

PCA

explained within the context of Face Recognition

Berrin Yanikoglu

FENS

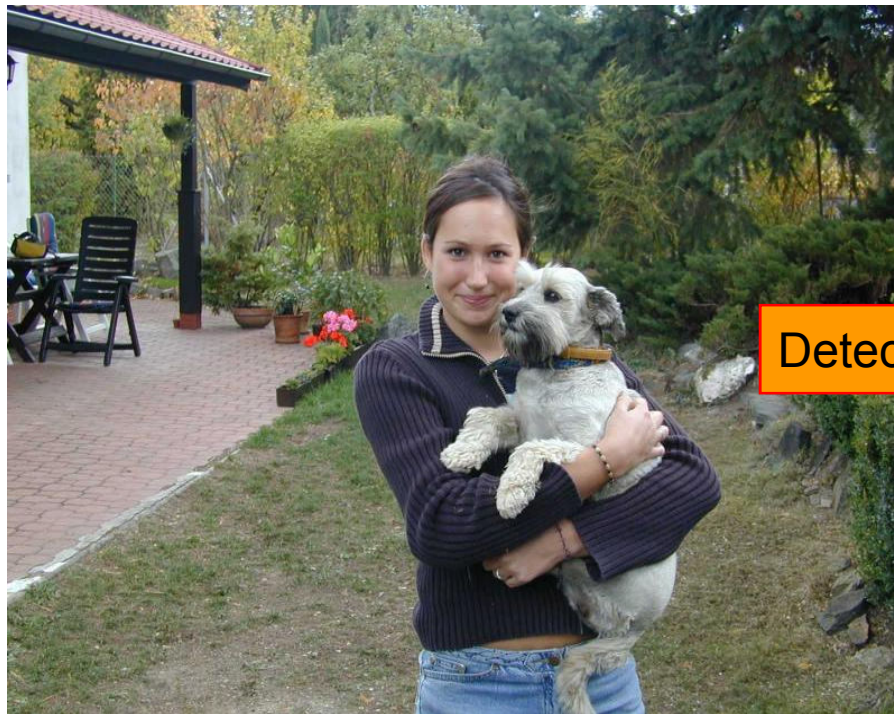
Computer Science & Engineering
Sabancı University

Updated Dec. 2012

Overview

- Definition: Face recognition, verification, tracking...
- **Feature subspaces: PCA**
- Side info: Interesting findings about human face recognition

Face detection and recognition



Detection



Recognition

“Sally”

Applications of Face Recognition

- Surveillance
- Digital photography
- Album organization

Recording

Detecting....

Matching with Database

Name: Alireza,
Date: 25 My 2007 15:45
Place: Main corridor

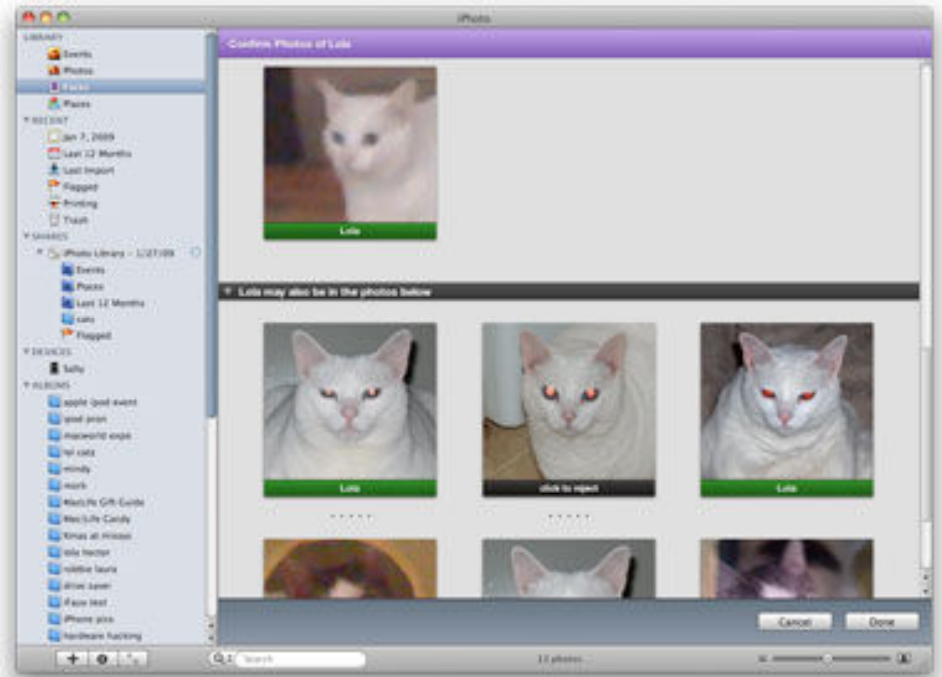
Name: **Unknown**
Date: 25 My 2007 15:45
Place: Main corridor

Report



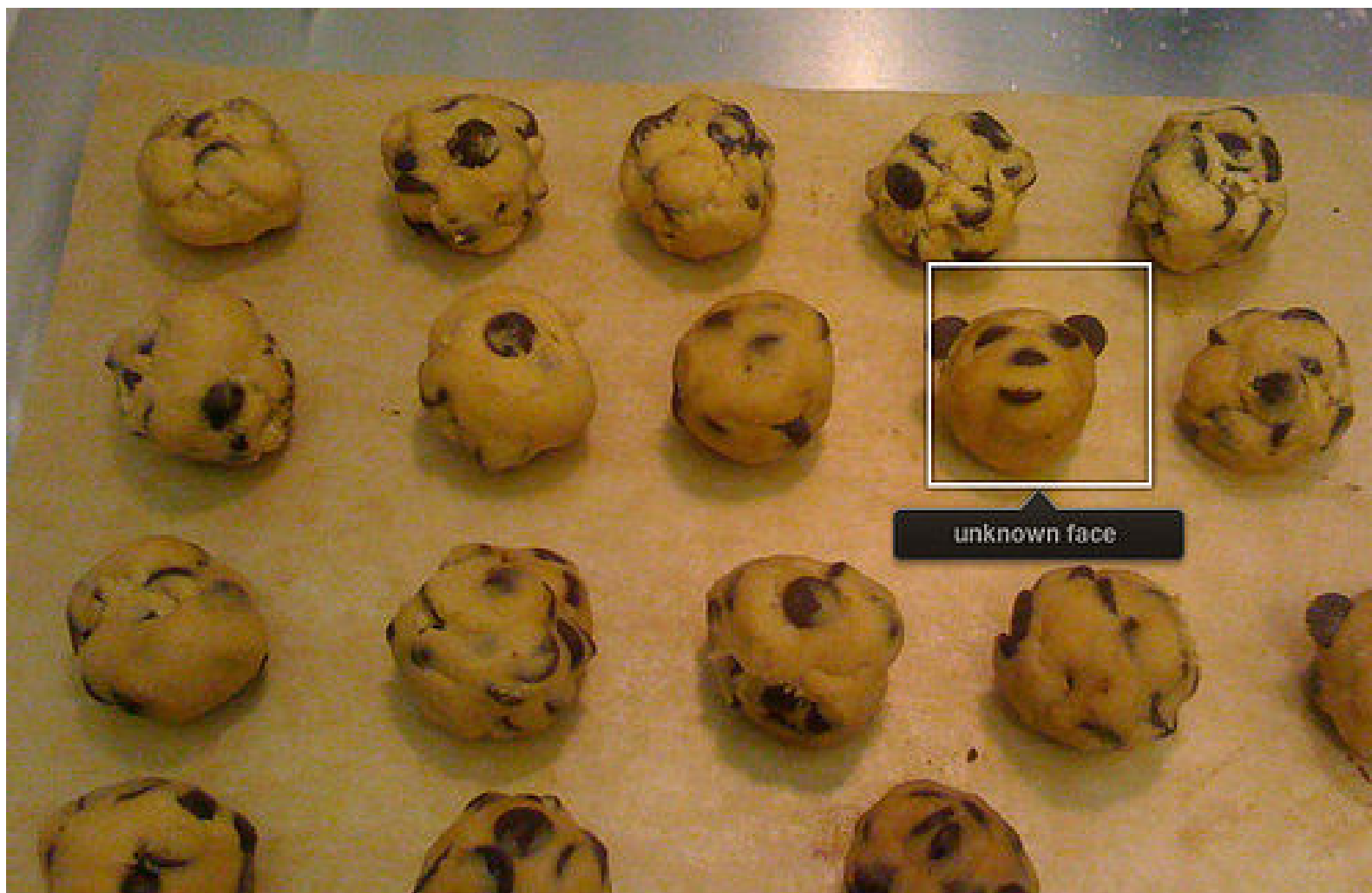
Consumer application: iPhoto 2009

- Can be trained to recognize pets!



http://www.maclife.com/article/news/iphotos_faces_recognizes_cats

Consumer application: iPhoto 2009



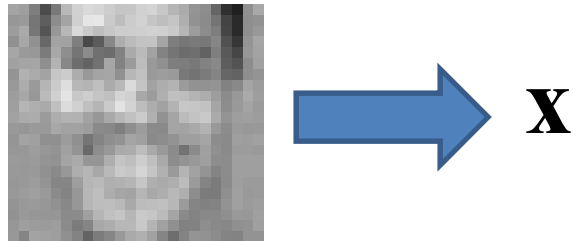
Error measure

- Face **Detection/Verification**
 - False Positives (%)
 - False Negatives (%)
- Face **Recognition**
 - Top-N rates (%)
 - Open/closed set problems
- Sources of variations:

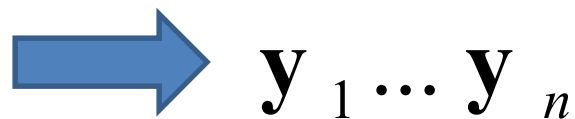


Face recognition

1. Treat pixels as a vector



2. Recognize face by nearest neighbor



$$\min_k \left\| \mathbf{y}_k^T - \mathbf{x} \right\|$$

The space of face images

- When viewed as vectors of pixel values, face images are extremely high-dimensional
 - 100x100 image = 10,000 dimensions
 - Large memory and computational requirements
- But very few 10,000-dimensional vectors are valid face images
- We want to reduce dimensionality and effectively model the subspace of face images



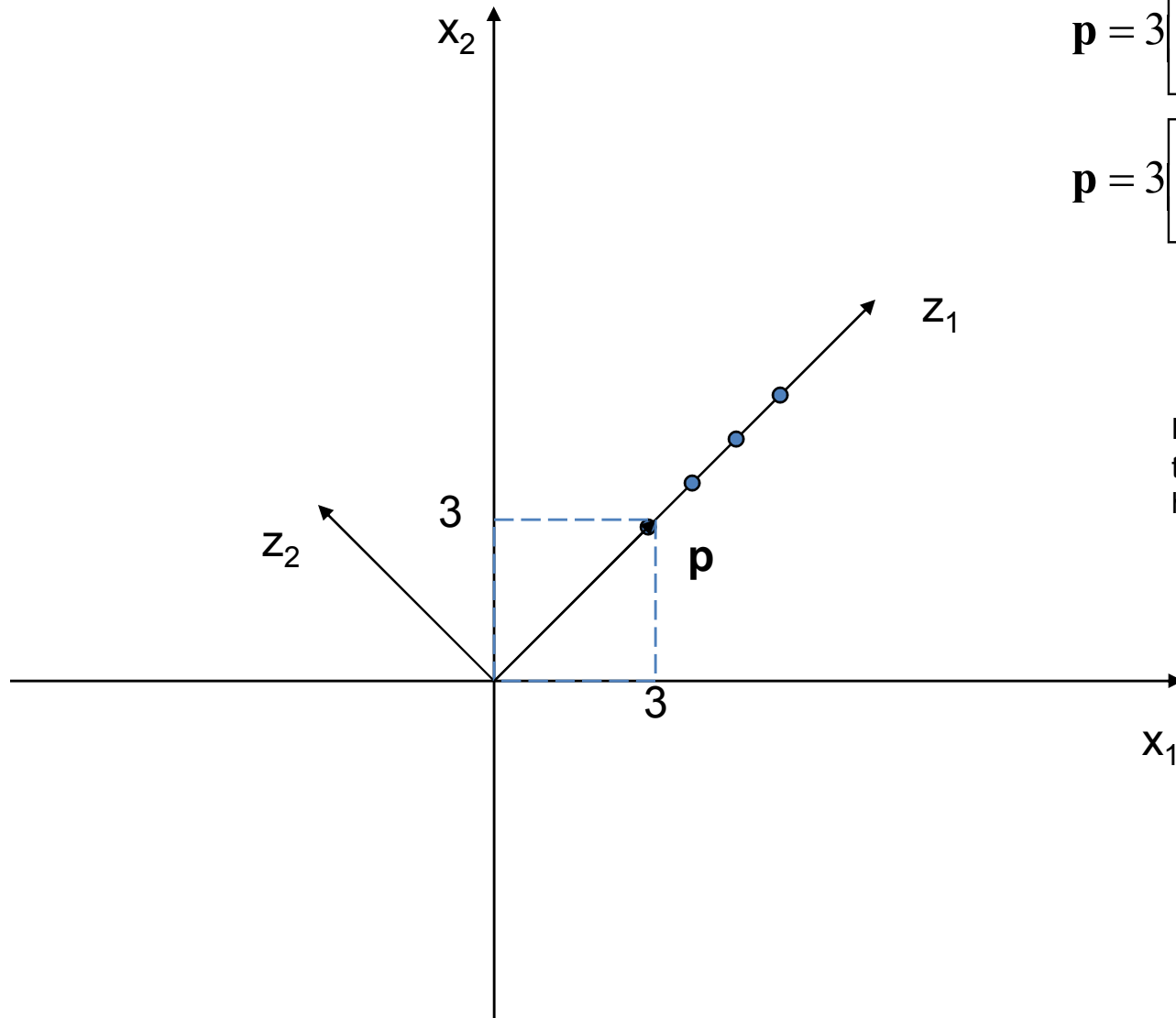
Principal Component Analysis (PCA)

- Pattern recognition in high-dimensional spaces
 - Problems arise when performing recognition in a high-dimensional space (**curse of dimensionality**).
 - Significant improvements can be achieved by first mapping the data into a *lower-dimensional sub-space*.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_N \end{bmatrix} \xrightarrow{\text{dimensionality reduction}} \mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \dots \\ z_K \end{bmatrix} \quad \text{where } K \ll N.$$

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

Change of basis

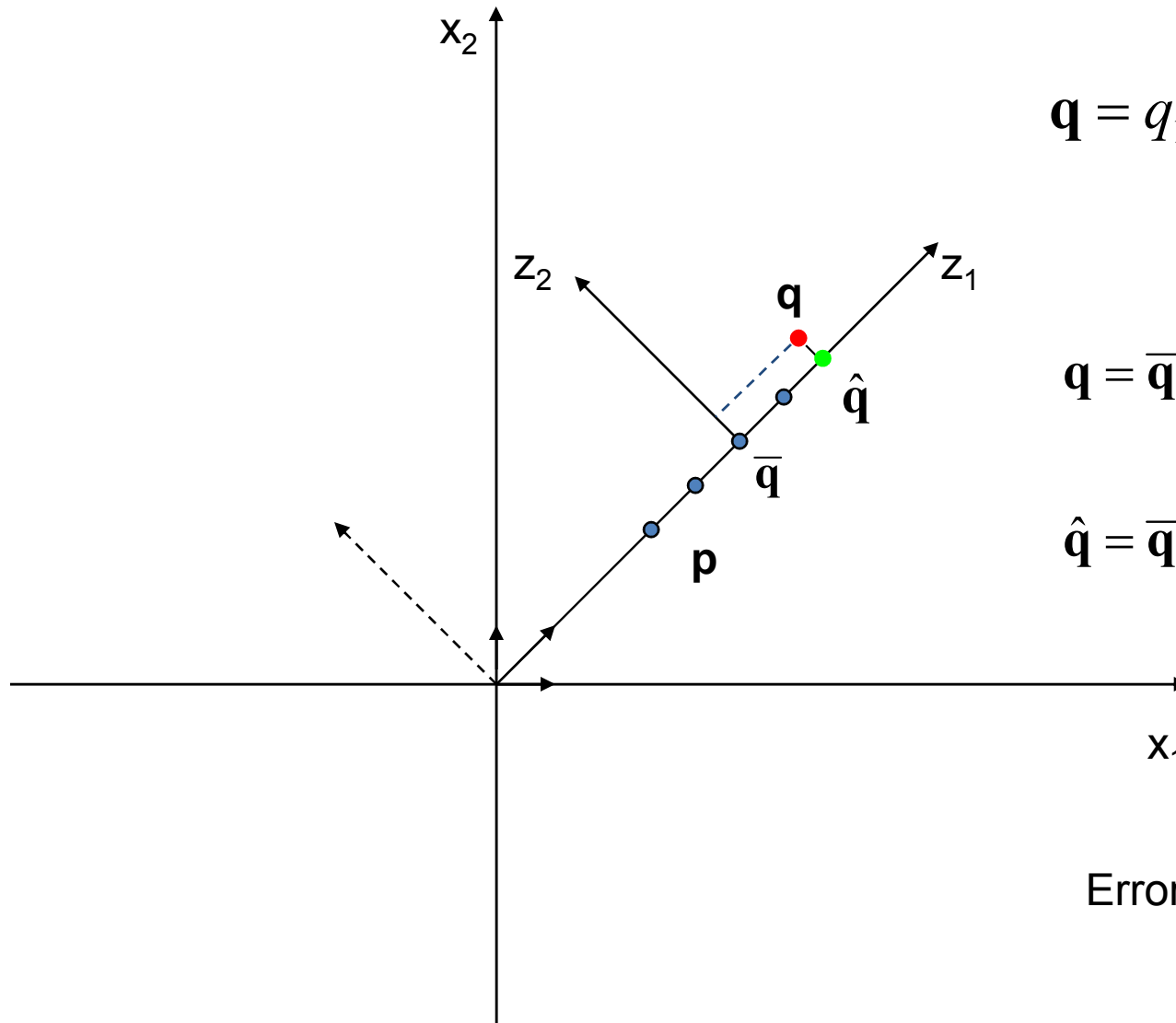


$$\mathbf{p} = 3 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 3 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{p} = 3 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + 0 \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

Note that the vector $[1 \ 1]$ is longer than the vectors $[1 \ 0]$ or $[0 \ 1]$; hence the coefficient is still 3.

Dimensionality reduction



$$\mathbf{q} = q_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + q_2 \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$\mathbf{q} = \bar{\mathbf{q}} + b_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + b_2 \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$\hat{\mathbf{q}} = \bar{\mathbf{q}} + b_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Error: $\| \mathbf{q} - \hat{\mathbf{q}} \|$

Principal Component Analysis (PCA)

- PCA allows us to compute a linear transformation that maps data from a high dimensional space to a lower dimensional sub-space.

$$z_1 = u_{11}x_1 + u_{12}x_2 + \dots + u_{1N}x_N$$

$$z_2 = u_{21}x_1 + u_{22}x_2 + \dots + u_{2N}x_N$$

...

$$z_K = u_{K1}x_1 + u_{K2}x_2 + \dots + u_{KN}x_N$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_N \end{bmatrix}$$

- In short,

$$\mathbf{z} = \mathbf{W}\mathbf{x} \quad \text{where} \quad \mathbf{W} = \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1N} \\ u_{21} & u_{22} & \dots & u_{2N} \\ \dots & \dots & \dots & \dots \\ u_{K1} & u_{K2} & \dots & u_{KN} \end{bmatrix}$$

Principal Component Analysis (PCA)

- Lower dimensionality basis

- Approximate vectors by finding a basis in an appropriate lower dimensional space.

(1) Higher-dimensional space representation:

$$\mathbf{X} = x_1 \mathbf{V}_1 + x_2 \mathbf{V}_2 + \cdots + x_N \mathbf{V}_N$$

$\mathbf{V}_1, \cdots, \mathbf{V}_N$ are the basis vectors of the N-dimensional space

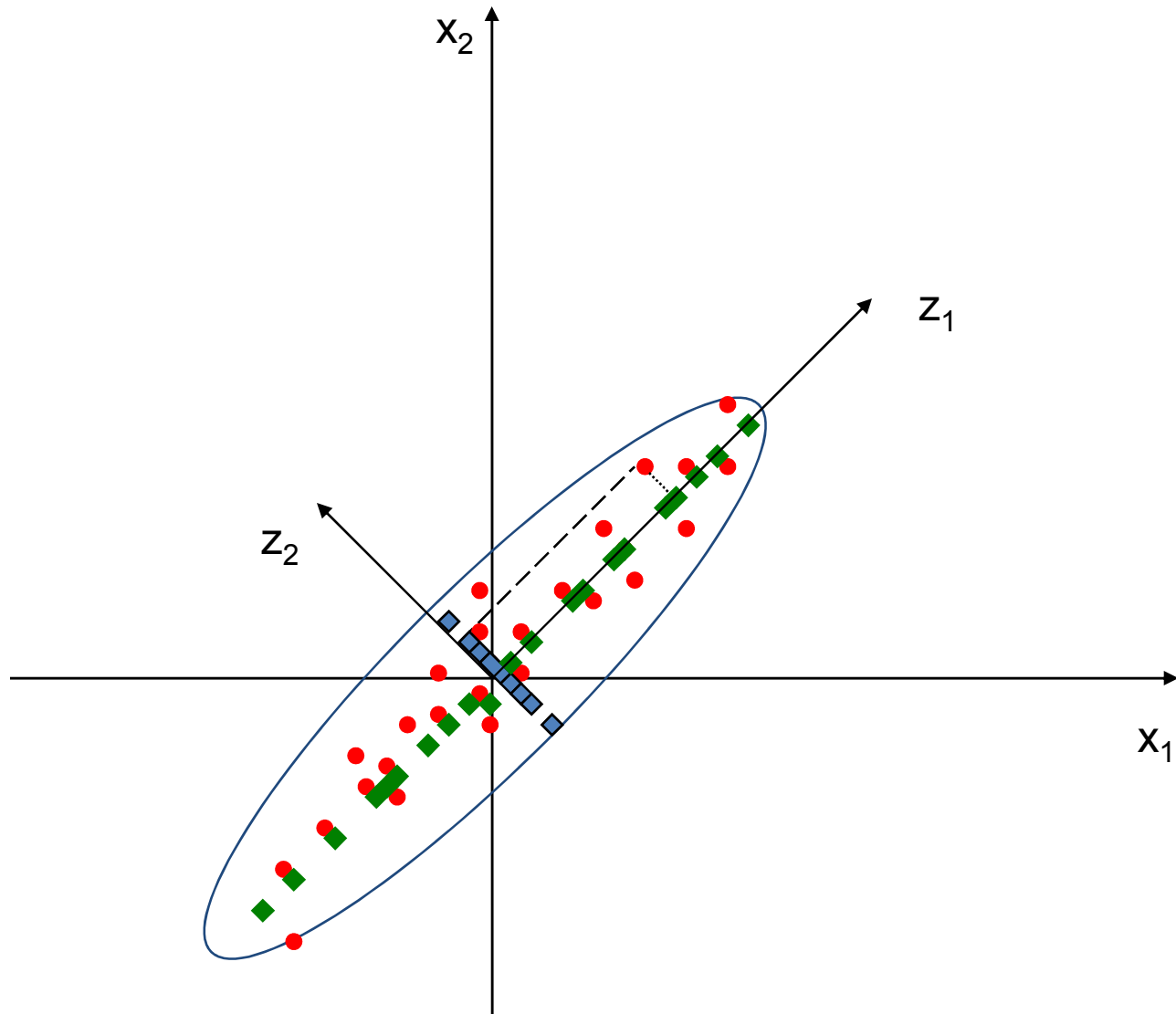
(2) Lower-dimensional space representation:

$$\hat{\mathbf{X}} = z_1 \mathbf{u}_1 + z_2 \mathbf{u}_2 + \cdots + z_K \mathbf{u}_K$$

$\mathbf{u}_1, \cdots, \mathbf{u}_K$ are the basis vectors of the K-dimensional space

Note: If $N=K$, then $\mathbf{X} = \hat{\mathbf{X}}$

Illustration for projection, variance and bases



Principal Component Analysis (PCA)

- Dimensionality reduction implies information loss !!
 - Want to preserve as much information as possible, that is:

$$\text{minimize } \|x - \hat{x}\| \text{ (error)}$$

- How to determine the best lower dimensional sub-space?

Principal Components Analysis (PCA)

- The projection of \mathbf{x} on the direction of \mathbf{u} is: $z = \mathbf{u}^T \mathbf{x}$
- Find the vector \mathbf{u} such that $Var(z)$ is maximized:

$$\begin{aligned} Var(z) &= Var(\mathbf{u}^T \mathbf{x}) \\ &= E[(\mathbf{u}^T \mathbf{x} - \mathbf{u}^T \boldsymbol{\mu})(\mathbf{u}^T \mathbf{x} - \mathbf{u}^T \boldsymbol{\mu})^T] \\ &= E[(\mathbf{u}^T \mathbf{x} - \mathbf{u}^T \boldsymbol{\mu})^2] && // \text{since } (\mathbf{u}^T \mathbf{x} - \mathbf{u}^T \boldsymbol{\mu}) \text{ is a scalar} \\ &= E[(\mathbf{u}^T \mathbf{x} - \mathbf{u}^T \boldsymbol{\mu})(\mathbf{u}^T \mathbf{x} - \mathbf{u}^T \boldsymbol{\mu})] \\ &= E[\mathbf{u}^T (\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T \mathbf{u}] \\ &= \mathbf{u}^T E[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T] \mathbf{u} \\ &= \mathbf{u}^T \boldsymbol{\Sigma} \mathbf{u} \end{aligned}$$

where $\boldsymbol{\Sigma} = E[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T]$ (covariance of \mathbf{x})

In other words, we see that maximizing $Var(z)$ is equivalent to maximizing $\mathbf{u}^T \boldsymbol{\Sigma} \mathbf{u}$ where \mathbf{u} is a candidate direction we can project the data and $\boldsymbol{\Sigma}$ is the covariance matrix of the original data.

- The next 3 slides show that the direction \mathbf{u} that maximizes $\text{Var}(z)$ is the eigenvectors of Σ .
 - *You are not responsible of understanding/knowing this derivation.*
- The eigenvectors with the largest eigenvalue results in the largest variance.
- As a result, **we start picking the new basis vectors** (new directions to project the data), **from the eigenvectors of the cov. matrix in order** (largest eigenvalue is first, then next largest etc.)
- In this process, we use **unit vectors** to represent each direction, to remove ambiguity.

- The following 3 slides require understanding of matrix operation, Lagrange multipliers and Eigenvalues.
- You are **are not required** in CS412/512 to understand this material, read only if interested.

Principal Component Analysis - Advanced

- Same thing, a bit more detailed:

Maximize $\frac{1}{N} \sum_{i=1}^N \underbrace{\mathbf{u}^T (\mathbf{x}_i - \mu)}_{\text{Projection of data point}} (\mathbf{u}^T (\mathbf{x}_i - \mu))^T$ subject to $\|\mathbf{u}\|=1$

$$= \mathbf{u}^T \underbrace{\left[\frac{1}{N} \sum_{i=1}^N (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T \right]}_{\text{Covariance matrix of data}} \mathbf{u}$$

$$= \mathbf{u}^T \Sigma \mathbf{u}$$

- Maximize $\text{Var}(z) = \mathbf{u}^T \Sigma \mathbf{u}$ subject to $\|\mathbf{u}\|=1$

$$\max_{\mathbf{u}_1} \mathbf{u}_1^T \Sigma \mathbf{u}_1 - \alpha (\mathbf{u}_1^T \mathbf{u}_1 - 1)$$

α, β :Lagrange multipliers

- Taking the derivative w.r.t \mathbf{u}_1 , and setting it equal to 0, we get:

$$\Sigma \mathbf{u}_1 = \alpha \mathbf{u}_1$$

$\Rightarrow \mathbf{u}_1$ is an eigenvector of Σ

- Choose the eigenvector with the largest eigenvalue for $\text{Var}(z)$ to be maximum
- Second principal component: Max $\text{Var}(z_2)$, s.t., $\|\mathbf{u}_2\|=1$ and it is orthogonal to \mathbf{u}_1

$$\max_{\mathbf{u}_2} \mathbf{u}_2^T \Sigma \mathbf{u}_2 - \alpha (\mathbf{u}_2^T \mathbf{u}_2 - 1) - \beta (\mathbf{u}_2^T \mathbf{u}_1 - 0)$$

- Similar analysis shows that, $\Sigma \mathbf{u}_2 = \alpha \mathbf{u}_2$
 $\Rightarrow \mathbf{u}_2$ is another eigenvector of Σ and so on.

- Maximize $\text{var}(z) = \mathbf{u}^T \Sigma \mathbf{u}$
- Consider the eigenvectors of Σ for which
- $\Sigma \mathbf{u} = \lambda \mathbf{u}$ where \mathbf{u} is an **eigenvector** of Σ and λ is the corresponding **eigenvalue**.
- Multiplying by \mathbf{u}^T :
$$\mathbf{u}^T \Sigma \mathbf{u} = \mathbf{u}^T \lambda \mathbf{u} = \lambda \mathbf{u}^T \mathbf{u} = \lambda \quad \text{for } \|\mathbf{u}\|=1.$$

=> Choose the eigenvector with the largest eigenvalue.

- So now that we know the new basis vectors, we need to project our old data which is centered at the origin, to find the new coordinates.
- This projection is nothing but finding the individual coordinates of a point in the Cartesian space.
 - The point $[3\ 4]$ has x-coord of 3 and y-coord of 4 because if we project it onto $[1\ 0]$ and $[0\ 1]$ those are the values we find.

Principal Component Analysis (PCA)

- Given: N data points $\mathbf{x}_1, \dots, \mathbf{x}_N$ in \mathbb{R}^d
- We want to find a new set of features that are linear combinations of original ones:

$$u(\mathbf{x}_i) = \mathbf{u}^T(\mathbf{x}_i - \boldsymbol{\mu})$$

($\boldsymbol{\mu}$: mean of data points)

- Note that the unit vector \mathbf{u} is in \mathbb{R}^d (has the same dimension as the original data).

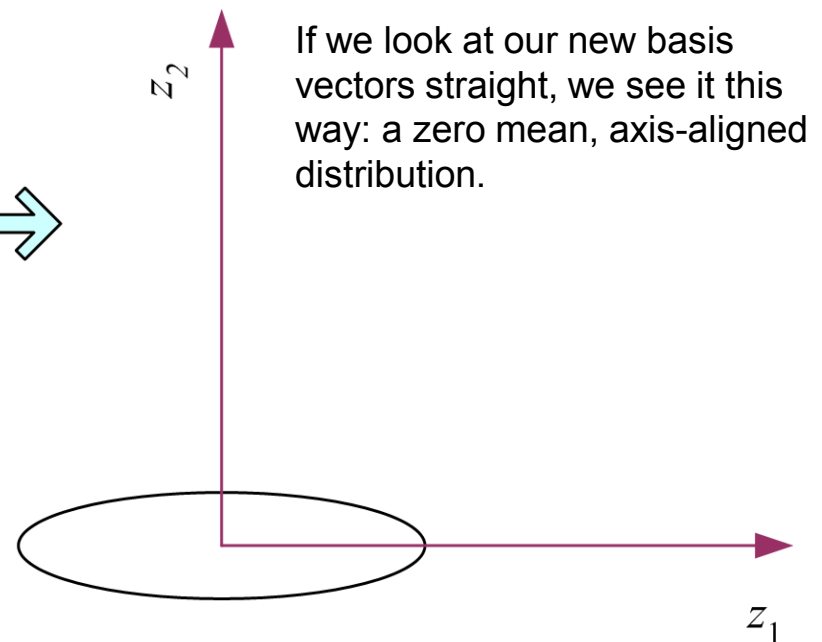
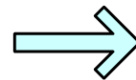
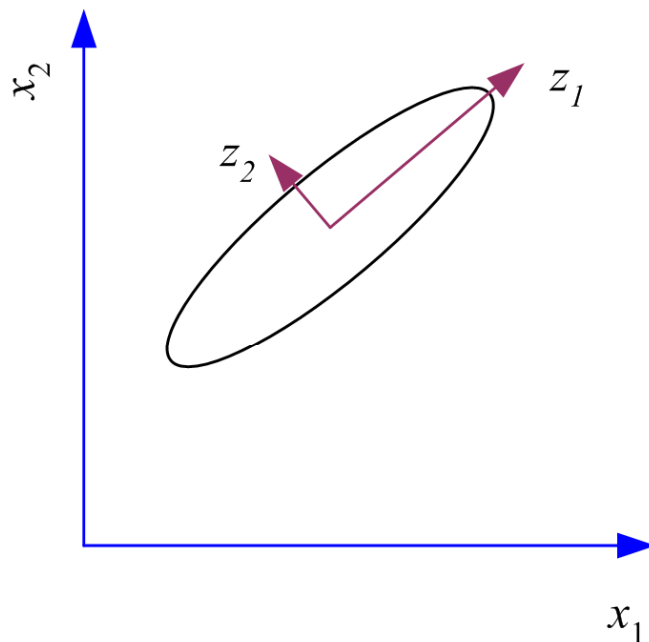
What PCA does

The transformation $\mathbf{z} = \mathbf{W}^T(\mathbf{x} - \boldsymbol{\mu})$

where the columns of \mathbf{W} are the eigenvectors of $\boldsymbol{\Sigma}$,

$\boldsymbol{\mu}$ is sample mean,

centers the data at the origin and rotates the axes



Eigenvalues of the covariance matrix - **Advanced**

The covariance matrix is symmetrical and it can always be diagonalized as:

$$\Sigma = W\Lambda W^T$$

where

- $W = [u_1, u_2, \dots, u_l]$ is the column matrix consisting of the **eigenvectors** of Σ ,
- $W^T = W^{-1}$
- Λ is the diagonal matrix whose elements are the **eigenvalues** of Σ .

Nice Summary of the PCA Algorithm

Principal Component Analysis (PCA)

- Methodology

- Suppose x_1, x_2, \dots, x_M are $N \times 1$ vectors

Step 1: $\bar{x} = \frac{1}{M} \sum_{i=1}^M x_i$

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

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

$$C = \frac{1}{M} \sum_{n=1}^M \Phi_n \Phi_n^T = AA^T$$

(sample **covariance** matrix, $N \times N$, 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, \dots, u_N

Principal Component Analysis (PCA)

- Methodology – cont.

- Since C is symmetric, u_1, u_2, \dots, 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):

$$x - \bar{x} = b_1 u_1 + b_2 u_2 + \dots + b_N u_N = \sum_{i=1}^N b_i u_i$$

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

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

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

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

Principal Component Analysis (PCA)

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

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

How many dimensions to select?

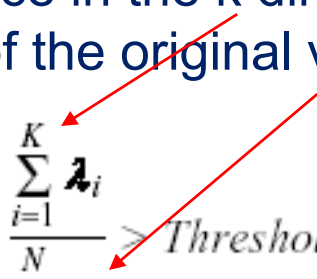
K should be $\ll N$

But what should be K?

Not covered until slide 42

Principal Component Analysis (PCA)

- How many principal components?
- By using more eigenvectors, we represent more of the variation in the original data.
 - If we discarded all but one dimension, the new data would have lost a lot of the original variation in the discarded dimensions.
- So, the rule used is considering to have some percentage of the original variance kept. The variance in each eigenvalue direction is λ_i , so we sum the variance in the k direction and we require that it surpasses say 90% of the original variation.

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


How to choose k ?

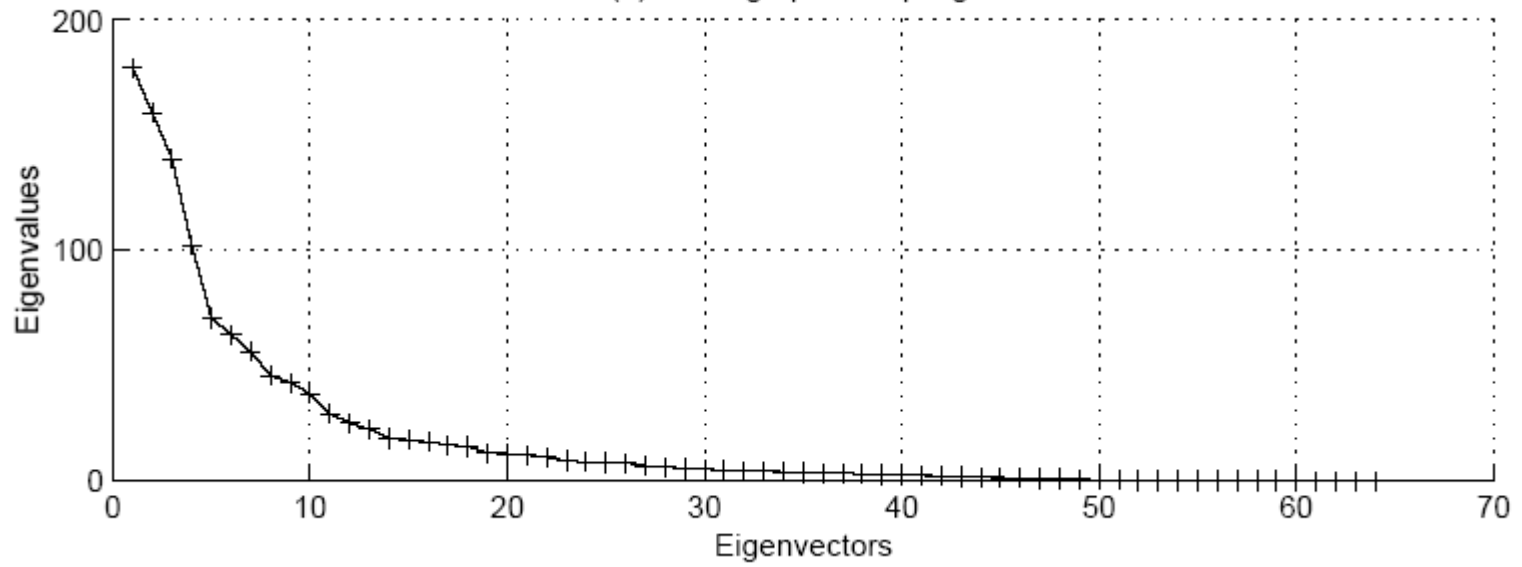
- Proportion of Variance (PoV) explained

$$\frac{\lambda_1 + \lambda_2 + \dots + \lambda_k}{\lambda_1 + \lambda_2 + \dots + \lambda_k + \dots + \lambda_d}$$

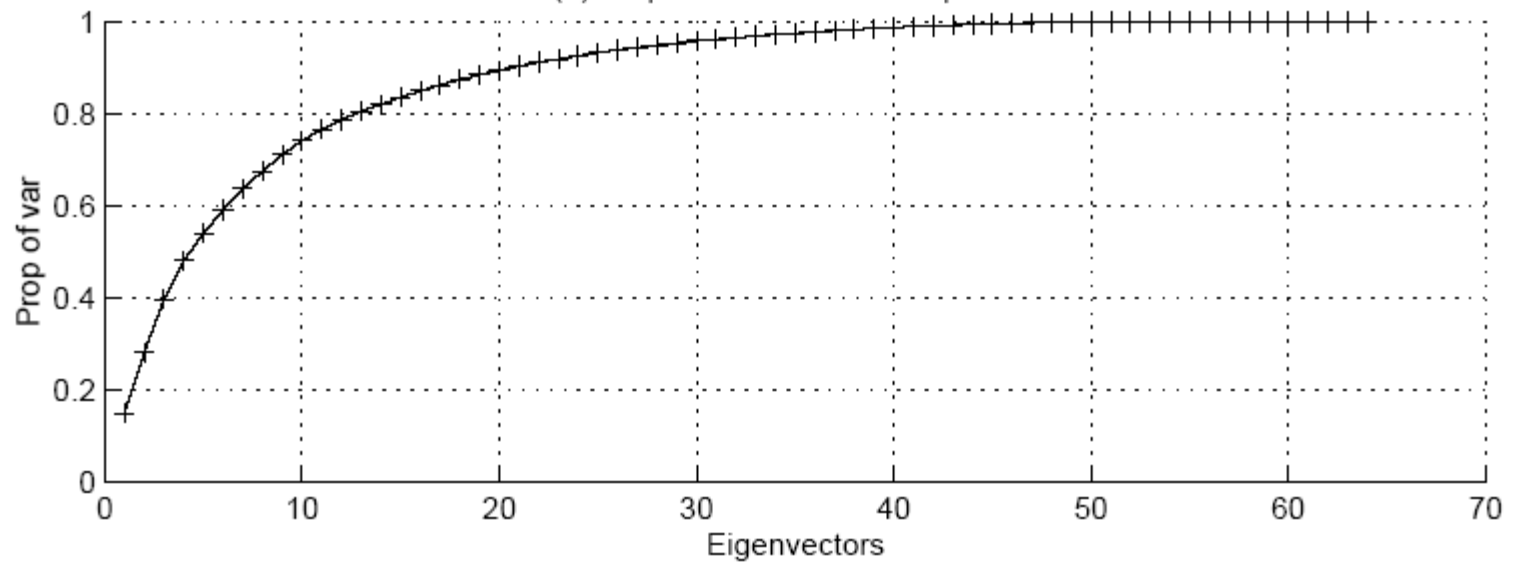
when λ_i are sorted in descending order

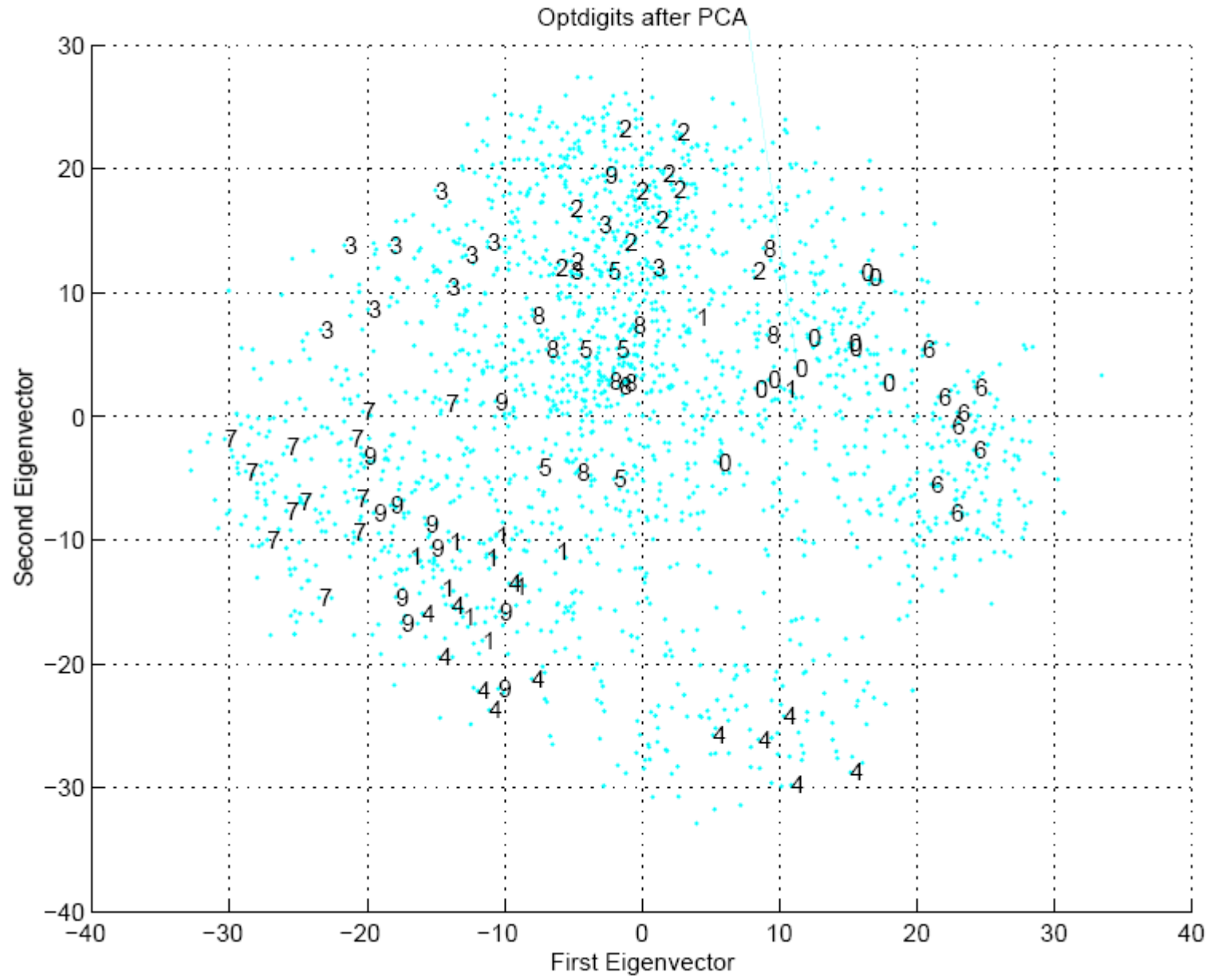
- Typically, stop at $\text{PoV} > 0.9$
- Scree graph plots of PoV vs k , stop at “elbow”

(a) Scree graph for Optdigits



(b) Proportion of variance explained





Principal Component Analysis (PCA)

- What is the error due to dimensionality reduction?
 - We saw above that an original vector x can be reconstructed using its principal components:

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

- It can be shown that the low-dimensional basis based on principal components minimizes the reconstruction error:

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

- It can be shown that the error is equal to:

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

Effect of units in computing variance

- What happens if our x_1 dimension is height and x_2 dimension is weight, but the height can be in cm (170cm, 190cm) or in meters (1.7m, 1.9m)...
- If the unit is centimeters the variance in the x_1 dimension will be larger than if we used meters.

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.
 - A common standardization method is to transform all the data to have zero mean and unit standard deviation, before applying PCA:

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

Eigenface Implementation

Eigenface Example

Eigenfaces example

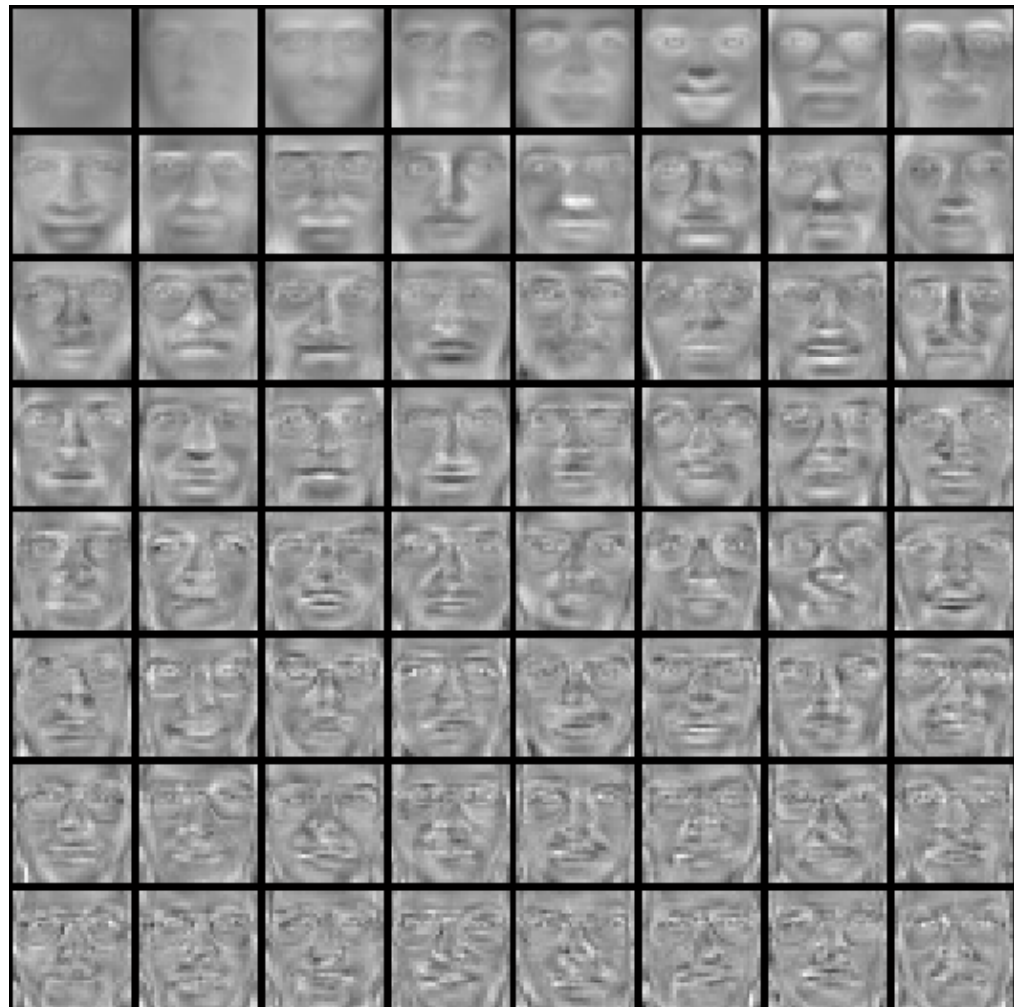
- Training images
- $\mathbf{x}_1, \dots, \mathbf{x}_N$



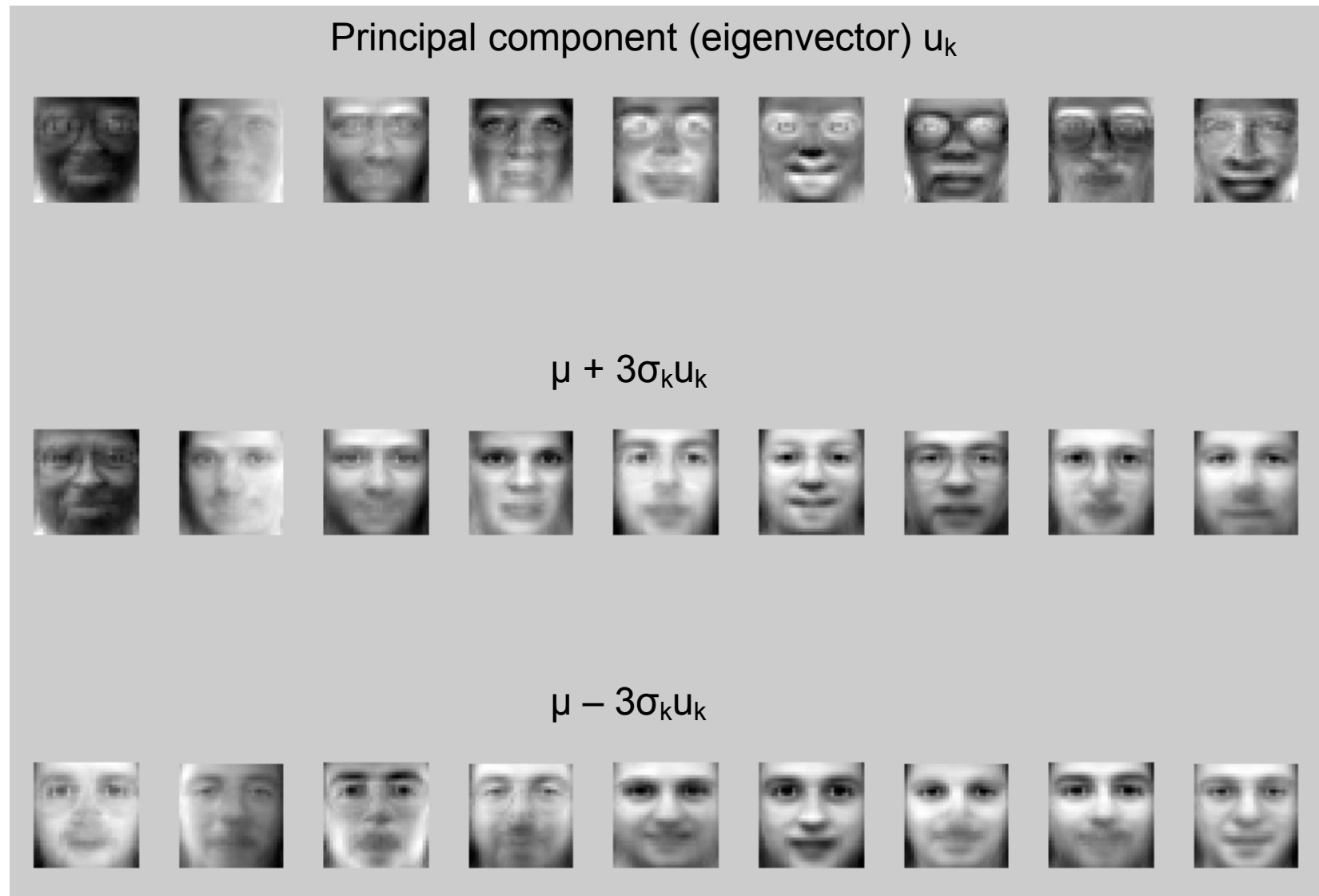
Eigenfaces example

Top eigenvectors: u_1, \dots, u_k

Mean: μ



Visualization of eigenfaces



Representation and reconstruction

- Face \mathbf{x} in “face space” coordinates:



$$\begin{aligned}\mathbf{x} &\longrightarrow [\mathbf{u}_1^T (\mathbf{x} - \mu), \dots, \mathbf{u}_k^T (\mathbf{x} - \mu)] \\ &= w_1, \dots, w_k\end{aligned}$$

Representation and reconstruction

- Face \mathbf{x} in “face space” coordinates:



$$\mathbf{x} \rightarrow [\mathbf{u}_1^T (\mathbf{x} - \mu), \dots, \mathbf{u}_k^T (\mathbf{x} - \mu)]$$
$$= w_1, \dots, w_k$$

- Reconstruction:



\approx



+



\mathbf{x}

\approx

μ

+

$w_1 u_1 + w_2 u_2 + w_3 u_3 + w_4 u_4 + \dots$

Reconstruction

P = 4



P = 200



P = 400



Eigenfaces are computed using the 400 face images from ORL face database. The size of each image is 92x112 pixels (x has ~10K dimension).

Recognition with eigenfaces

Process labeled training images

- Find mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
- Find k principal components (eigenvectors of $\boldsymbol{\Sigma}$) $\mathbf{u}_1, \dots, \mathbf{u}_k$
- Project each training image \mathbf{x}_i onto subspace spanned by principal components:
 $(w_{i1}, \dots, w_{ik}) = (\mathbf{u}_1^T(\mathbf{x}_i - \boldsymbol{\mu}), \dots, \mathbf{u}_k^T(\mathbf{x}_i - \boldsymbol{\mu}))$

Given novel image \mathbf{x}

- Project onto subspace:
 $(w_1, \dots, w_k) = (\mathbf{u}_1^T(\mathbf{x} - \boldsymbol{\mu}), \dots, \mathbf{u}_k^T(\mathbf{x} - \boldsymbol{\mu}))$
- Optional: check reconstruction error $\hat{\mathbf{x}} - \mathbf{x}$ to determine whether image is really a face
- Classify as closest training face in k -dimensional subspace

PCA

- General dimensionality reduction technique
- Preserves most of variance with a much more compact representation
 - Lower storage requirements (eigenvectors + a few numbers per face)
 - Faster matching
- What are the problems for face recognition?

Limitations

Global appearance method:

- not robust at all to misalignment
- not very robust to background variation, scale



Principal Component Analysis (PCA)

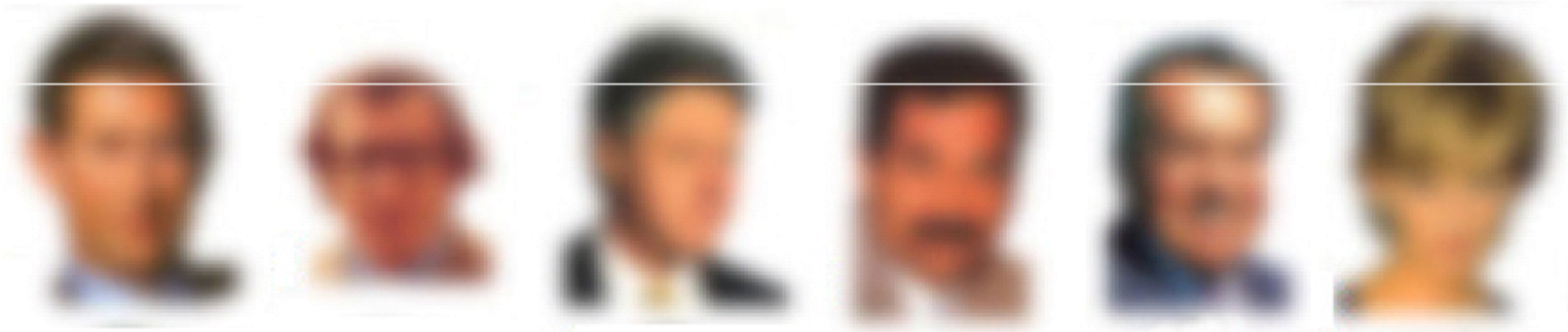
- Problems
 - Background (de-emphasize the outside of the face – e.g., by multiplying the input image by a 2D Gaussian window centered on the face)
 - Lighting conditions (performance degrades with light changes)
 - Scale (performance decreases quickly with changes to head size)
 - multi-scale eigenspaces
 - 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

Face recognition by humans

Face recognition by humans: 20 results
(2005)

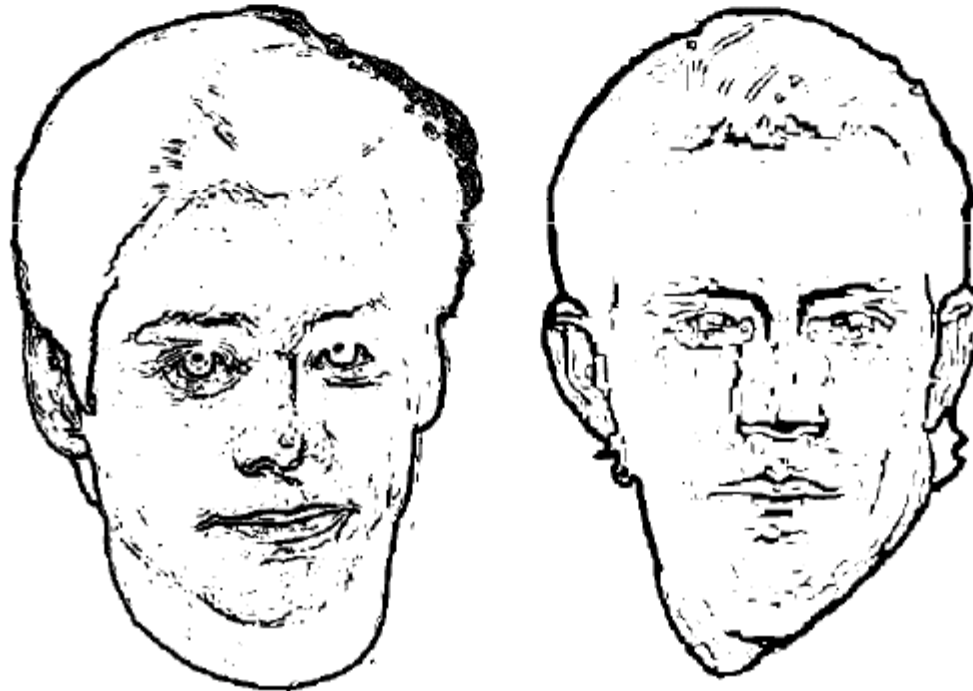
Result 1

- ▶ Humans can recognize faces in extremely low resolution images.



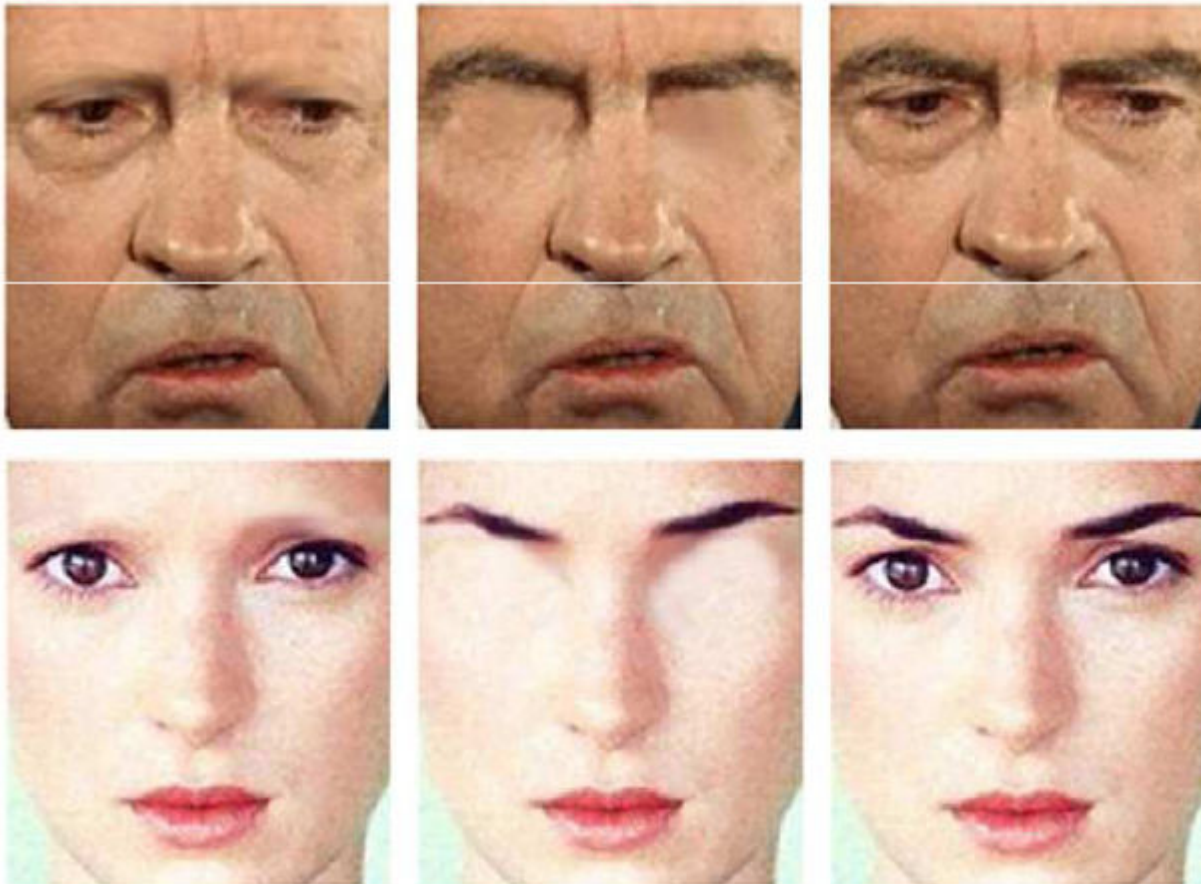
Result 3

- ▶ High-frequency information by itself does not lead to good face recognition performance



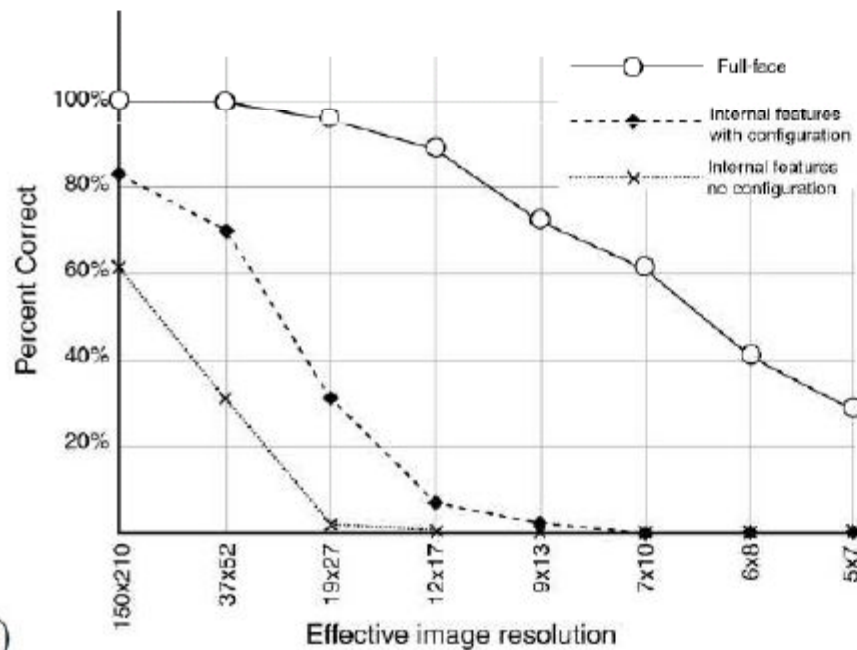
Result 5

- ▶ Eyebrows are among the most important for recognition



Result 6

- ▶ Both internal and external facial cues are important and they exhibit non-linear interactions



(a)



(b)

Result 7

- ▶ The important configural relations appear to be independent across the width and height dimensions



Result 8

- ▶ Vertical inversion dramatically reduces recognition performance



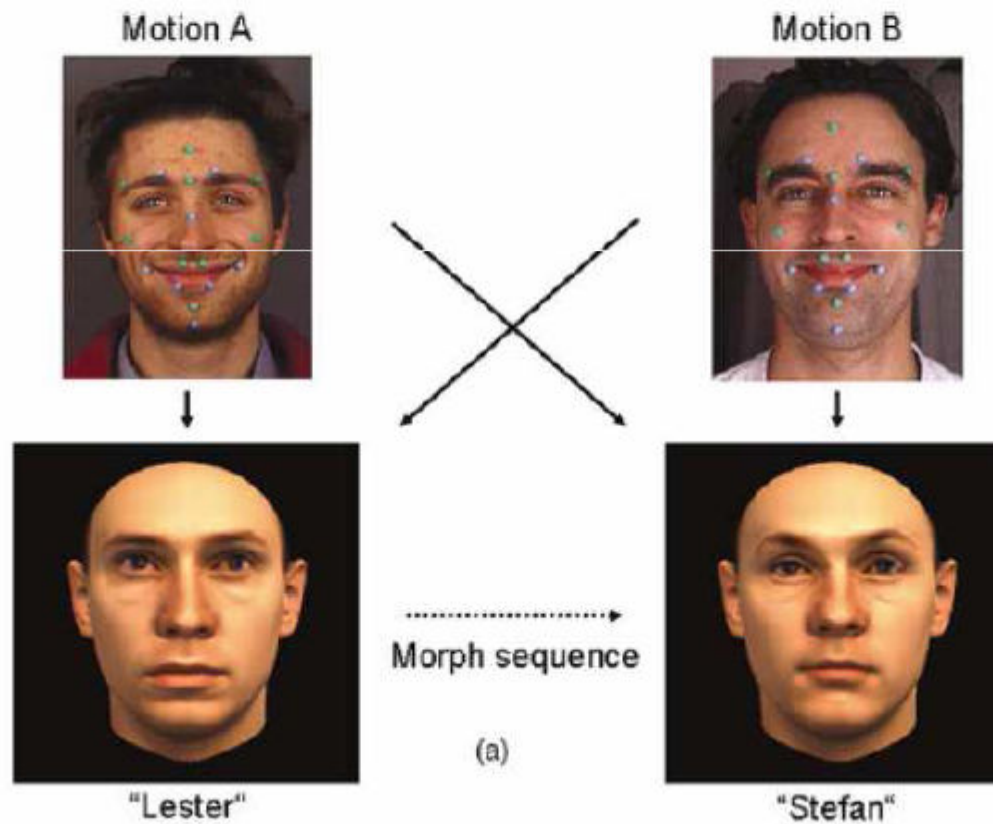
Result 12

- ▶ Contrast polarity inversion dramatically impairs recognition performance, possibly due to compromised ability to use pigmentation cues



Result 15

- ▶ Motion of faces appears to facilitate subsequent recognition

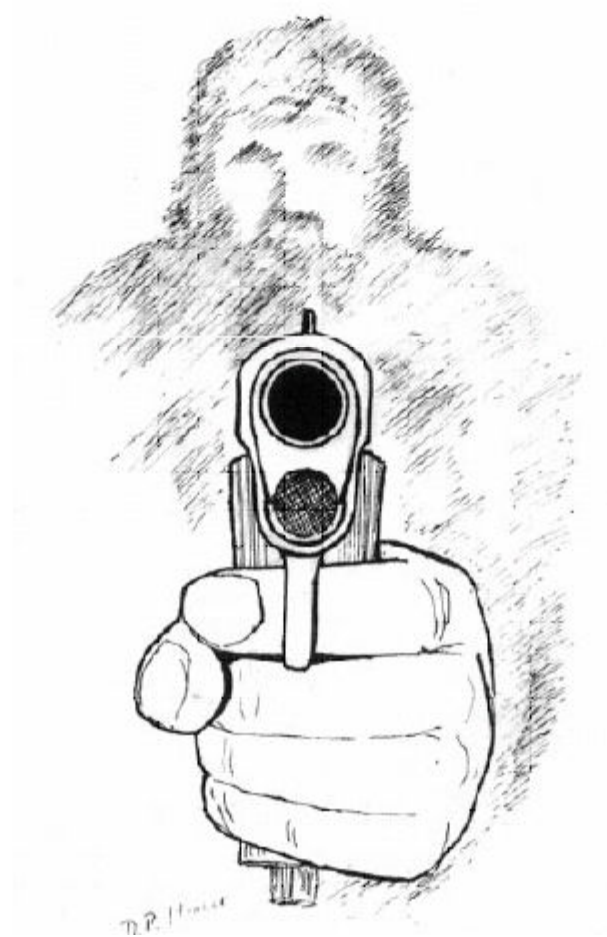


Result 17: Vision progresses from piecemeal to holistic

Age	Correct responses (%)			
	Faces		Houses	
	Upright	Inverted	Upright	Inverted
6	69	64	71	58*†
8	81	67	74	64
10	89	68‡	73	77

Result 20

- ▶ Human memory for briefly seen faces is rather poor



Things to remember

- PCA is a generally useful dimensionality reduction technique
 - But not ideal for discrimination
- FLD better for discrimination, though only ideal under Gaussian data assumptions
- Computer face recognition works very well under controlled environments – still room for improvement in general conditions