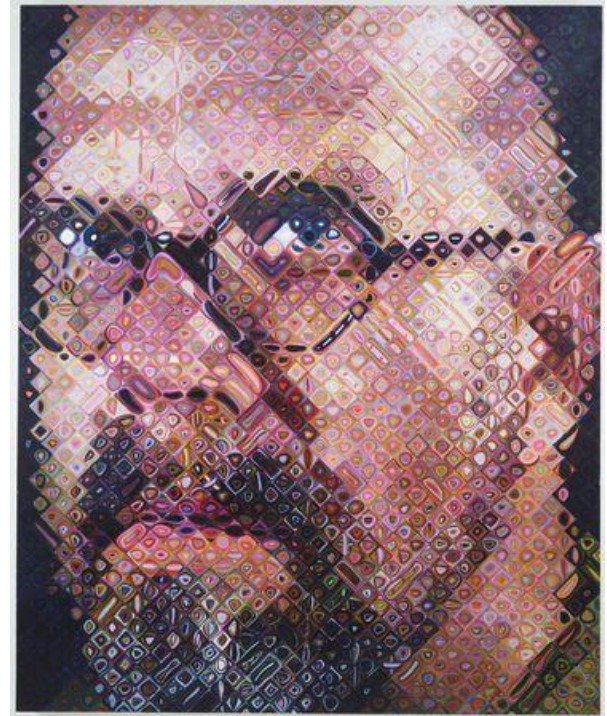


# Face Recognition and Feature Subspaces



Lucas by Chuck Close



Chuck Close, self portrait

Computer Vision (CS 543 / ECE 549)

University of Illinois

Derek Hoiem

# This class: face recognition

- Two methods: “Eigenfaces” and “Fisherfaces”
  - Feature subspaces: PCA and FLD
- Look at results from recent vendor test
- Recent method: DeepFace
- Look at interesting findings about human face recognition
  
- Key ideas
  - Nearest neighbor classifiers
  - PCA and feature extraction

# Applications of Face Recognition

- Surveillance



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

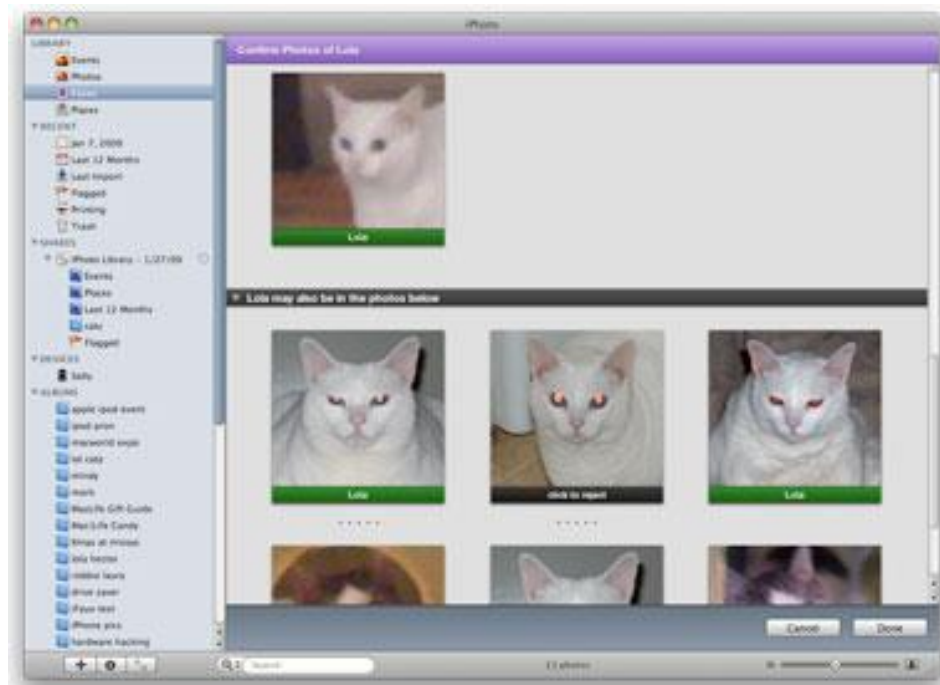
# Applications of Face Recognition

- Album organization: iPhoto 2009



<http://www.apple.com/ilife/iphoto/>

- Can be trained to recognize pets!



[http://www.maclife.com/article/news/iphotos\\_faces\\_recognizes\\_cats](http://www.maclife.com/article/news/iphotos_faces_recognizes_cats)

# Facebook friend-tagging with auto-suggest

## We've Suggested Tags for Your Photos

We've automatically grouped together similar pictures and suggested the names of friends who might appear in them. This lets you quickly label your photos and notify friends who are in this album.

## Tag Your Friends

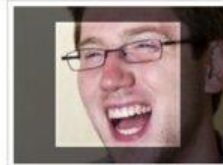
This will quickly label your photos and notify the friends you tag. [Learn more](#)



Who is this?



Who is this?



Who is this?



Who is this?



Who is this?



Who is this?



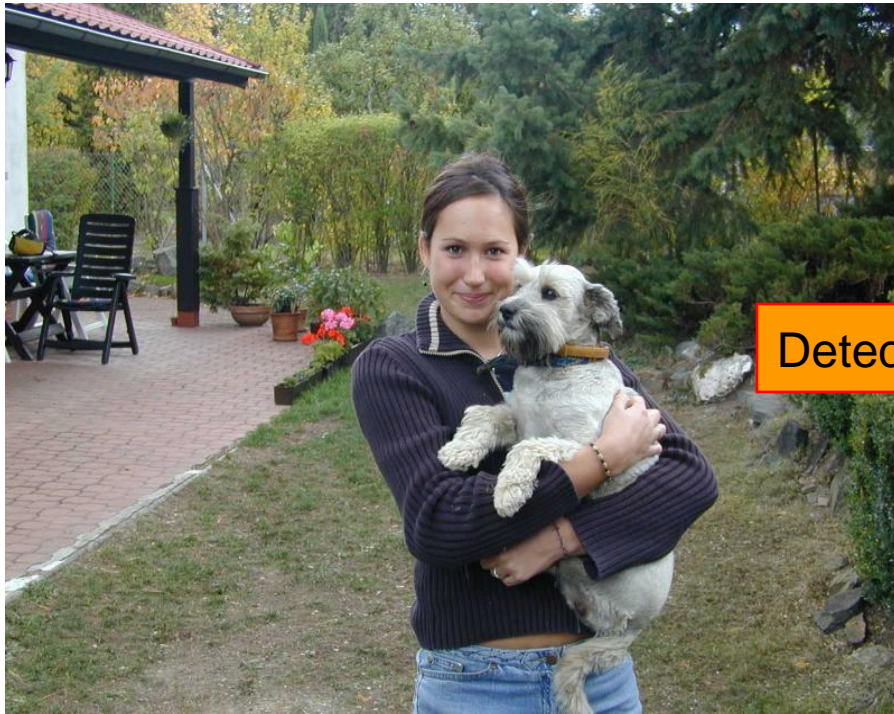
Francis Luu



[Skip Tagging Friends](#)

[Save Tags](#)

Face recognition: once you've detected and cropped a face, try to recognize it



"Sally"

# Face recognition: overview

- Typical scenario: few examples per face, identify or verify test example
- What's hard: changes in expression, lighting, age, **occlusion**, **viewpoint**
- Basic approaches (all nearest neighbor)
  1. Project into a new subspace (or kernel space) (e.g., "Eigenfaces"=PCA)
  2. Measure face features
  3. Make 3d face model, compare shape+appearance (e.g., AAM)

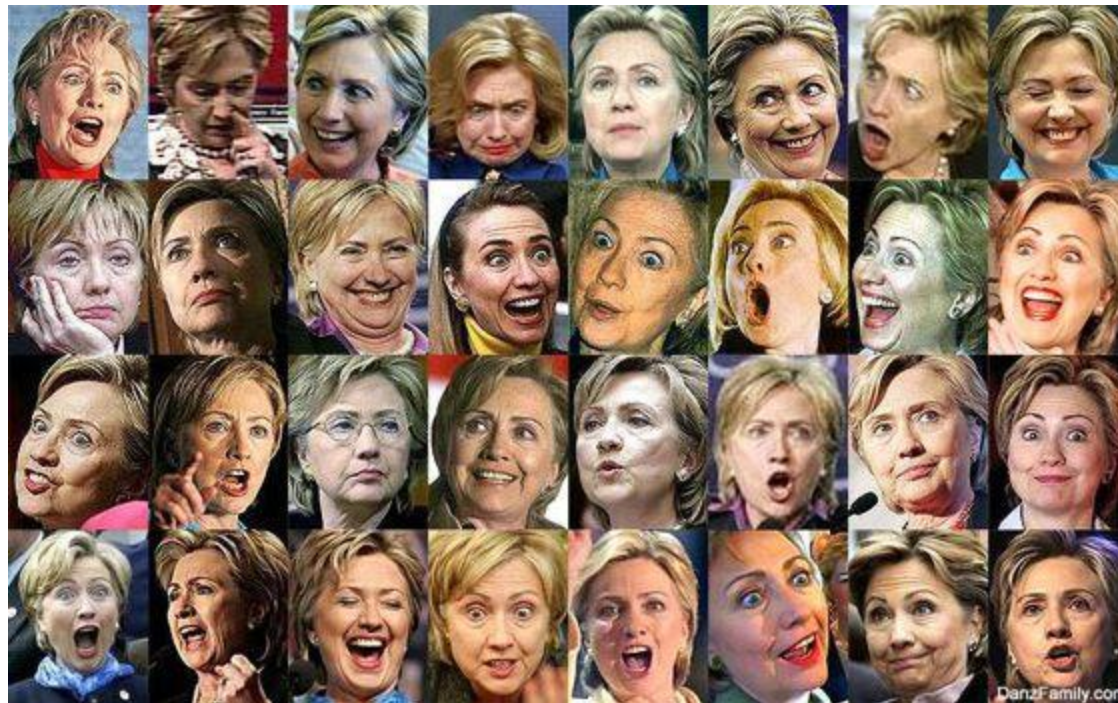


# Typical face recognition scenarios

- Verification: a person is claiming a particular identity; verify whether that is true
  - E.g., security
- Closed-world identification: assign a face to one person from among a known set
- General identification: assign a face to a known person or to “unknown”

# What makes face recognition hard?

## Expression



# What makes face recognition hard?

## Lighting



# What makes face recognition hard?

Occlusion



# What makes face recognition hard?

## Viewpoint



# Simple idea for face recognition

1. Treat face image as a vector of intensities



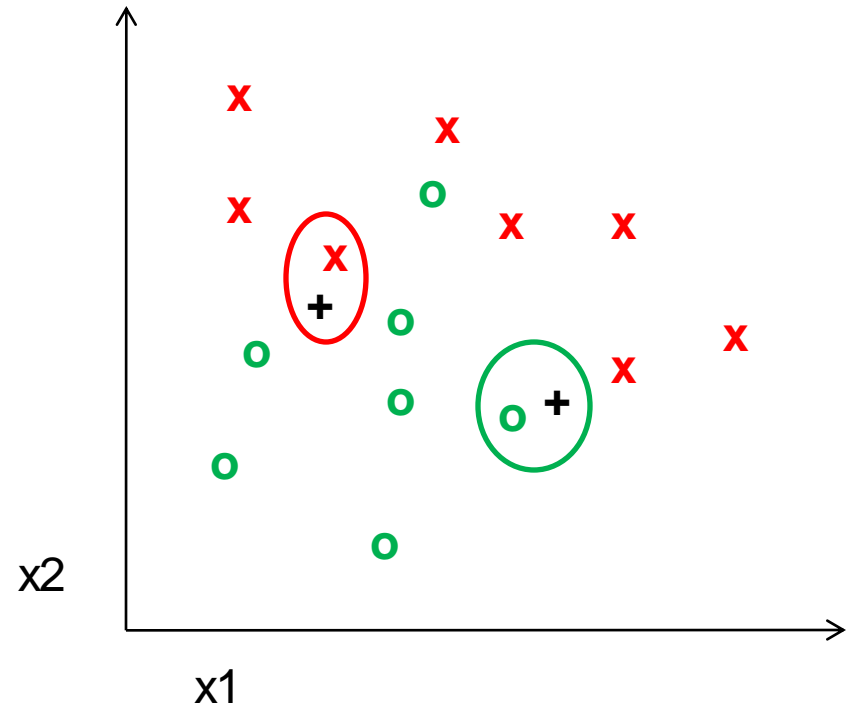
2. Recognize face by nearest neighbor in database



$$k = \underset{k}{\operatorname{argmin}} \|\mathbf{y}_k - \mathbf{x}\|$$

# Nearest neighbor classifier

- Label test sample with label of most similar training sample
- Good choice if you have few examples per class
- No training time, once feature representation is determined



# The space of all face images

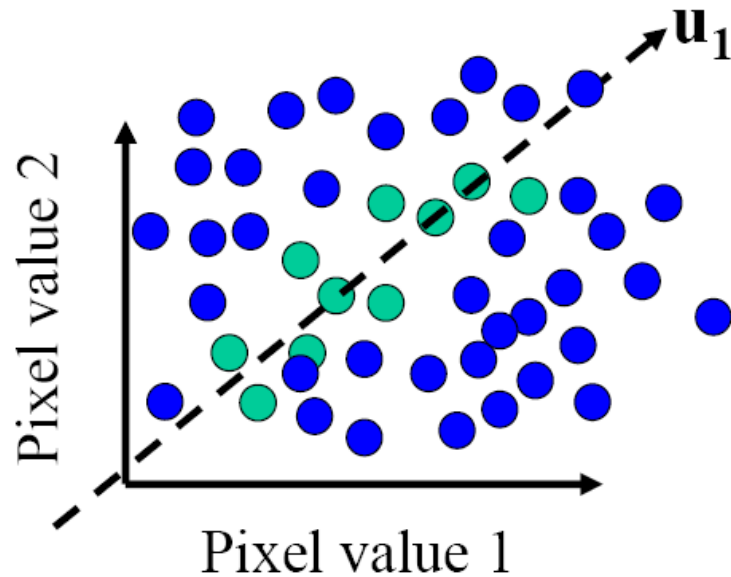
- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
  - Slow and lots of storage
- But very few 10,000-dimensional vectors are valid face images
- We want to effectively model the subspace of face images





# The space of all face images

- Eigenface idea: construct a low-dimensional linear subspace that best explains the variation in the set of face images



● A face image

● A (non-face) image

# 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)

- Choose unit vector  $\mathbf{u}$  in  $\mathbb{R}^d$  that captures the most data variance

# Principal Component Analysis

- Direction that maximizes the variance of the projected data:

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  $\|\mathbf{u}\|=1$

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

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

The direction that maximizes the variance is the eigenvector associated with the largest eigenvalue of  $\Sigma$  (can be derived using Raleigh's quotient or Lagrange multiplier)

# Implementation issue

- Covariance matrix is huge ( $M^2$  for  $M$  pixels)
- But typically # examples  $\ll M$
- Simple trick
  - $\mathbf{X}$  is  $M \times N$  matrix of normalized training data
  - Solve for eigenvectors  $\mathbf{u}$  of  $\mathbf{X}^T \mathbf{X}$  instead of  $\mathbf{X} \mathbf{X}^T$
  - Then  $\mathbf{X} \mathbf{u}$  is eigenvector of covariance  $\mathbf{X} \mathbf{X}^T$
  - Need to normalize each vector of  $\mathbf{X} \mathbf{u}$  into unit length

# Eigenfaces (PCA on face images)

1. Compute the principal components (“eigenfaces”) of the covariance matrix

$$X = [(x_1 - \mu) \ (x_2 - \mu) \ \dots \ (x_n - \mu)]$$

$$[U, \lambda] = \text{eig}(X^T X)$$

$$V = XU$$

2. Keep K eigenvectors with largest eigenvalues

$$V = V(:, \text{largest}_{\text{eig}})$$

3. Represent all face images in the dataset as linear combinations of eigenfaces

- Perform nearest neighbor on these coefficients

$$X_{pca} = V(:, \text{largest}_{\text{eig}})^T X$$

# Eigenfaces example

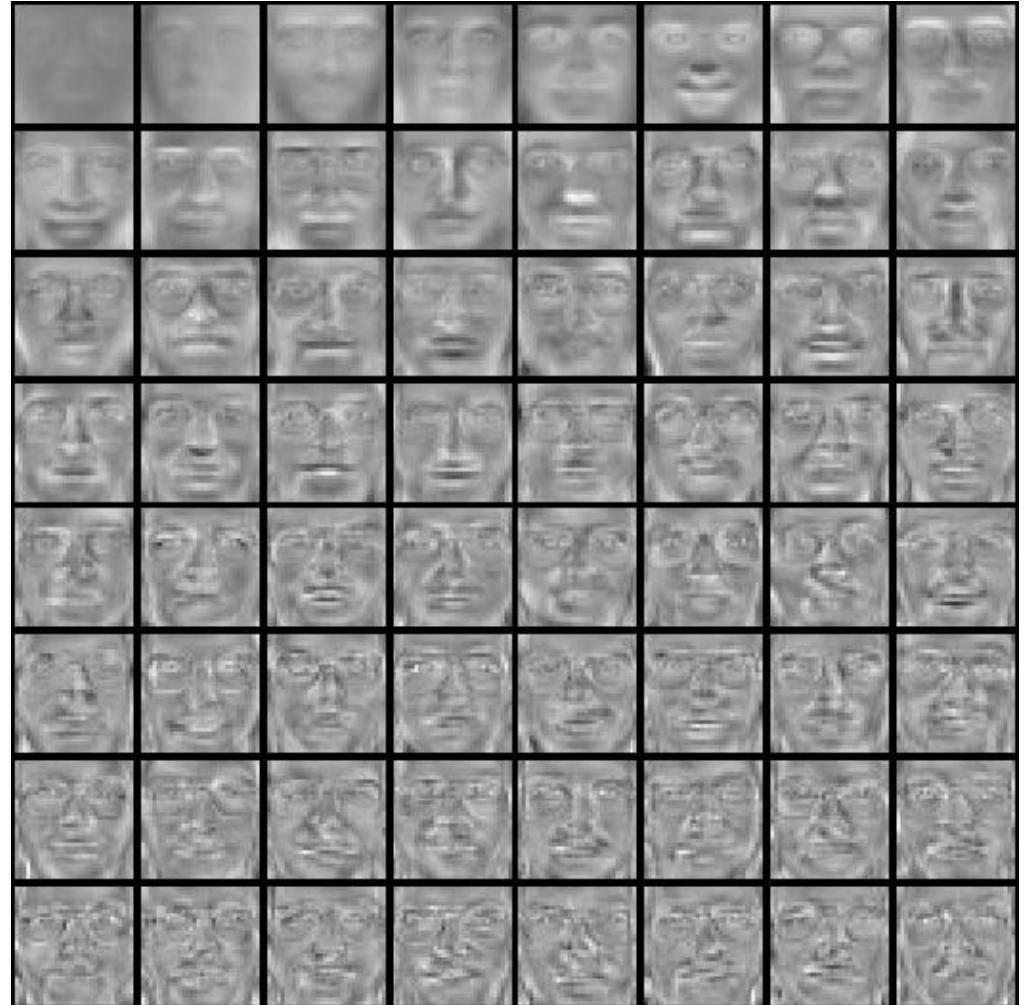
- Training images
- $\mathbf{x}_1, \dots, \mathbf{x}_N$



# Eigenfaces example

Top eigenvectors:  $u_1, \dots, u_k$

Mean:  $\mu$



# Visualization of eigenfaces

Principal component (eigenvector)  $u_k$



$\mu + 3\sigma_k u_k$



$\mu - 3\sigma_k u_k$





# Representation and reconstruction

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



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

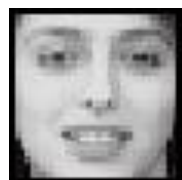
# Representation and reconstruction

- 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:



=



+



$$\hat{\mathbf{x}} = \mu + w_1 \mathbf{u}_1 + w_2 \mathbf{u}_2 + w_3 \mathbf{u}_3 + w_4 \mathbf{u}_4 + \dots$$

# Reconstruction

$P = 4$



$P = 200$

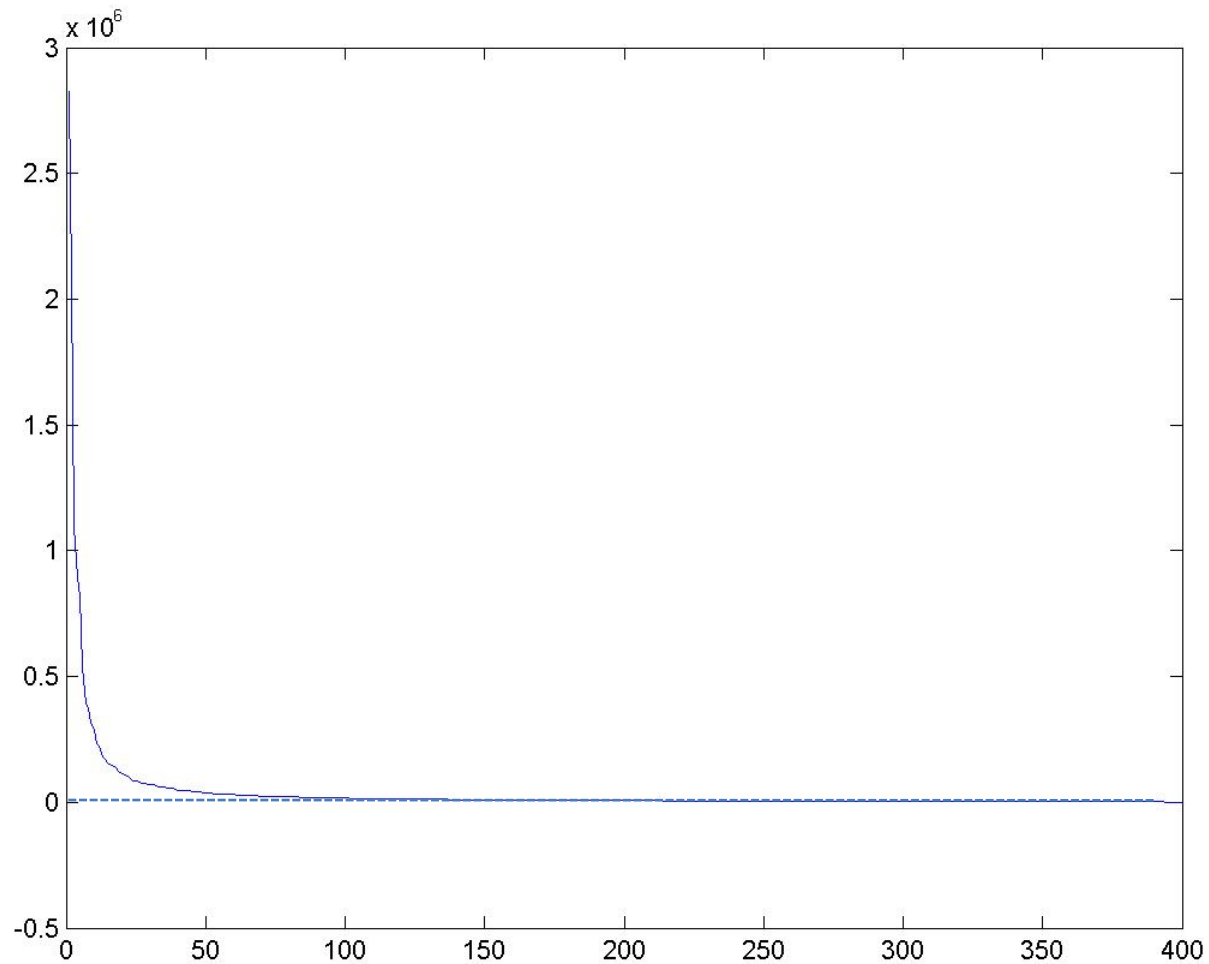


$P = 400$



After computing eigenfaces using 400 face images from ORL face database

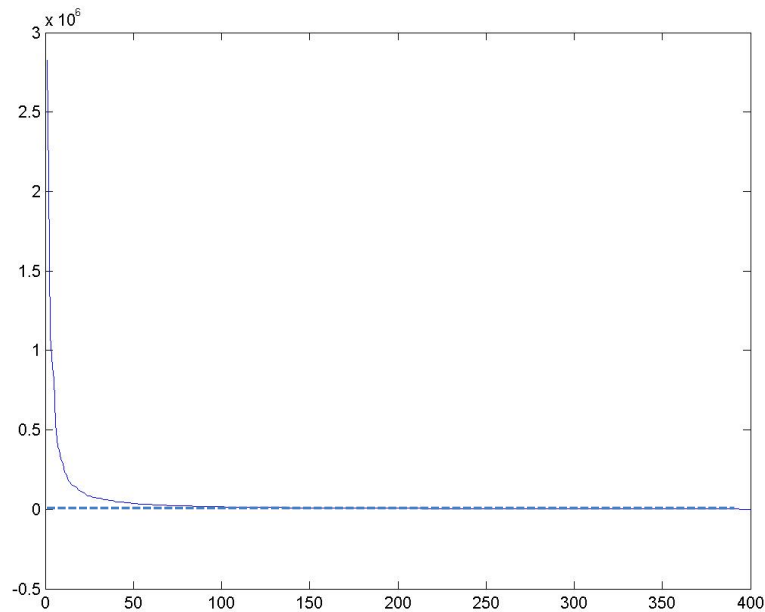
# Eigenvalues (variance along eigenvectors)



# Note

Preserving variance (minimizing MSE) does not necessarily lead to qualitatively good reconstruction.

$P = 200$



# 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  $\mathbf{x} - \hat{\mathbf{x}}$  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

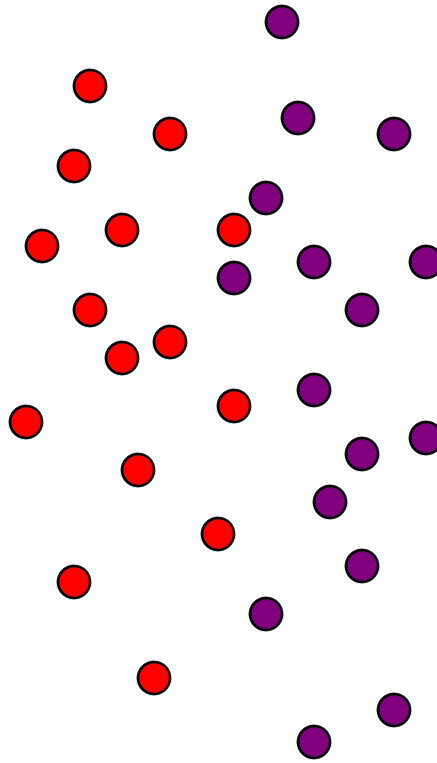
Global appearance method: not robust to misalignment, background variation





# Limitations

- The direction of maximum variance is not always good for classification

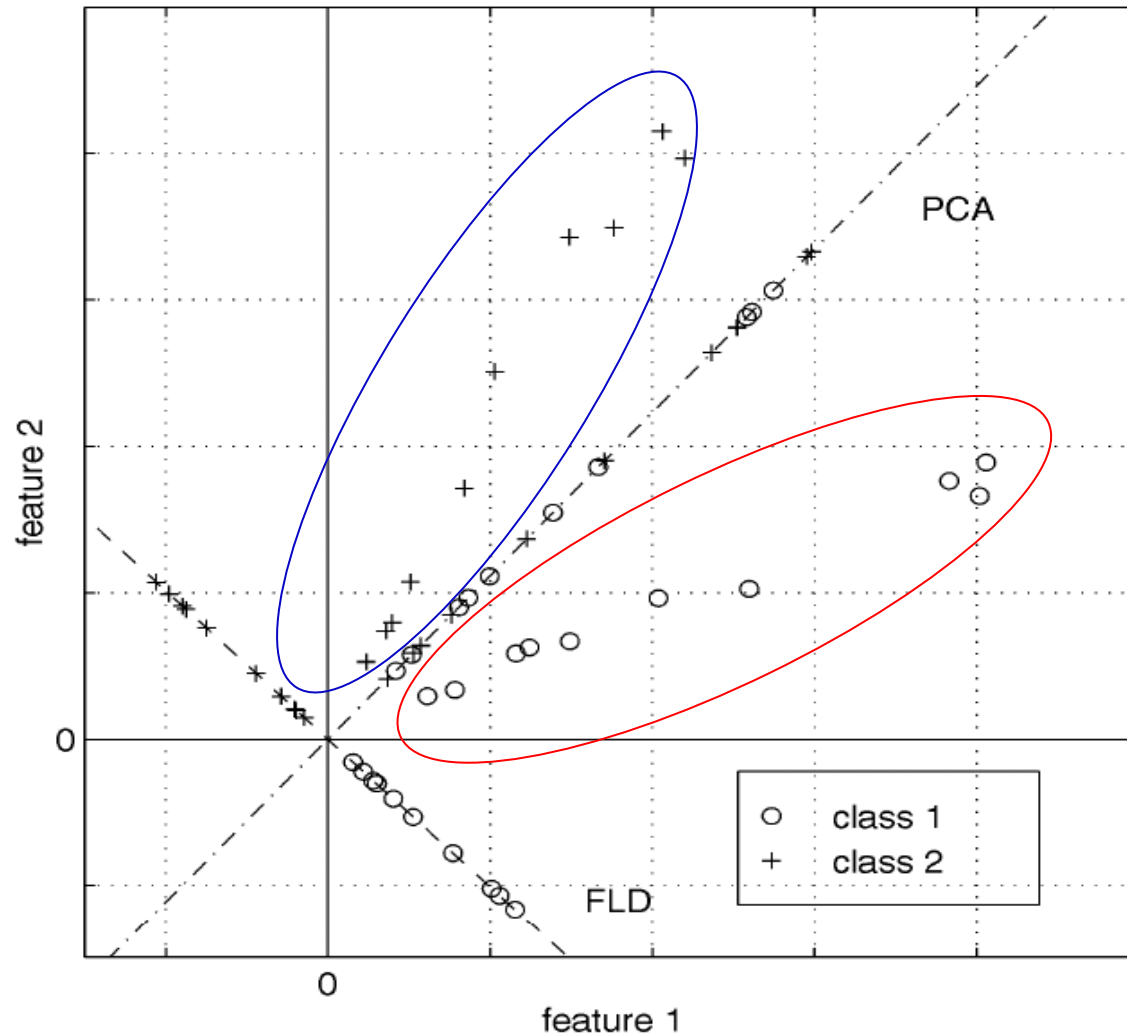


# A more discriminative subspace: FLD

- Fisher Linear Discriminants → “Fisher Faces”
- PCA preserves maximum variance
- FLD preserves discrimination
  - Find projection that maximizes scatter between classes and minimizes scatter within classes

Reference: [Eigenfaces vs. Fisherfaces, Belheumer et al., PAMI 1997](#)

# Comparing with PCA



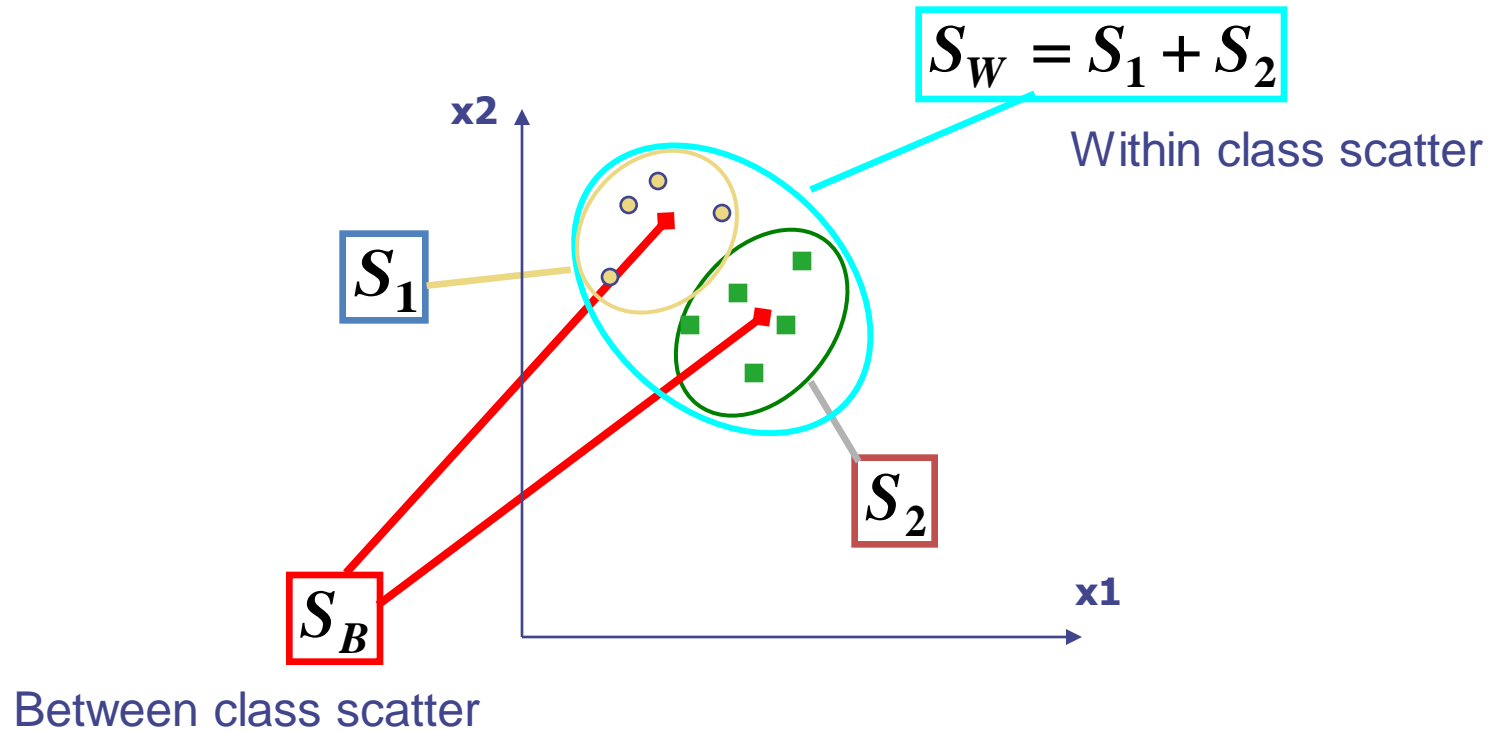
# Variables

- N Sample images:  $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$
- c classes:  $\{\mathcal{X}_1, \dots, \mathcal{X}_c\}$
- Average of each class:  $\mu_i = \frac{1}{N_i} \sum_{\mathbf{x}_k \in \mathcal{X}_i} \mathbf{x}_k$
- Average of all data:  $\mu = \frac{1}{N} \sum_{k=1}^N \mathbf{x}_k$

# Scatter Matrices

- Scatter of class  $i$ : 
$$S_i = \sum_{x_k \in \mathcal{X}_i} (x_k - \mu_i)(x_k - \mu_i)^T$$
- Within class scatter: 
$$S_W = \sum_{i=1}^c S_i$$
- Between class scatter: 
$$S_B = \sum_{i=1}^c N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

# Illustration



# Mathematical Formulation

- After projection
  - Between class scatter  $y_k = W^T x_k$
  - Within class scatter  $\tilde{S}_B = W^T S_B W$
  - Within class scatter  $\tilde{S}_W = W^T S_W W$

- Objective:

$$W_{opt} = \arg \max_W \frac{|\tilde{S}_B|}{|\tilde{S}_W|} = \arg \max_W \frac{|W^T S_B W|}{|W^T S_W W|}$$

- Solution: Generalized Eigenvectors

$$S_B w_i = \lambda_i S_W w_i \quad i = 1, \dots, m$$

- Rank of  $W_{opt}$  is limited
  - Rank( $S_B$ )  $\leq$  |C|-1
  - Rank( $S_W$ )  $\leq$  N-C

# Recognition with FLD

- Use PCA to reduce dimensions to N-C

$$W_{pca} = \text{pca}(X)$$

- Compute within-class and between-class scatter matrices for PCA coefficients

$$S_i = \sum_{x_k \in \mathcal{X}_i} (x_k - \mu_i)(x_k - \mu_i)^T \quad S_W = \sum_{i=1}^c S_i \quad S_B = \sum_{i=1}^c N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

- Solve generalized eigenvector problem

$$W_{fld} = \arg \max_W \frac{|W^T S_B W|}{|W^T S_W W|} \quad S_B w_i = \lambda_i S_W w_i \quad i = 1, \dots, m$$

- Project to FLD subspace (c-1 dimensions)

$$W_{opt}^T = W_{fld}^T W_{pca}^T \quad \hat{x} = W_{opt}^T x$$

- Classify by nearest neighbor

Note: x in step 2 refers to PCA coef; x in step 4 refers to original data

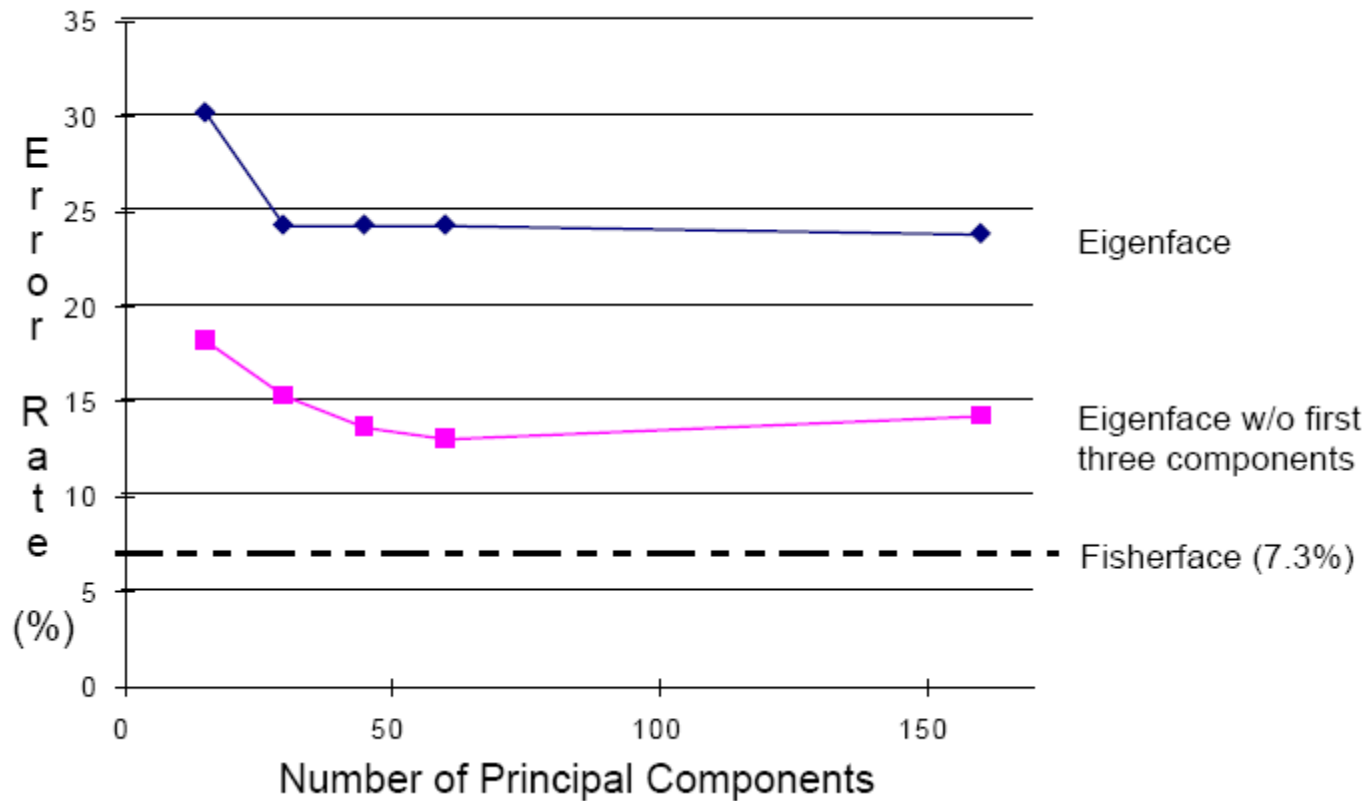


# Results: Eigenface vs. Fisherface

- Input: 160 images of 16 people
- Train: 159 images
- Test: 1 image
- Variation in Facial Expression, Eyewear, and Lighting



# Eigenfaces vs. Fisherfaces

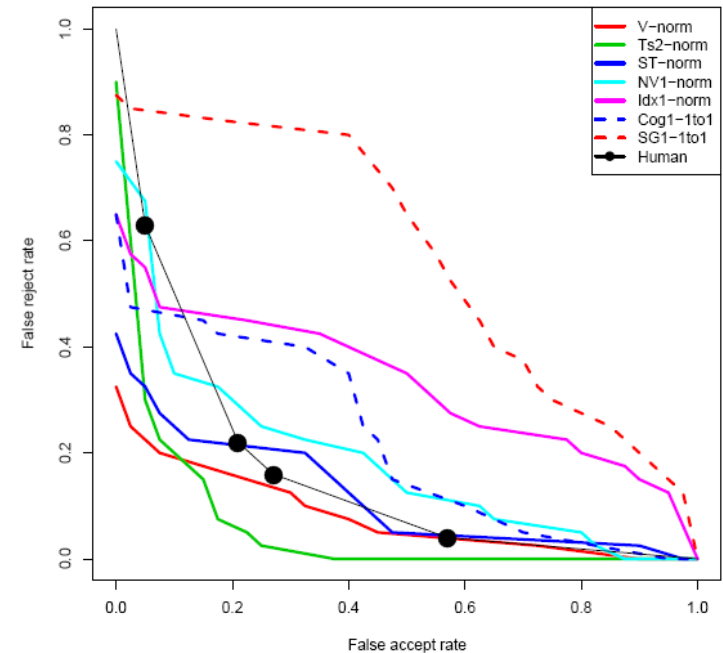
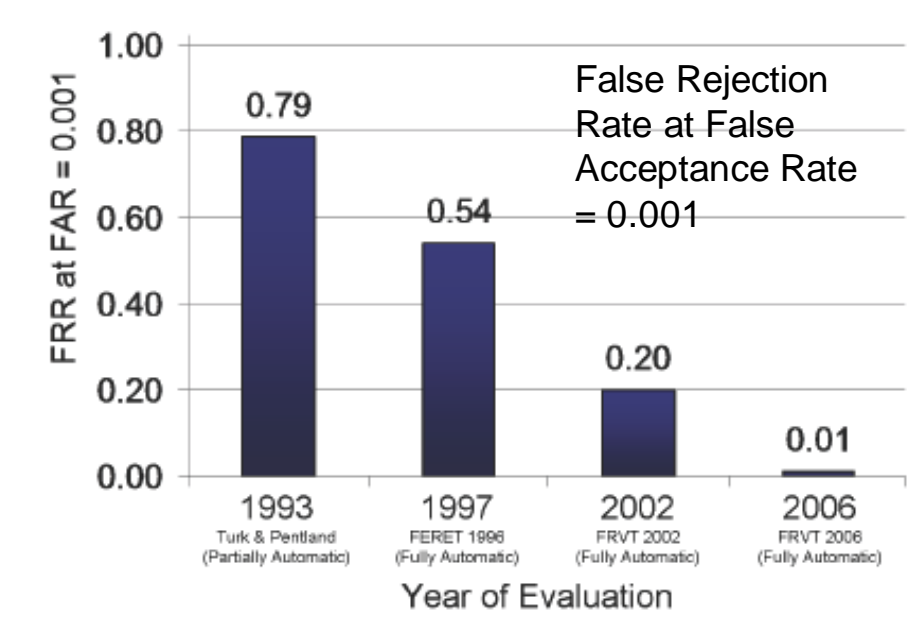


# Large scale comparison of methods

- [FRVT 2006 Report](#)
- Not much (or any) information available about methods, but gives idea of what is doable



# FVRT Challenge: interesting findings



- Left: Major progress since Eigenfaces
- Right: Computers outperformed humans in controlled settings (cropped frontal face, known lighting, aligned)
- Humans outperform greatly in less controlled settings (viewpoint variation, no crop, no alignment, change in age, etc.)

# State-of-the-art Face Recognizers

- Most recent research focuses on “faces in the wild”, recognizing faces in normal photos
  - Classification: assign identity to face
  - Verification: say whether two people are the same
- Important steps
  1. Detect
  2. Align
  3. Represent
  4. Classify

# DeepFace: Closing the Gap to Human-Level Performance in Face Verification

Yaniv Taigman

Ming Yang

Marc'Aurelio Ranzato

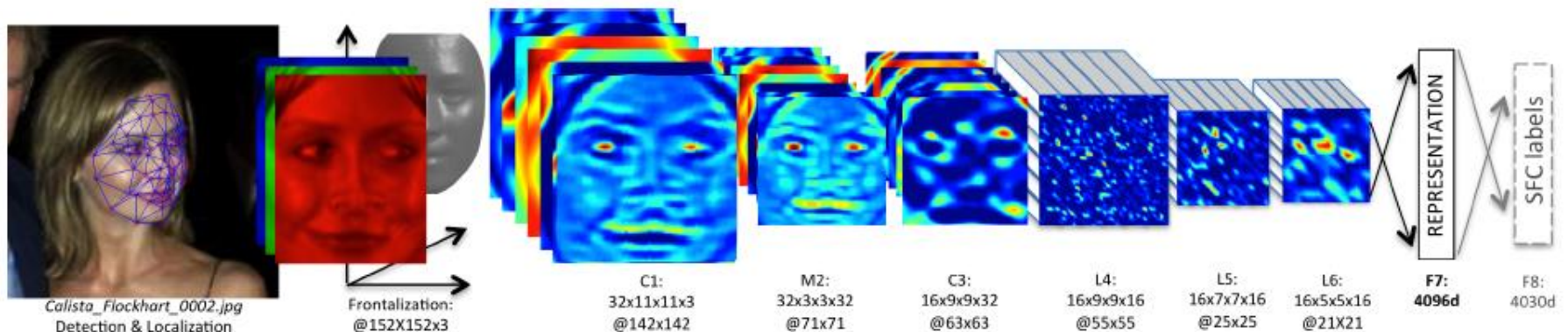
Lior Wolf

Facebook AI Research  
Menlo Park, CA, USA

{yaniv, mingyang, ranzato}@fb.com

Tel Aviv University  
Tel Aviv, Israel

wolf@cs.tau.ac.il



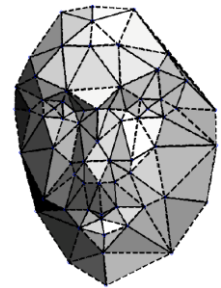
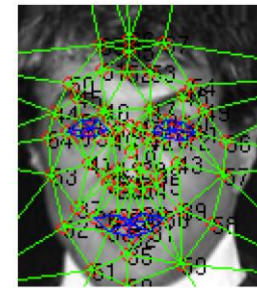
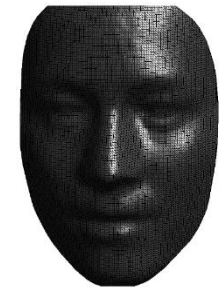
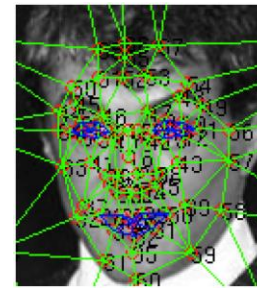
[DeepFace: Closing the Gap to Human-Level Performance in Face Verification](#)

Taigman, Yang, Ranzato, & Wolf (Facebook, Tel Aviv), CVPR 2014

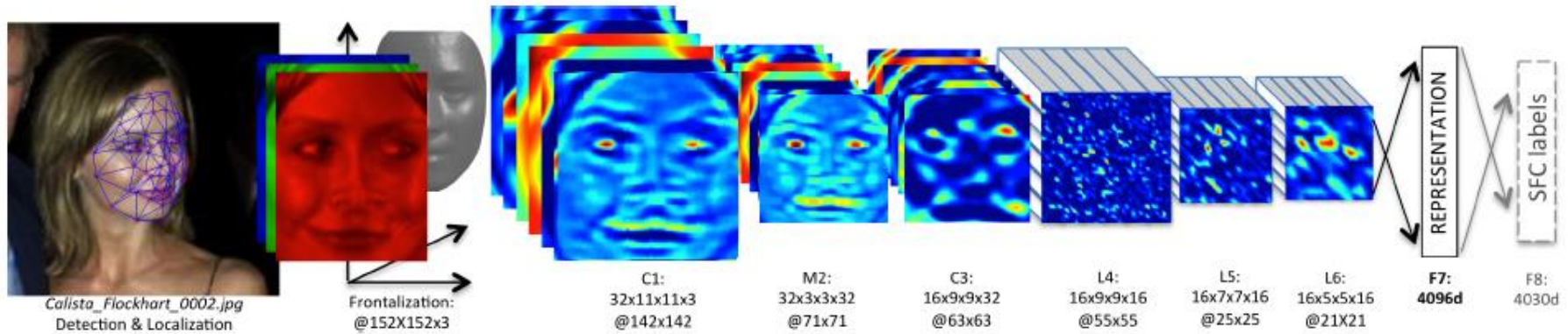
Following slides adapted from Daphne Tsatsoulis

# Face Alignment

1. Detect a face and 6 fiducial markers using a support vector regressor (SVR)
2. Iteratively scale, rotate, and translate image until it aligns with a target face
3. Localize 67 fiducial points in the 2D aligned crop
4. Create a generic 3D shape model by taking the average of 3D scans from the USF Human-ID database and manually annotate the 67 anchor points
5. Fit an affine 3D-to-2D camera and use it to direct the warping of the face



# Train DNN classifier on aligned faces



Architecture (deep neural network classifier)

- Two convolutional layers (with one pooling layer)
- 3 locally connected and 2 fully connected layers
- > 120 million parameters

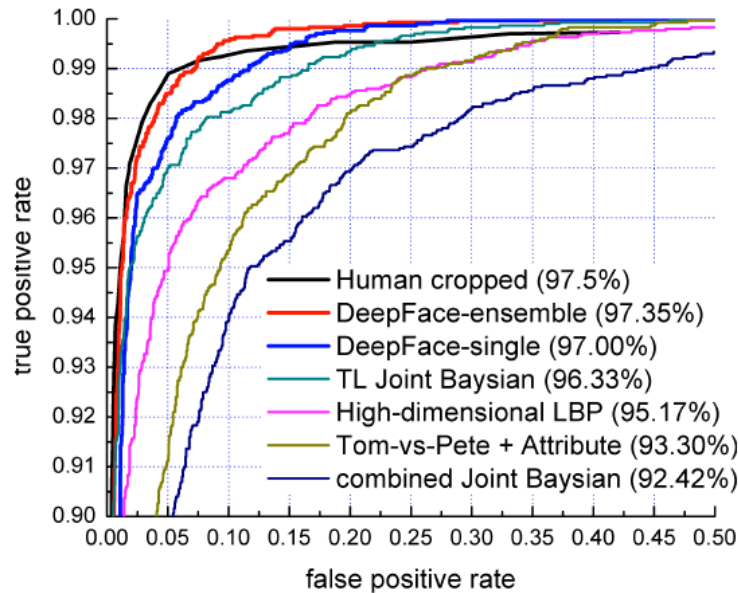
Train on dataset with 4400 individuals, ~1000 images each

- Train to identify face among set of possible people

Verification is done by comparing features at last layer for two faces



# Results: Labeled Faces in the Wild Dataset



Method	Accuracy $\pm$ SE	Protocol
Joint Bayesian [6]	0.9242 $\pm$ 0.0108	restricted
Tom-vs-Pete [4]	0.9330 $\pm$ 0.0128	restricted
High-dim LBP [7]	0.9517 $\pm$ 0.0113	restricted
TL Joint Bayesian [5]	0.9633 $\pm$ 0.0108	restricted
DeepFace-single	<b>0.9592</b> $\pm$ 0.0029	unsupervised
DeepFace-single	<b>0.9700</b> $\pm$ 0.0028	restricted
DeepFace-ensemble	<b>0.9715</b> $\pm$ 0.0027	restricted
DeepFace-ensemble	<b>0.9735</b> $\pm$ 0.0025	unrestricted
Human, cropped	0.9753	

Performs similarly to humans!

(note: humans would do better with uncropped faces)

Experiments show that alignment is crucial (0.97 vs 0.88) and that deep features help (0.97 vs. 0.91)

# Face recognition by humans

[Face recognition by humans: 20 results \(2005\)](#)

## Result 1

---

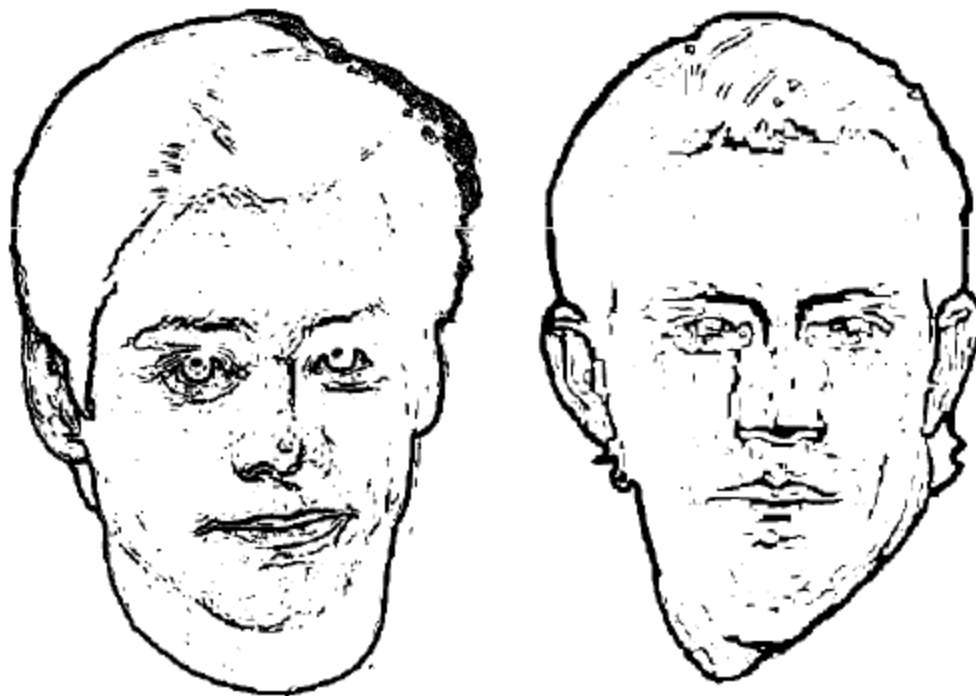
- ▶ Humans can recognize faces in extremely low resolution images.



## Result 3

---

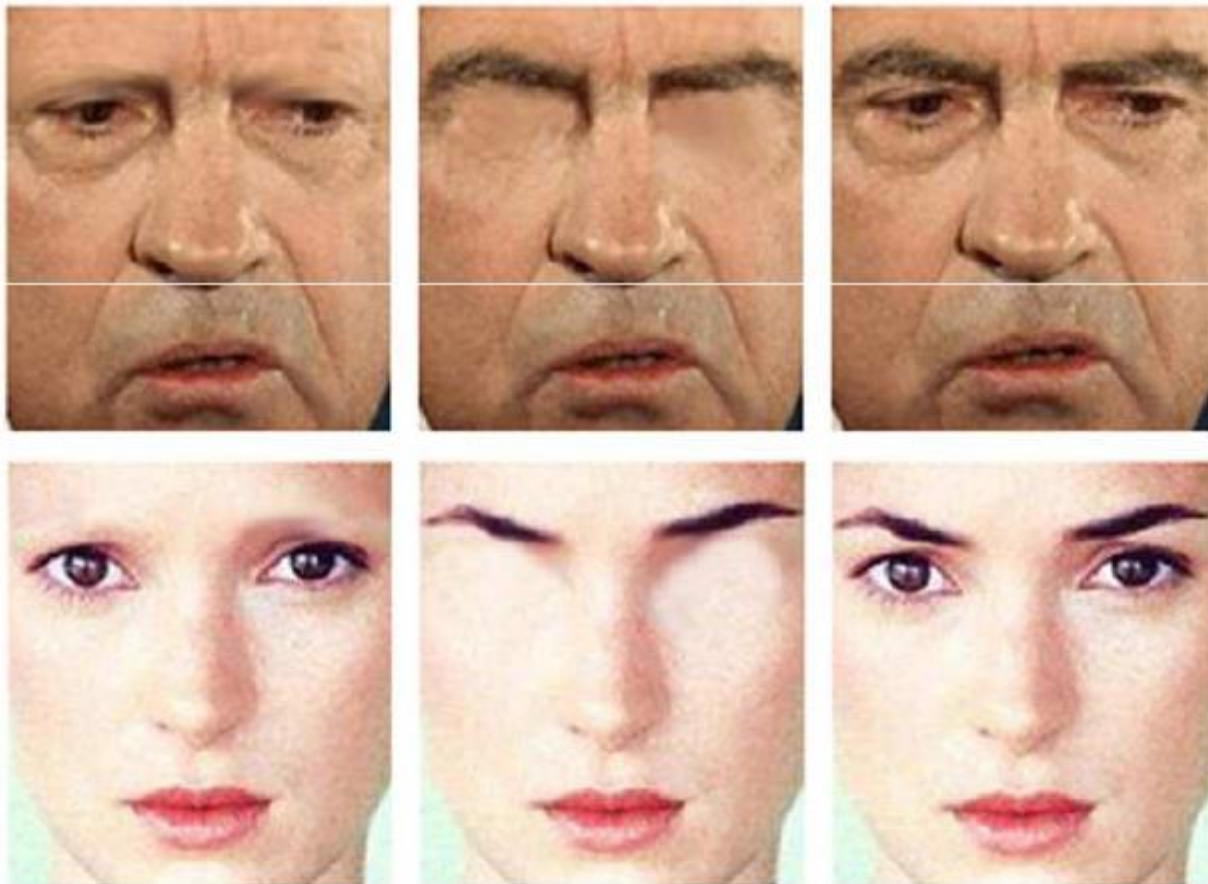
- ▶ High-frequency information by itself does not lead to good face recognition performance



## Result 5

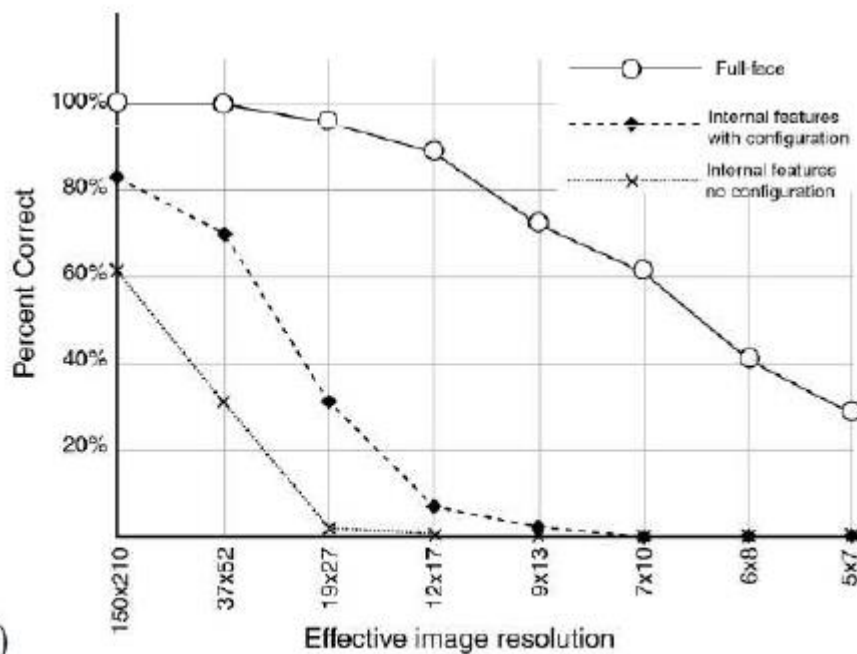
---

- ▶ 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

---

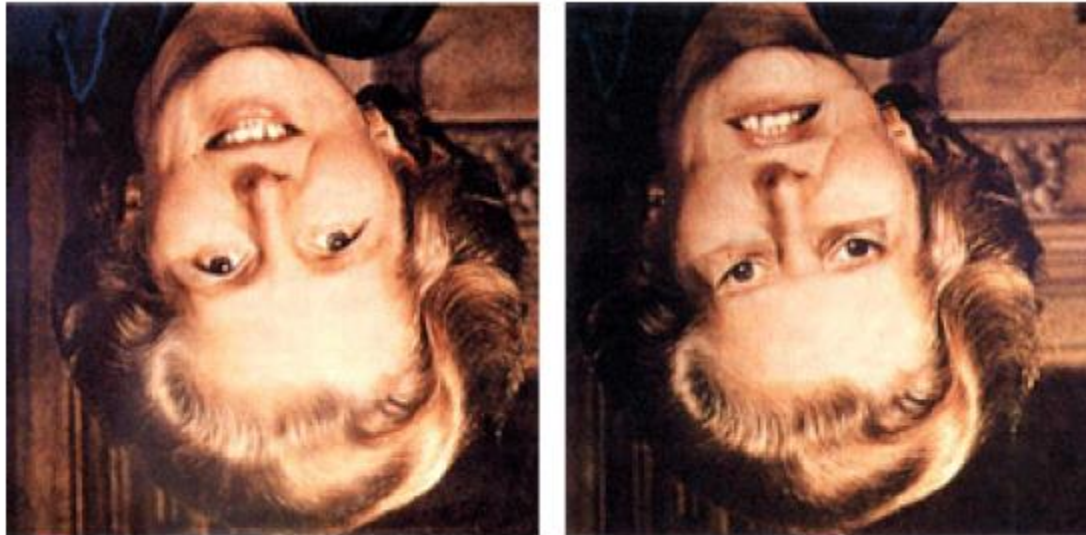
- ▶ The important configural relations appear to be independent across the width and height dimensions



## Result 8

---

- ▶ Vertical inversion dramatically reduces recognition performance





## Result 12

---

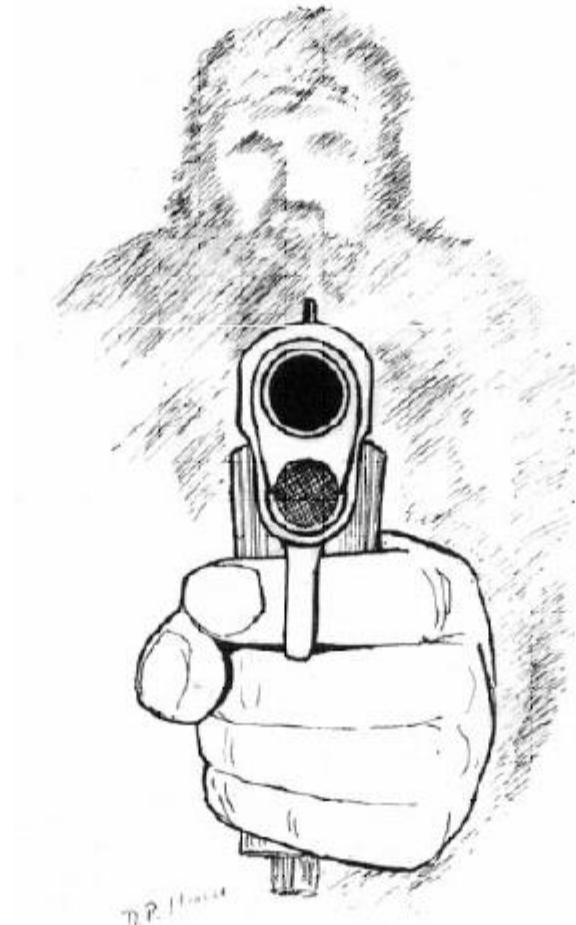
- ▶ Contrast polarity inversion dramatically impairs recognition performance, possibly due to compromised ability to use pigmentation cues



# Result 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
- Simple nearest neighbor classifiers can be effective, but features matter
  - FLD, Deep networks
- Computer face recognition works very well under controlled environments (since 2006)
- Also starting to perform at human level in uncontrolled settings (recent progress: better alignment, features, more data)

# Next week

- Image categorization, features, and classifiers