cdots w_K u_K \ (K \ll N^2)$$
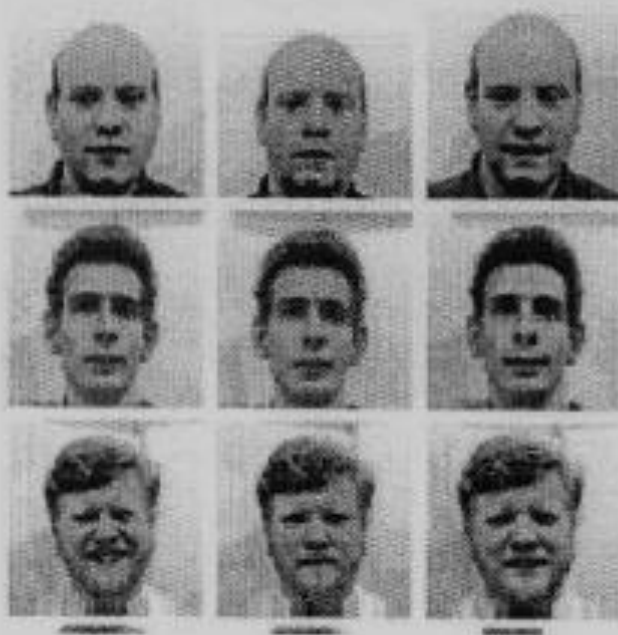
N x N image $\quad \rightsquigarrow \quad$ N^2 x 1 vector
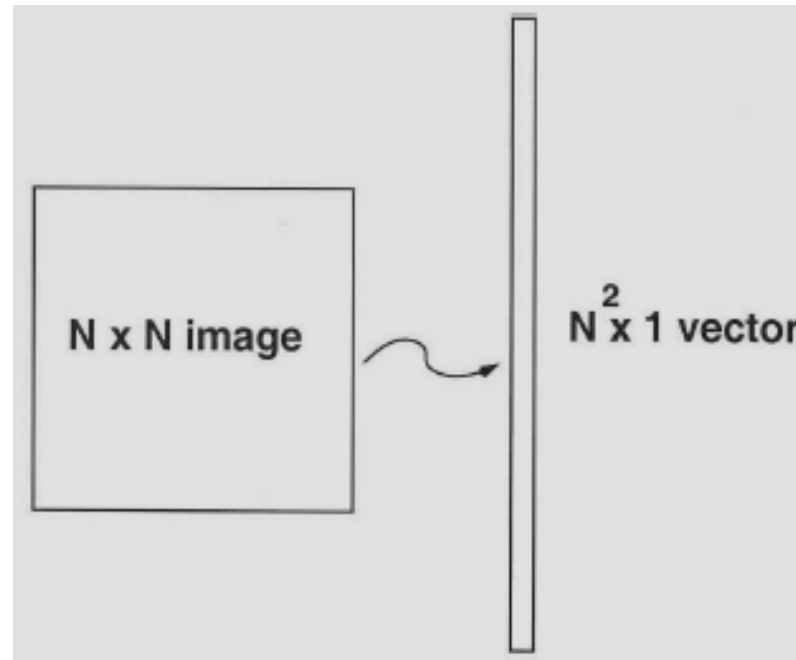
Principal Component Analysis (PCA)

- Computation of the eigenfaces

Step 1: obtain face images $I_1, I_2, ..., I_M$ (training faces)

(**very important:** the face images must be *centered* and of the same *size*)

Step 2: represent every image I_i as a vector Γ_i

Principal Component Analysis (PCA)

- Computation of the eigenfaces – cont.

Step 3: compute the average face vector Ψ:

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i$$

Step 4: subtract the mean face:

$$\Phi_i = \Gamma_i - \Psi$$

Step 5: compute the covariance matrix C:

$$C = \frac{1}{M} \sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T \quad (N^2 \text{x} N^2 \text{ matrix})$$

$$\text{where } A = [\Phi_1 \ \Phi_2 \ \cdots \ \Phi_M] \quad (N^2 \text{x} M \text{ matrix})$$

Principal Component Analysis (PCA)

- Computation of the eigenfaces – cont.

Step 6: compute the eigenvectors u_i of AA^T

The matrix AA^T is very large --> not practical !!

Step 6.1: consider the matrix $A^T A$ ($M \times M$ matrix)

Step 6.2: compute the eigenvectors v_i of $A^T A$

$$A^T Av_i = \mu_i v_i$$

What is the relationship between u_i and v_i?

$$A^T Av_i = \mu_i v_i => AA^T Av_i = \mu_i Av_i =>$$

$$CAv_i = \mu_i Av_i \text{ or } Cu_i = \mu_i u_i \text{ where } \boxed{u_i = Av_i}$$

Thus, AA^T and $A^T A$ have the same eigenvalues and their eigenvectors are related as follows: $u_i = Av_i$!!

Principal Component Analysis (PCA)

- Computation of the eigenfaces – cont.

Note 1: AA^T can have up to N^2 eigenvalues and eigenvectors.

Note 2: $A^T A$ can have up to M eigenvalues and eigenvectors.

Note 3: The M eigenvalues of $A^T A$ (along with their corresponding eigenvectors) correspond to the M *largest* eigenvalues of AA^T (along with their corresponding eigenvectors).

Step 6.3: compute the M best eigenvectors of AA^T: $u_i = Av_i$

(**important:** normalize u_i such that $\|u_i\| = 1$)

Step 7: keep only K eigenvectors (corresponding to the K largest eigenvalues)

Principal Component Analysis (PCA)

- Representing faces onto this basis

- Each face (minus the mean) Φ_i in the training set can be represented as a linear combination of the best K eigenvectors:
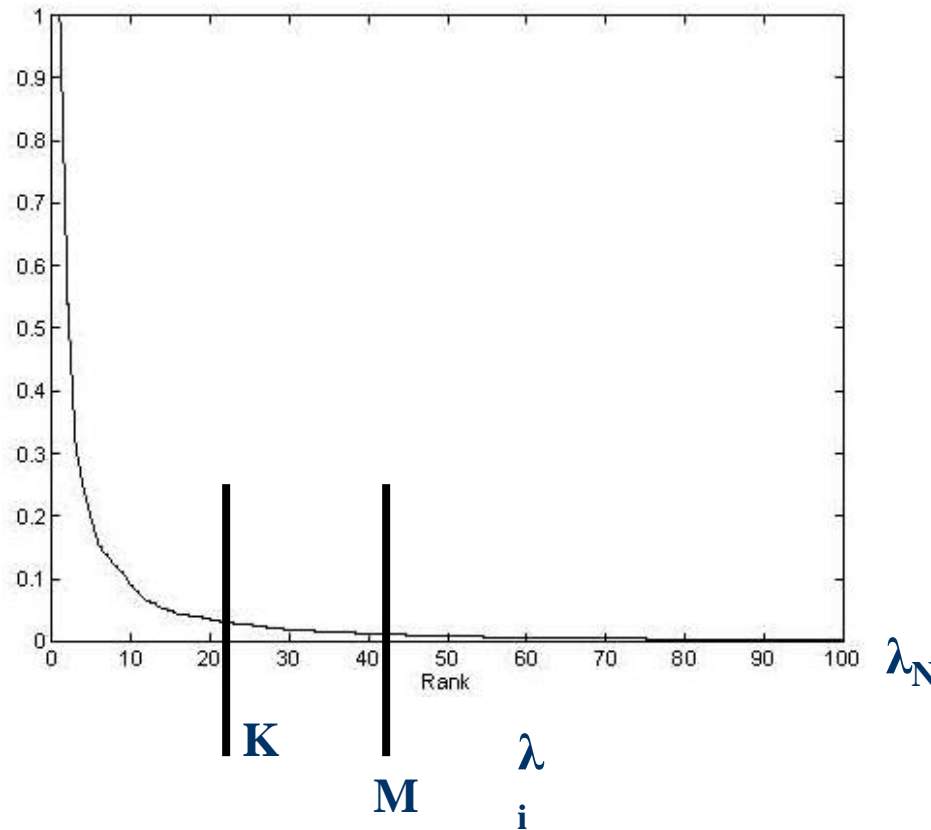
$$\hat{\Phi}_i = \sum_{j=1}^{K} w_j u_j, \quad (w_j = u_j^T \Phi_i)$$

(we call the u_j's *eigenfaces*)

= 0.9571 *   - 0.1945 *   + 0.0461 *   0.0586 *

Principal Component Analysis (PCA)

- Eigenvalue spectrum

Principal Component Analysis (PCA)

- Representing faces onto this basis – cont.

- Each normalized training face Φ_i is represented in this basis by a vector:

$$\Omega_i = \begin{bmatrix} w_1^i \\ w_2^i \\ \dots \\ w_K^i \end{bmatrix}, \quad i = 1, 2, \dots, M$$

Principal Component Analysis (PCA)

- Face Recognition Using Eigenfaces

- Given an unknown face image Γ (centered and of the same size like the training faces) follow these steps:

Step 1: normalize Γ: $\Phi = \Gamma - \Psi$

Step 2: project on the eigenspace

$$\hat{\Phi} = \sum_{i=1}^{K} w_i u_i \quad (w_i = u_i^T \Phi)$$

Step 3: represent Φ as: $\Omega = \begin{bmatrix} w_1 \\ w_2 \\ \dots \\ w_K \end{bmatrix}$

Step 4: find $e_r = \min_l \| \Omega - \Omega^l \|$

Step 5: if $e_r < T_r$, then Γ is recognized as face l from the training set.

Principal Component Analysis (PCA)

- Face Recognition Using Eigenfaces – cont.

    - The distance e_r is called _distance within the face space_ (_difs_)

    - Comment: we can use the common Euclidean distance to compute e_r, however, it has been reported that the _Mahalanobis distance_ performs better:

$$\|\Omega - \Omega^k\| = \sum_{i=1}^{K} \frac{1}{\lambda_i} (w_i - w_i^k)^2$$

(variations along all axes are treated as equally signifi cant)

Principal Component Analysis (PCA)

- Face Detection Using Eigenfaces

- Given an unknown image Γ

  Step 1: compute $\Phi = \Gamma - \Psi$

  Step 2: compute $\hat{\Phi} = \sum_{i=1}^{K} w_i u_i \quad (w_i = u_i^T \Phi)$

  Step 3: compute $e_d = \| \Phi - \hat{\Phi} \|$

  Step 4: if $e_d < T_d$, then Γ is a face.

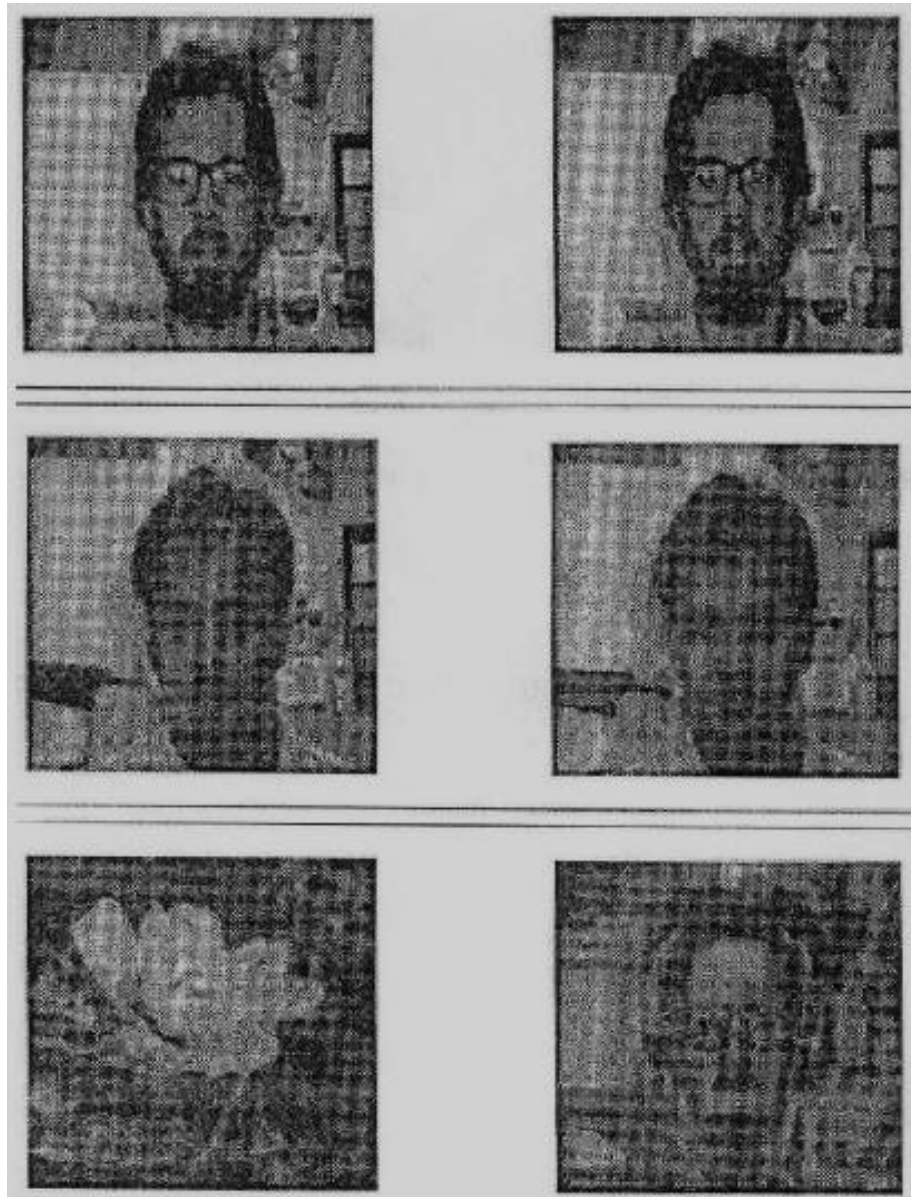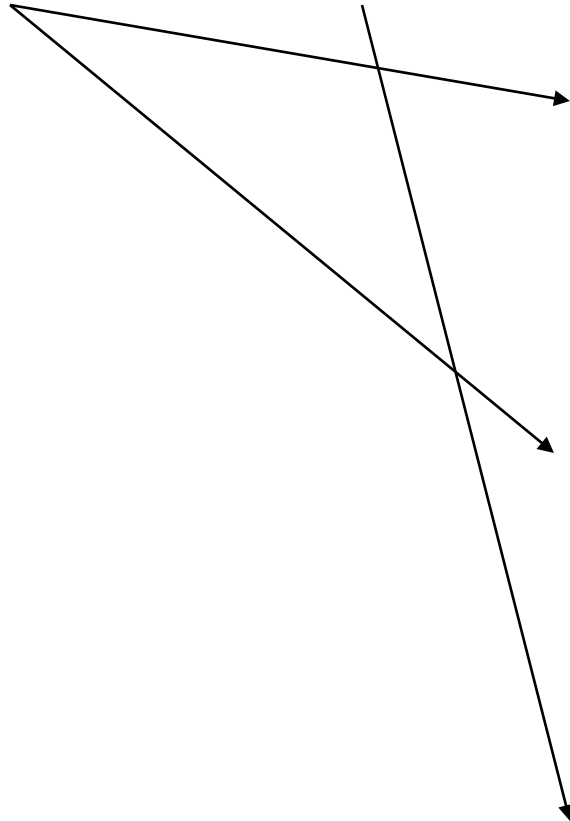- The distance e_d is called distance from face space (dffs)

Principal Component Analysis (PCA)
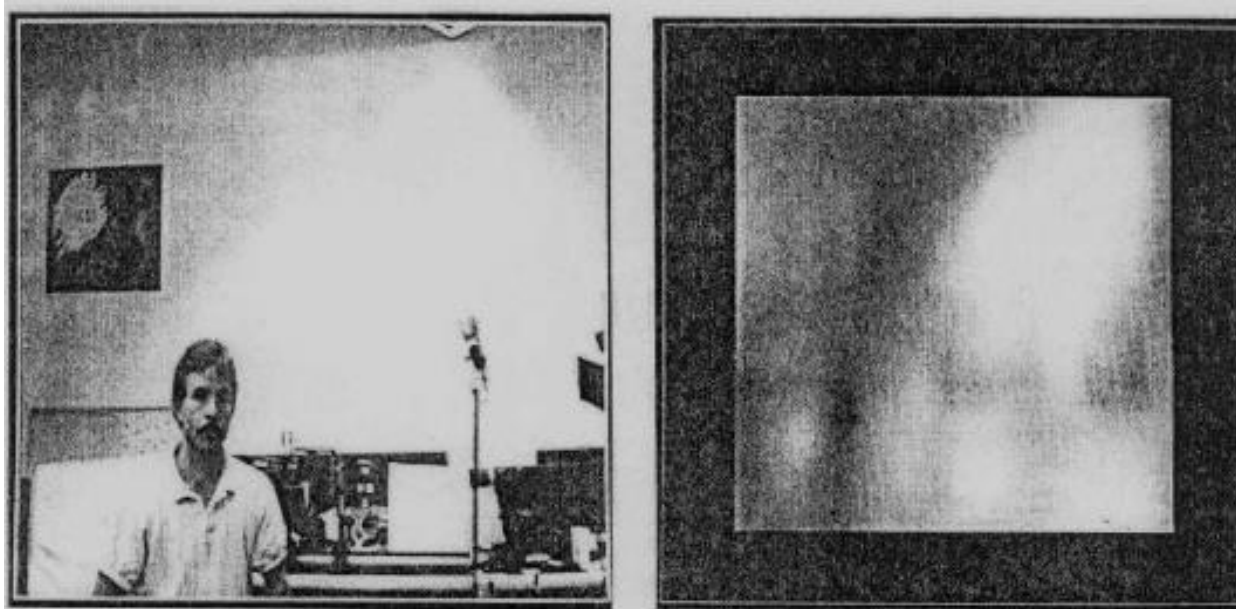
- Face Detection Using Eigenfaces – cont.
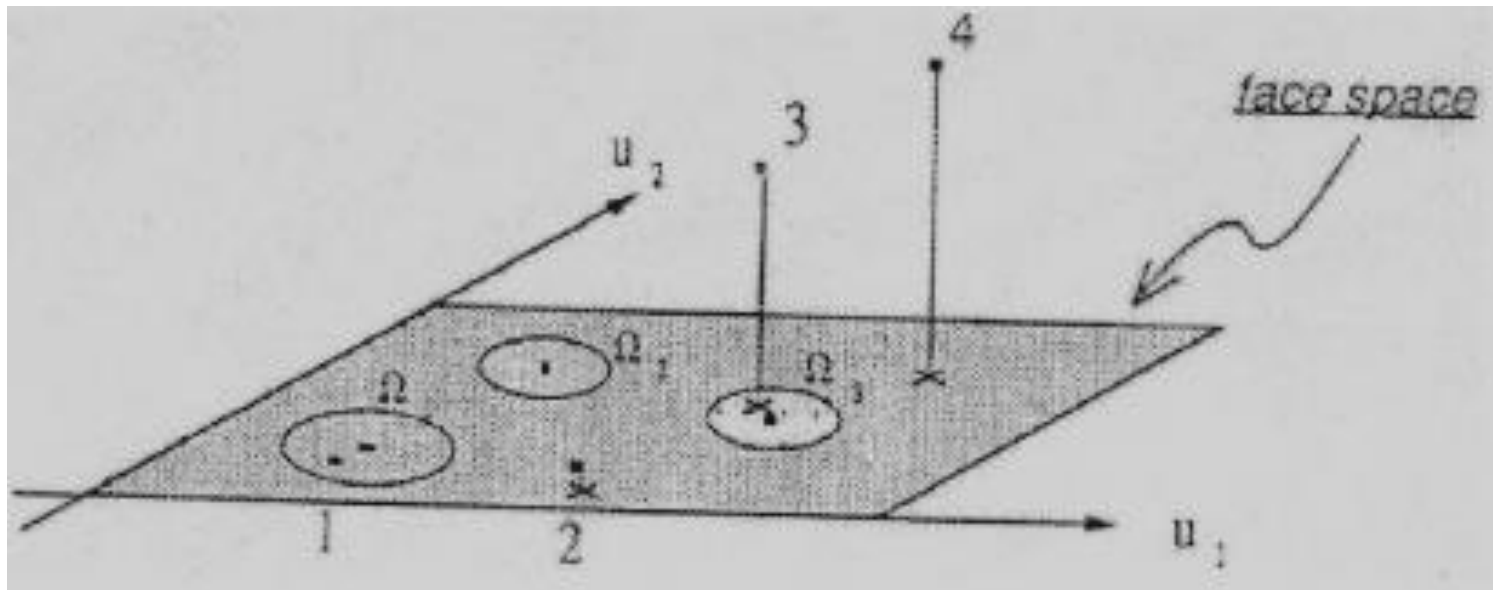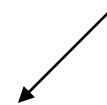
Principal Component Analysis (PCA)

- Reconstruction of faces and non-faces

Principal Component Analysis (PCA)

- Applications
  - Face detection, tracking, and recognition

dffs

Principal Component Analysis (PCA)

- Problems in face recognition:

  - **Background** (de-emphasize the outside of the face – e.g., by multiplying the input image by a 2D Gaussian window centered on the face)
  - **Lighting conditions** (performance degrades with light changes)
  - **Scale** (performance decreases quickly with changes to head size)
    - multi-scale eigenspaces
    - scale input image to multiple sizes
  - **Orientation** (performance decreases but not as fast as with scale changes)
    - plane rotations can be handled
    - out-of-plane rotations are more difficult to handle

Principal Component Analysis (PCA)

- Dataset

  - 16 subjects
  - 3 orientations, 3 sizes
  - 3 lighting conditions, 6 resolutions (512x512 ... 16x16)
  - **Total number of images**: 2,592

Principal Component Analysis (PCA)

- Experiment 1

  - Used various sets of 16 images for training
    - One image/person, taken under the same conditions
  - Classify the rest images as one of the 16 individuals
  - 7 eigenfaces were used
  - No rejections (i.e., no threshold on *difs*)

  - Performed a large number of experiments and averaged the results:
    - 96% correct averaged over **light variation**
    - 85% correct averaged over **orientation variation**
    - 64% correct averaged over **size variation**

Principal Component Analysis (PCA)

- ## Experiment 2

  - Considered rejections (i.e., by thresholding *difs*)
  - There is a tradeoff between correct recognition and rejections.
  - Adjusting the threshold to achieve 100% recognition accuracy resulted in:
    - 19% rejections while **varying lighting**
    - 39% rejections while **varying orientation**
    - 60% rejections while **varying size**

- ## Experiment 3

  - Reconstruction using partial information

Principal Component Analysis (PCA)

- PCA and classification

  - PCA is **not** always an optimal dimensionality-reduction procedure for classification purposes.

- Multiple classes and PCA

  - Suppose there are C classes in the training data.
  - PCA is based on the sample covariance which characterizes the scatter of the entire data set, *irrespective of class-membership*.
  - The projection axes chosen by PCA might not provide good discrimination power.

Principal Component Analysis (PCA)

- PCA and classification (cont'd)
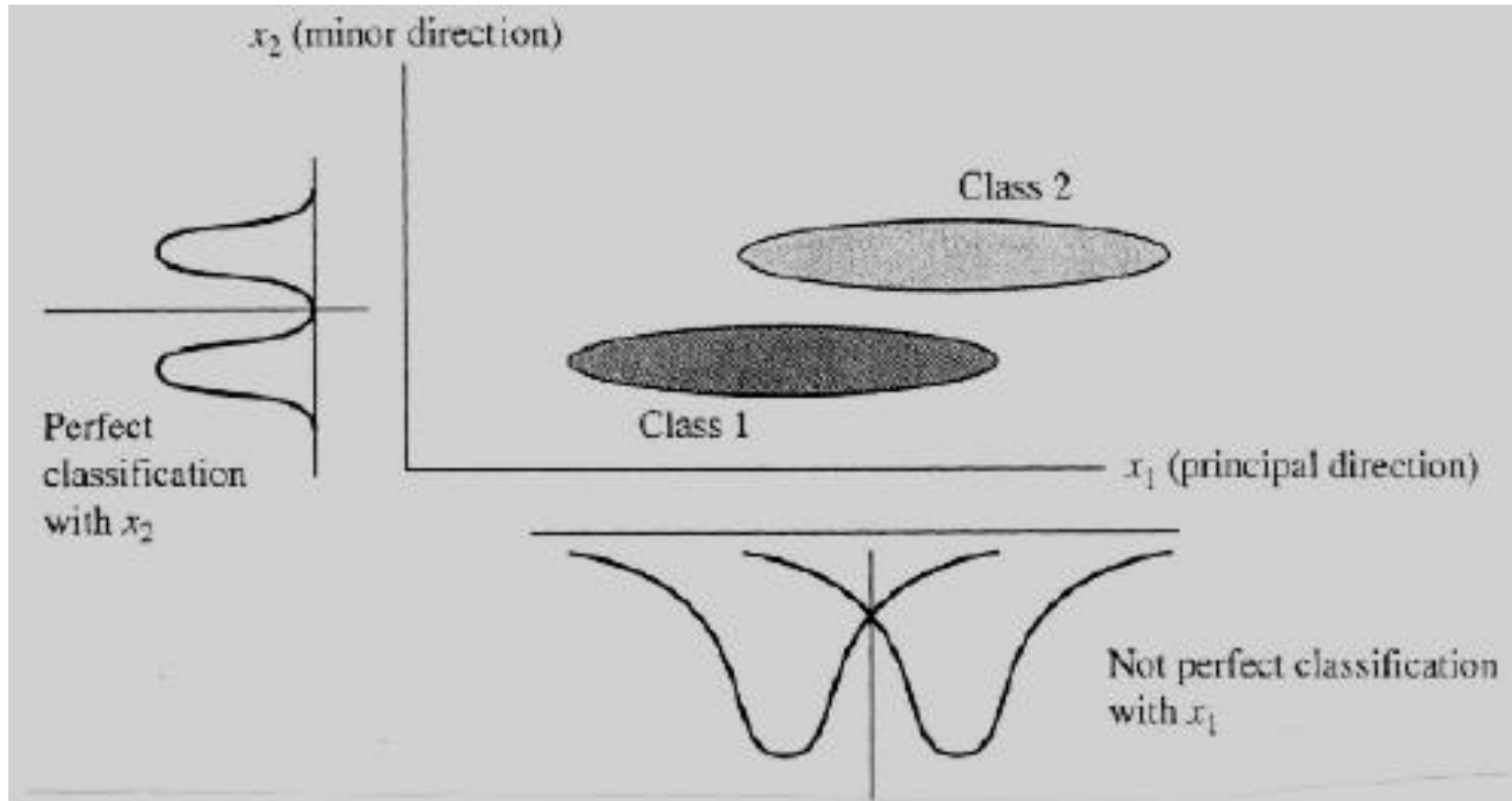
Linear Discriminant Analysis (LDA)

- What is the goal of LDA?

  - Perform dimensionality reduction **"while preserving as much of the class discriminatory information as possible"**.

  - Seeks to find directions along which the classes are best separated.

  - Takes into consideration the scatter *within-classes* but also the scatter *between-classes*.

  - More capable of distinguishing image variation due to <u>identity</u> from variation due to other sources such as <u>illumination</u> and <u>expression</u>.
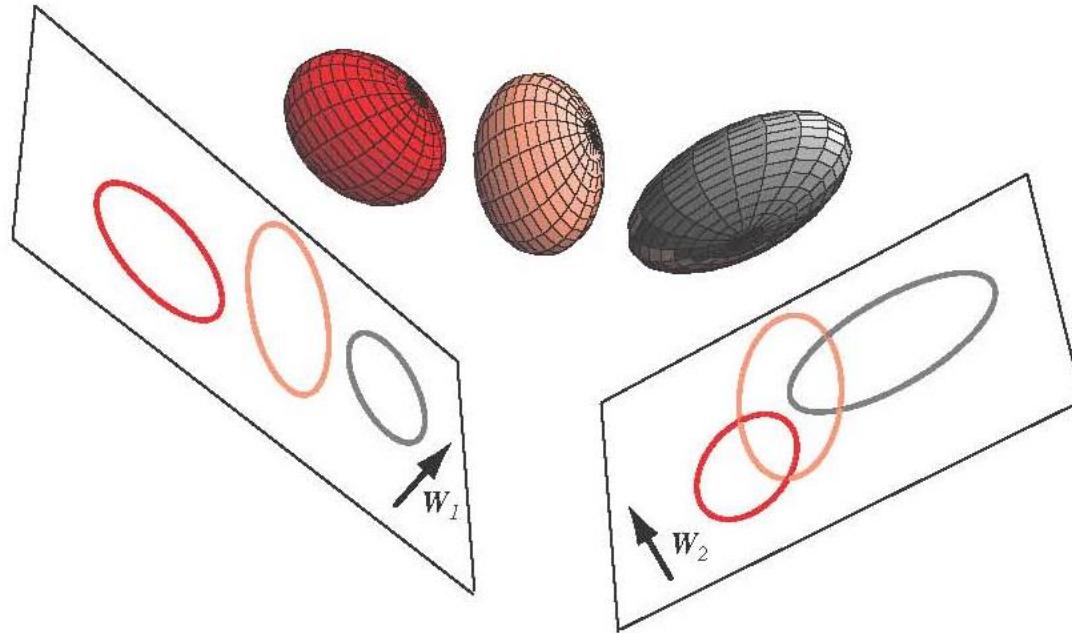
Linear Discriminant Analysis (LDA)

FIGURE 3.6. Three three-dimensional distributions are projected onto two-dimensional subspaces, described by a normal vectors W_1 and W_2. Informally, multiple discriminant methods seek the optimum such subspace, that is, the one with the greatest separation of the projected distributions for a given total within-scatter matrix, here as associated with W_1. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Linear Discriminant Analysis (LDA)

- Notation

- Suppose there are C classes
- Let $\boldsymbol{\mu}_i$ be the mean vector of class i, $i = 1, 2, \ldots, C$
- Let M_i be the number of samples within class i, $i = 1, 2, \ldots, C$,
- Let $M = \sum_{i=0}^{C} M_i$ be the total number of samples. and

Within-class scatter matrix:

$$S_w = \sum_{i=1}^{C} \sum_{j=1}^{M_i} (x_j - \boldsymbol{\mu}_i)(x_j - \boldsymbol{\mu}_i)^T$$

Between-class scatter matrix:

(each sub-matrix has rank 1 or less, i.e., outer product of two vectors)

(S_b has at most rank C-1)

$$S_b = \sum_{i=1}^{C} (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T$$

$$\boldsymbol{\mu} = 1/C \sum_{i=1}^{C} \boldsymbol{\mu}_i \quad \text{(mean of entire data set)}$$

Linear Discriminant Analysis (LDA)

- Methodology

projection matrix

$$\mathbf{y} = U^T \mathbf{x}$$

- LDA computes a transformation that maximizes the between-class scatter while minimizing the within-class scatter:

$$\max \frac{|U^T S_b U|}{|U^T S_w U|} = \max \frac{|\tilde{S}_b|}{|\tilde{S}_w|}$$

products of eigenvalues !

$\tilde{S}_b, \tilde{S}_w$: scatter matrices of the projected data $\mathbf{y}$

Linear Discriminant Analysis (LDA)

- Linear transformation implied by LDA

  - The LDA solution is given by the eigenvectors of the _generalized eigenvector problem_:

  $$S_B u_k = \lambda_k S_w u_k$$

  - The linear transformation is given by a matrix _U_ whose columns are the eigenvectors of the above problem (i.e., called _Fisherfaces_).

  $$\begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_K \end{bmatrix} = \begin{bmatrix} u_1^T \\ u_2^T \\ \dots \\ u_K^T \end{bmatrix} (x - \boldsymbol{\mu}) = U^T (x - \boldsymbol{\mu})$$

  - **Important:** Since S_b has at most rank C-1, the max number of eigenvectors with non-zero eigenvalues is C-1 (i.e., max dimensionality of sub-space is C-1)

Linear Discriminant Analysis (LDA)

- Does S_w^{-1} always exist?

  - If S_w is non-singular, we can obtain a conventional eigenvalue problem by writing:

$$S_w^{-1} S_B u_k = \lambda_k u_k$$

  - In practice, S_w is often singular since the data are image vectors with large dimensionality while the size of the data set is much smaller ($M << N$)

Linear Discriminant Analysis (LDA)

- Does S_w^{-1} always exist? – cont.
  - To alleviate this problem, we can use PCA first:

    1) PCA is first applied to the data set to reduce its dimensionality.

    $$\begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_N \end{bmatrix} --> PCA --> \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_K \end{bmatrix}$$

    2) LDA is then applied to find the most discriminative directions:

    $$\begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_K \end{bmatrix} --> LDA --> \begin{bmatrix} z_1 \\ z_2 \\ \dots \\ z_{C-1} \end{bmatrix}$$

Linear Discriminant Analysis (LDA)

- Case Study: Using Discriminant Eigenfeatures for Image Retrieval

  − D. Swets, J. Weng, "Using Discriminant Eigenfeatures for Image Retrieval", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 831-836, 1996.

- Content-based image retrieval

  − The application being studied here is *query-by-example* image retrieval.

  − The paper deals with the problem of *selecting a good set of image features* for content-based image retrieval.

Linear Discriminant Analysis (LDA)

- Assumptions
  - "Well-framed" images are required as input for training and query-by-example test probes.
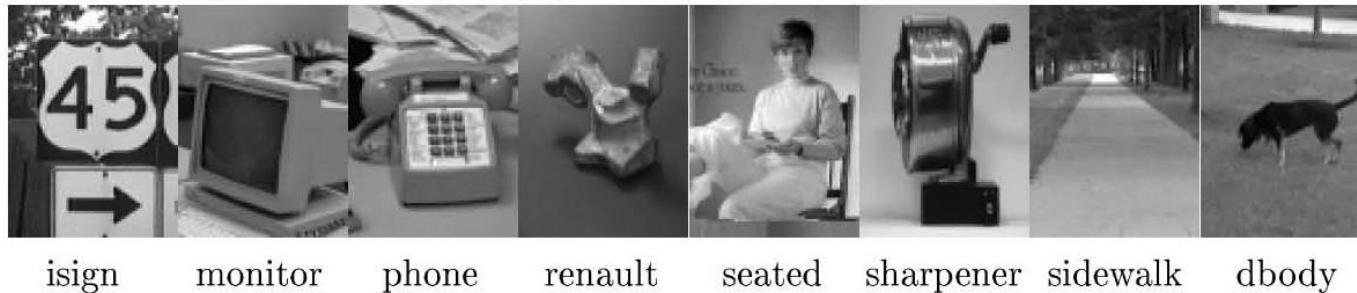 - Only a small variation in the size, position, and orientation of the objects in the images is allowed.

isign   monitor   phone   renault   seated   sharpener   sidewalk   dbody

Figure 6. Representatives of training images from some of the various classes learned. Objects of interest are at the center of the fovea images. In the learning phase for this experiment, the training images were generated using manual extraction of the areas of interest.

Linear Discriminant Analysis (LDA)

- Some terminology
  - <u>Most Expressive Features (MEF)</u>: the features (projections) obtained using PCA.
  - <u>Most Discriminating Features (MDF)</u>: the features (projections) obtained using LDA.

- Numerical problems
  - When computing the eigenvalues/eigenvectors of $S_w^{-1}S_B u_k = \lambda_k u_k$ numerically, the computations can be <u>unstable</u> since $S_w^{-1}S_B$ is not always symmetric.
  - See paper for a way to find the eigenvalues/eigenvectors in a stable way.

Linear Discriminant Analysis (LDA)

- Factors unrelated to classification
  - MEF vectors show the tendency of PCA to capture major variations in the training set such as lighting direction.
  - MDF vectors discount those factors unrelated to classification.

MEF 1   MEF 2   MEF 3   MEF 4   MDF 1   MDF 2   MDF 3   MDF 4
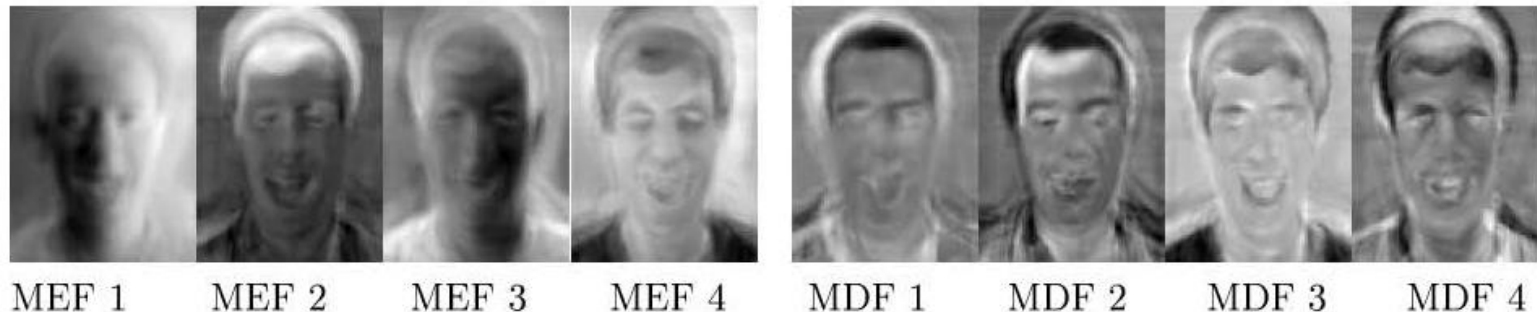
Figure 2. A sample of MEF and MDF vectors treated as images. The MEF vectors show the tendency of the principal components to capture major variations in the training set, such as lighting direction. The MDF vectors show the ability of the MDFs to discount those factors unrelated to classification. The training images used to produce these vectors are courtesy of the Weizmann Institute.

Linear Discriminant Analysis (LDA)

- ## Clustering effect
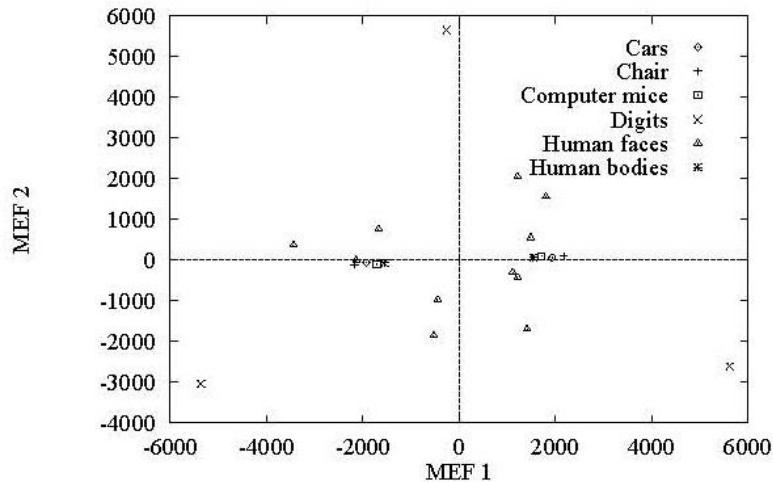
(a) MEF space
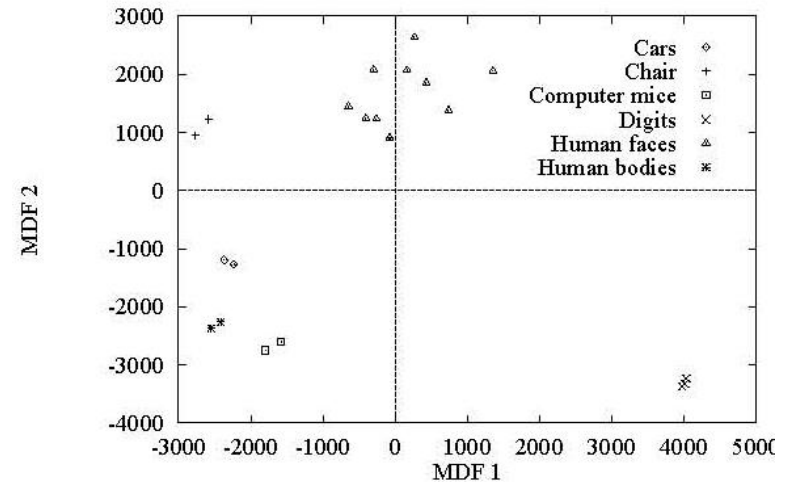
(b) MDF space

- ## Methodology
    1) Generate the set of MEFs/MDFs for each image in the training set.
    2) Given an query image, compute its MEFs/MDFs using the same procedure.
    3) Find the **k closest neighbors** for retrieval (e.g., using Euclidean distance).

Linear Discriminant Analysis (LDA)

- Experiments and results
  - <u>Face images</u>
    - A set of face images was used with 2 expressions, 3 lighting conditions.
    - Testing was performed using a disjoint set of images:
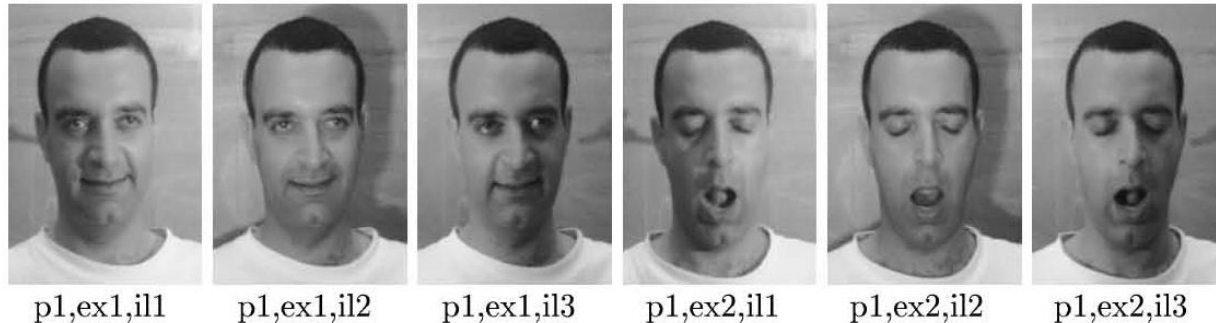   - One image, randomly chosen, from each individual.

p1,ex1,il1   p1,ex1,il2   p1,ex1,il3   p1,ex2,il1   p1,ex2,il2   p1,ex2,il3

Figure 5. A sample of the Weizmann Institute face data. The frontal images for an individual contain two expressions; for each expression, a set of three images with differing lighting conditions forms the set of images available for an individual.

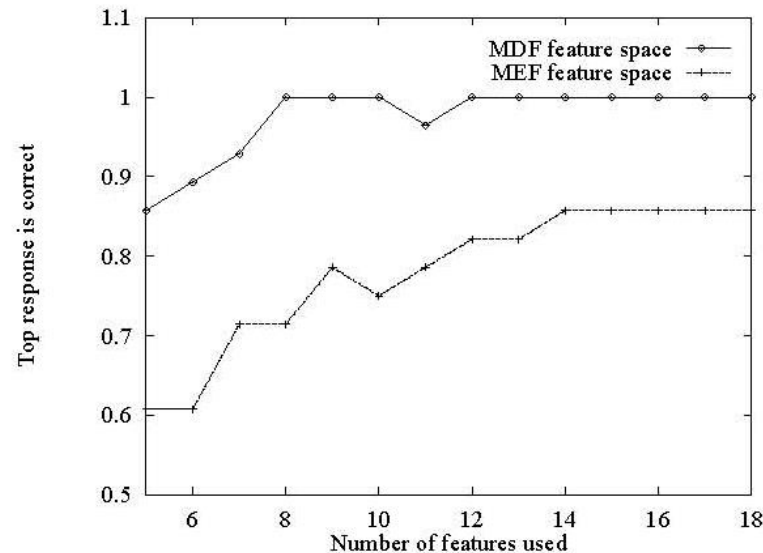# Linear Discriminant Analysis (LDA)

Figure 4. The performance of the system for different numbers of MEF and MDF features, respectively. The number of features from the subspace used was varied to show how the MDF subspace outperforms the MEF subspace. 95% of the variance for the MDF subspace was attained when 15 features were used; 95% of variance for the MEF subspace did not occur until 37 features were used. Using 95% of the MEF variance resulted in an 89% recognition rate, and that rate was not improved using more features.

Linear Discriminant Analysis (LDA)

- – <u>Examples of correct search probes</u>
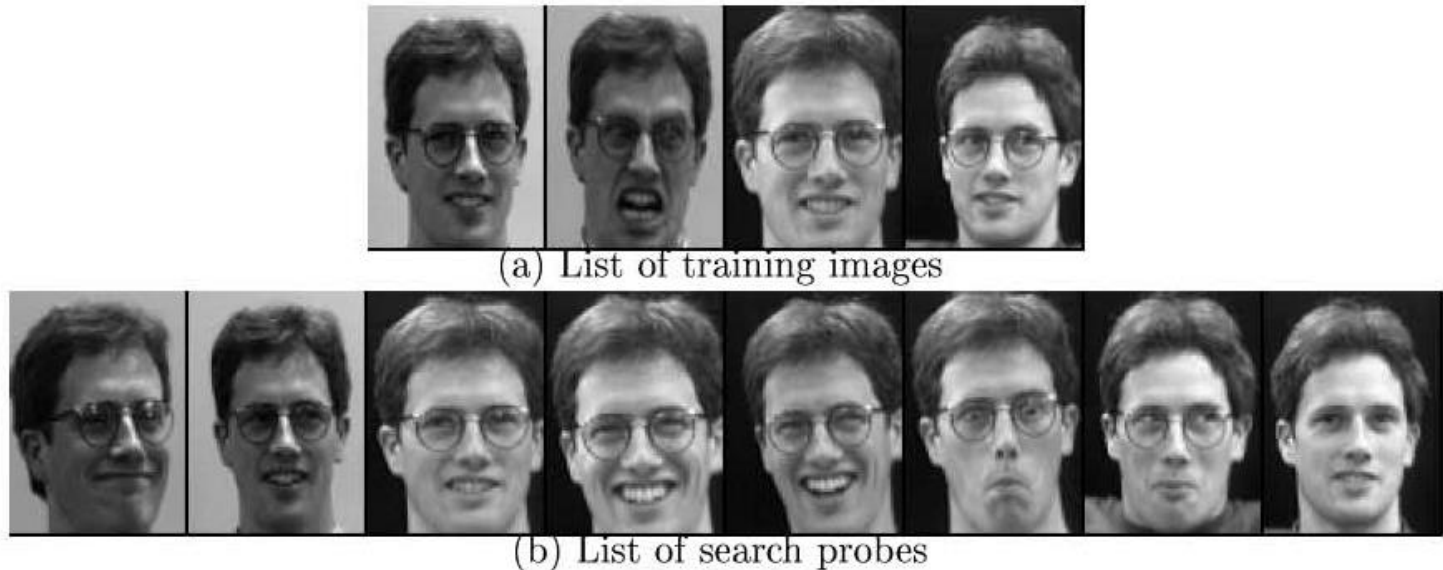

(a) List of training images

(b) List of search probes

Figure 8. Example of how well within-class variation is handled. The system correctly retrieved images from the class defined by the training samples for each of the search probes.

Linear Discriminant Analysis (LDA)
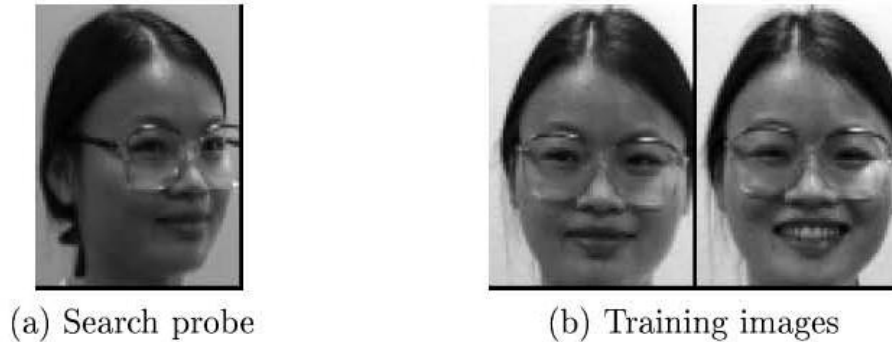
– Example of a failed search probe

(a) Search probe   (b) Training images

Figure 7. Example of a failed search probe. The retrieval failed to select the appropriate class due to a lack of 3D rotation in the set of training images.

Linear Discriminant Analysis (LDA)

- Case Study: PCA versus LDA
  - A. Martinez, A. Kak, "PCA versus LDA", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228-233, 2001.

- Is LDA always better than PCA?

  - There has been a tendency in the computer vision community to prefer LDA over PCA.
  - This is mainly because LDA deals directly with discrimination between classes while PCA does not pay attention to the underlying class structure.
  - Main results of this study:

    (1) When the training set is small, PCA can outperform LDA.
    (2) When the number of samples is large and representative for each class, LDA outperforms PCA.

Linear Discriminant Analysis (LDA)

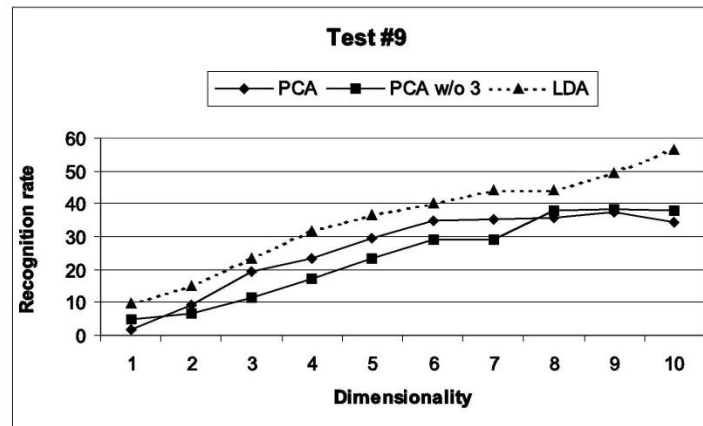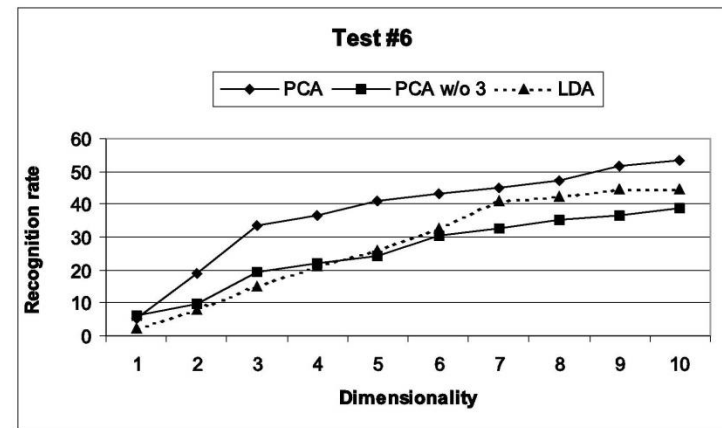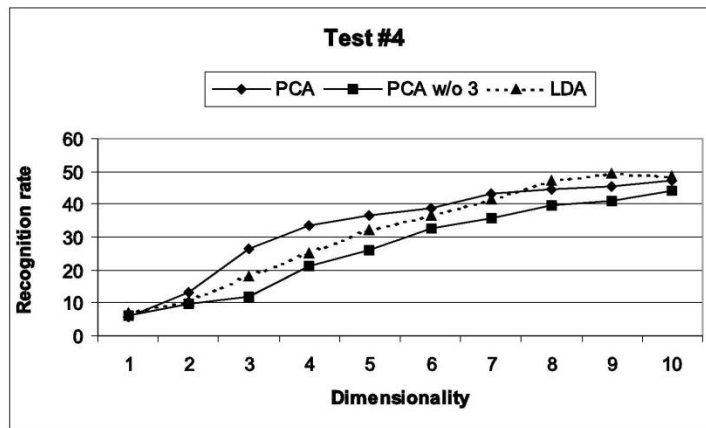- Is LDA always better than PCA? – cont.

Fig. 3. Images of one subject in the AR face database. The images (a)-(m) were taken during one session and the images (n)-(z) at a different session.

Linear Discriminant Analysis (LDA)

- Is LDA always better than PCA? – cont.

LDA is not always better when training set is small

Linear Discriminant Analysis (LDA)

- Is LDA always better than PCA? – cont.

LDA outperforms PCA when training set is large