COS 429: COMPUTER VISON

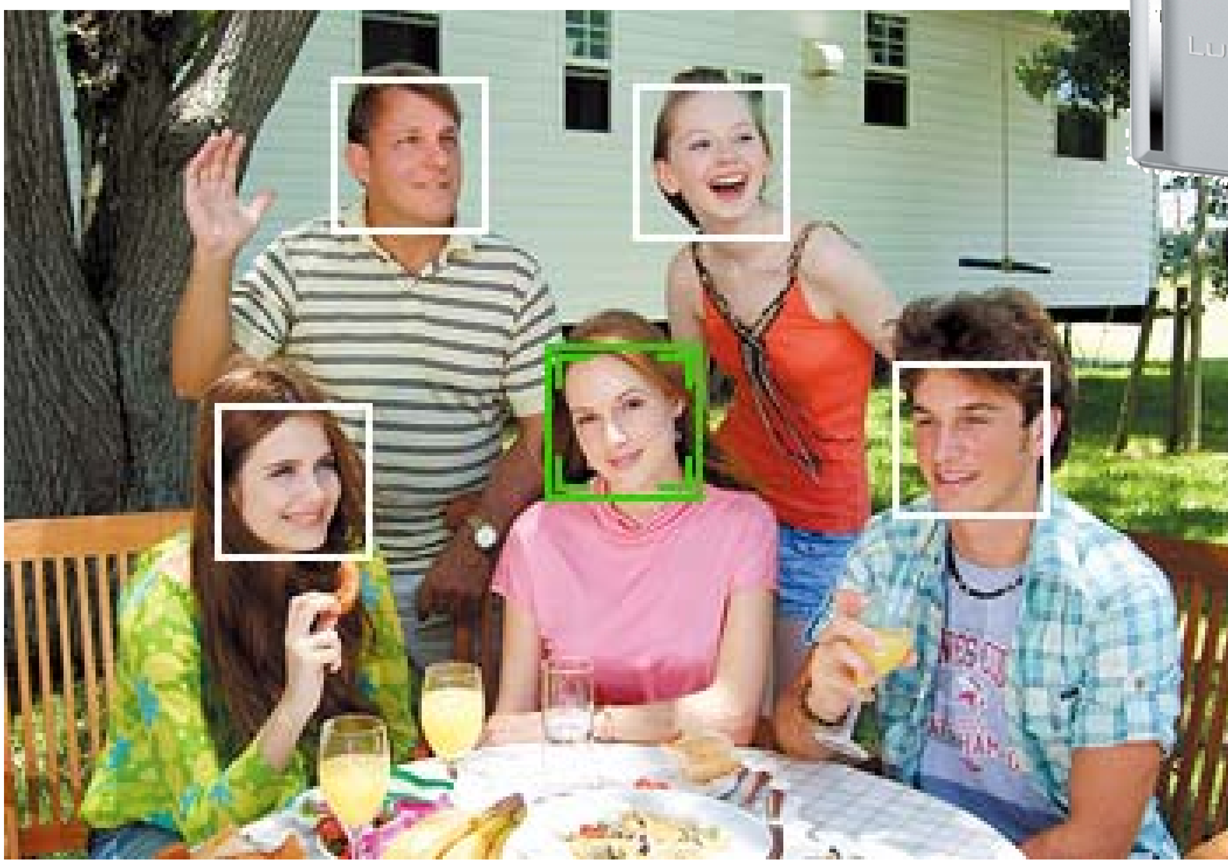
Face Recognition

- Intro to recognition
- PCA and Eigenfaces
- LDA and Fisherfaces
- Face detection: Viola & Jones
- (Optional) generic object models for faces: the Constellation Model

Reading: Turk & Pentland, ???

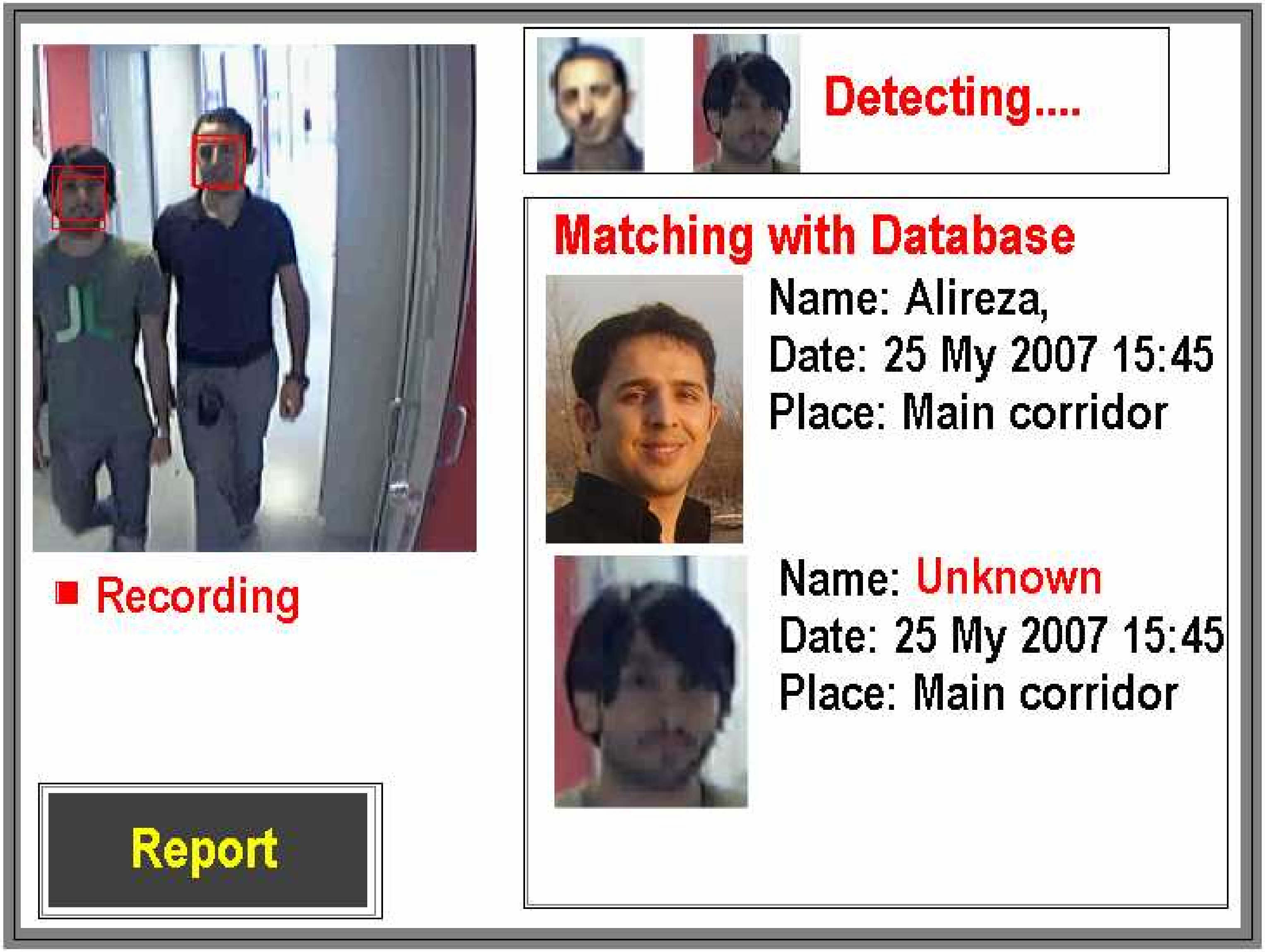
Face Recognition

- Digital photography



Face Recognition

- Digit
- Surv

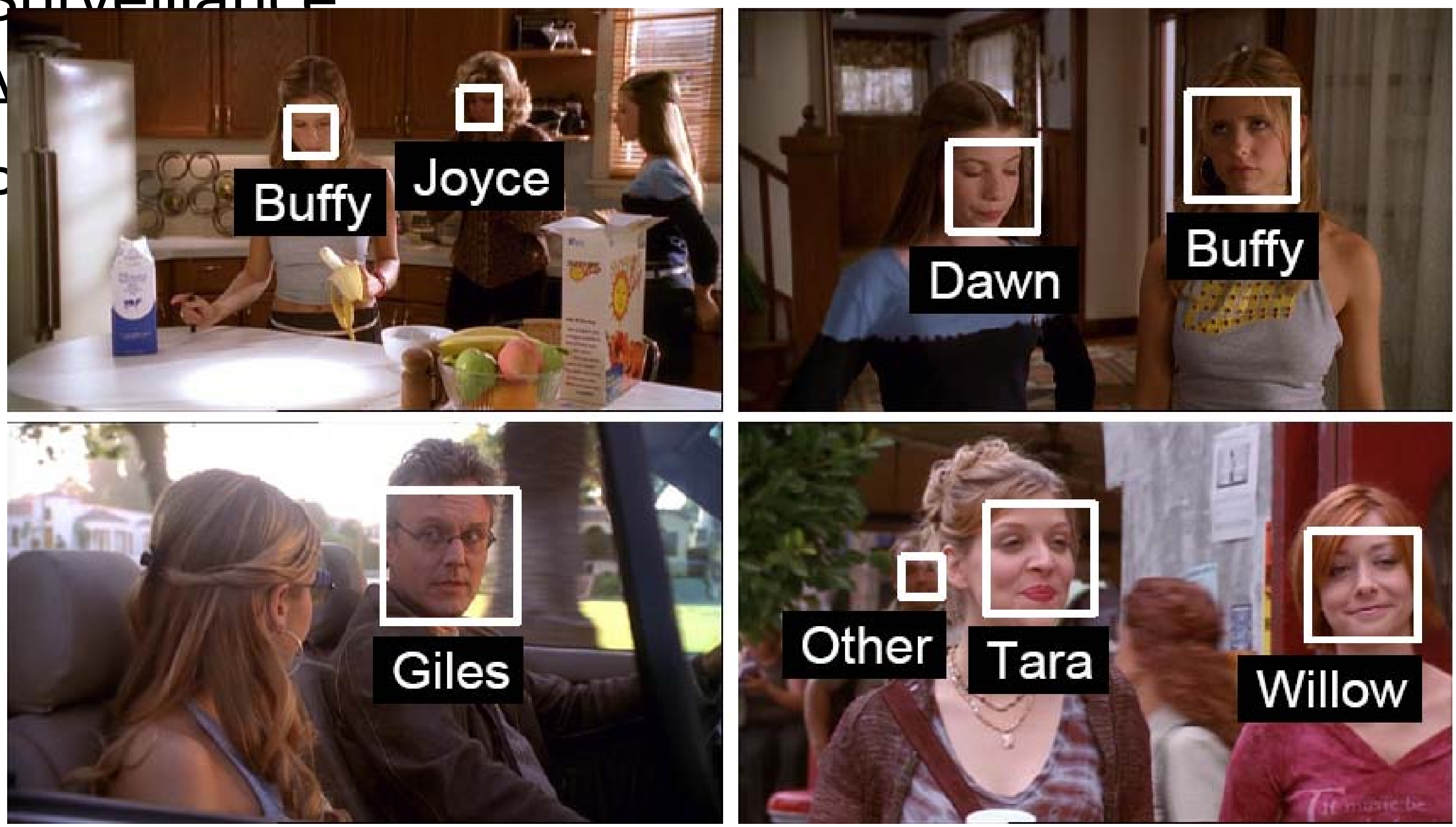


Face Recognition

- Digital photography
- Surveillance
- Album organization

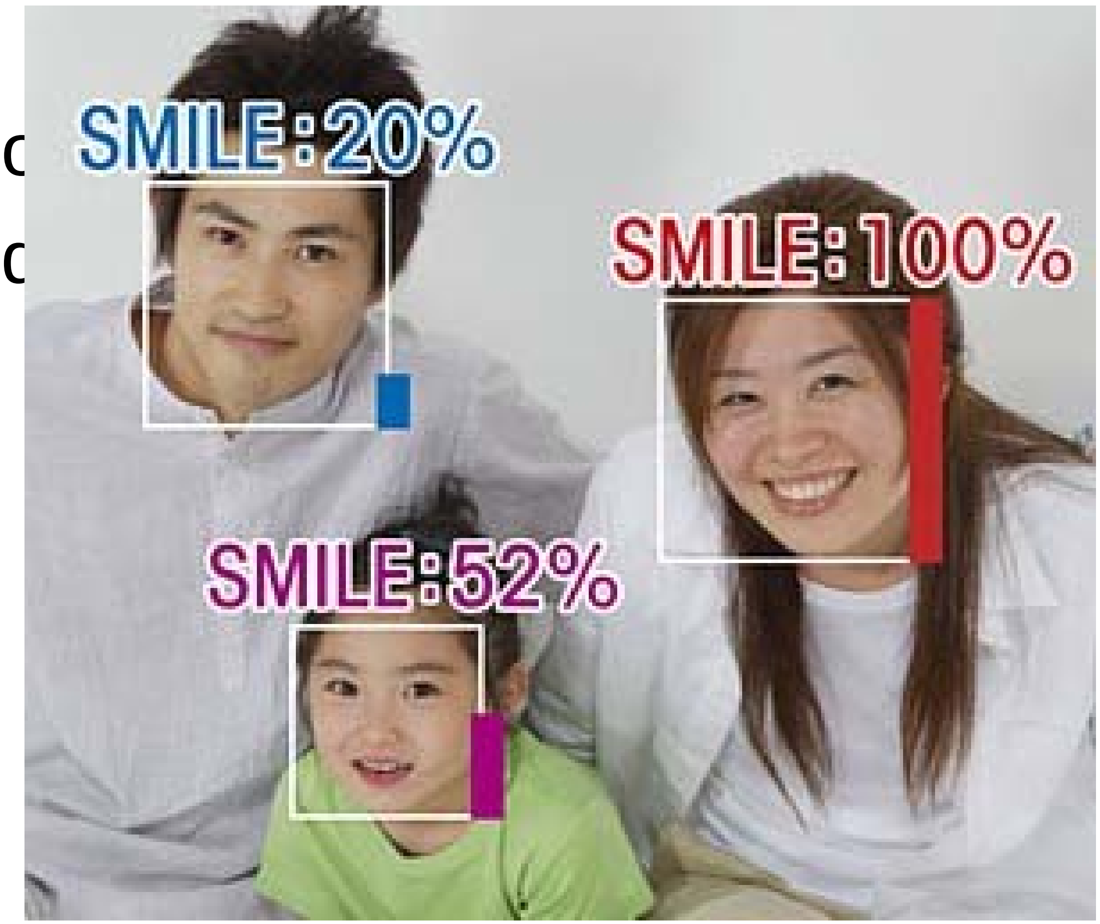
Face Recognition

- Digital photography
- Surveillance
- A
- P



Face Recognition

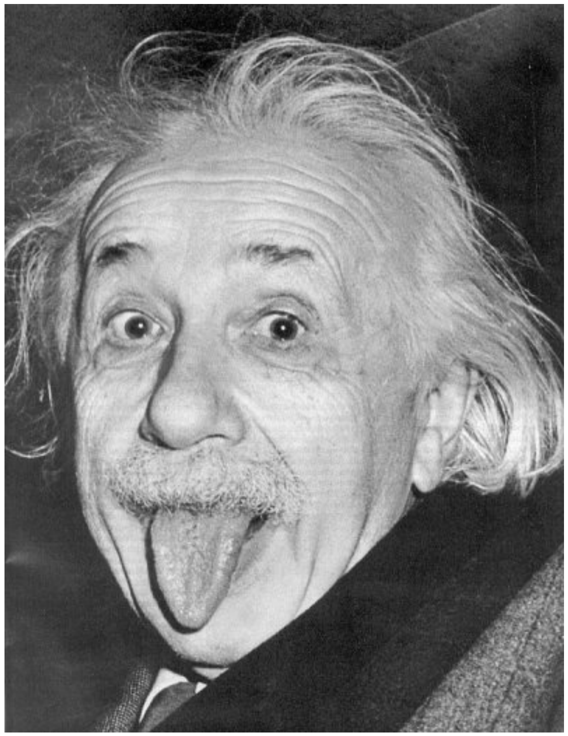
- Digital photography
- Surveillance
- Album organizati
- Person tracking/i
- Emotions and expressions



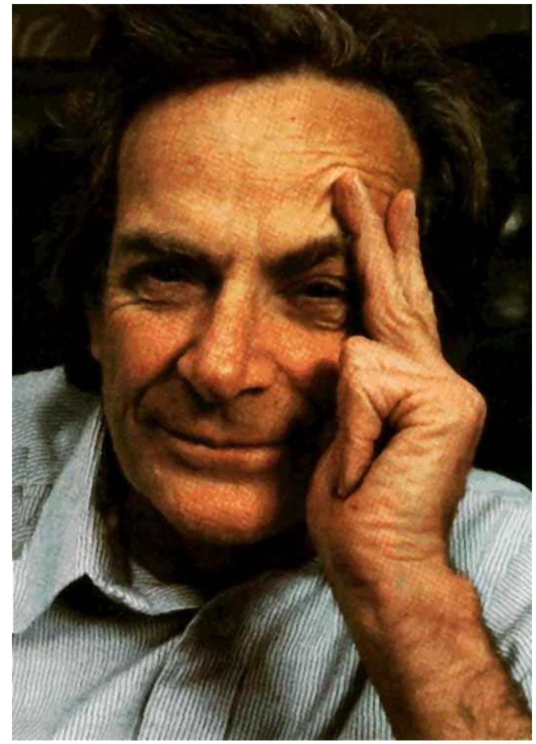
Face Recognition

- Digital photography
- Surveillance
- Album organization
- Person tracking/id.
- Emotions and expressions
- Security/warfare
- Tele-conferencing
- Etc.

What's 'recognition'?

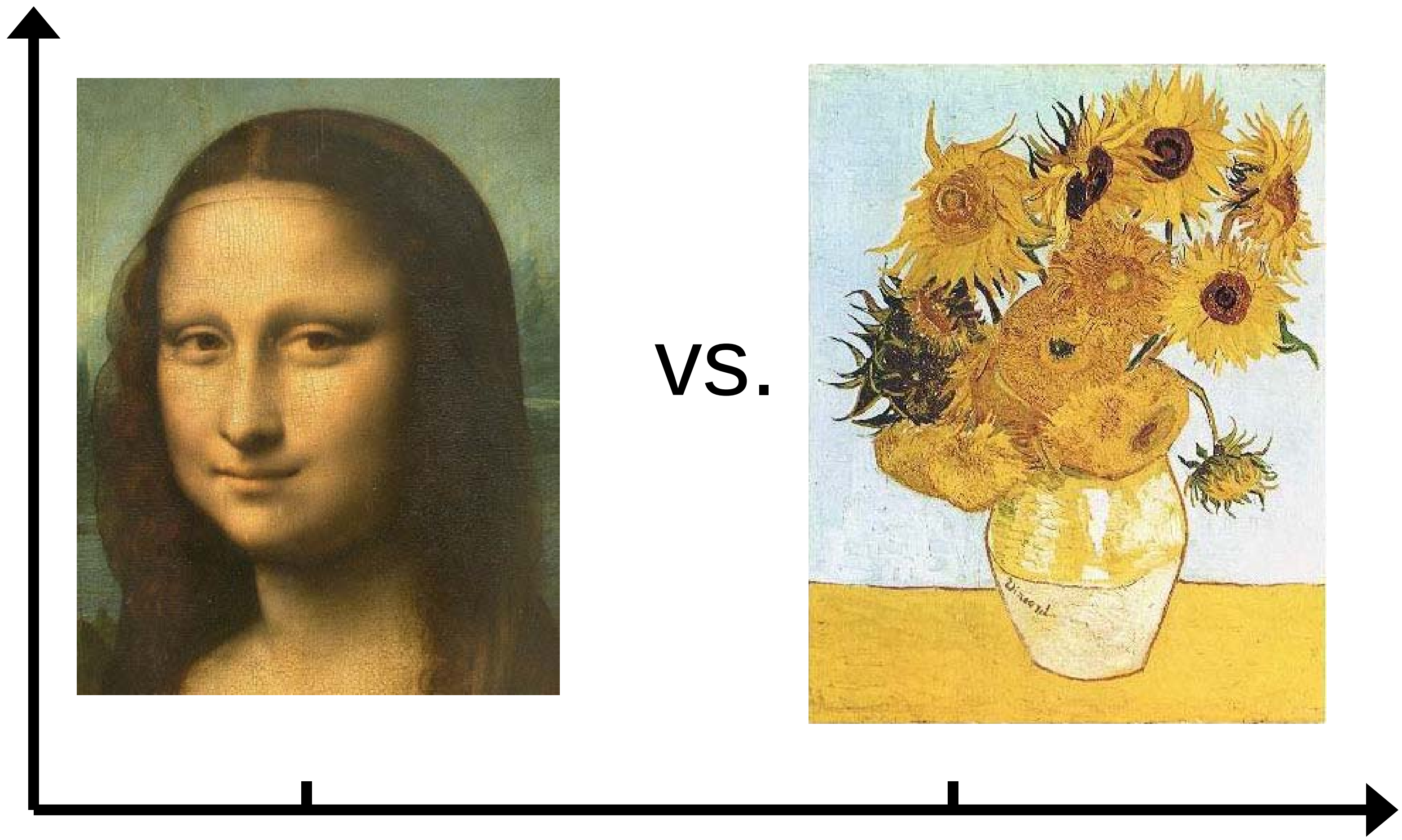


vs.



Identification or Discrimination

What's 'recognition'?



Identification or Discrimination

Categorization or Classification

Wh ’?

Yes, there are faces

No localization

Identification or Discrimination

Categorization or Classification

Wh ’?

Yes, there is John Lennon

No localization

Identification or Discrimination

Categorization or Classification

Wh ’?

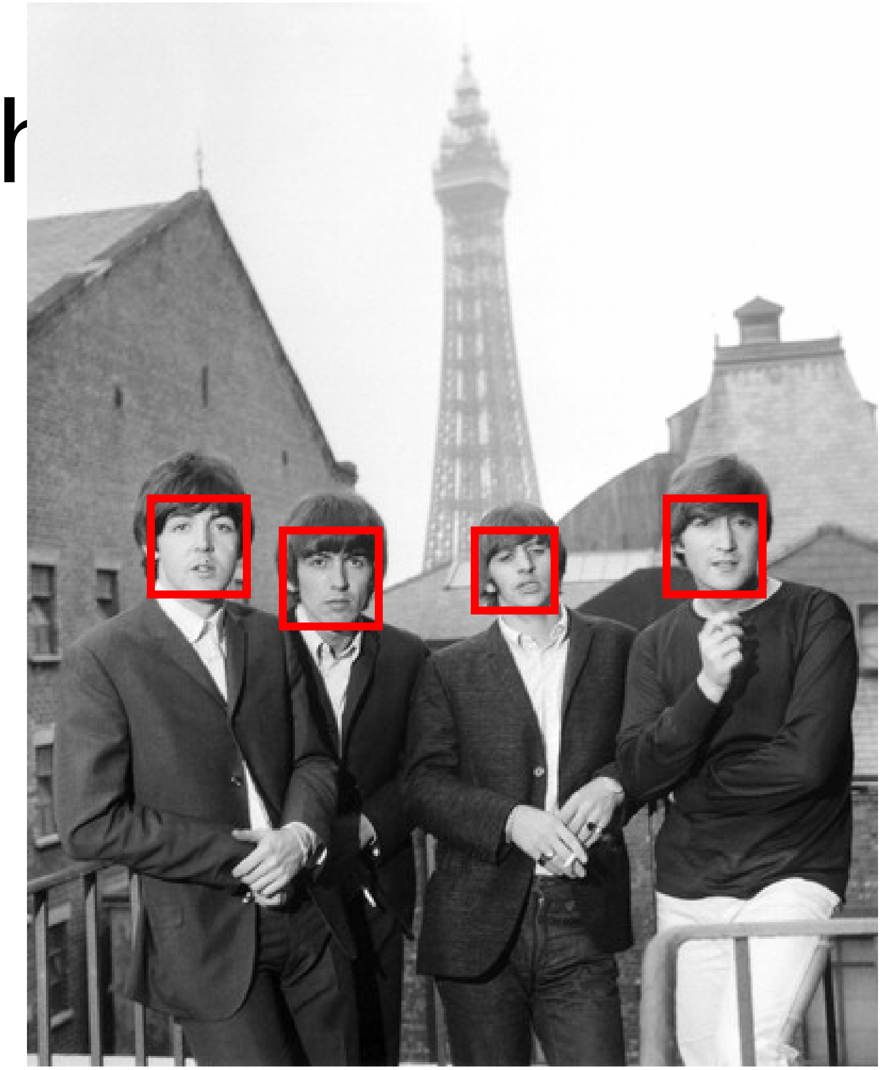
Detection or Localizatoin

No localization

Identification or Discrimination

Categorization or Classification

Wh ’?



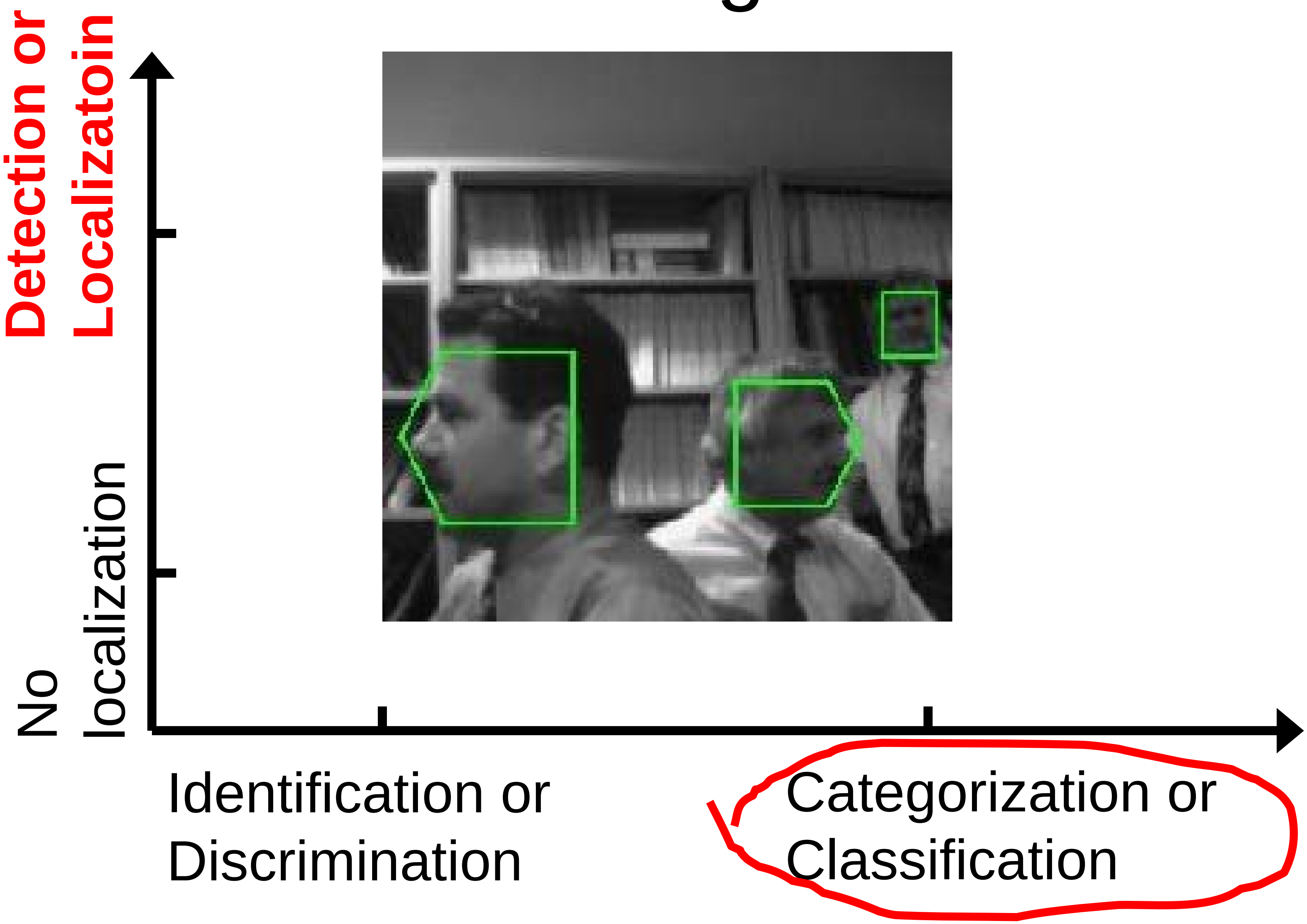
Detection or Localizatoin

No localization

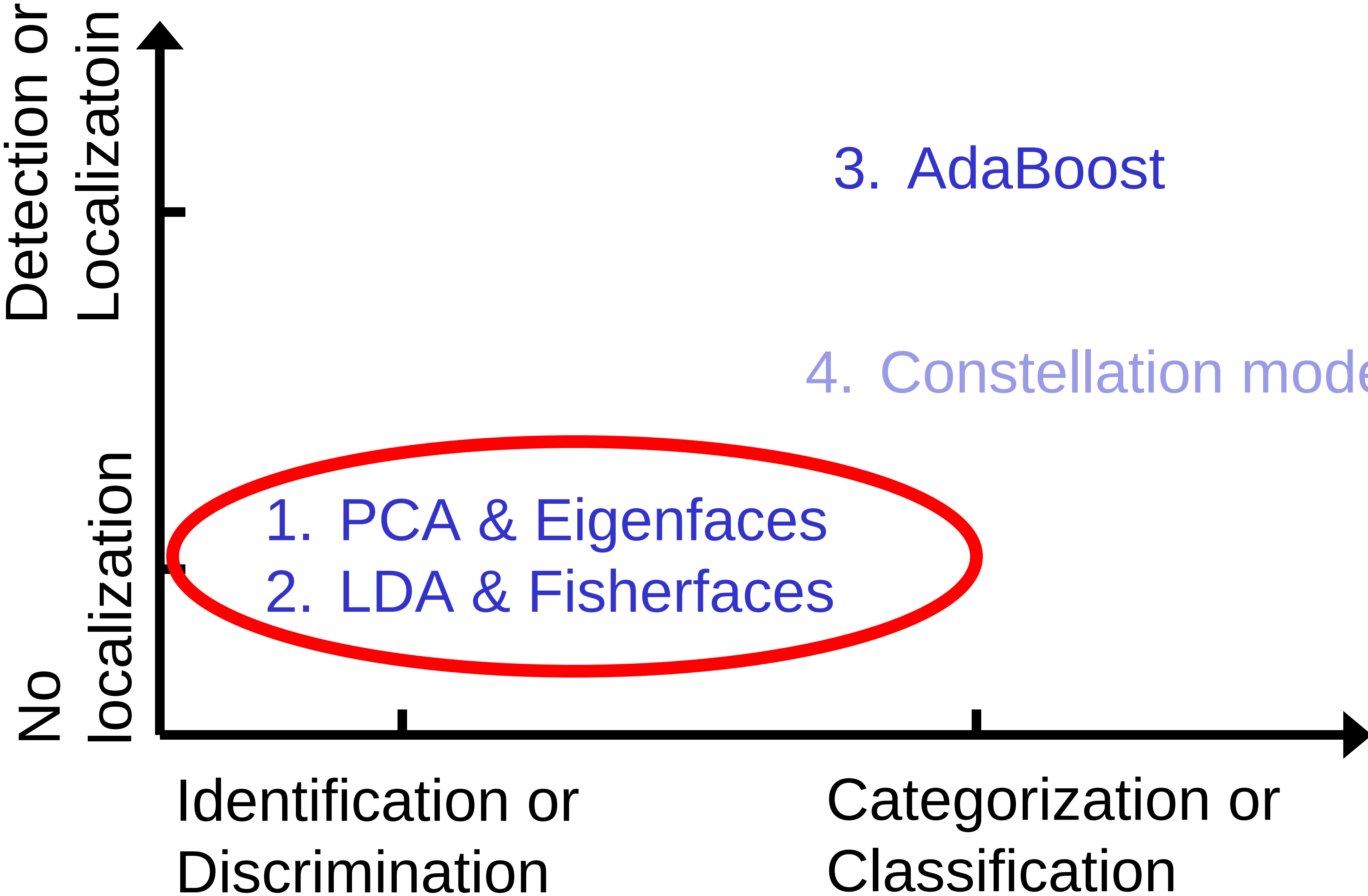
Identification or Discrimination

Categorization or Classification

What's 'recognition'?



Today’s agenda



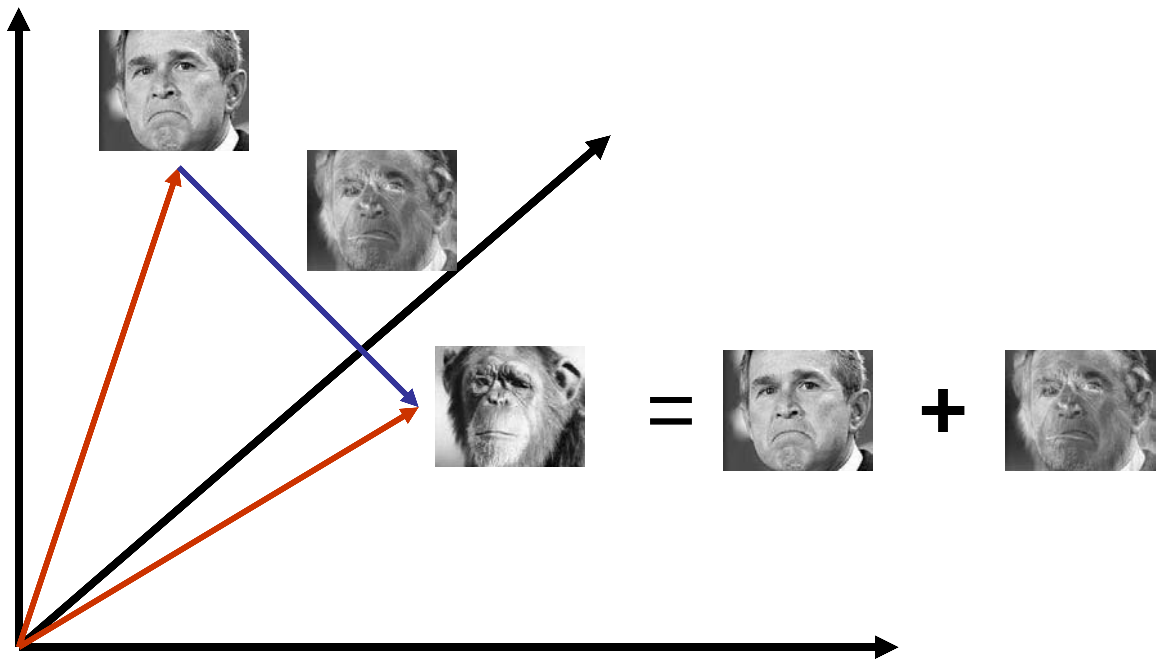
Eigenfaces and Fishfaces

- Introduction
- Techniques
  - Principle Component Analysis (PCA)
  - Linear Discriminant Analysis (LDA)
- Experiments

References:

1. Turk and Penland, Eigenfaces for Recognition, 1991
2. Belhumeur, Hespanha and Kriegman, Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection

The Space of Faces



- An image is a point in a high dimensional space
  - An N x M image is a point in R^{NM}
  - We can define vectors in this space as we did in the 2D case

[Thanks to Chuck Dyer, Steve Seitz, Nishino]

Key Idea

- Images in the possible set $\chi = \{\hat{x}_{RL}^{P}\}$ are highly correlated.
- So, compress them to a low-dimensional subspace that captures key appearance characteristics of the visual DOFs.
- EIGENFACES: [Turk and Pentland]

USE PCA!

- Two simple but useful techniques

For example, a generative graphical model:

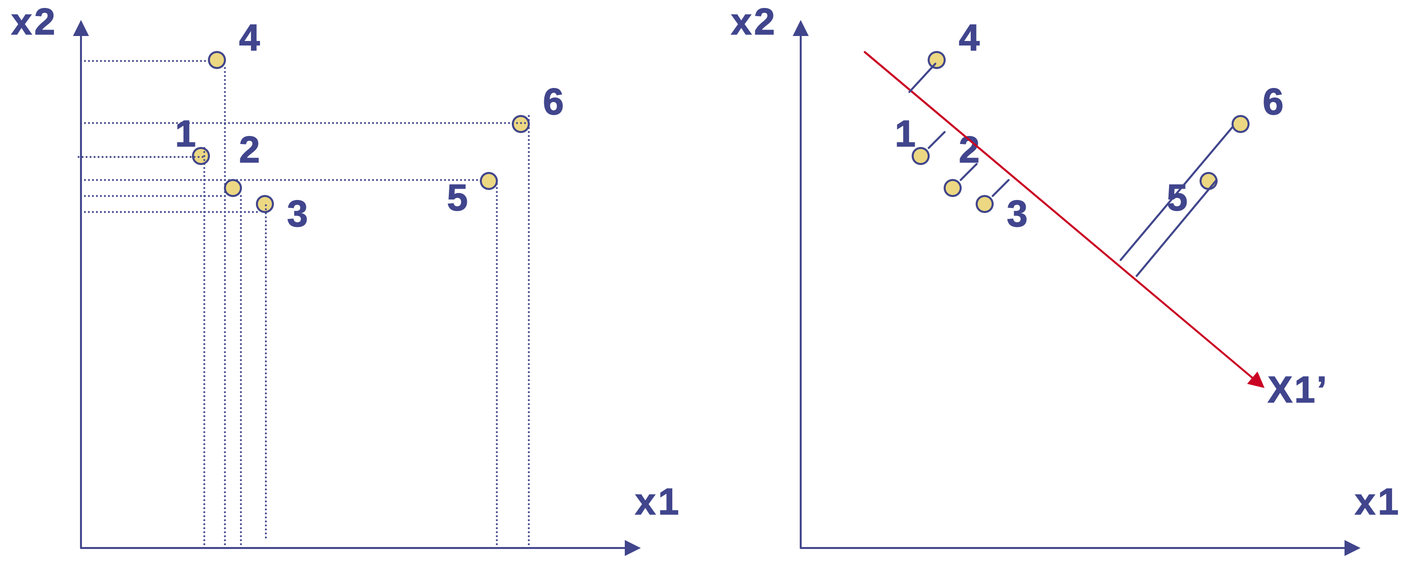
$$P(identity,image) = P(identiy|image)\boxed{P(image)}$$

Preprocessing model
(can be performed by PCA)

Principal Component Analysis (PCA)

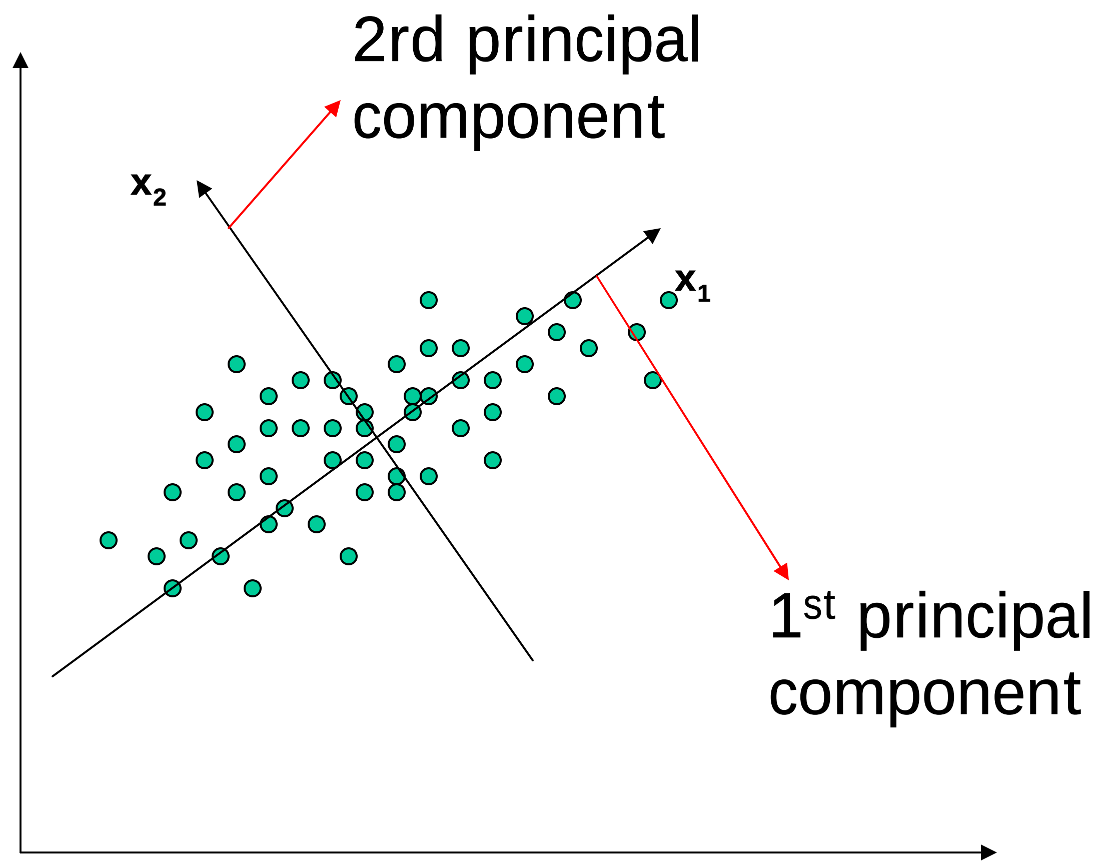
- PCA is used to determine the most representing features among data points.
  - It computes the p-dimensional subspace such that the projection of the data points onto the subspace has the largest variance among all p-dimensional subspaces.

Illustration of PCA



One projection PCA projection

Illustration of PCA



Eigenface for Face Recognition

- PCA has been used for face image representation/compression, face recognition and many others.
- Compare two faces by projecting the images into the subspace and measuring the EUCLIDEAN distance between them.

Mathematical Formulation

Find a transformation, W,

$$\mathbf{y}_k = W^T \mathbf{x}_k \qquad k = 1, 2, \ldots, N$$

m-dimensional | **Orthonormal** $W \in \mathbb{R}^{n \times m}$ | **n-dimensional**

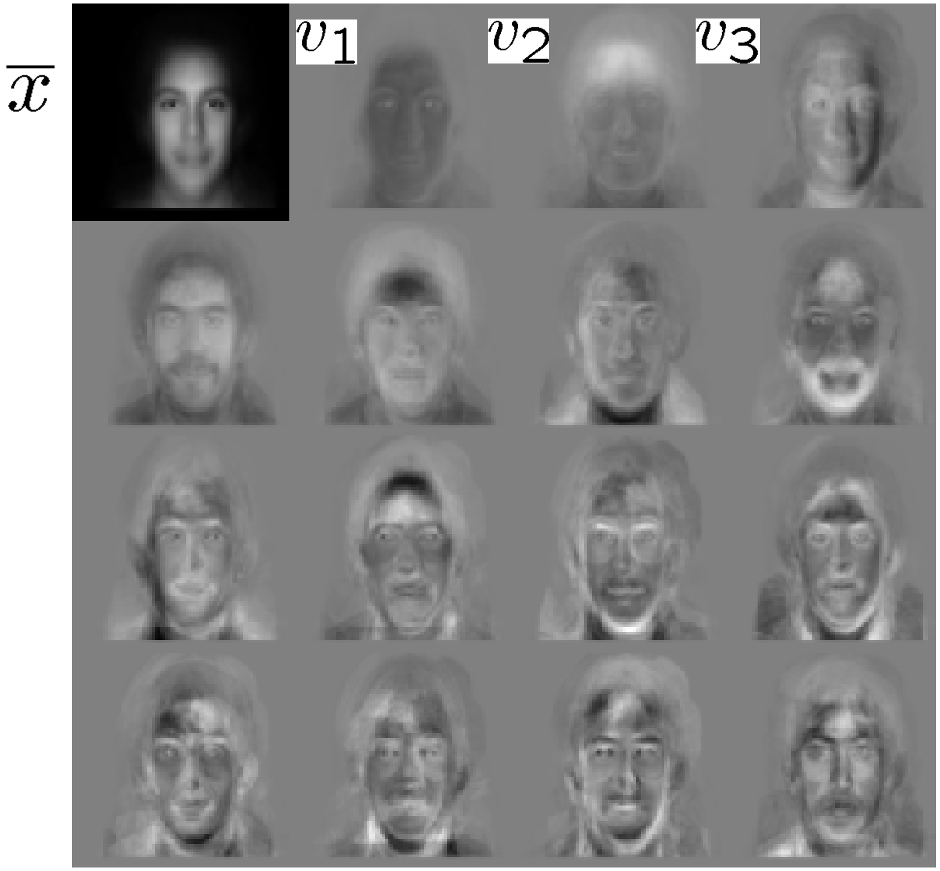
Total scatter matrix: $$S_T = \sum_{k=1}^{N} (\mathbf{x}_k - \mu)(\mathbf{x}_k - \mu)^T$$

$$W_{opt} = \arg\max_W \left| W^T S_T W \right| = \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \ldots & \mathbf{w}_m \end{bmatrix}$$

W_{opt} corresponds to m eigen-vectors of S_T

Eigenfaces

- PCA extracts the eigenvectors of **A**
  - Gives a set of vectors $\mathbf{v_1}$, $\mathbf{v_2}$, $\mathbf{v_3}$, ...
  - Each one of these vectors is a direction in face space
    - what do these look like?

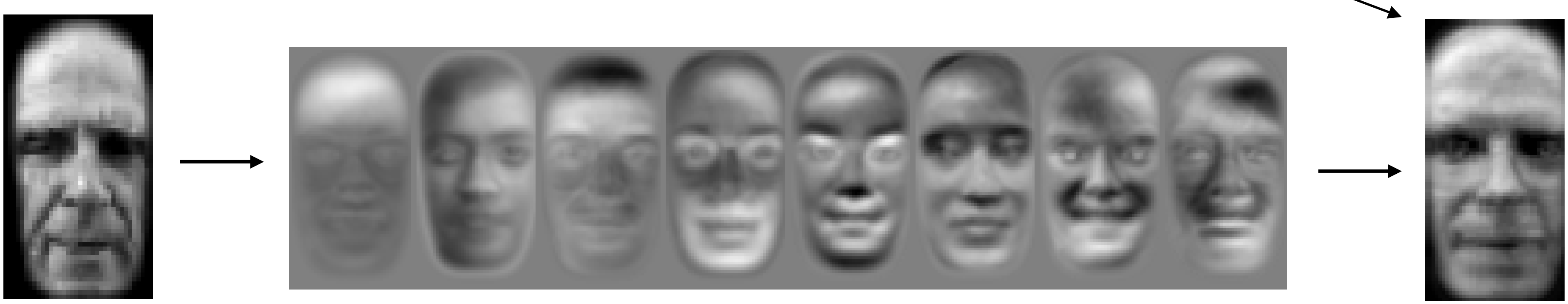


Projecting onto the Eigenfaces

- The eigenfaces $\mathbf{v_1}$, ..., $\mathbf{v_K}$ span the space of faces
  - A face is converted to eigenface coordinates by

$$\mathbf{x} \rightarrow (\underbrace{(\mathbf{x}-\overline{\mathbf{x}}) \cdot \mathbf{v_1}}_{a_1},\ \underbrace{(\mathbf{x}-\overline{\mathbf{x}}) \cdot \mathbf{v_2}}_{a_2}, \ldots,\ \underbrace{(\mathbf{x}-\overline{\mathbf{x}}) \cdot \mathbf{v_K}}_{a_K})$$

$$\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K}$$



$\mathbf{x}$ $a_1\mathbf{v_1}$ $a_2\mathbf{v_2}$ $a_3\mathbf{v_3}$ $a_4\mathbf{v_4}$ $a_5\mathbf{v_5}$ $a_6\mathbf{v_6}$ $a_7\mathbf{v_7}$ $a_8\mathbf{v_8}$

Algorithm

Training

1. Align training images $x_1, x_2, \ldots, x_N$

Note that each image is formulated into a long vector!

2. Compute average face $u = 1/N \sum x_i$

3. Compute the difference image $\varphi_i = x_i - u$

Algorithm

4. Compute the covariance matrix (total scatter matrix)

$$S_T = 1/N \sum \varphi_i \varphi_i^T = BB^T, B=[\varphi_1, \varphi_2 \ldots \varphi_N]$$

5. Compute the eigenvectors of the covariance matrix , W

Testing

1. Projection in Eigenface

   Projection $\omega_i = W(X - u)$, $W = \{eigenfaces\}$

2. Compare projections

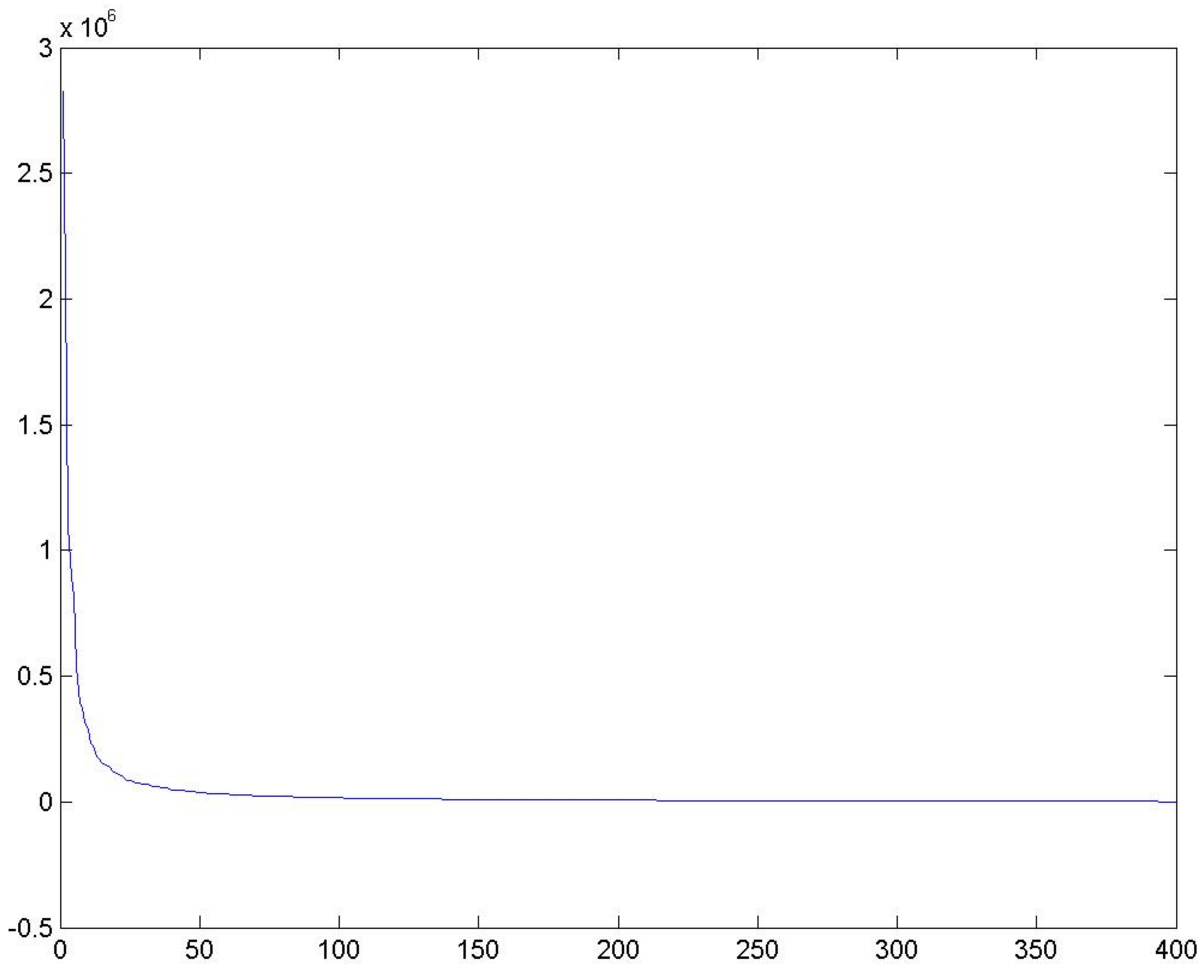
Illustration of Eigenfaces

- The visualization of eigenvectors:

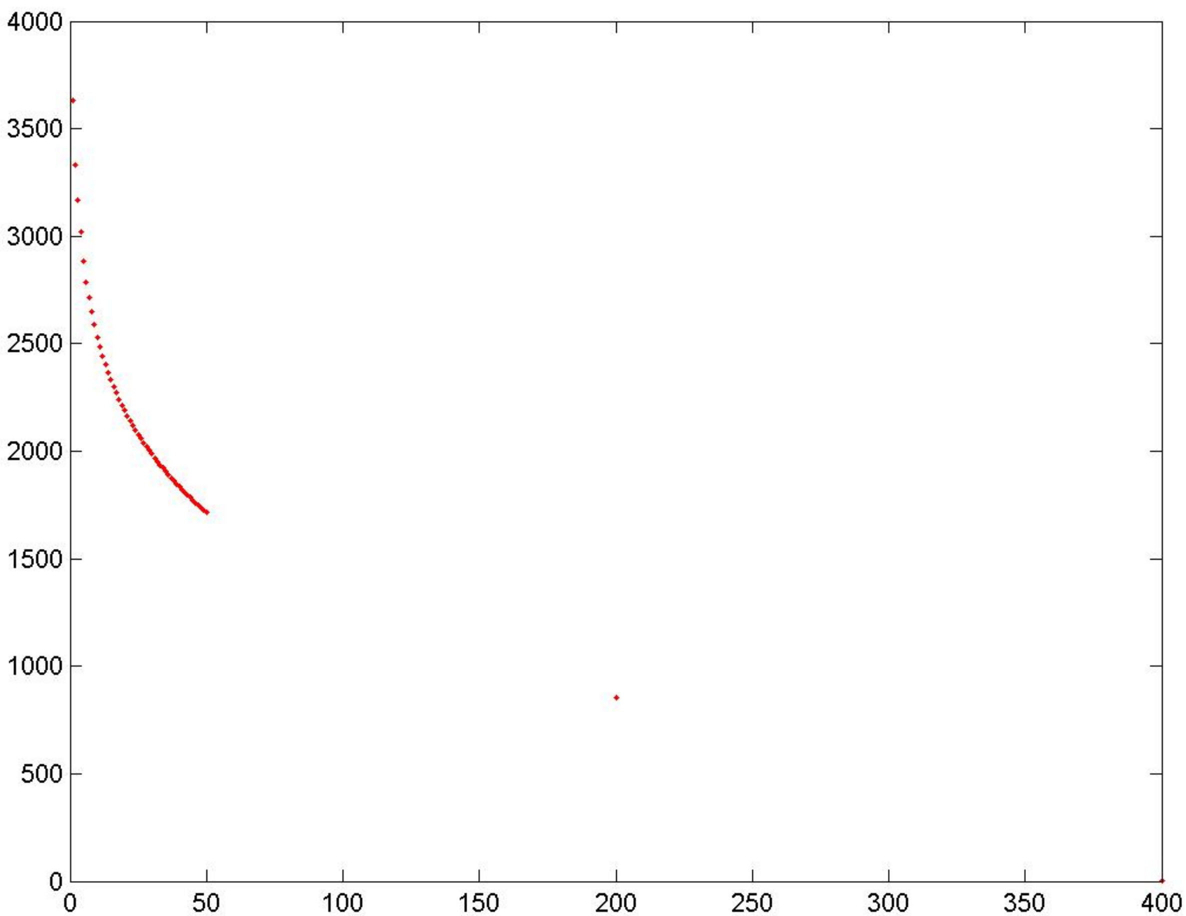
These are the first 4 eigenvectors from a training set of 400 images (ORL Face Database). They look like faces, hence called Eigenface.

Eigenfaces look somewhat like generic faces.

Eigenvalues



Reconstruction and Errors



imensionality.

ıd hence less

Summary for PCA and Eigenface

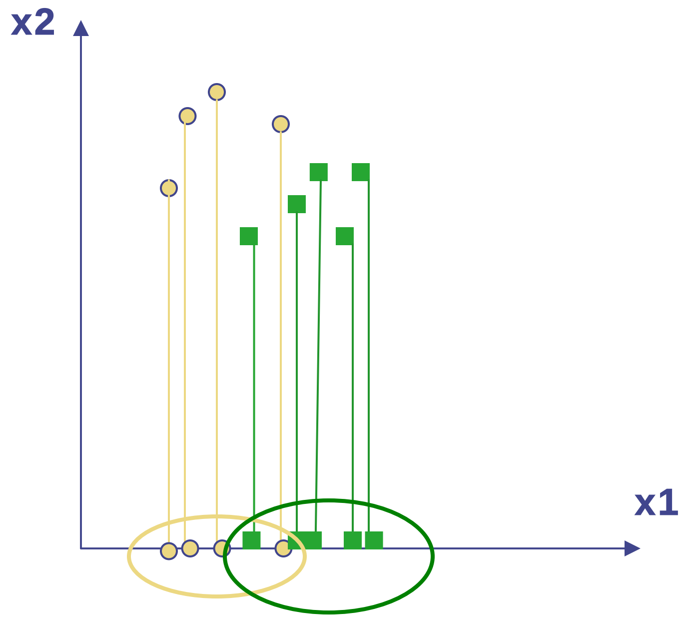
- Non-iterative, globally optimal solution
- PCA projection is **optimal for reconstruction** from a low dimensional basis, but **may NOT be optimal for discrimination…**

Linear Discriminant Analysis (LDA)

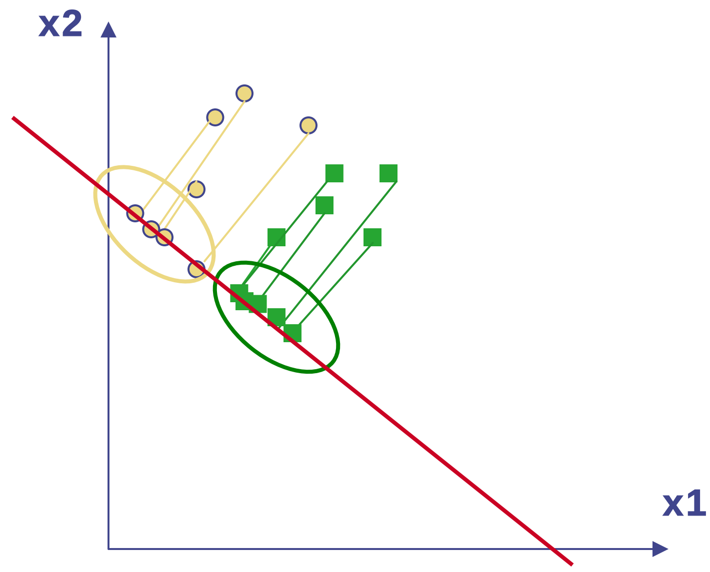
- Using Linear Discriminant Analysis (LDA) or Fisher's Linear Discriminant (FLD)
- Eigenfaces attempt to maximise the scatter of the training images in face space, while Fisherfaces attempt to maximise the **between class scatter**, while minimising the **within class scatter**.

Illustration of the Projection

- Using two classes as example:

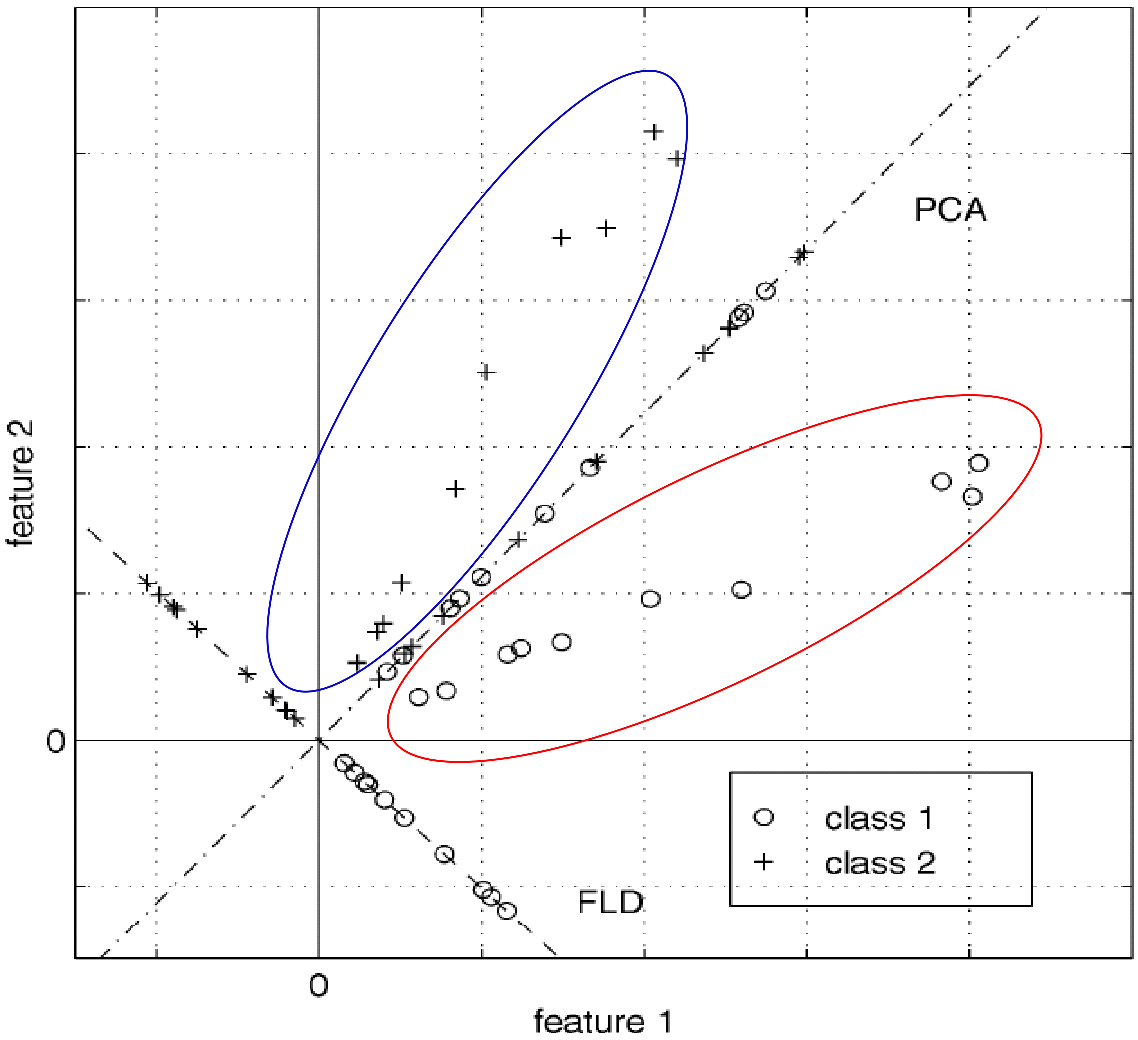


Poor Projection



Good Projection

Comparing with PCA



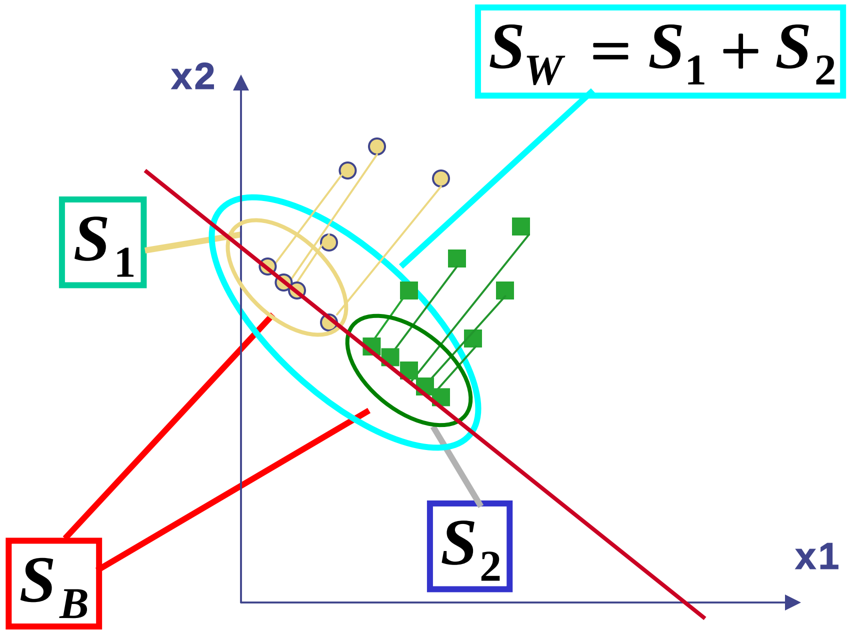
Variables

- N Sample images: $\{x_1, \cdots, x_N\}$
- c classes: $\{\chi_1, \cdots, \chi_c\}$
- Average of each class: $\mu_i = \frac{1}{N_i} \sum_{x_k \in \chi_i} x_k$
- Total average: $\mu = \frac{1}{N} \sum_{k=1}^{N} x_k$

Scatters

- Scatter of class i: $$\boxed{S_i = \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T}$$
- Within class scatter: $$\boxed{S_W = \sum_{i=1}^{c} S_i}$$
- Between class scatter: $$\boxed{S_B = \sum_{i=1}^{c} |\chi_i| (\mu_i - \mu)(\mu_i - \mu)^T}$$
- Total scatter: $$\boxed{S_T = S_W + S_B}$$

Illustration



Mathematical Formulation (1)

- After projection: $\boxed{y_k = W^T x_k}$

- Between class scatter (of y's): $\boxed{\tilde{S}_B = W^T S_B W}$
- Within class scatter (of y's): $\boxed{\tilde{S}_W = W^T S_W W}$

Mathematical Formulation (2)

- The desired projection:

$$\boxed{W_{opt} = \arg\max_{W} \frac{\left|\tilde{S}_B\right|}{\left|\tilde{S}_W\right|} = \arg\max_{W} \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|}}$$

- How is it found ? → Generalized Eigenvectors

$$\boxed{S_B w_i = \lambda_i S_W w_i \qquad i = 1, \ldots, m}$$

- Data dimension is much larger than the number of samples $n >> N$
- The matrix S_W is singular: $\boxed{Rank(S_W) \le N - c}$

Fisherface (PCA+FLD)

- Project with PCA to $N-c$ space

$$\boxed{z_k = W_{pca}{}^T x_k}$$

$$\boxed{W_{pca} = \arg\max_{W} \left| W^T S_T W \right|}$$

- Project with FLD to $c-1$ space

$$\boxed{y_k = W_{fld}{}^T z_k}$$

$$\boxed{W_{fld} = \arg\max_{W} \frac{\left| W^T W_{pca}^T S_B W_{pca} W \right|}{\left| W^T W_{pca}^T S_W W_{pca} W \right|}}$$

Illustration of FisherFace

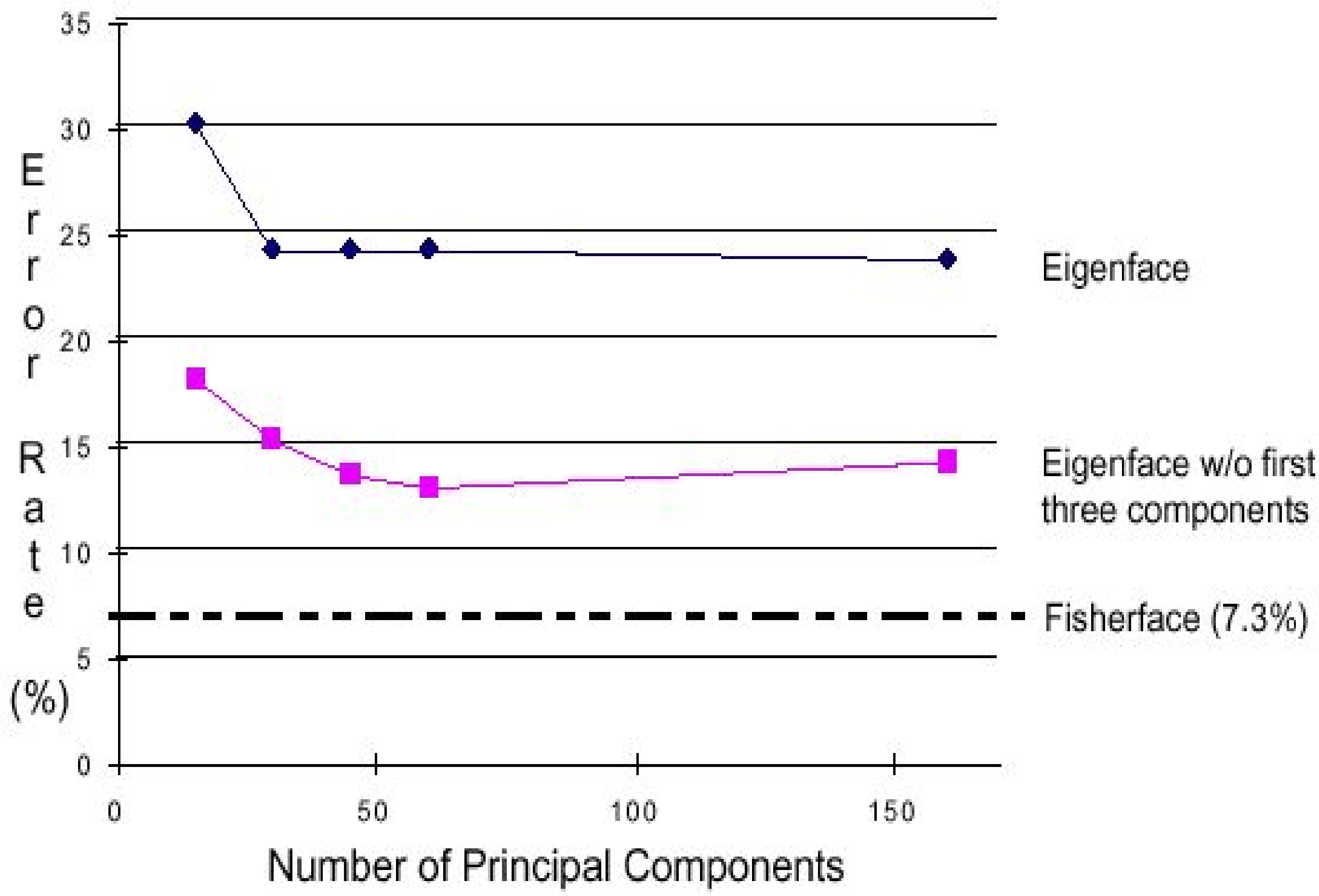
- Fisherface

Results: Eigenface vs. Fisherface (1)

- Input: 160 images of 16 people
- Train: 159 images
- Test: 1 image

- Variation in Facial Expression, Eyewear, and Lighting

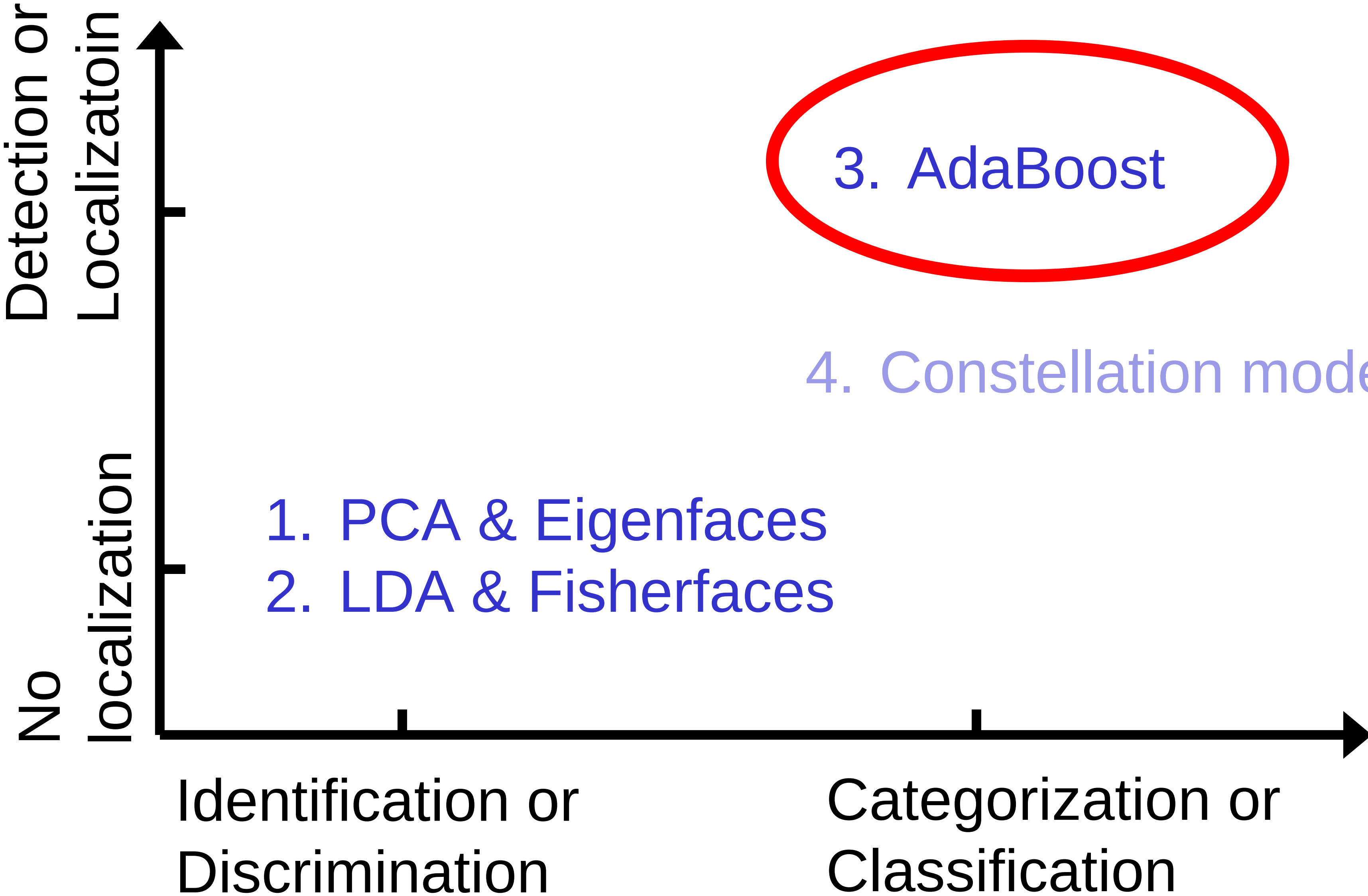
Eigenface vs. Fisherface (2)



discussion

- Removing the first three principal components results in better performance under variable lighting conditions
- The Firsherface methods had error rates lower than the Eigenface method for the small datasets tested.

Today's agenda



Robust Face Detection Using AdaBoost

- Brief intro on (Ada)Boosting
- Viola & Jones, 2001
  - Weak detectors: Haar wavelets
  - Integral image
  - Cascade
  - Exp. & Res.

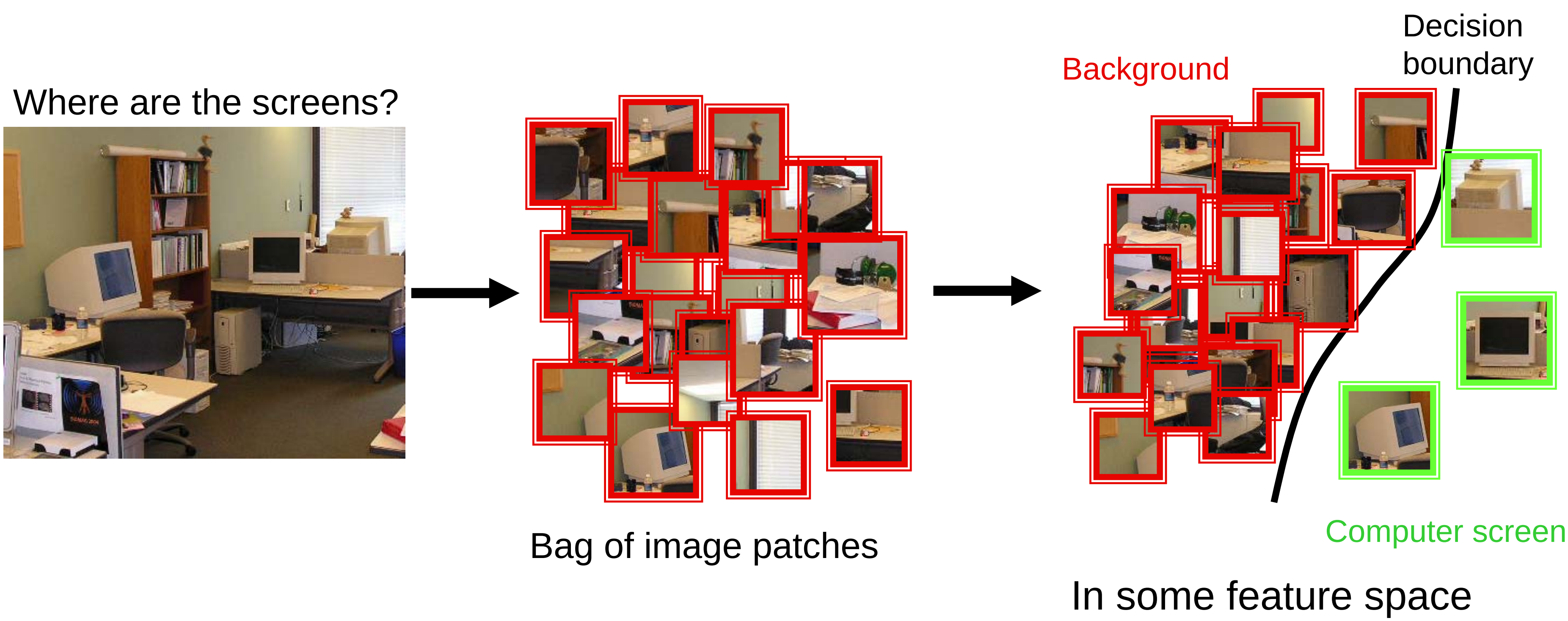
Reference:
P. Viola and M. Jones (2001) Robust Real-time Object Detection, IJCV.

Discriminative methods

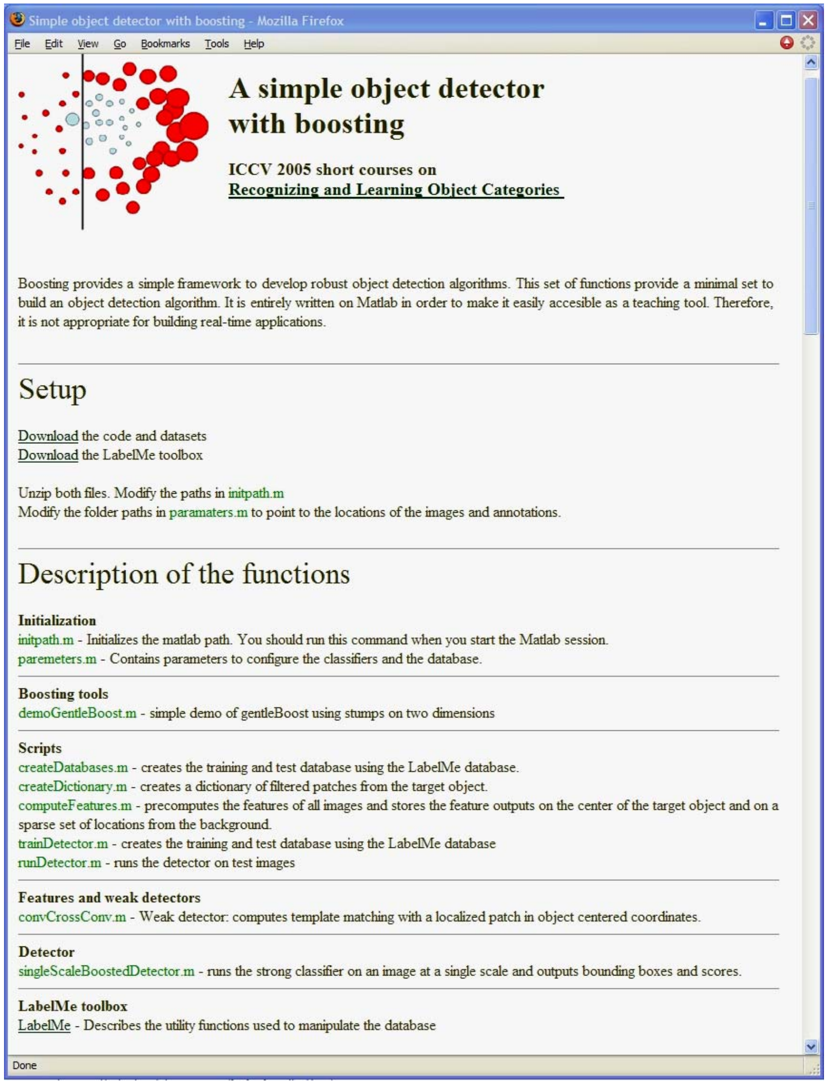
Object detection and recognition is formulated as a classification problem.

The image is partitioned into a set of overlapping windows

… and a decision is taken at each window about if it contains a target object or not.



A simple object detector with Boosting



Simple object detector with boosting - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

A simple object detector with boosting

ICCV 2005 short courses on
Recognizing and Learning Object Categories

Boosting provides a simple framework to develop robust object detection algorithms. This set of functions provide a minimal set to build an object detection algorithm. It is entirely written on Matlab in order to make it easily accesible as a teaching tool. Therefore, it is not appropriate for building real-time applications.

Setup

Download the code and datasets
Download the LabelMe toolbox

Unzip both files. Modify the paths in initpath.m
Modify the folder paths in paramaters.m to point to the locations of the images and annotations.

Description of the functions

Initialization
initpath.m - Initializes the matlab path. You should run this command when you start the Matlab session.
paremeters.m - Contains parameters to configure the classifiers and the database.

Boosting tools
demoGentleBoost.m - simple demo of gentleBoost using stumps on two dimensions

Scripts
createDatabases.m - creates the training and test database using the LabelMe database.
createDictionary.m - creates a dictionary of filtered patches from the target object.
computeFeatures.m - precomputes the features of all images and stores the feature outputs on the center of the target object and on a sparse set of locations from the background.
trainDetector.m - creates the training and test database using the LabelMe database
runDetector.m - runs the detector on test images

Features and weak detectors
convCrossConv.m - Weak detector: computes template matching with a localized patch in object centered coordinates.

Detector
singleScaleBoostedDetector.m - runs the strong classifier on an image at a single scale and outputs bounding boxes and scores.

LabelMe toolbox
LabelMe - Describes the utility functions used to manipulate the database

Done

http://people.csail.mit.edu/torralba/iccv2005/

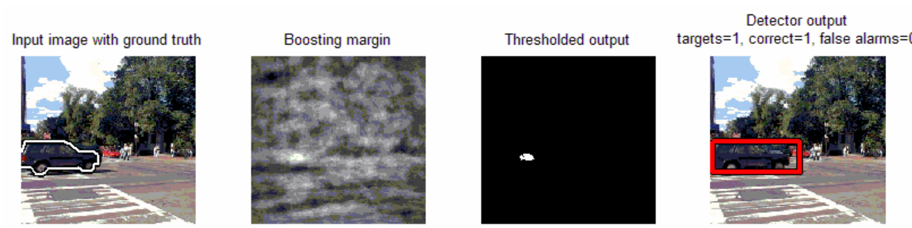
Download

- Toolbox for manipulating dataset
- Code and dataset

Matlab code

- Gentle boosting
- Object detector using a part based model

Dataset with cars and computer monitors



Why boosting?

- A simple algorithm for learning robust classifiers
  - Freund & Shapire, 1995
  - Friedman, Hastie, Tibshhirani, 1998
- Provides efficient algorithm for sparse visual feature selection
  - *Tieu & Viola, 2000*
  - *Viola & Jones, 2003*
- Easy to implement, not requires external optimization tools.

Boosting

- Defines a classifier using an additive model:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \dots$$

Strong classifier

Weak classifier

Weight

Features vector

Boosting

- Defines a classifier using an additive model:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \ldots$$

Strong classifier

Features vector

Weight

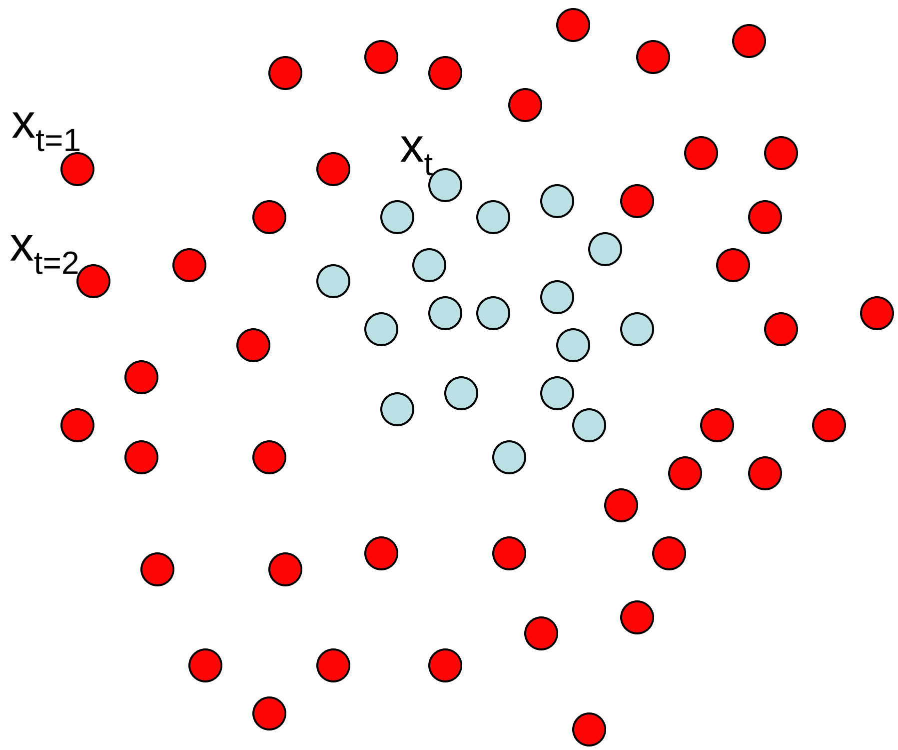
Weak classifier

- We need to define a family of weak classifiers

$f_k(x)$ from a family of weak classifiers

Boosting

- It is a sequential procedure:



Each data point has a class label:

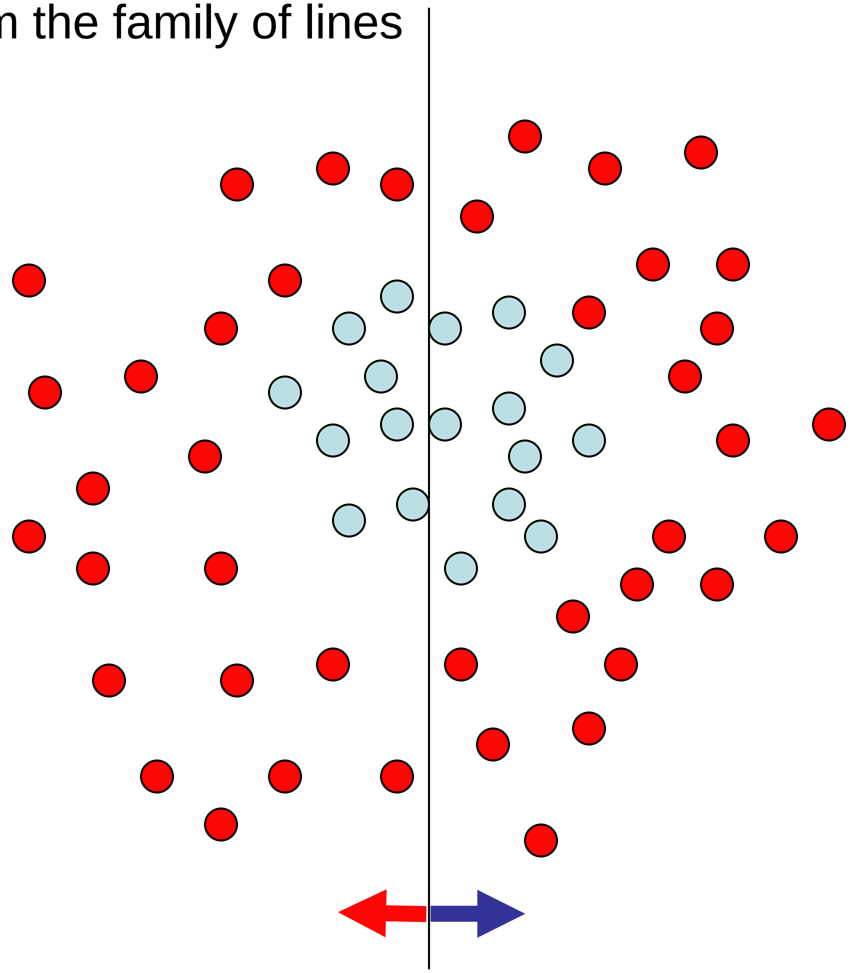
$$y_t = \begin{cases} +1 \; (\;) \\ -1 \; (\;) \end{cases}$$

and a weight:

$w_t = 1$

Toy example

Weak learners from the family of lines



Each data point has

a class label:

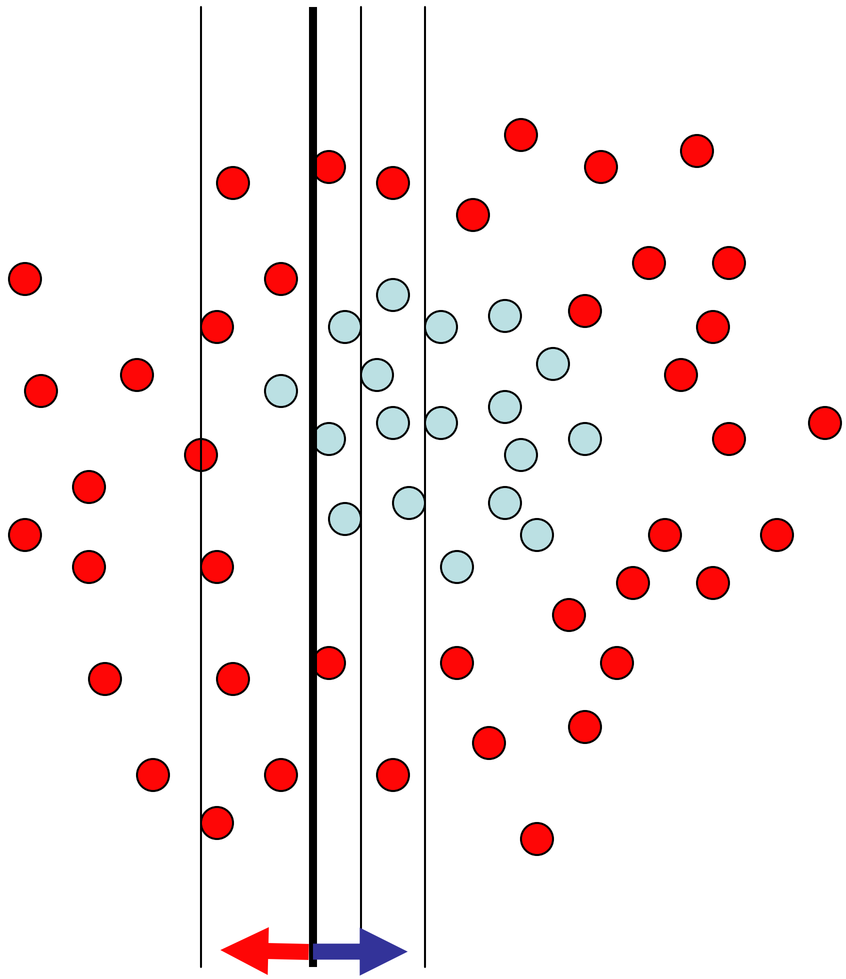
$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases}$$

and a weight:

$w_t = 1$

h => p(error) = 0.5 it is at chance

Toy example



Each data point has

a class label:

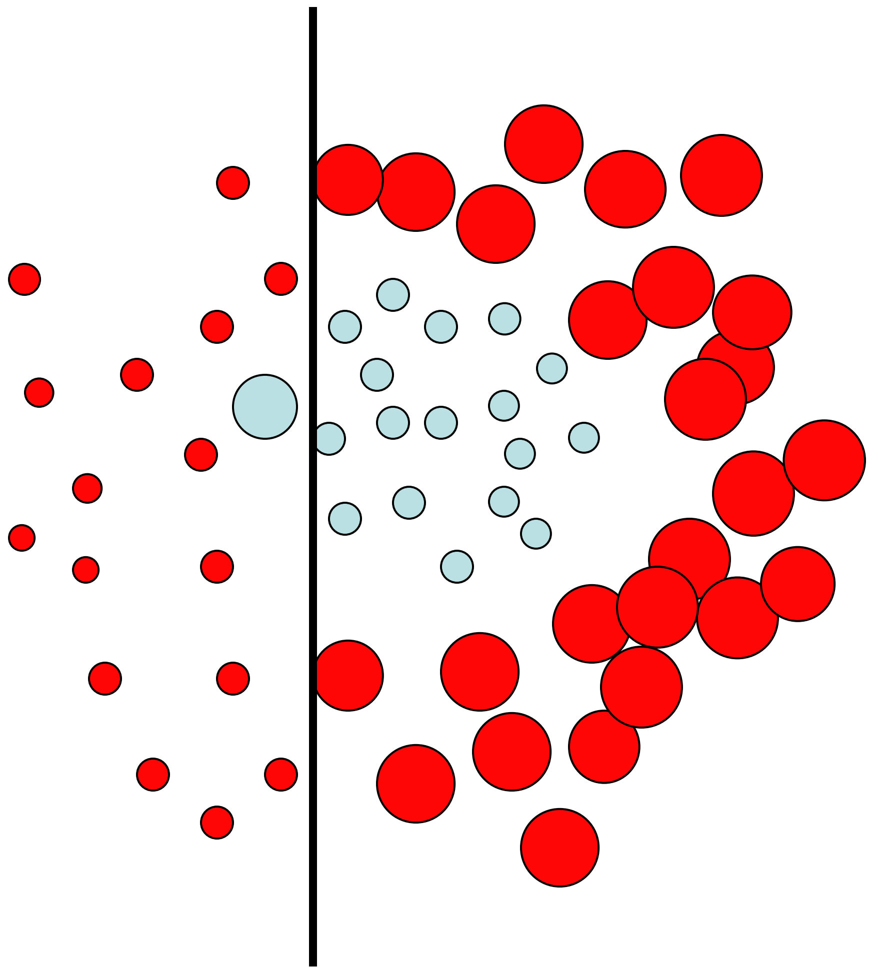
$$y_t = \begin{cases} +1 \text{ (red)} \\ -1 \text{ (light blue)} \end{cases}$$

and a weight:

$w_t = 1$

This is a ‘**weak classifier**’: It performs slightly better than chance.

Toy example



Each data point has

a class label:

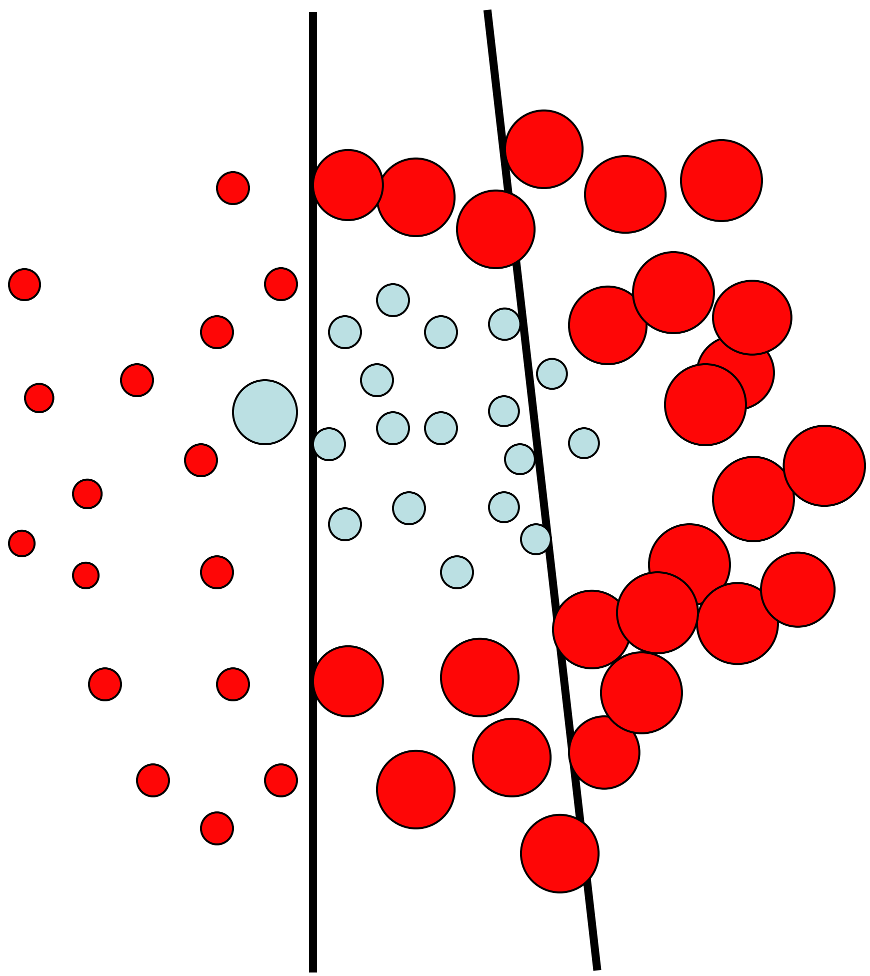
$$y_t = \begin{cases} +1 & (\;) \\ -1 & (\;) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has

a class label:

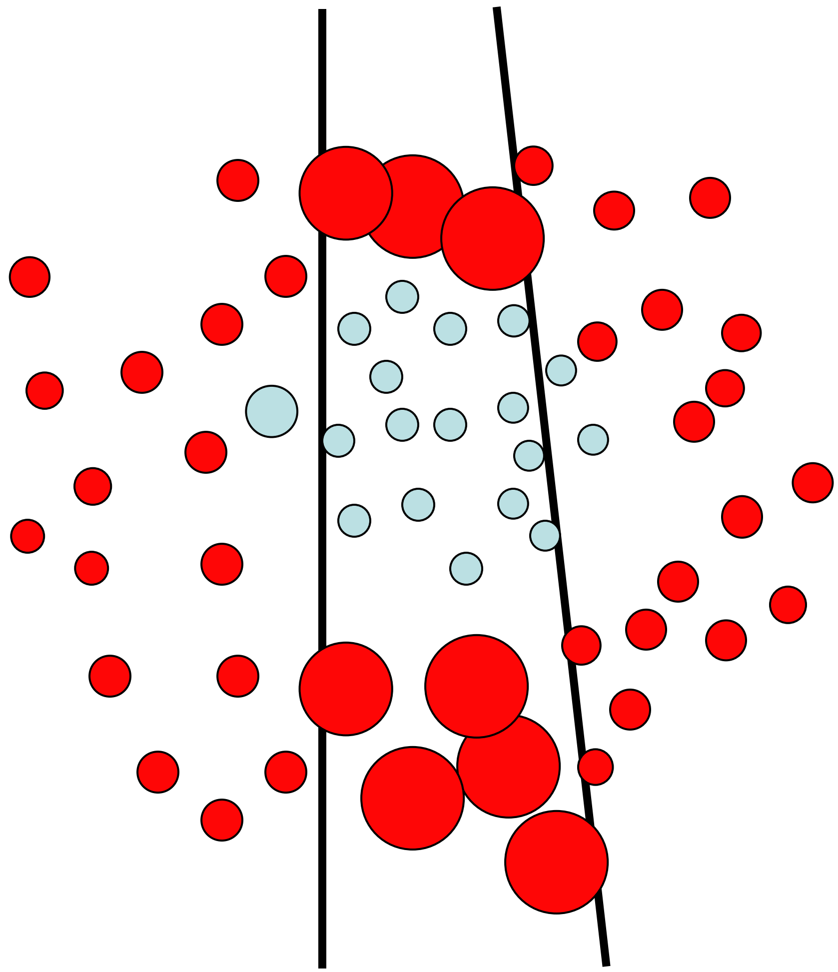
$$y_t = \begin{cases} +1 \ (\bullet) \\ -1 \ (\circ) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has

a class label:

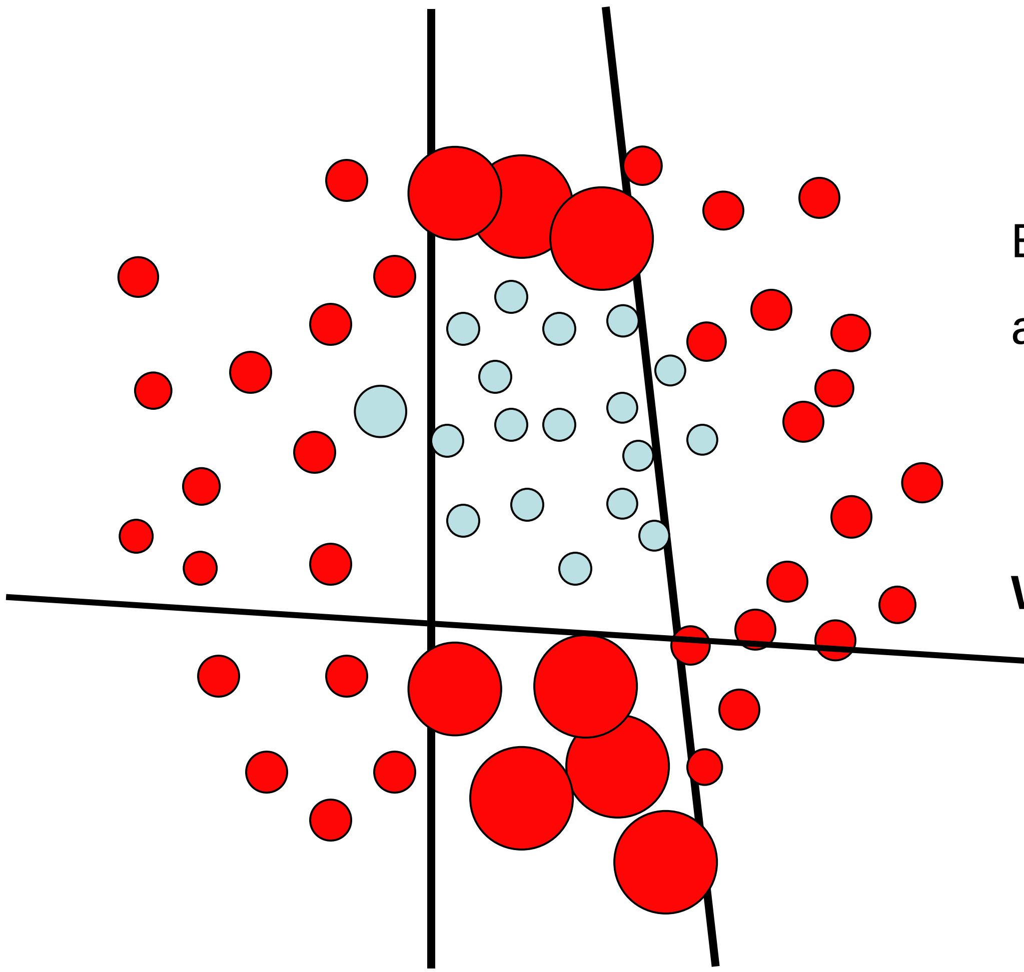
$$y_t = \begin{cases} +1 \ (\ ) \\ -1 \ (\ ) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has

a class label:

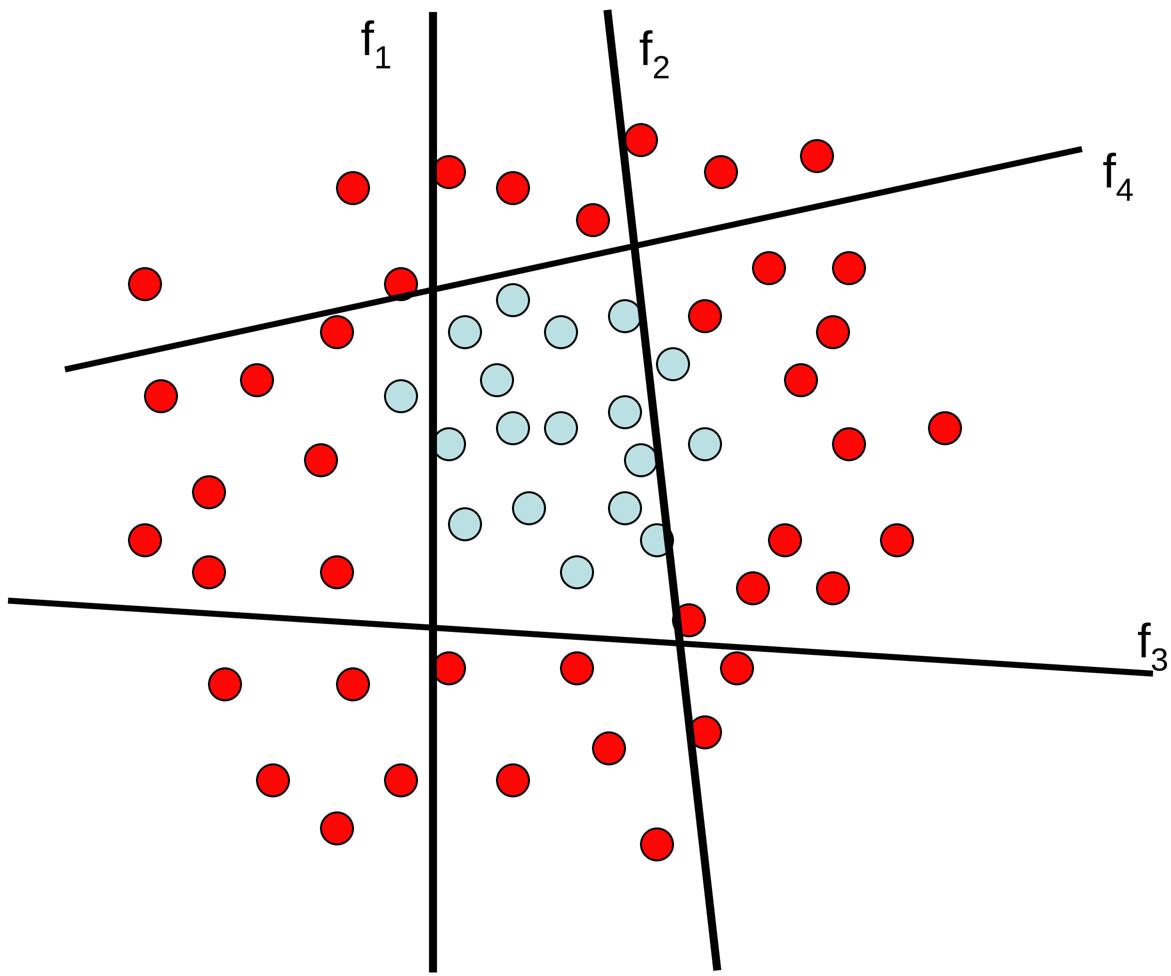
$$y_t = \begin{cases} +1 \ (\bullet) \\ -1 \ (\circ) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



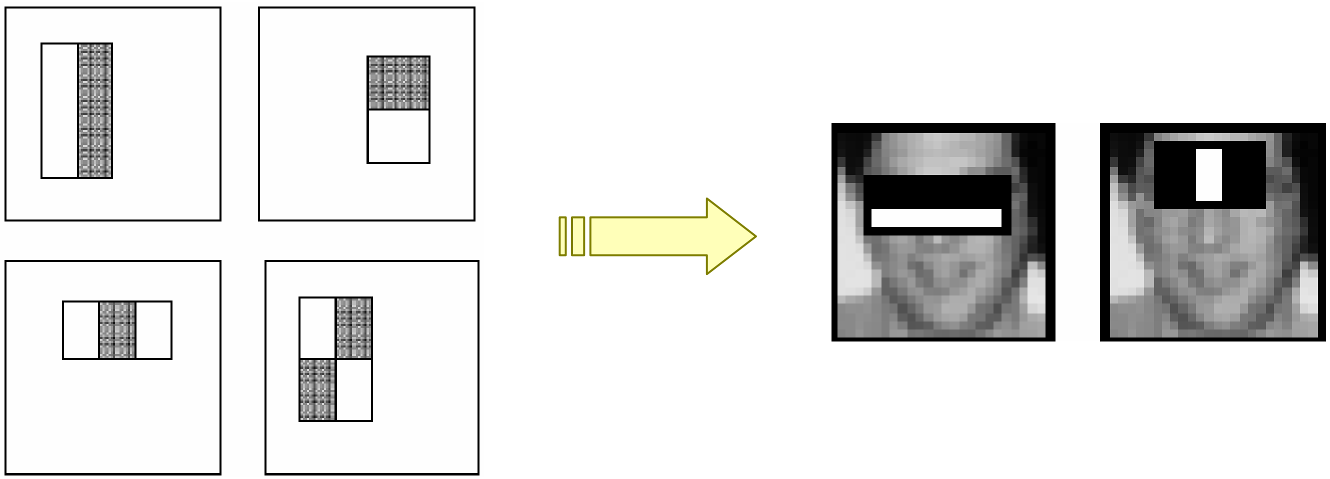
The strong (non- linear) classifier is built as the combination of all the weak (linear) classifiers.

Real-time Face Detection

- Integral Image
  - New image representation to compute the features very quickly
- AdaBoost
  - Selecting a small number of important feature
- Cascade
  - A method for combining classifiers
  - Focussing attention on promising regions of the image
- Implemented on 700MHz Intel Pentium Ⅲ, face detection proceeds at 15f/s.
  - Working only with a single grey scale image

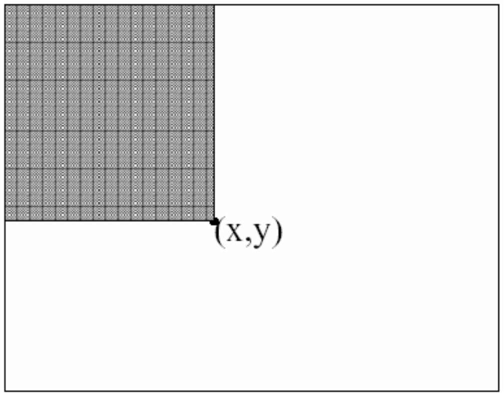
Features

- Three kinds of rectangle features

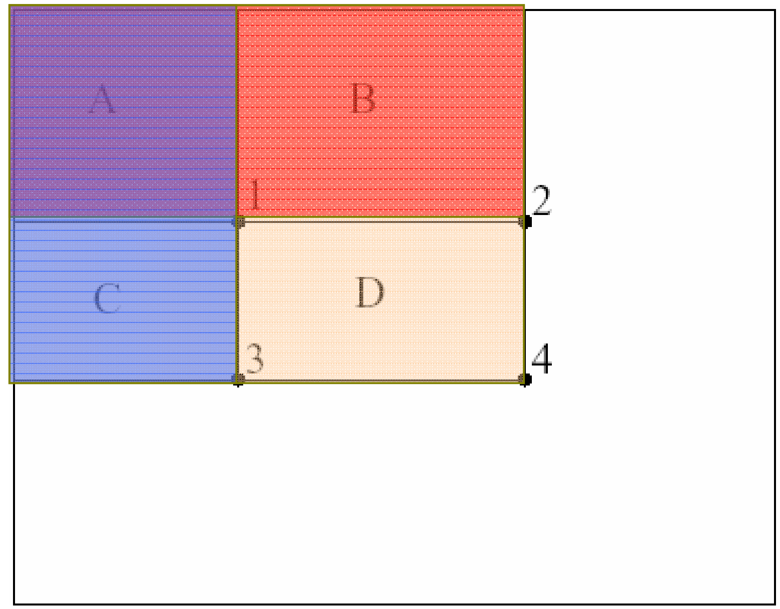


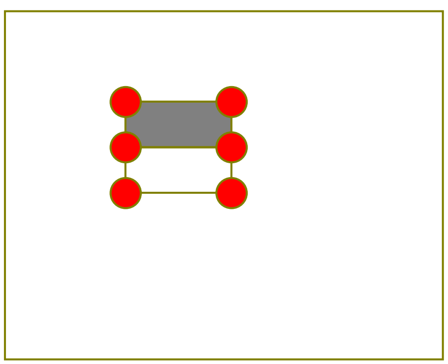
- The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the gray rectangles

Integral Image



$$ii(x, y) = \sum_{x' \le x, y' \le y} i(x', y')$$

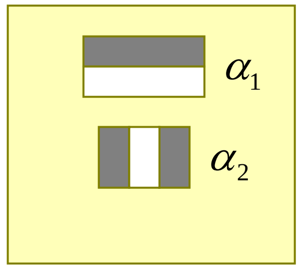




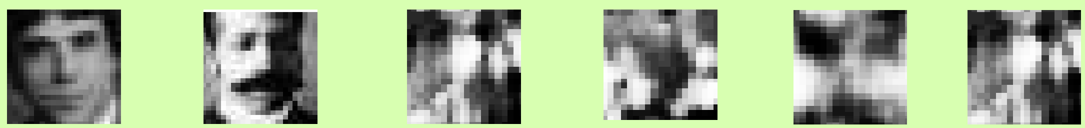
The sum within D=4-(2+3)+1

Learning Classification Function (1)

- Selecting a small number of important features



①



$(x_1, 1)$ $(x_2, 1)$ $(x_3, 0)$ $(x_4, 0)$ $(x_5, 0)$ $(x_6, 0)$ $\cdots\cdots (x_n, y_n)$

② Initialize weights $w_{1,i} = \frac{1}{2l}, \frac{1}{2m}$ for $y_i = 1, 0$ respectively

# of face

# of nonface

Learning Classification Function (2)

③ For t=1,....,T

a. Normalize the weights $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$

$h_j(x) = \begin{cases} 1 & \text{if } f_j(x) > \theta_j \\ 0 & \text{otherwise} \end{cases}$

b. For each feature, j $\varepsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$

c. Choose the classifier, h_t with the lowest error ε_t

d. Update the weights $w_{t+1,i} = w_{t,}$ 1 or 0 $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$

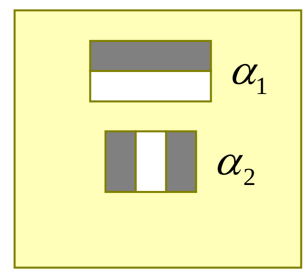
Learning Classification Function (3)

④ The final strong classifier is

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

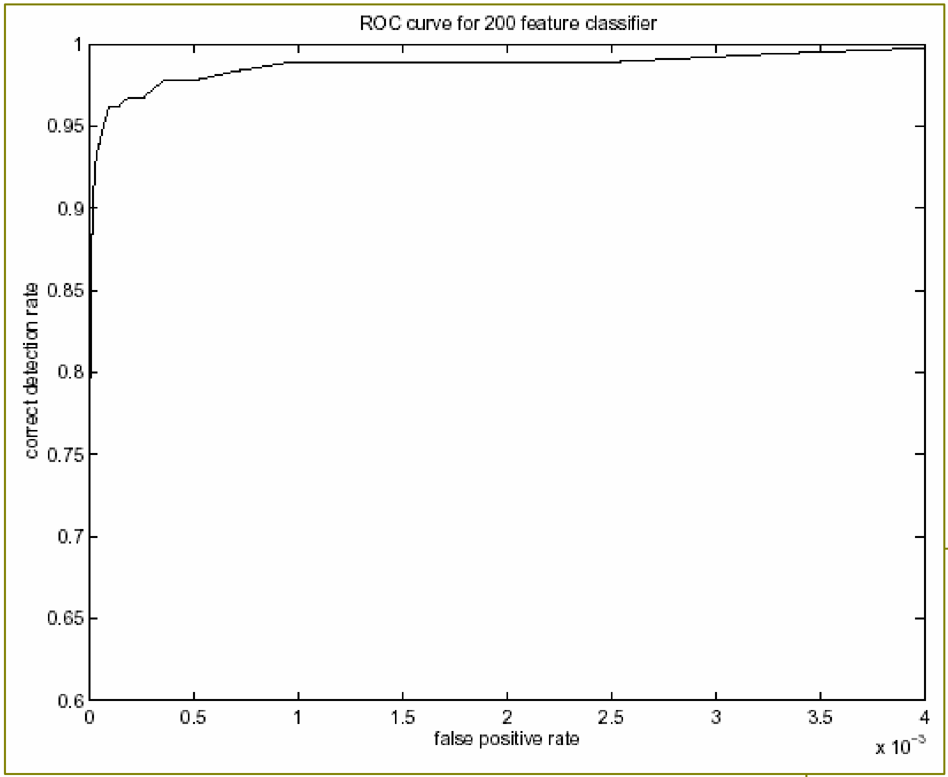
$$\alpha_t = \log \frac{1}{\beta_t}$$

☞The final hypothesis is a weighted linear combination of the T hypotheses where the weights are inversely proportional to the training errors

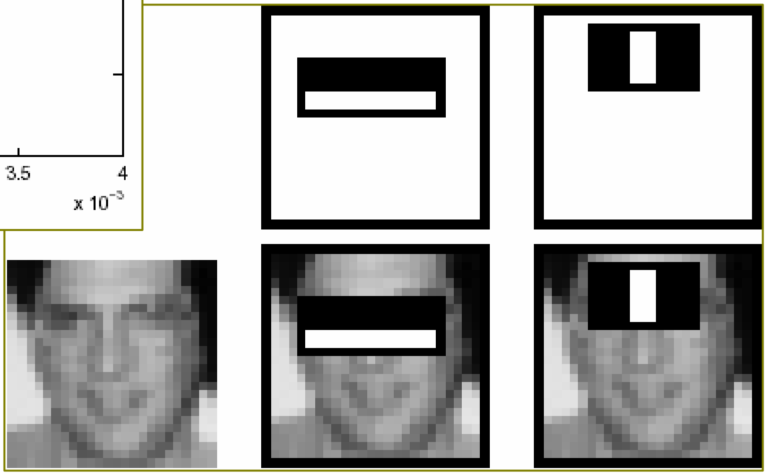


Learning Results

- 200 features
- Detection rate: 95% – false positive 1/14,804



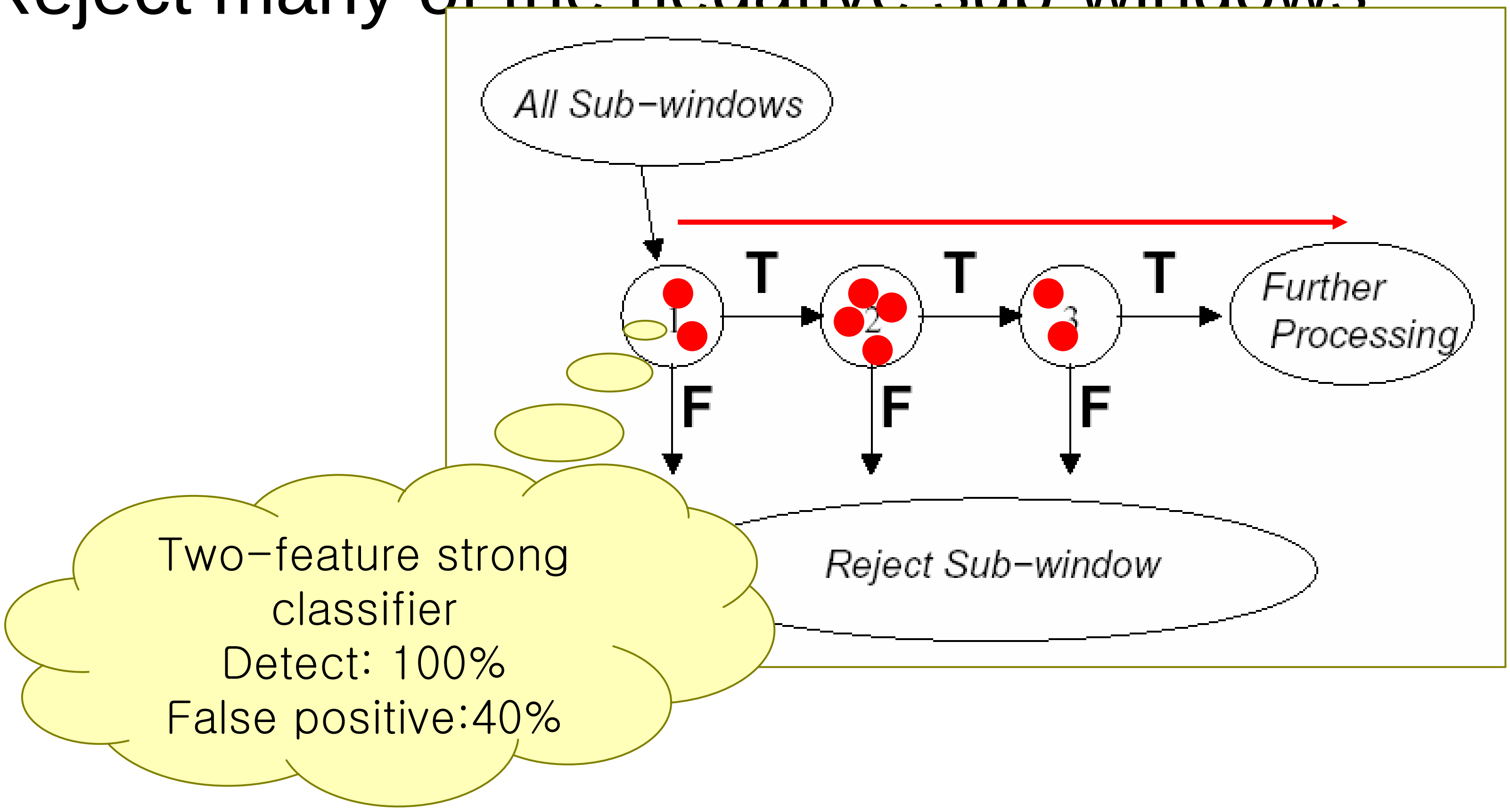
The first and second features selected by AdaBoost



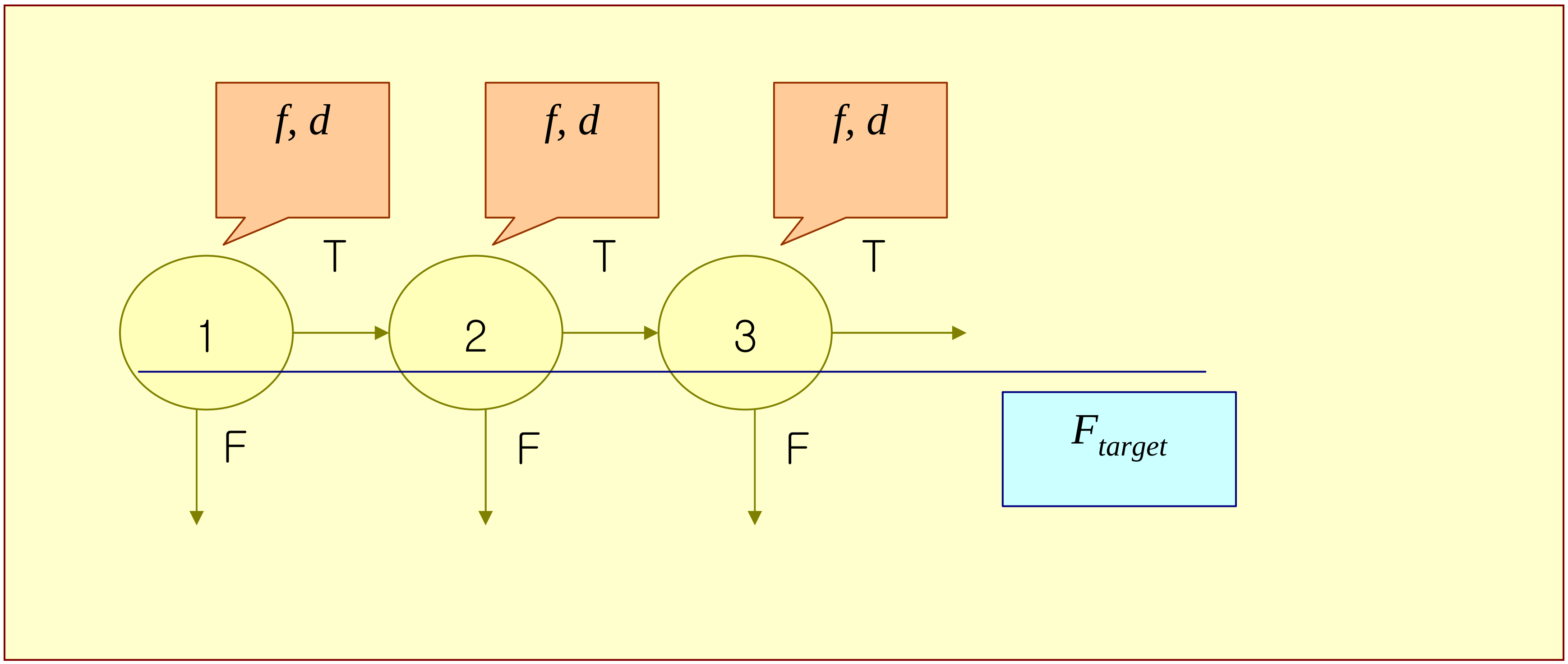
Requiring 0.7 seconds to scan 384x288 pixel image

The Attentional Cascade

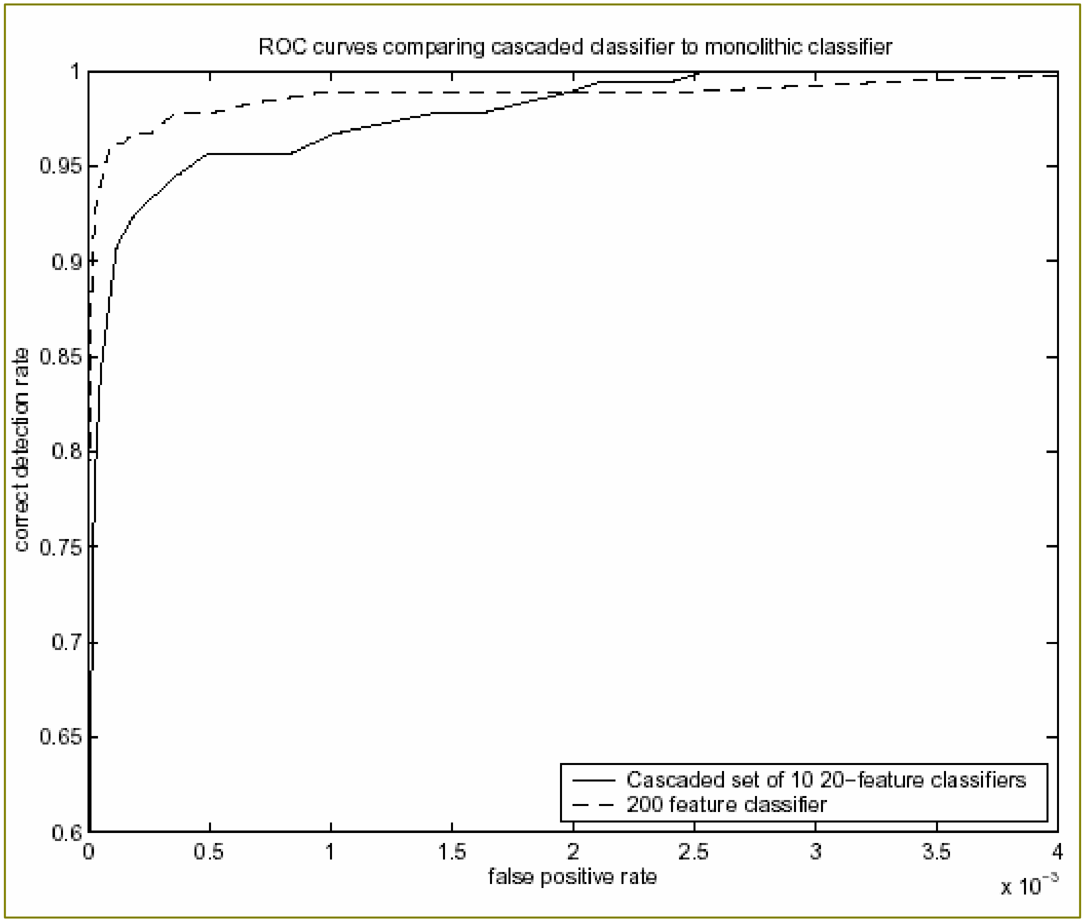
- Reject many of the negative sub-windows



A Cascaded Detector



Detector Cascade Discussion

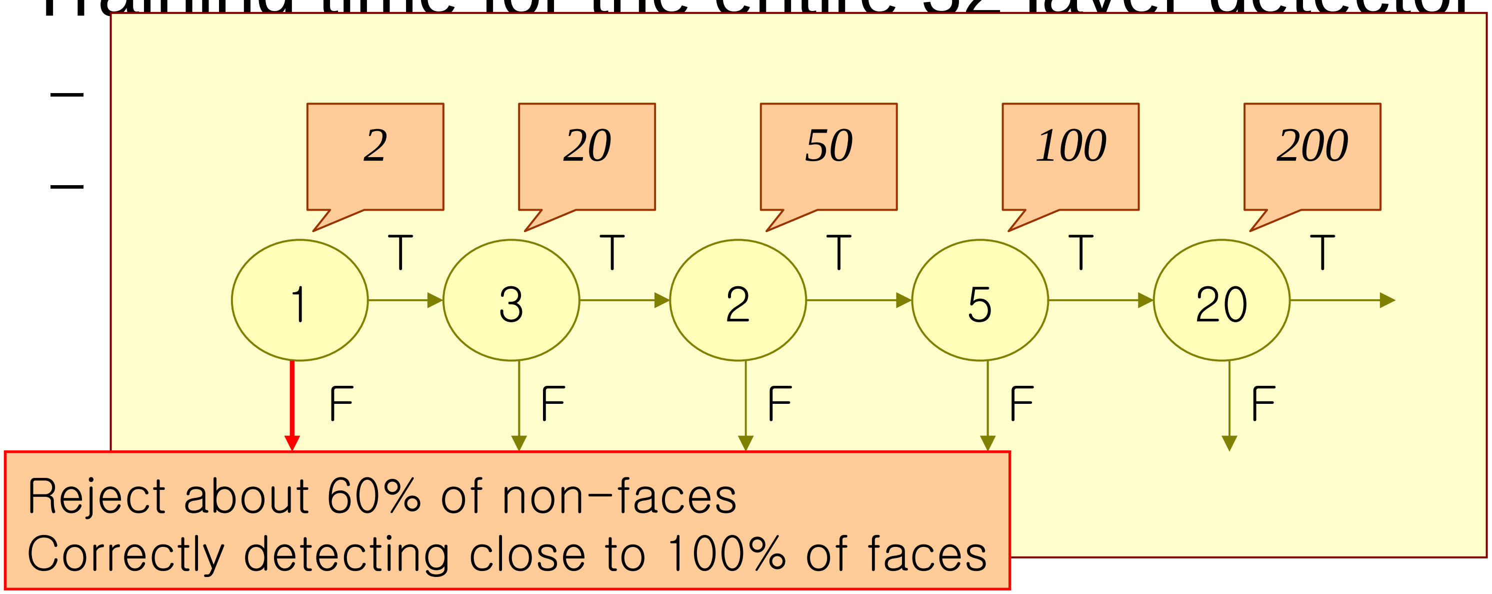


☞ The speed of the cascaded classifier is almost 10 times faster

Experimental Results (1)

- Training dataset
  - Face training set: 4916 h
  - Scaled and aligned to a b

- Structure of the detector cascade
  - 32layer, 4297 feature
- Training time for the entire 32 layer detector
  -
  -



Face Image Databases

- Databases for face recognition can be best utilized as training sets
  - Each image consists of an individual on a uniform and uncluttered background
- Test Sets for face detection
  - MIT, CMU (frontal, profile), Kodak

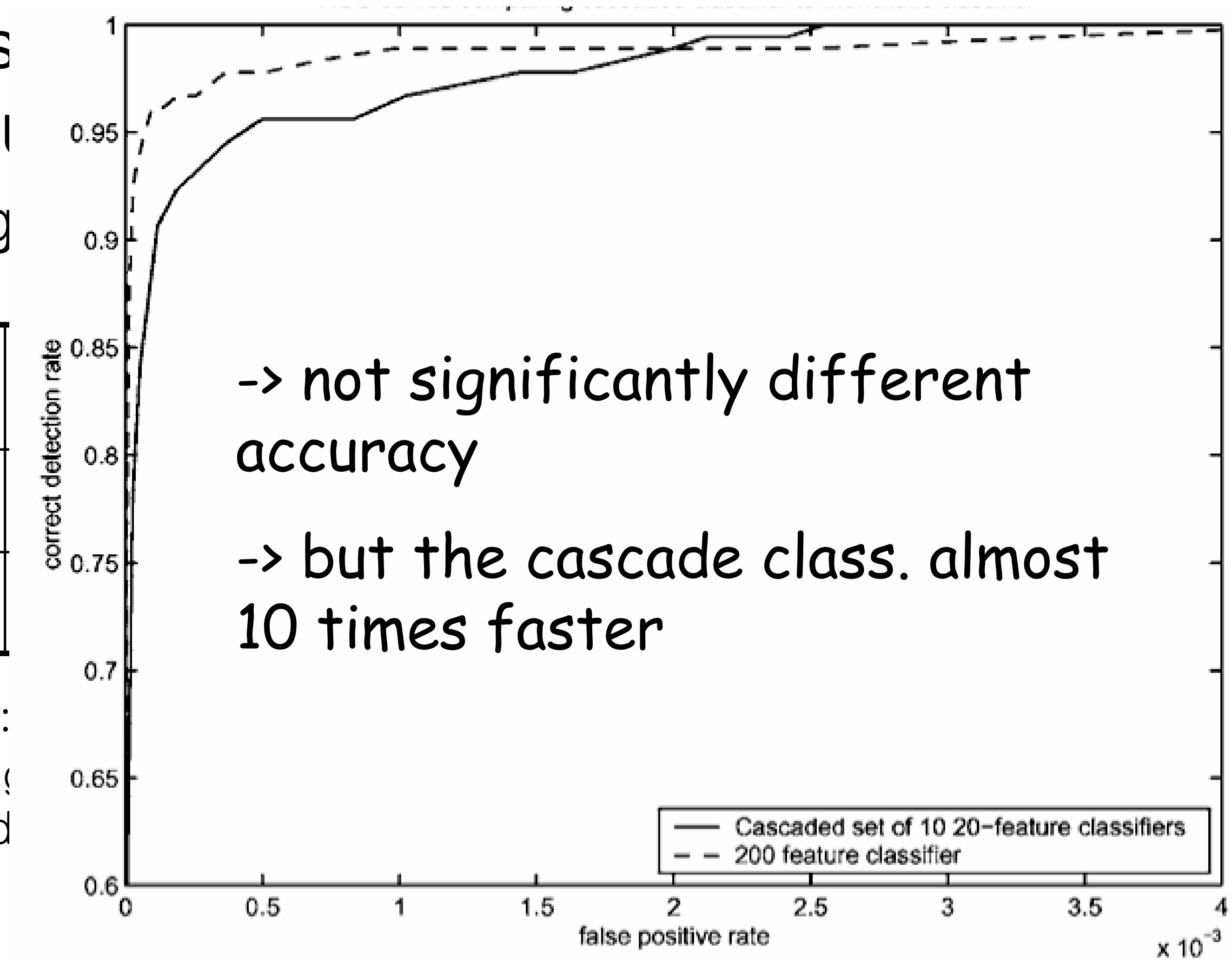
Training dataset: 4916 images

Experimental Results

- Test datas
  – MIT+CM
  – 130 imag

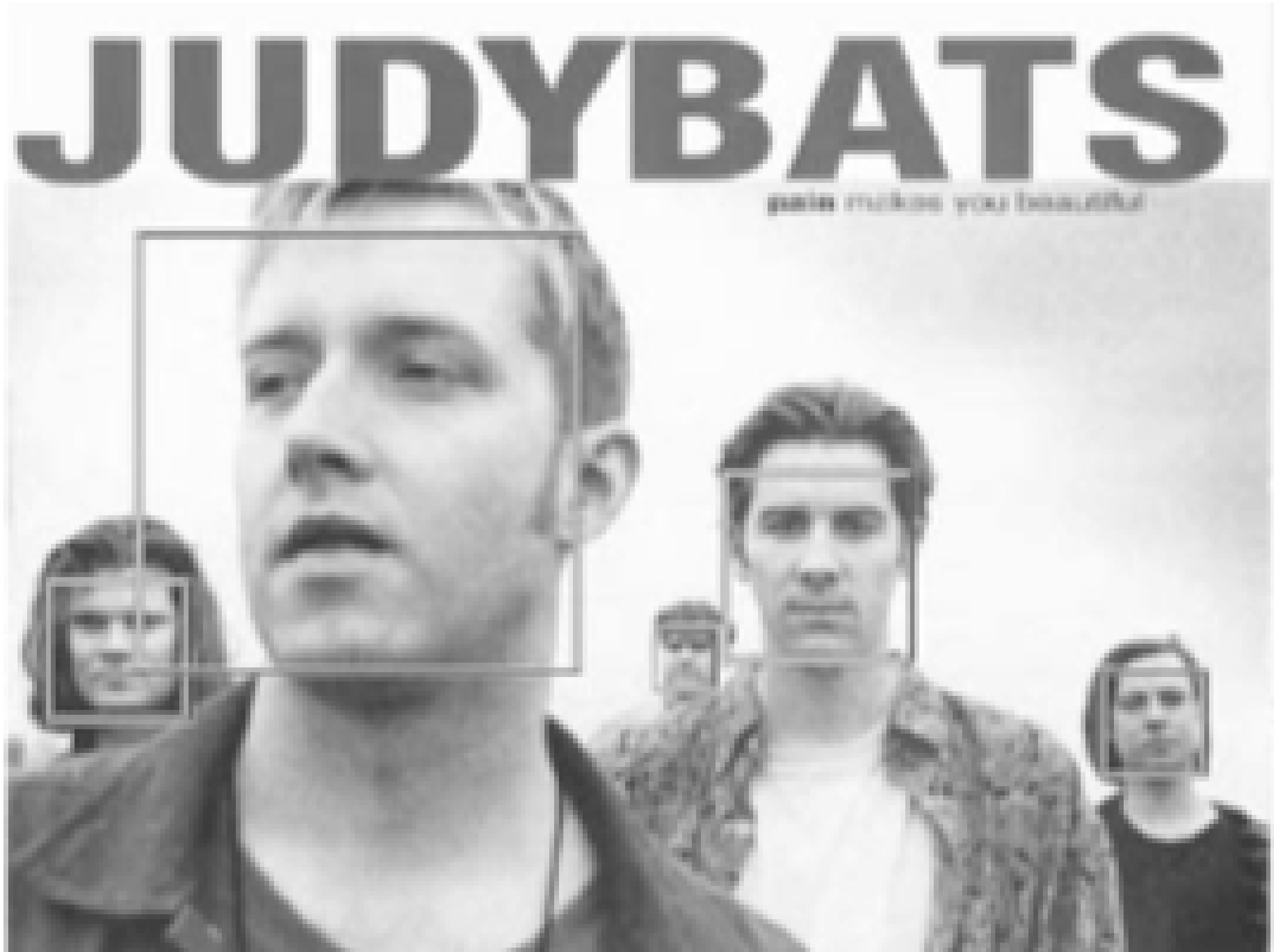
False detection
AdaBoost
Neural-net

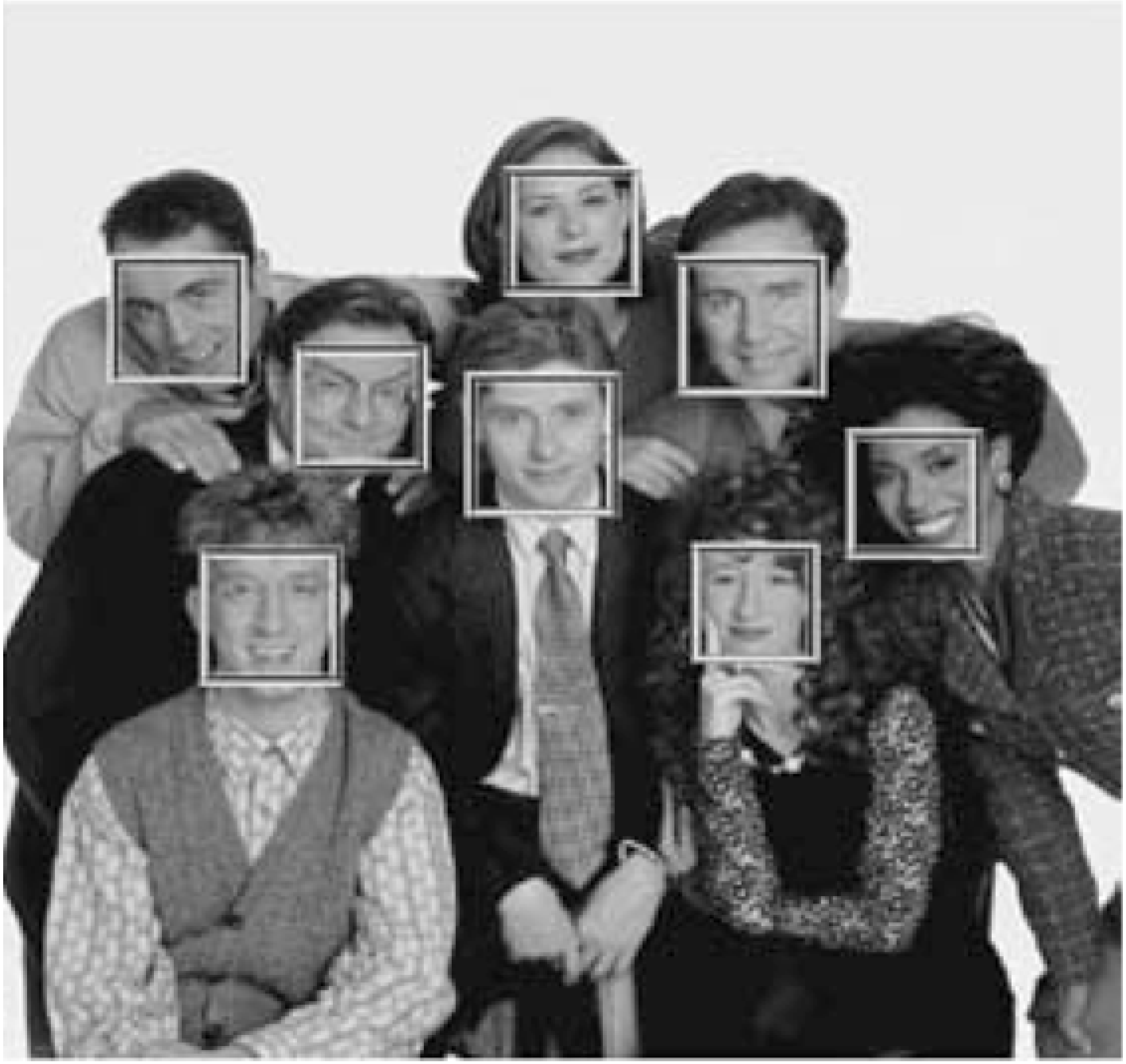
MIT test set:
Sung & pog
AdaBoost: d

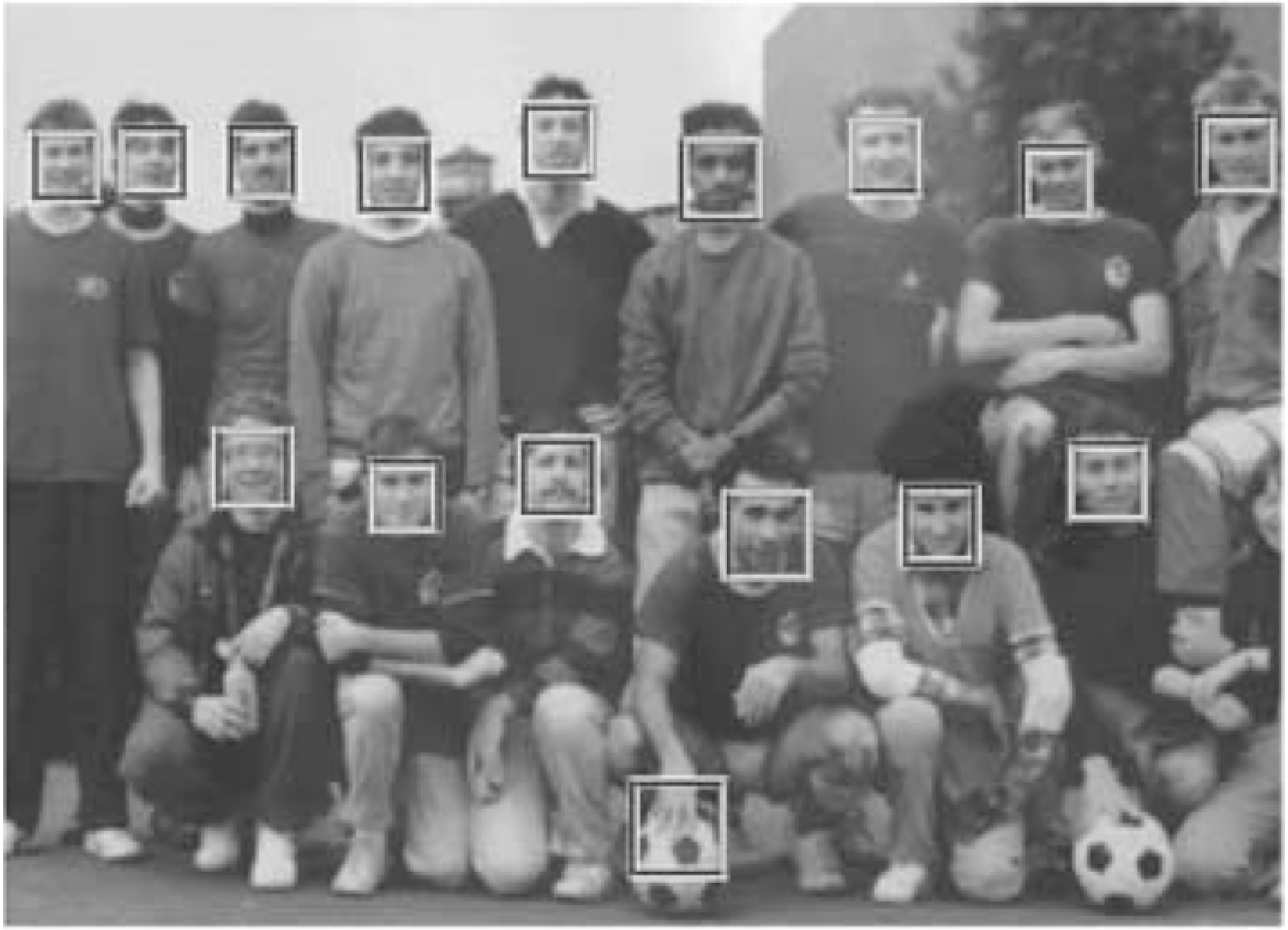


-> not significantly different accuracy

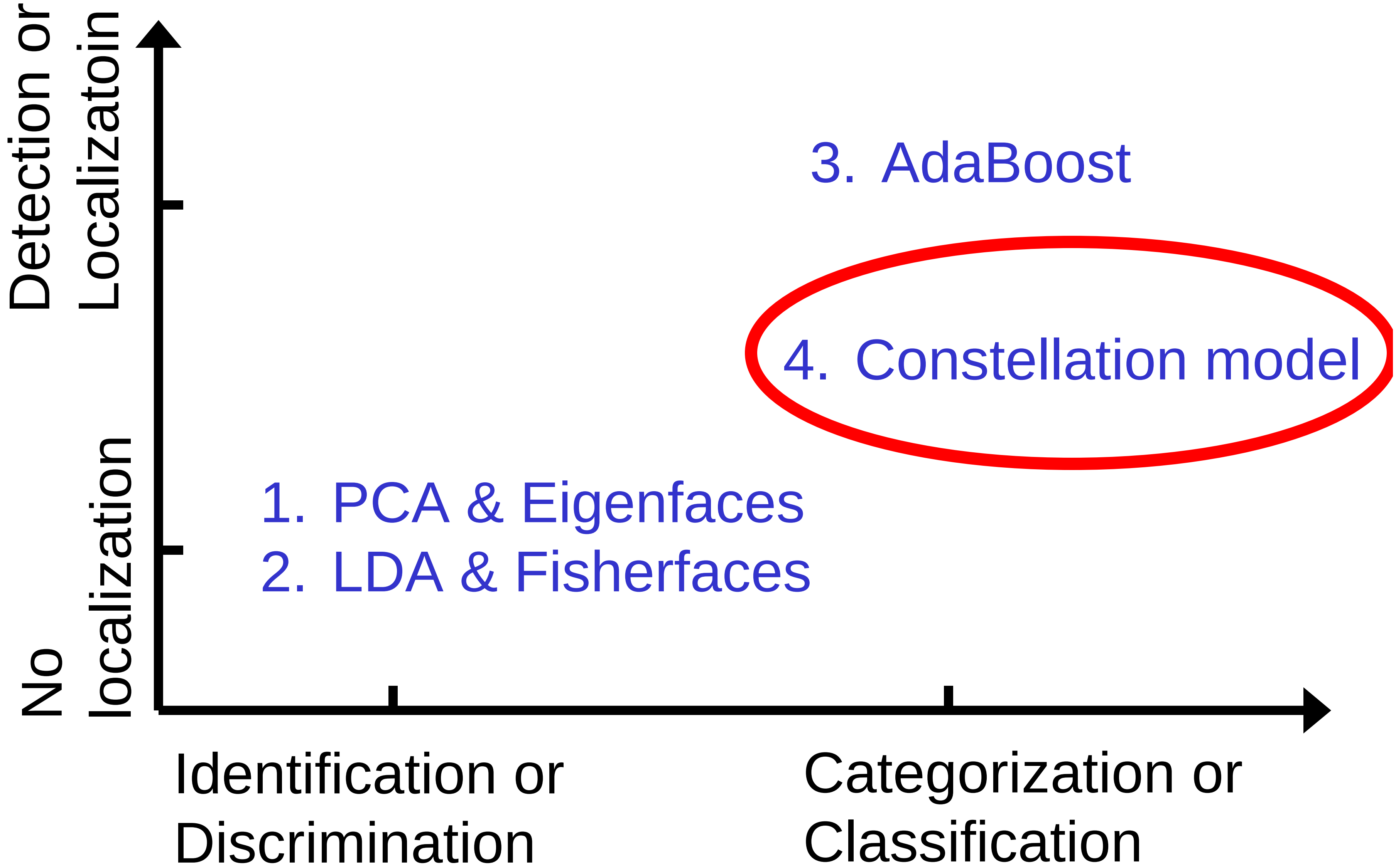
-> but the cascade class. almost 10 times faster



JUDYBATS



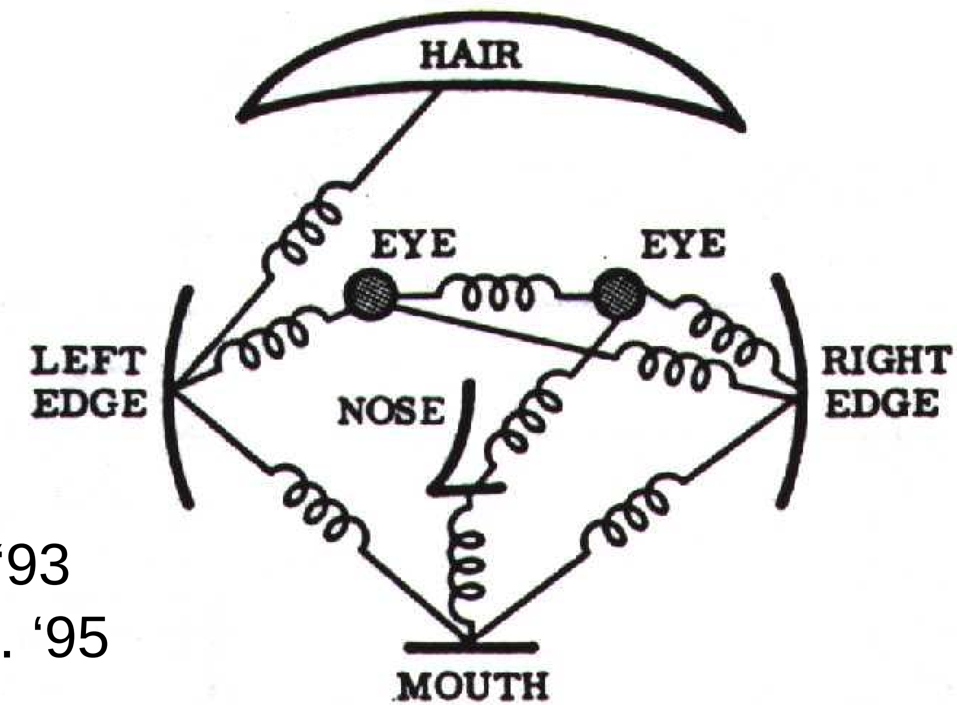


Today’s agenda



Parts and Structure Literature

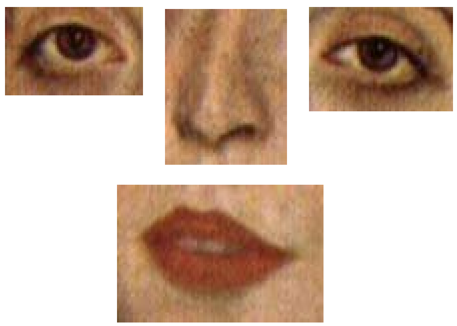
- Fischler & Elschlager 1973



- Yuille '91
- Brunelli & Poggio '93
- Lades, v.d. Malsburg et al. '93
- Cootes, Lanitis, Taylor et al. '95
- Amit & Geman '95, '99
- et al. Perona '95, '96, '98, '00, '03
- Huttenlocher et al. '00
- Agarwal & Roth '02

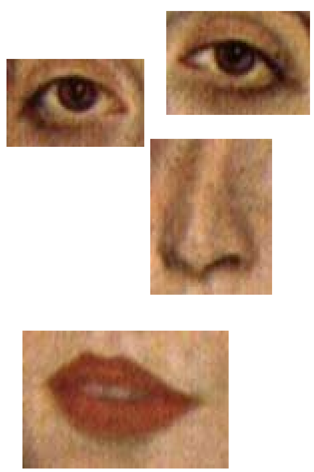
etc…

Deformations



A

B

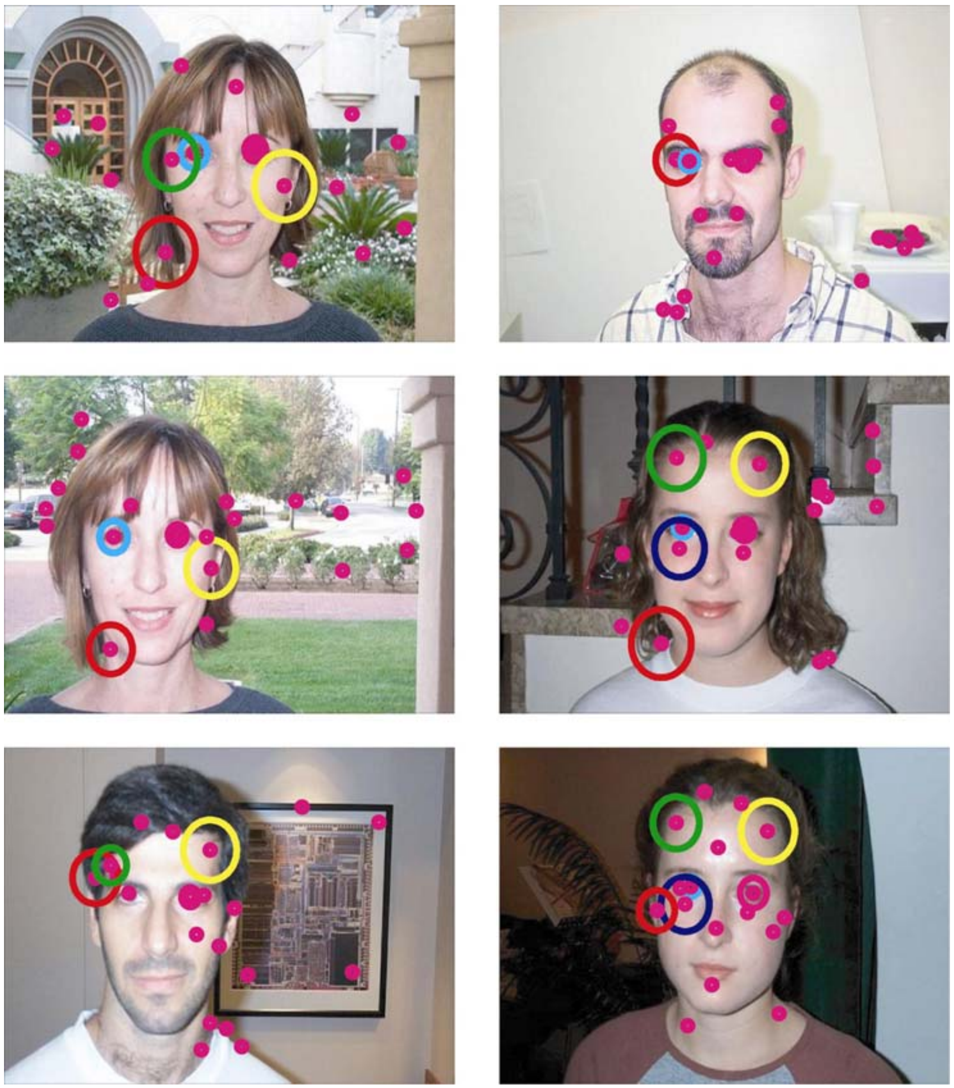


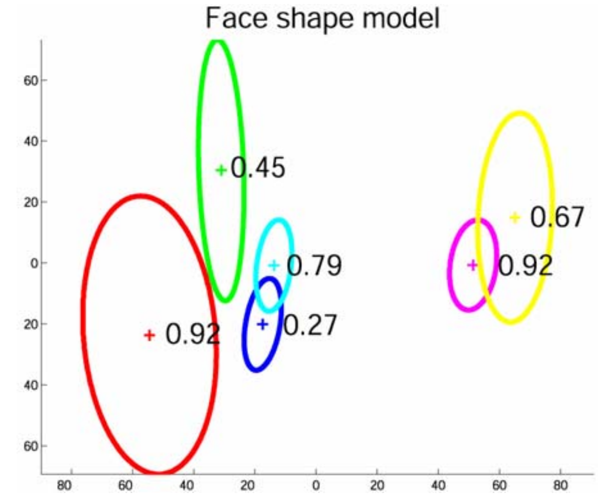
C

D

Background clutter

Frontal faces





Face images


correct
correct
correct
correct
correct
correct
correct
correct
correct
correct
correct
correct
correct
correct
correct
correct
correct
incorrect
incorrect
incorrect

3D Object recognition – Multiple mixture components

3D Orientation Tuning