

# Recognition, Detection, and Classification (Part 2)

Introduction to Computer Vision I

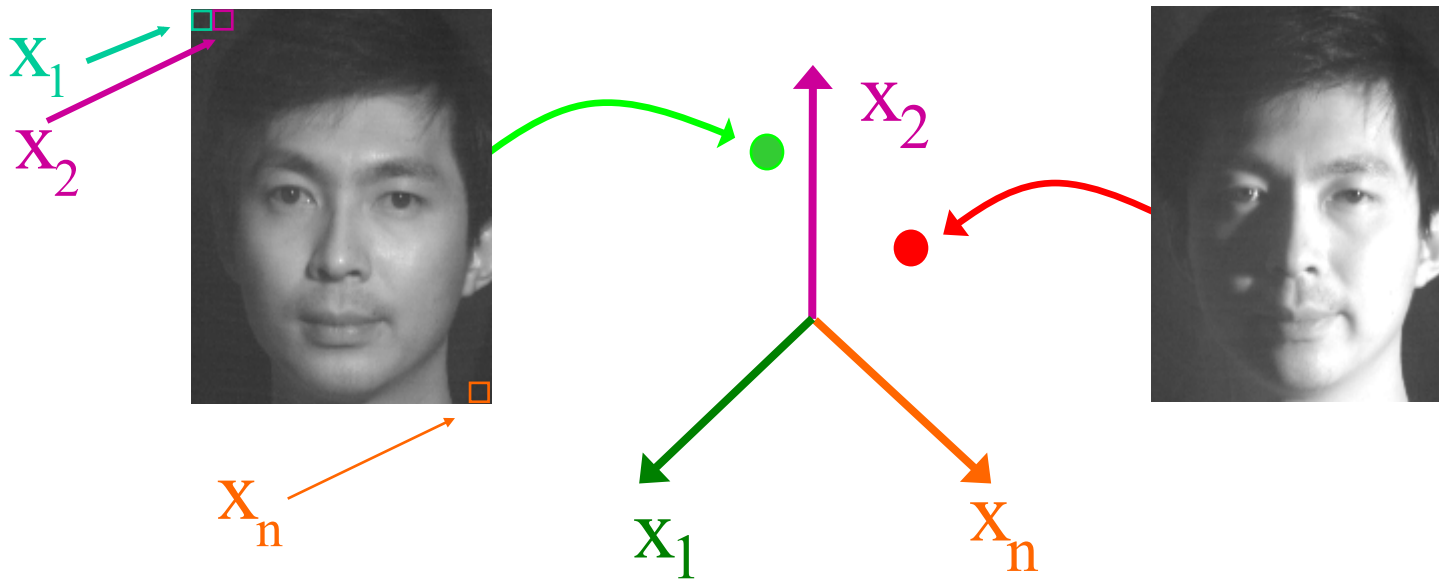
CSE 152A

Lecture 14

# Announcements

- Assignment 3 is due today, 11:59 PM
- Quiz 3 is Feb 26
- Assignment 4 will be released Mar 1
  - Due Mar 10, 11:59 PM
- Reading:
  - Szeliski
    - Sections 5.1 (intro), 5.1.1, 5.1.2, 5.2.3, 6.2 (intro), and 6.2.1

# Simplest feature: Image as a Feature Vector



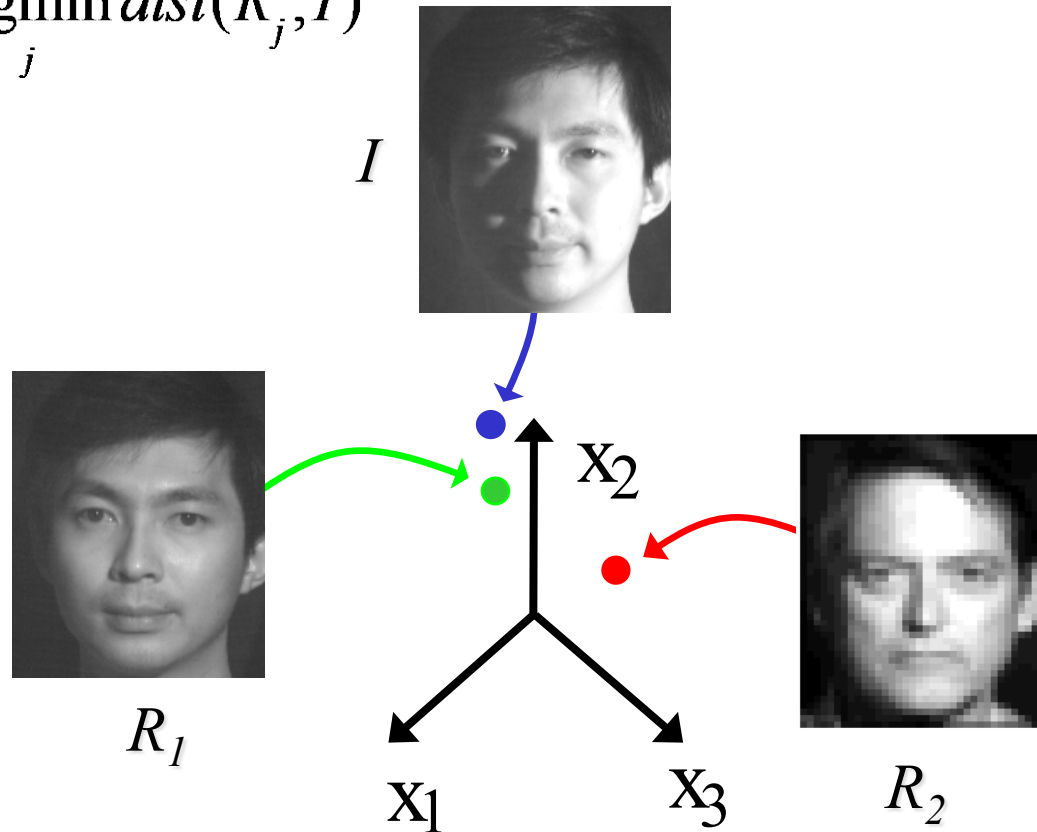
- Consider an  $n$ -pixel image to be a point in an  $n$ -dimensional space,  $\mathbf{x} \in \mathbf{R}^n$
- Each pixel value  $x_i$  is a coordinate of  $\mathbf{x}$

# Nearest Neighbor Classifier

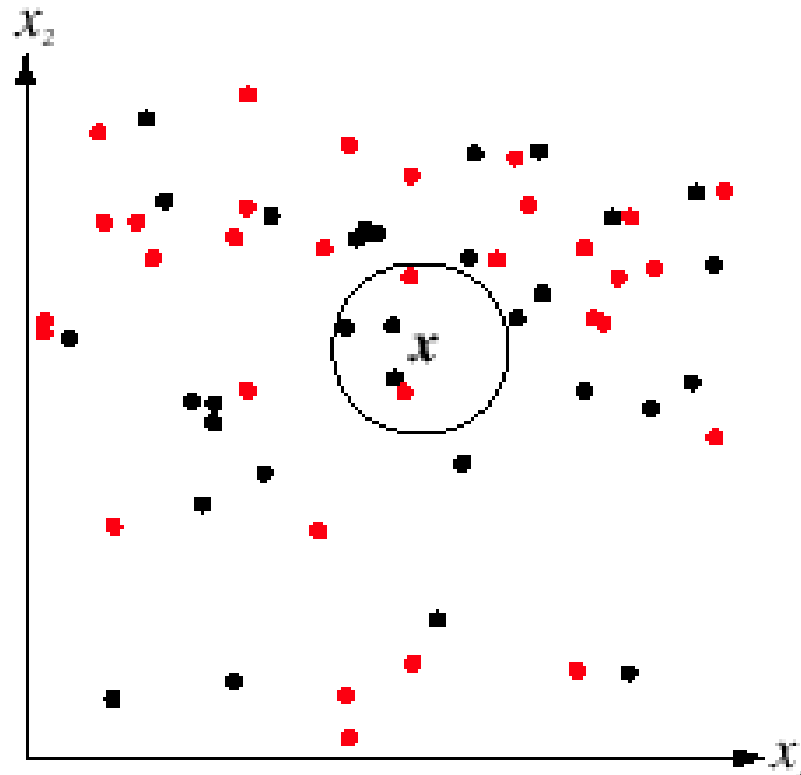
n classes (different people)

$\{R_j : j=1, \dots, n\}$  : set of training images, one per person

$$label = \underset{j}{\operatorname{argmin}} \operatorname{dist}(R_j, I)$$

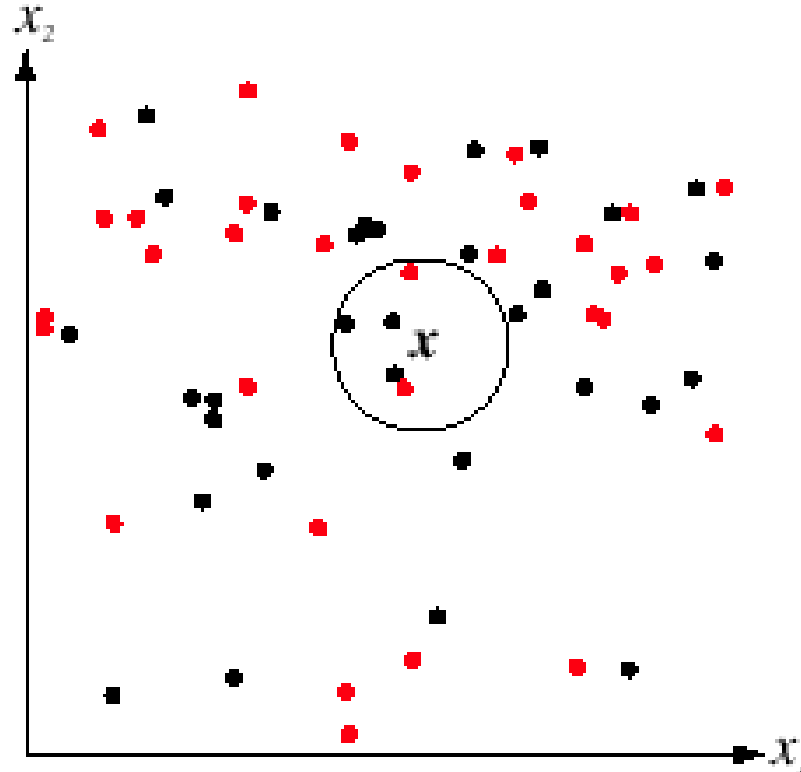


# k Nearest Neighbor Classification



**FIGURE 4.15.** The  $k$ -nearest-neighbor query starts at the test point  $\mathbf{x}$  and grows a spherical region until it encloses  $k$  training samples, and it labels the test point by a majority vote of these samples. In this  $k = 5$  case, the test point  $\mathbf{x}$  would be labeled the category of the black points. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

# Nearest Neighbor Classification



- k Nearest Neighbor with  $k=1$
- It can be shown that as number of samples approaches infinity, the error rate of nearest neighbor is at most twice that of the optimal Bayesian classifier. (Why is even close? Training samples are drawn from  $P(x | \omega_i)P(\omega_i)$  )

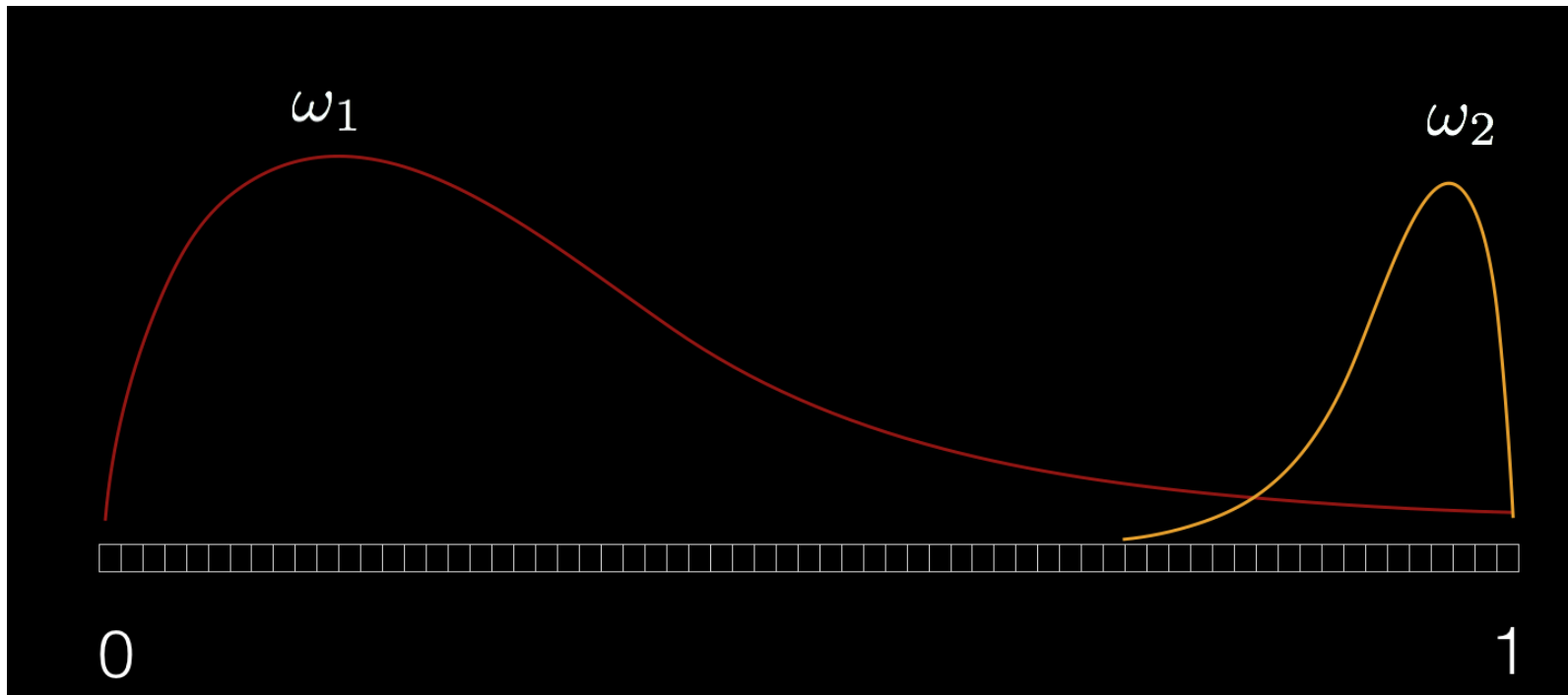
# Comments on Nearest Neighbor

- Sometimes called “Template Matching”
- Variations on distance function (e.g.  $L_1$ , robust distances)
- Multiple templates per class- perhaps many training images per class
- Expensive to compute  $k$  distances, especially when each image is big ( $N$  dimensional)
- May not generalize well to unseen examples of class
- Will it apply well to other features (height, weight?)
- Some solutions:
  - Bayesian classification
  - Dimensionality reduction

- If we want to build a minimum-error rate classifier, then we need a very good estimate of  $P(\omega_i | \mathbf{x})$
- How do we do this?



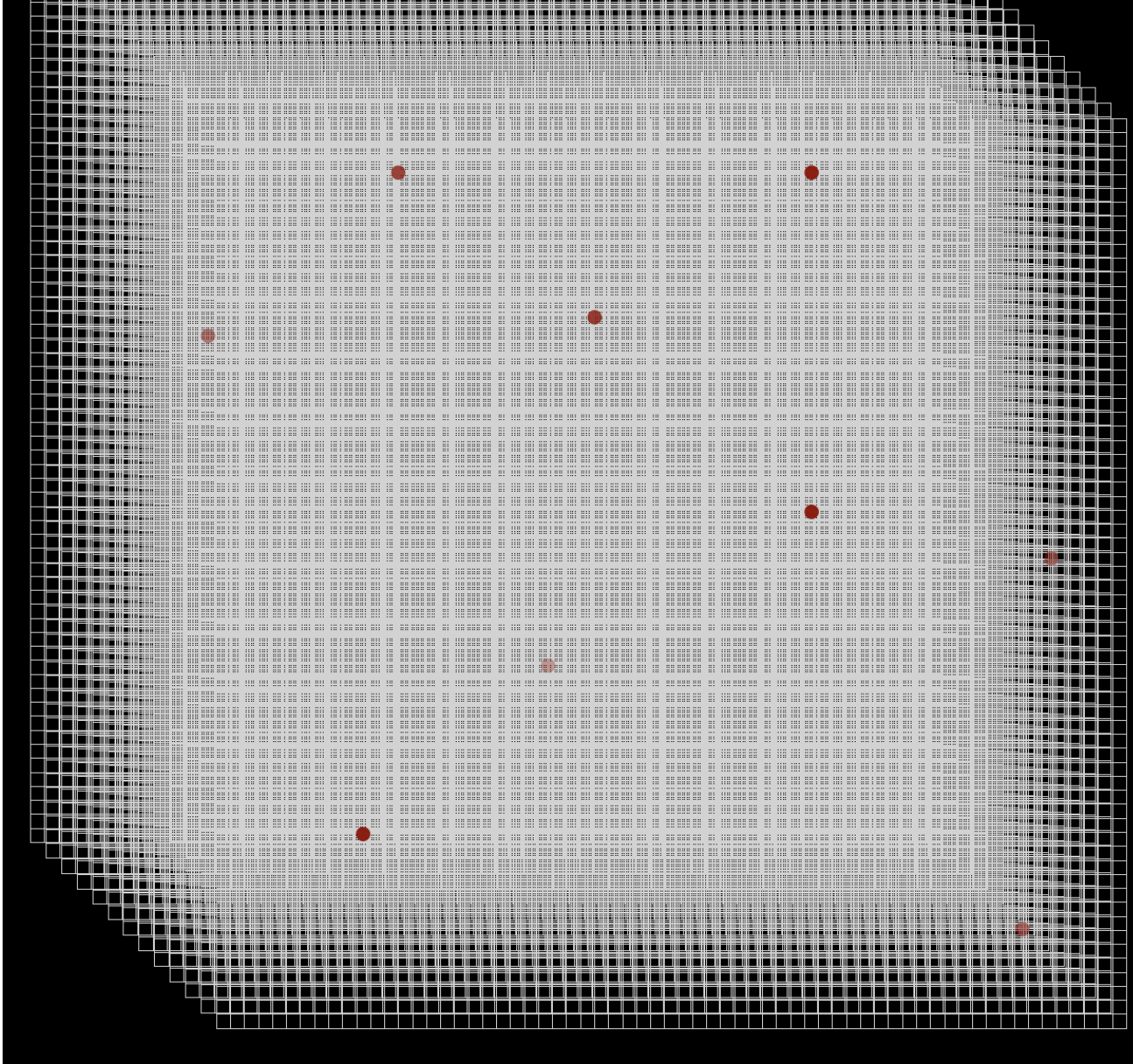
- Let's say our feature space is just 1-dimensional and our feature  $\mathbf{x} \in [0,1]$
- Let's say we have **10,000 training samples** from which to estimate our *a posteriori* probabilities
- We could estimate these probabilities using a histogram in which we divided the interval into 100 evenly spaced bins



- On average each bin would have 100 samples
- We could estimate  $P(\mathbf{x} | \omega_i)$  as the number of samples from class  $i$  that fall in the same bin that falls into divided by the total number of samples in that bin

But this plan does not scale as we increase the dimensionality of the feature space!

- Let's say our feature space is 3 dimensional and our feature  $\mathbf{x} \in [0,1]^3$
- Let's say we still have **10,000 training samples** from which to estimate our *a posteriori* probabilities
- If we estimate these probabilities using a histogram in which we divide the volume into the same width bins as before...



On average each bin would only have 0.01 samples!

We're not going to be able to estimate probabilities well

# Curse of Dimensionality

## Dimensionality Reduction

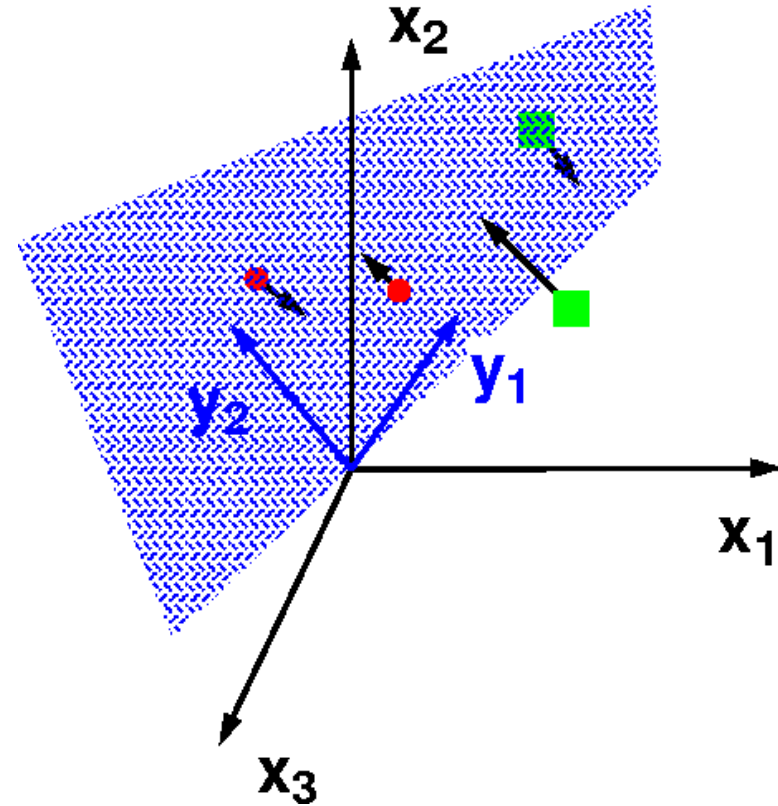
# An idea:

Represent the set of images as a linear subspace

What is a linear subspace?

Let  $V$  be a vector space and let  $W$  be a subset of  $V$ . Then  $W$  is a subspace if and only if:

1. The null vector  $\mathbf{0}$  is in  $W$
  2. If  $\mathbf{u}$  and  $\mathbf{v}$  are elements of  $W$ , then any linear combination of  $\mathbf{u}$  and  $\mathbf{v}$  is an element of  $W$ ;  $a\mathbf{u} + b\mathbf{v} \in W$
  3. If  $\mathbf{u}$  is an element of  $W$  and  $c$  is a scalar, then the scalar product  $c\mathbf{u} \in W$
- A  $k$ -dimensional subspace is spanned by  $k$  linearly independent vectors. It is spanned by a  $k$ -dimensional orthogonal basis



Example: A 2-D linear subspace of  $\mathbf{R}^3$  spanned by  $y_1$  and  $y_2$

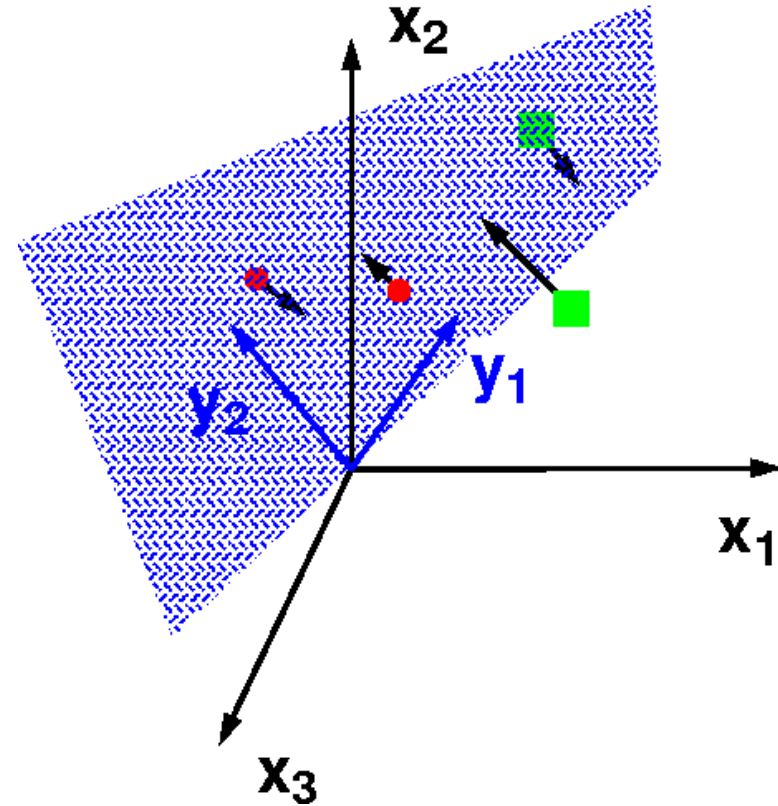
# Linear Subspaces & Linear Projection

- A  $d$ -pixel image  $\mathbf{x} \in \mathbf{R}^d$  can be projected to a low-dimensional feature space  $\mathbf{y} \in \mathbf{R}^k$  by

$$\mathbf{y} = W\mathbf{x}$$

where  $W$  is an  $k$  by  $d$  matrix

- Each training image is projected to the subspace
- Recognition is performed in  $\mathbf{R}^k$  using, for example, nearest neighbor
- How do we choose a good  $W$ ?



Example: A 2-D linear subspace of  $\mathbf{R}^3$  spanned by  $y_1$  and  $y_2$

# How do we choose a good $W$ ?

- Drop dimensions (feature selection)
- Random projections
- Principal Component Analysis
- Linear Discriminant Analysis
- Independent Component Analysis
- Or non-linear dimensionality reduction



## Principal Component Analysis (PCA)

Assume we have a set of  $n$  feature vectors  $\mathbf{x}_i$  ( $i = 1, \dots, n$ ) in  $\mathbb{R}^d$ . Write

$$\boldsymbol{\mu} = \frac{1}{n} \sum_i \mathbf{x}_i$$

$$\boldsymbol{\Sigma} = \frac{1}{n-1} \sum_i (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$$

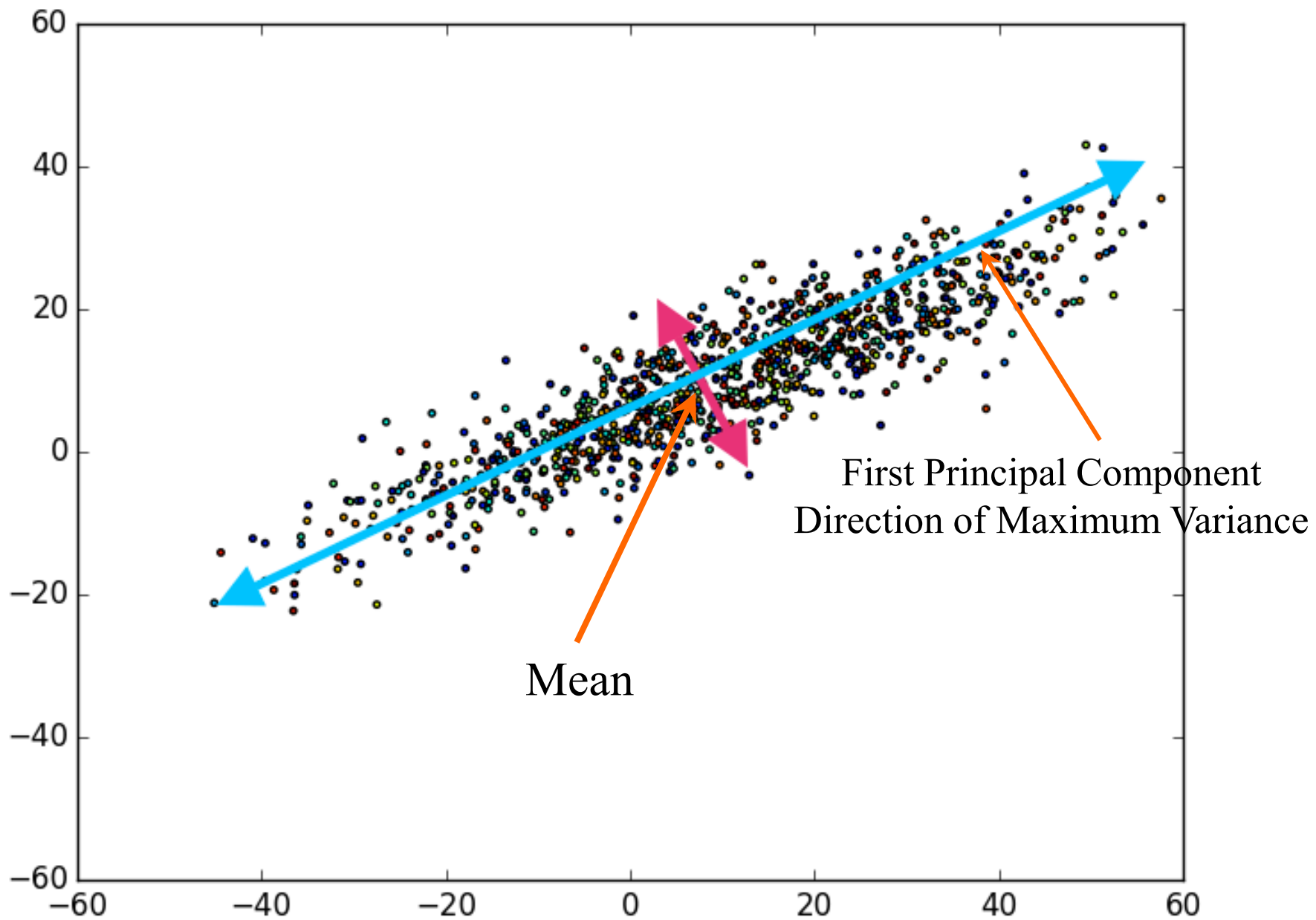
The unit eigenvectors of  $\boldsymbol{\Sigma}$  — which we write as  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_d$ , where the order is given by the size of the eigenvalue and  $\mathbf{v}_1$  has the largest eigenvalue — give a set of features with the following properties:

- They are independent.
- Projection onto the basis  $\{\mathbf{v}_1, \dots, \mathbf{v}_k\}$  gives the  $k$ -dimensional set of linear features that preserves the most variance.

**Algorithm 22.5:** *Principal components analysis identifies a collection of linear features that are independent, and capture as much variance as possible from a dataset.*

**Eigen decomposition of covariance matrix.**

**Alternative: singular value decomposition of (mean-deviation form of) data matrix.**



# Singular value decomposition and its relationship to eigen decomposition

- Any  $m$  by  $n$  matrix  $A$  may be factored such that

$$A = U\Sigma V^T$$

$$[m \times n] = [m \times m][m \times n][n \times n]$$

- $U$ :  $m$  by  $m$ , orthogonal matrix
  - Columns of  $U$  are the eigenvectors of  $AA^T$
- $V$ :  $n$  by  $n$ , orthogonal matrix,
  - columns are the eigenvectors of  $A^T A$
- $\Sigma$ :  $m$  by  $n$ , diagonal with non-negative entries ( $\sigma_1, \sigma_2, \dots, \sigma_s$ ) with  $s = \min(m, n)$  are called the called the singular values
  - **Singular values are the square roots of eigenvalues of both  $AA^T$  and  $A^T A$**
  - *Result of SVD algorithm:*  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_s$

# SVD Properties

- $r = \text{Rank}(A) = \#$  of non-zero singular values.
- $U, V$  give an orthonormal bases for the subspaces of  $A$ :
  - 1st  $r$  columns of  $U$ : Column space of  $A$
  - Last  $m - r$  columns of  $U$ : Left nullspace of  $A$
  - 1st  $r$  columns of  $V$ : Row space of  $A$
  - 1st  $n - r$  columns of  $V$ : (Right) nullspace of  $A$
- *For some  $d$  where  $d \leq r$ , the first  $d$  column of  $U$  provide the best  $d$ -dimensional basis for columns of  $A$  in least squares sense.*

# Performing PCA with SVD

- Singular values of  $A$  are the square roots of eigenvalues of  $AA^T$  (and  $A^T A$ )
- Columns of  $U$  are corresponding Eigenvectors of  $AA^T$
- And  $\sum_{i=1}^n a_i a_i^T = [a_1 \ a_2 \ \dots \ a_n][a_1 \ a_2 \ \dots \ a_n]^T = AA^T$
- Covariance matrix is:

$$\Sigma = \frac{1}{n-1} \sum_{i=1}^n (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T$$

- So, ignoring  $1/(n-1)$ , subtract mean image  $\boldsymbol{\mu}$  from each input image, create a  $d$  by  $n$  data matrix, and perform thin SVD on the data matrix.  $D = [\mathbf{x}_1 - \boldsymbol{\mu} \mid \mathbf{x}_2 - \boldsymbol{\mu} \mid \dots \mid \mathbf{x}_n - \boldsymbol{\mu}]$

# Economy SVD

- Any  $m$  by  $n$  matrix  $A$  may be factored such that

$$A = U\Sigma V^T$$

$$[m \times n] = [m \times m][m \times n][n \times n]$$

- If  $m > n$ , then one can view  $\Sigma$  as: (i.e., more pixels than images)

$$\begin{bmatrix} \Sigma' \\ 0 \end{bmatrix}$$

- Where  $\Sigma' = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_s)$  with  $s = \min(m, n)$ , and lower matrix is  $(n - m \text{ by } m)$  of zeros.
- Alternatively, you can write:

$$A = U'\Sigma'V^T$$

# PCA for recognition

## Modeling

1. Given a collection of  $n$  training images  $x_i$ , represent each one as a  $d$ -dimensional column vector
2. Compute the mean image and covariance matrix
3. Compute  $k$  Eigenvectors of the covariance matrix corresponding to the  $k$  largest Eigenvalues and form matrix  $W^T = [u_1, u_2, \dots, u_k]$  (Or perform using SVD)
  - Note that the Eigenvectors are images
4. Project the training images to the  $k$ -dimensional Eigenspace.  
 $y_i = Wx_i$

## Recognition

1. Given a test image  $x$ , project the vectorized image to the Eigenspace by  $y = Wx$
2. Perform classification of  $y$  to the projected training images

# Example: Eigenfaces



Training  
images

[ Turk, Pentland 91]



# Eigenfaces



Mean Image



Basis Images

# Accuracy of PCA + K-NN

KNN + 3,072 Features	33.86
KNN + 200 PCA Comp.	36.54
KNN + 75 PCA Comp.	39.77
KNN + 50 PCA Comp.	40.12
KNN + 40 PCA Comp.	40.93
<b>KNN + 30 PCA Comp.</b>	<b>41.78</b>
KNN + 25 PCA Comp.	41.57
KNN + 15 PCA Comp.	38.75
KNN + 10 PCA Comp.	34.93

# Difficulties with PCA

- Projection may suppress important detail
  - smallest variance directions may not be unimportant
- Method does not take discriminative task into account
  - typically, we wish to compute features that allow good discrimination
  - not the same as largest variance or minimizing reconstruction error

# Fisherfaces: Class specific linear projection

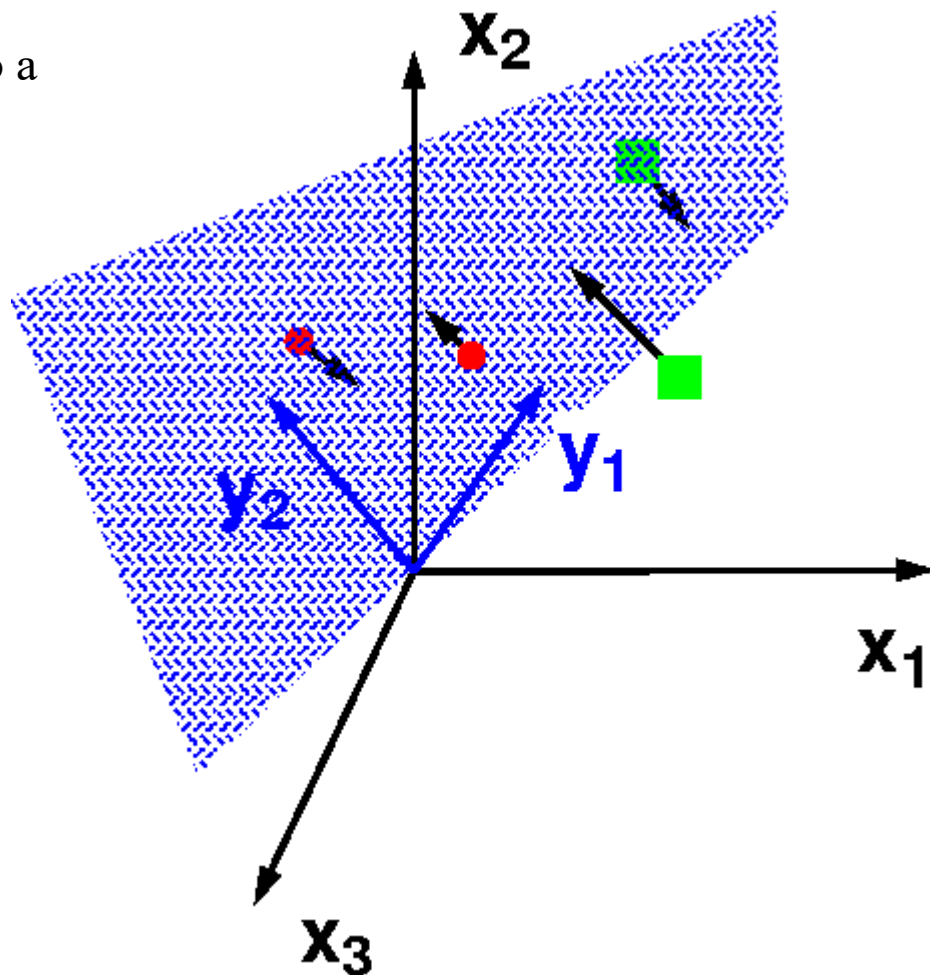
P. Belhumeur, J. Hespanha, D. Kriegman, *Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection*, PAMI, July 1997, pp. 711--720.

- An  $n$ -pixel image  $\mathbf{x} \in \mathbf{R}^d$  can be projected to a low-dimensional feature space  $\mathbf{y} \in \mathbf{R}^k$  by

$$\mathbf{y} = W\mathbf{x}$$

where  $W$  is an  $k$  by  $d$  matrix

- Recognition is performed using nearest neighbor in  $\mathbf{R}^k$
- How do we choose a good  $W$ ?



# PCA & Fisher's Linear Discriminant

- Between-class scatter

$$S_B = \sum_{i=1}^c |\chi_i| (\mu_i - \mu)(\mu_i - \mu)^T$$

- Within-class scatter

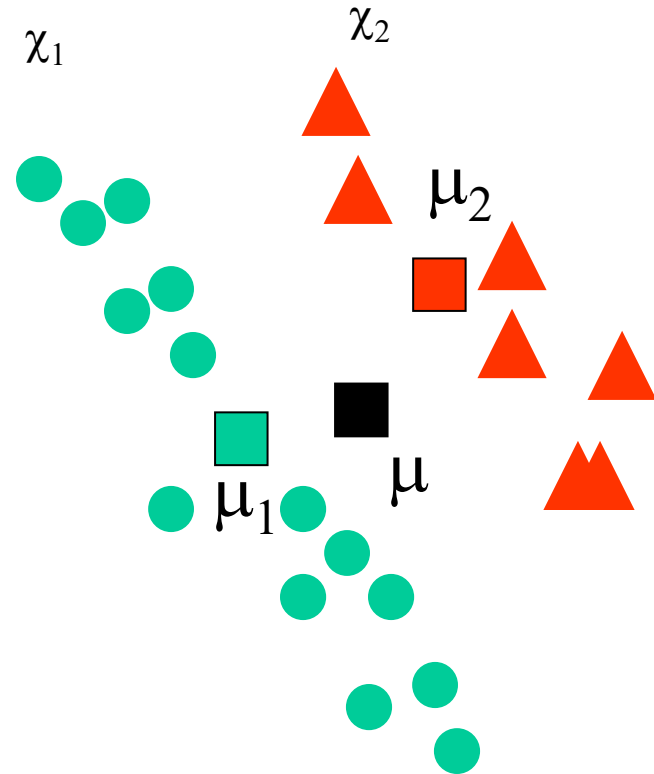
$$S_W = \sum_{i=1}^c \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T$$

- Total scatter

$$S_T = \sum_{i=1}^c \sum_{x_k \in \chi_i} (x_k - \mu)(x_k - \mu)^T = S_B + S_W$$

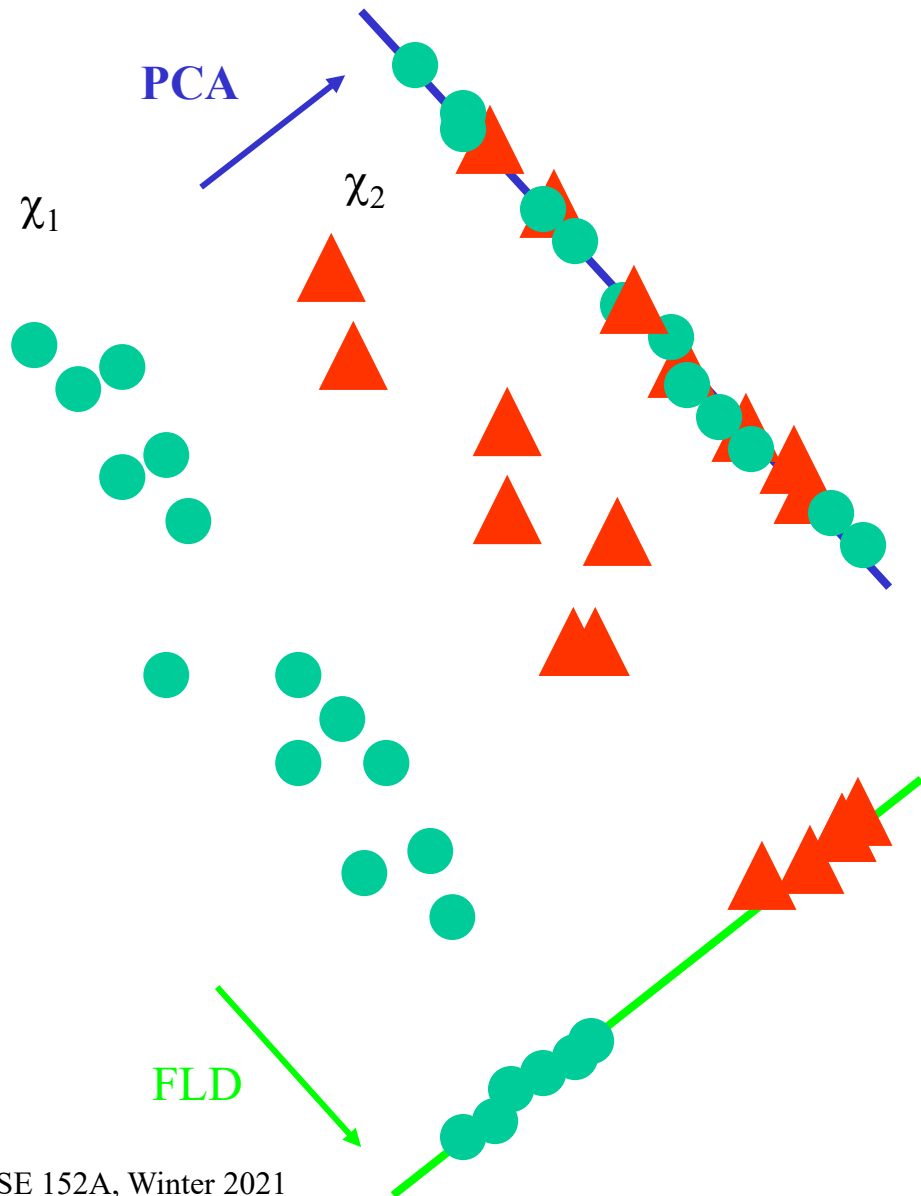
- Where

- $c$  is the number of classes
- $\mu_i$  is the mean of class  $\chi_i$
- $|\chi_i|$  is number of samples of  $\chi_i$ .



If the data points  $x_i$  are projected by  $y_i = Wx_i$  and the scatter of  $x_i$  is  $S$ , then the scatter of the projected points  $y_i$  is  $W^T S W$

# PCA & Fisher's Linear Discriminant



- PCA (Eigenfaces)

$$W_{PCA} = \arg \max_W |W^T S_T W|$$

Maximizes projected total scatter

- Fisher's Linear Discriminant

$$W_{fld} = \arg \max_W \frac{|W^T S_B W|}{|W^T S_W W|}$$

Maximizes ratio of projected between-class to projected within-class scatter

# Computing the Fisher Projection Matrix

$$\begin{aligned} W_{opt} &= \arg \max_W \frac{|W^T S_B W|}{|W^T S_W W|} \\ &= [\mathbf{w}_1 \quad \mathbf{w}_2 \quad \dots \quad \mathbf{w}_m] \end{aligned} \quad (4)$$

where  $\{\mathbf{w}_i \mid i = 1, 2, \dots, m\}$  is the set of generalized eigenvectors of  $S_B$  and  $S_W$  corresponding to the  $m$  largest generalized eigenvalues  $\{\lambda_i \mid i = 1, 2, \dots, m\}$ , i.e.,

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \quad i = 1, 2, \dots, m$$

- There are at most  $c-1$  non-zero generalized Eigenvalues, so  $m \leq c-1$

# Fisherfaces

$$W = W_{fld} W_{PCA}$$

$$W_{PCA} = \arg \max_W |W^T S_T W|$$

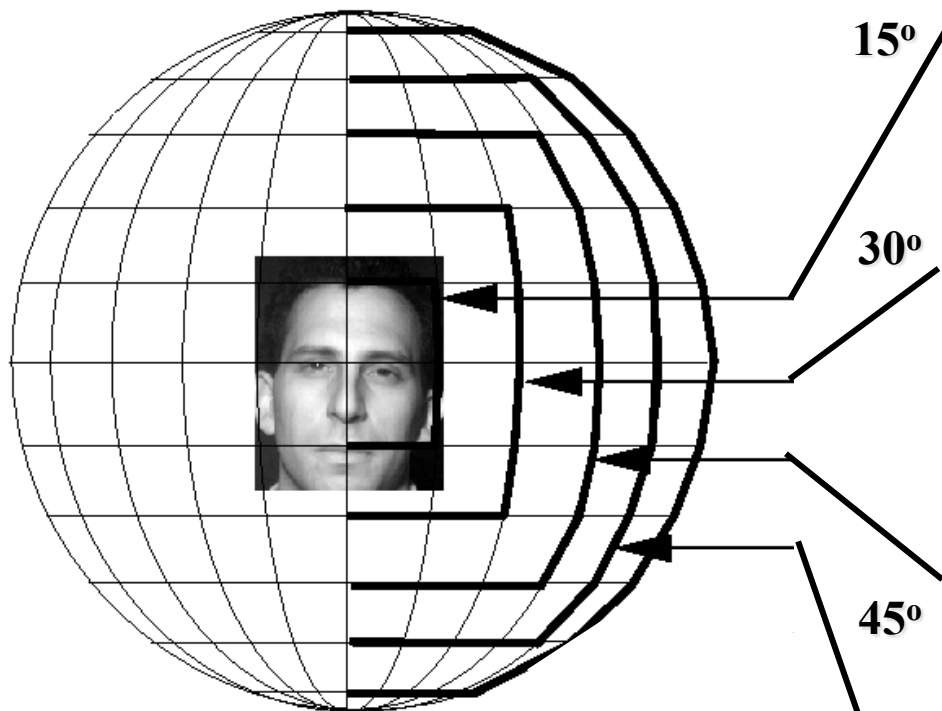
$$W_{fld} = \arg \max_W \frac{|W^T W_{PCA}^T S_B W_{PCA} W|}{|W^T W_{PCA}^T S_W W_{PCA} W|}$$

- Since  $S_W$  is rank  $N-c$ , project training set to subspace spanned by first  $N-c$  principal components of the training set.
- Apply FLD to  $N-c$  dimensional subspace yielding  $c-1$  dimensional feature space.

- Fisher's Linear Discriminant projects away the within-class variation (lighting, expressions) found in training set.
- Fisher's Linear Discriminant preserves the separability of the classes.

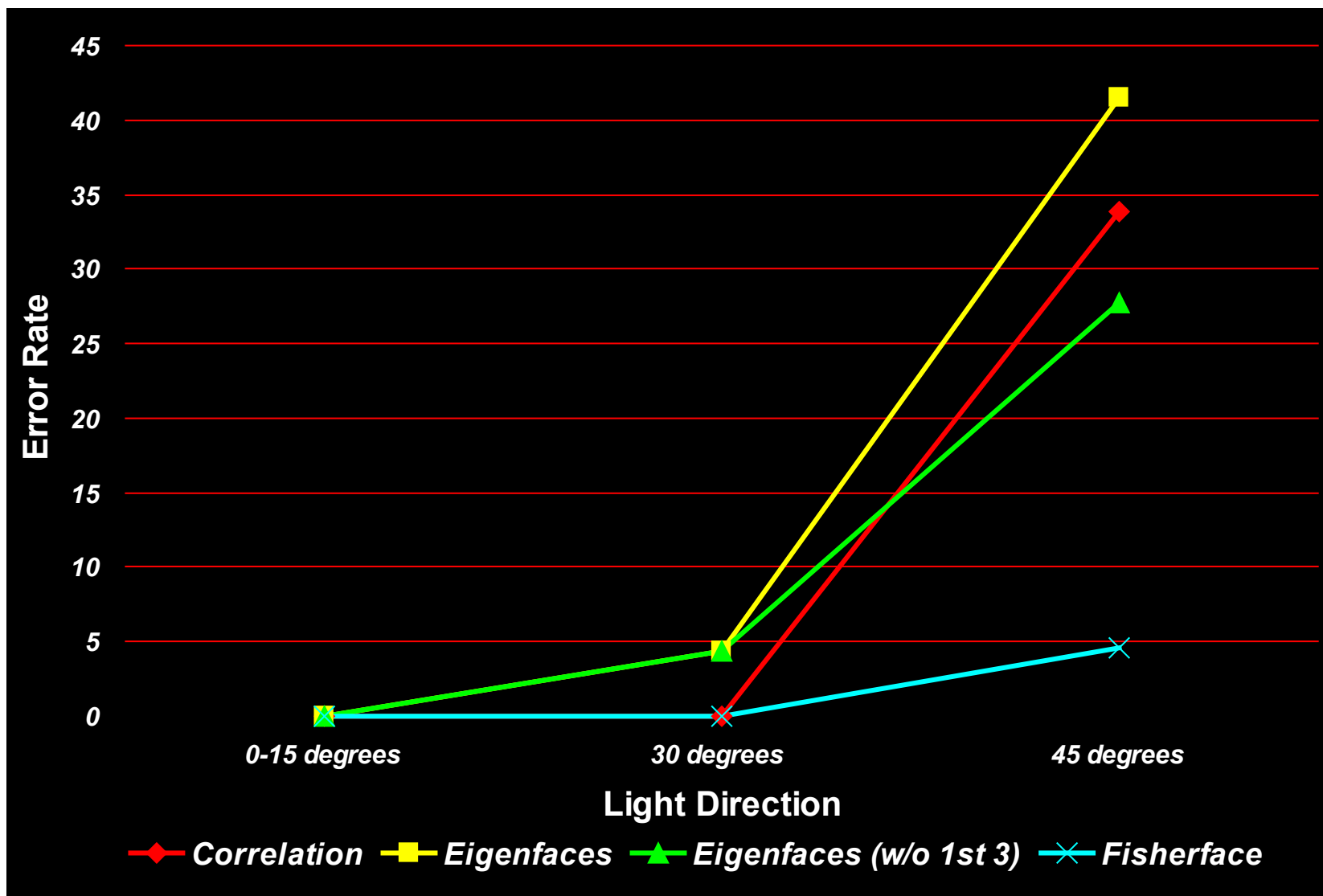


# Harvard Face Database

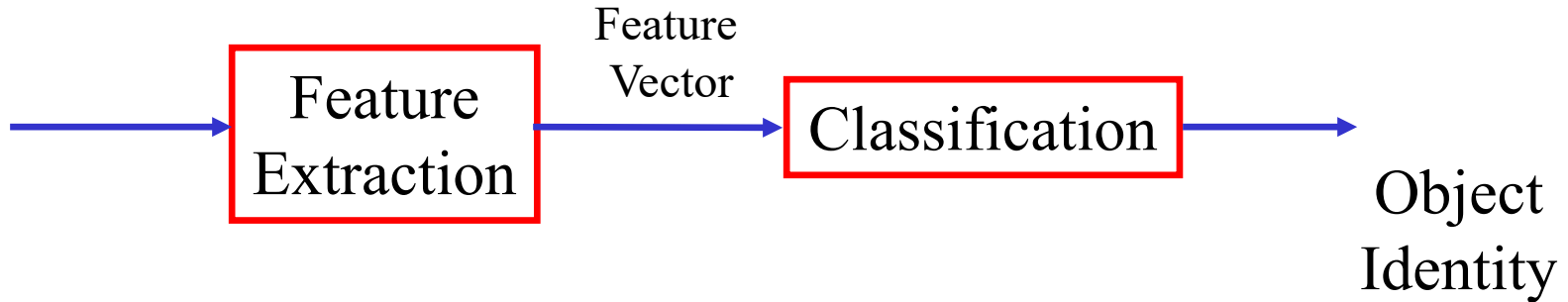


- 10 individuals
- 66 images per person
- Train on 6 images at  $15^\circ$
- Test on remaining images

# Recognition Results: Lighting Extrapolation



# Where have we been so far



- Supervised classification
  - Feature space
  - Nearest Neighbor (kth nearest neighbor)
  - Bayesian (MAP) classifier
- Curse of dimensionality
- Dimensionality reduction
  - Principal component analysis
  - Fisher's linear discriminant

# Next Lectures

- Neural networks
- Reading:
  - Szeliski
    - Section 5.3