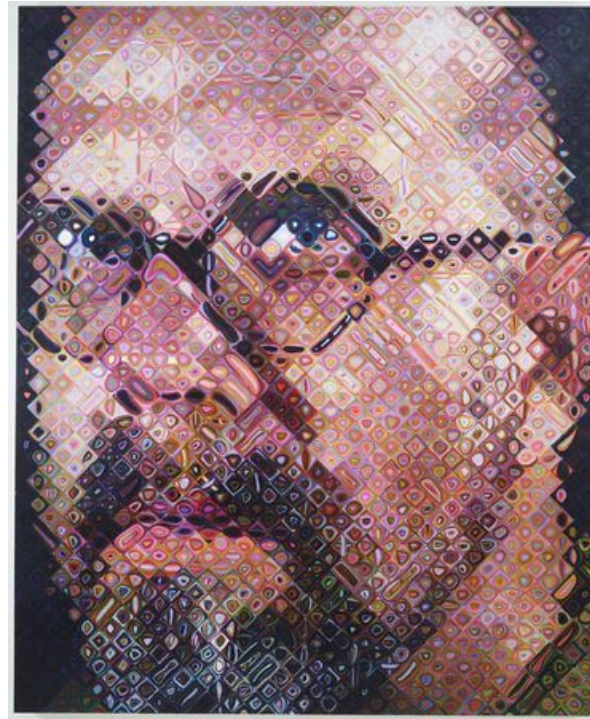
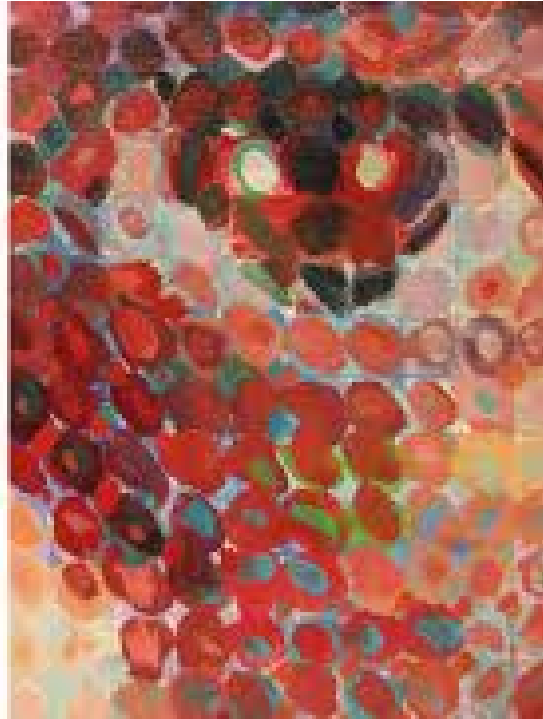


# Detection, Recognition, and Transformation of Faces



Lucas by Chuck Close



Chuck Close, self portrait

# Face detection and recognition



Detection

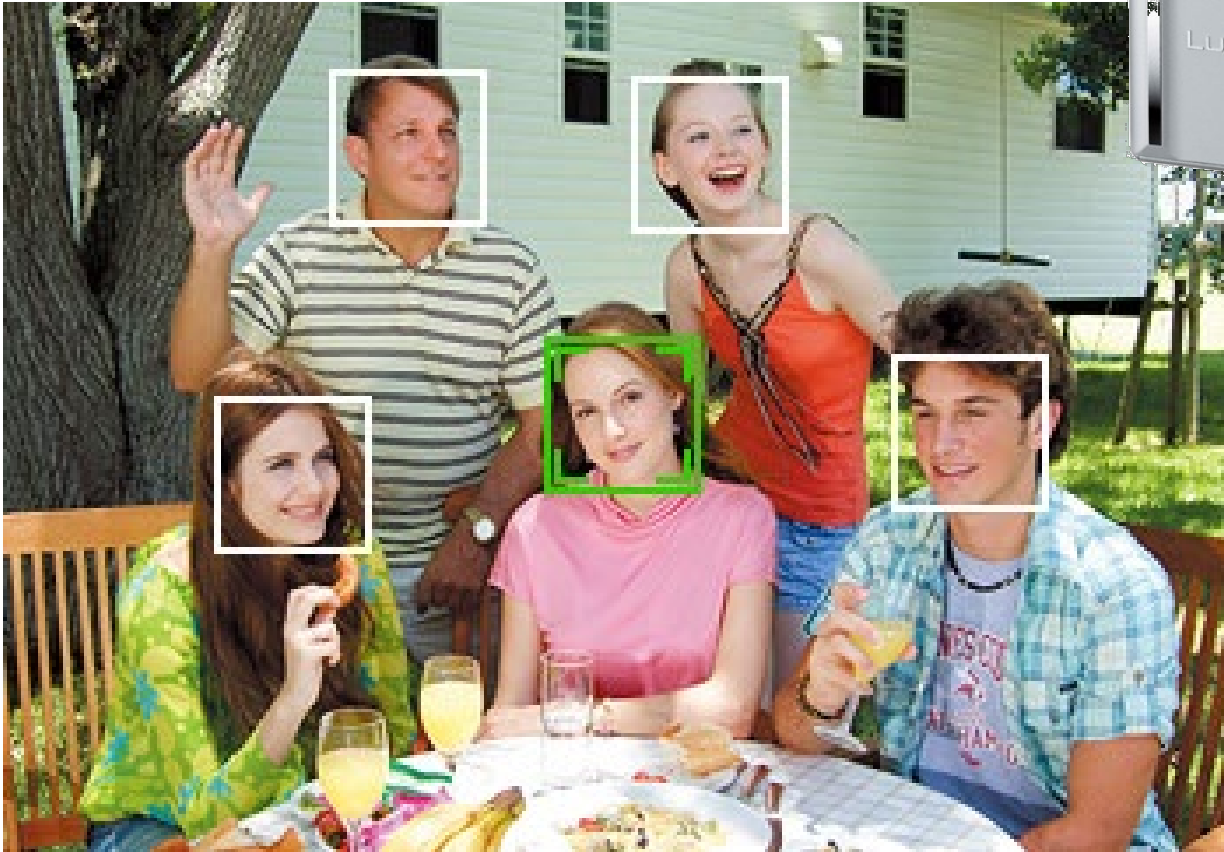


Recognition

"Sally"

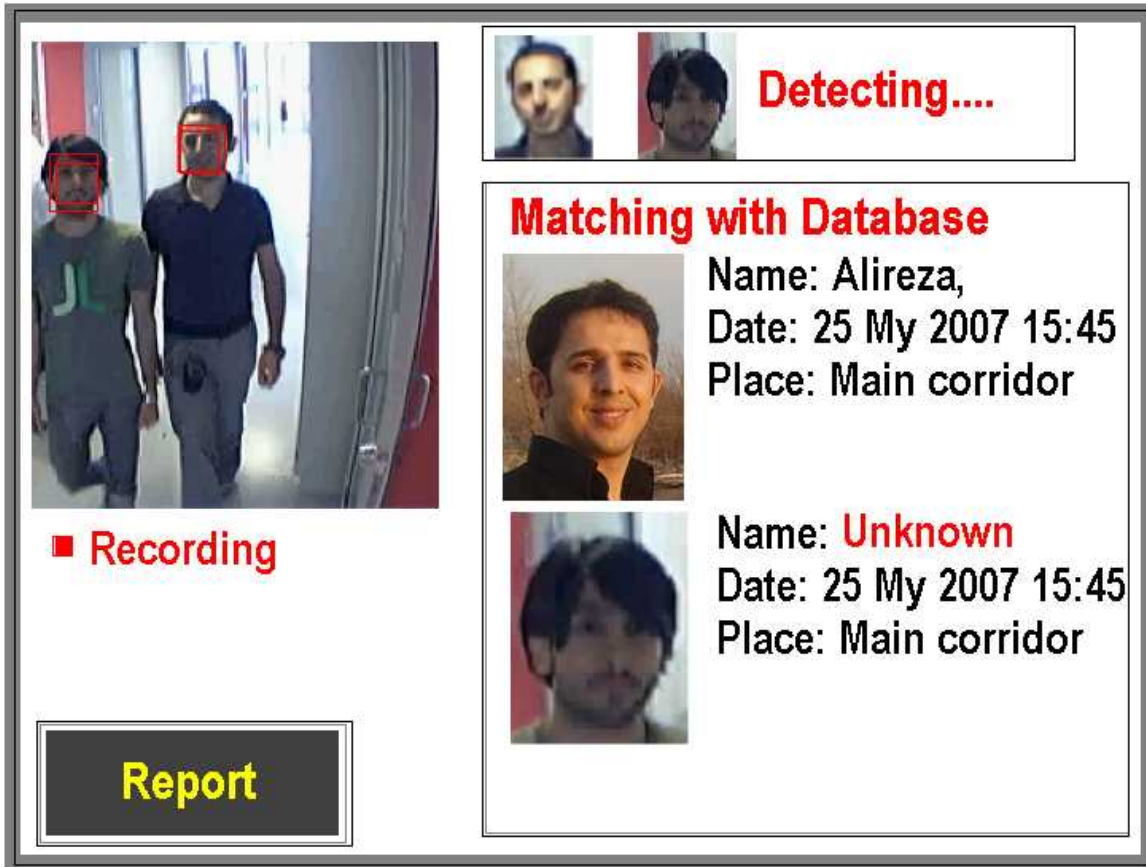
# Applications of Face Recognition

- Digital photography



# Applications of Face Recognition

- Digital photography
- 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

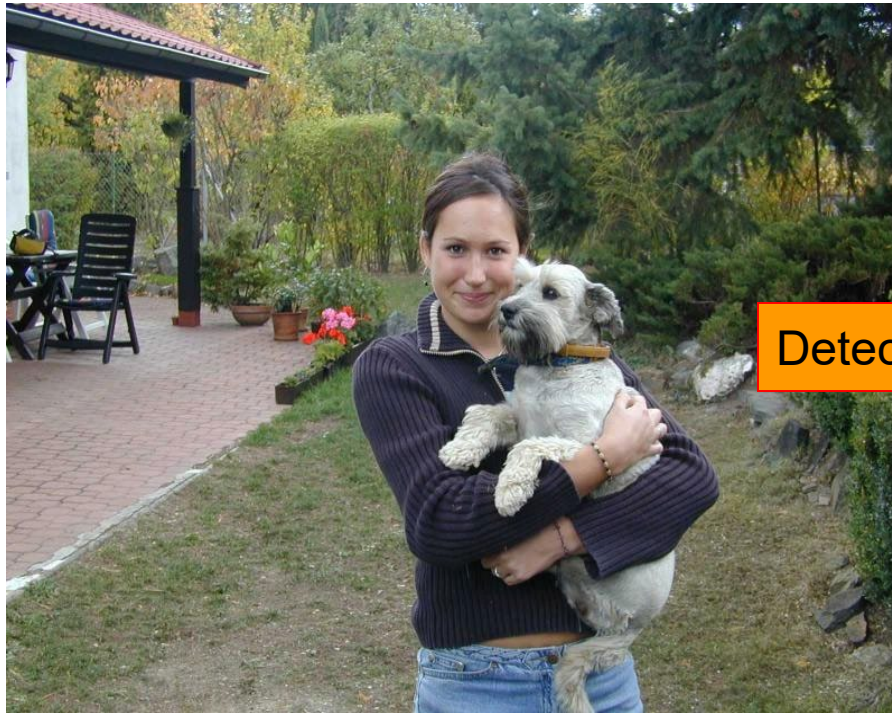


# Applications of Face Recognition

- Digital photography
- Surveillance
- Album organization



# Face detection



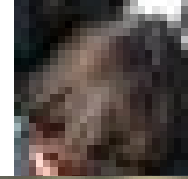
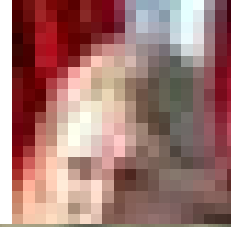
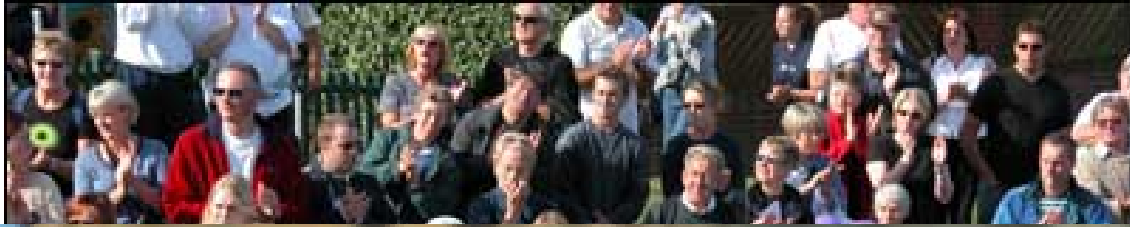
Detection



# What does a face look like?



# What does a face look like?





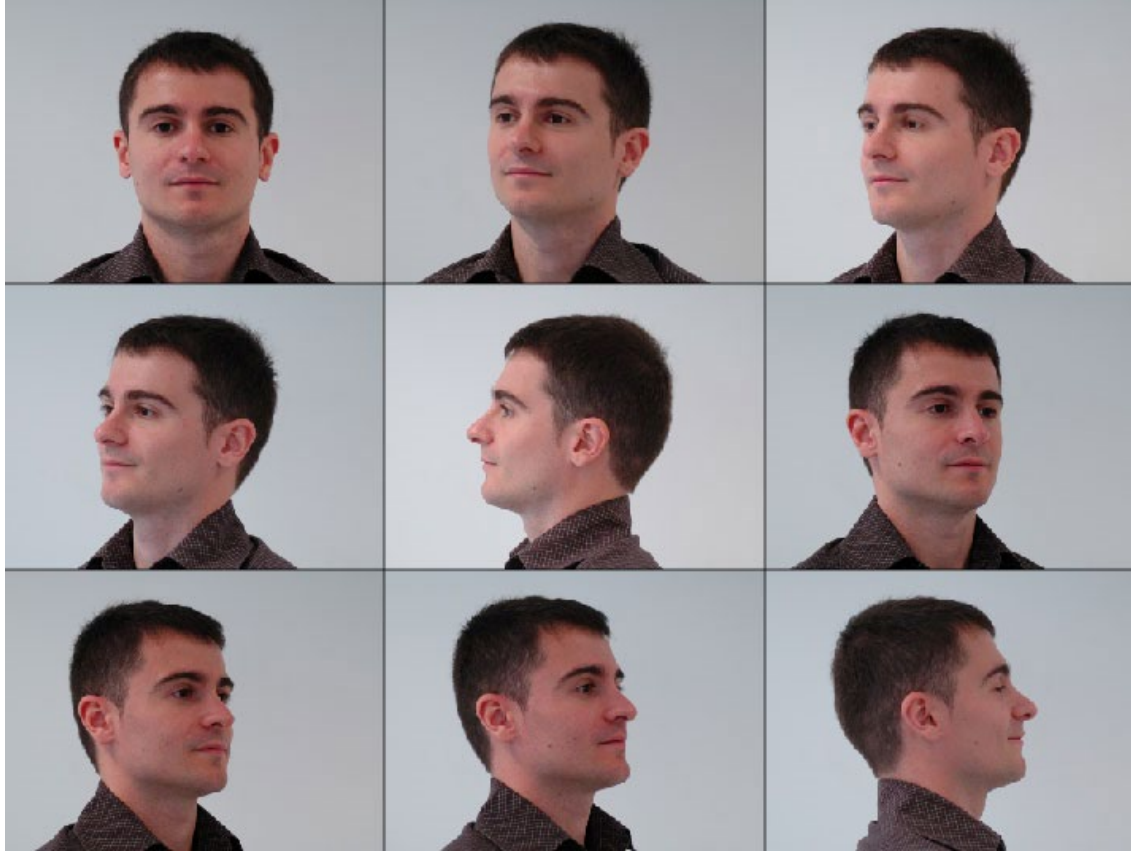
# What makes face detection hard?

## Expression



# What makes face detection hard?

## Viewpoint



# What makes face detection hard?

Occlusion



# What makes face detection and recognition hard?

## Coincidental textures





# Consumer application: iPhoto 2009

- Things iPhoto thinks are faces



# How to find faces anywhere in an image?

- Filter Image with a face?

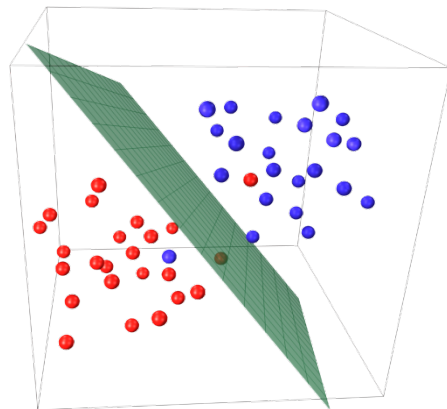


# Train a Filter

Positive Training Images



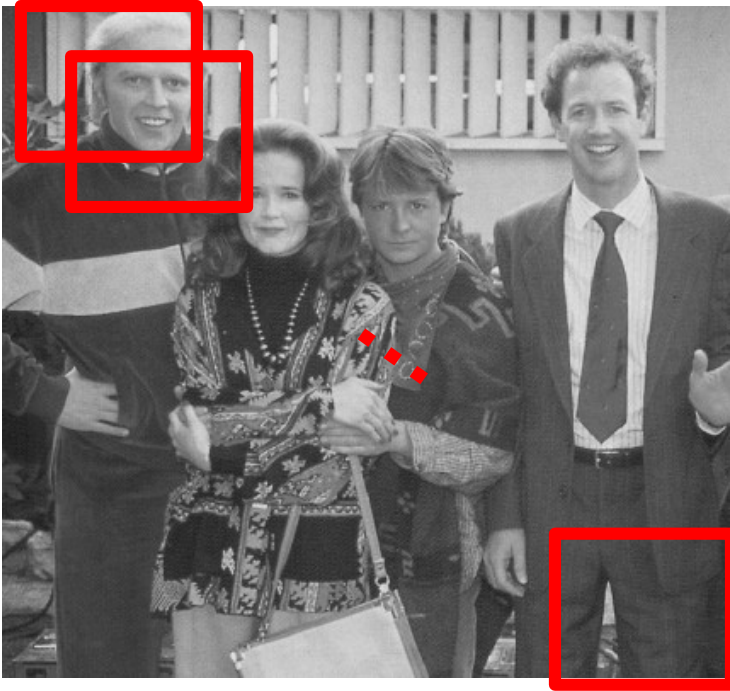
Negative Training Images



**SVM**



# Face detection: sliding windows



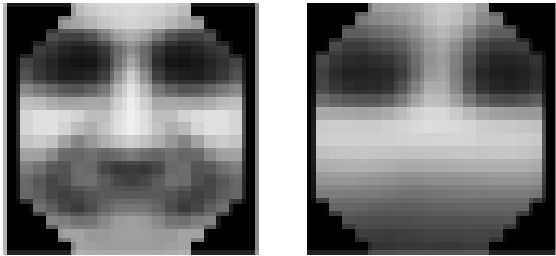
Filter/Template



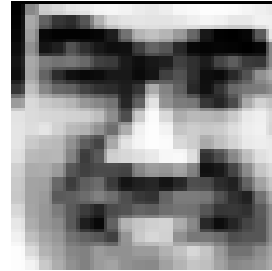
Multiple scales



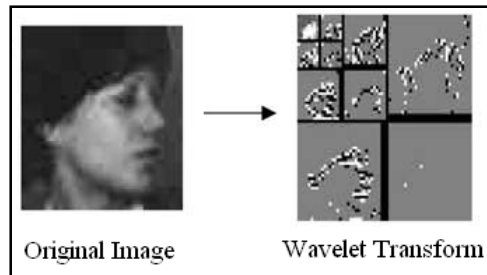
# What features?



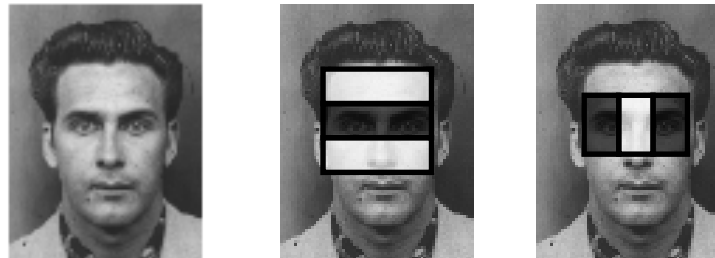
Exemplars  
(Sung Poggio 1994)



Intensity Patterns (with NNs)  
(Rowley Baluja Kanade 1996)



Edge (Wavelet) Pyramids  
(Schneiderman Kanade 1998)

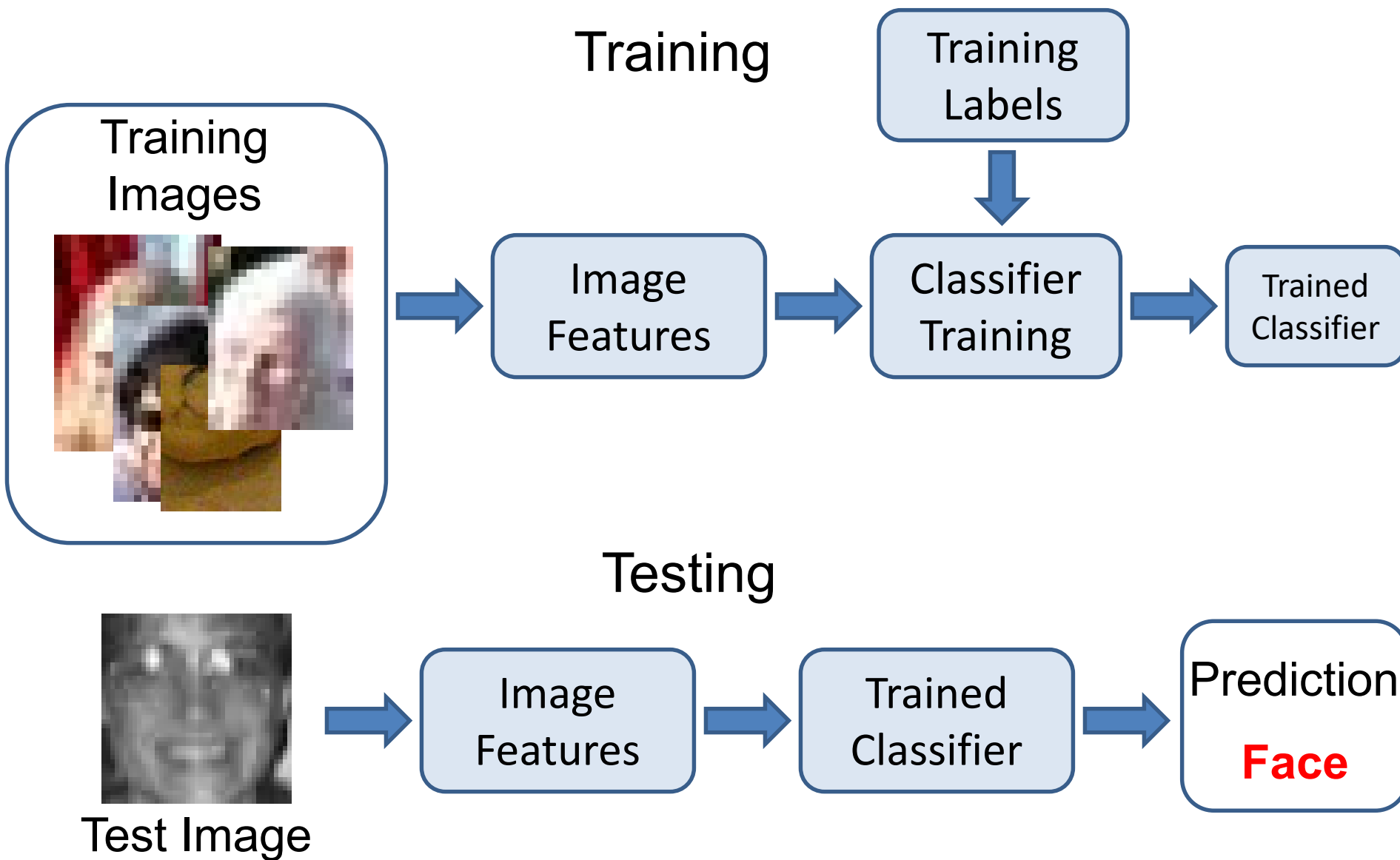


Haar Filters  
(Viola Jones 2000)

# How to classify?

- Many ways
  - Neural networks
  - Adaboost
  - SVMs
  - Nearest neighbor

# Face classifier



# Face Detection: State of the Art

## RetinaFace: Single-stage Dense Face Localisation in the Wild

Jiankang Deng<sup>\* 1,2,4</sup>    Jia Guo<sup>\* 2</sup>    Yuxiang Zhou<sup>1</sup>

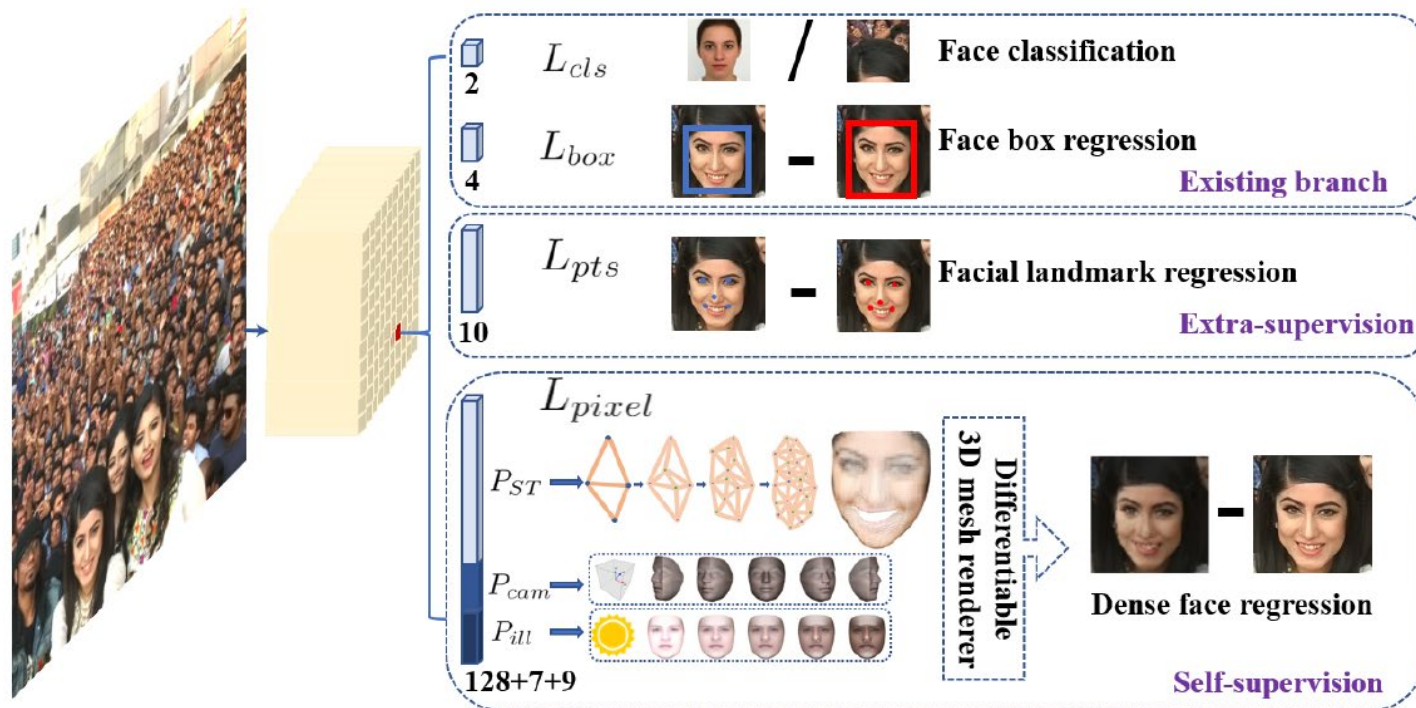
Jinke Yu<sup>2</sup>    Irene Kotsia<sup>3</sup>    Stefanos Zafeiriou<sup>1,4</sup>

<sup>1</sup>Imperial College London

<sup>2</sup>InsightFace

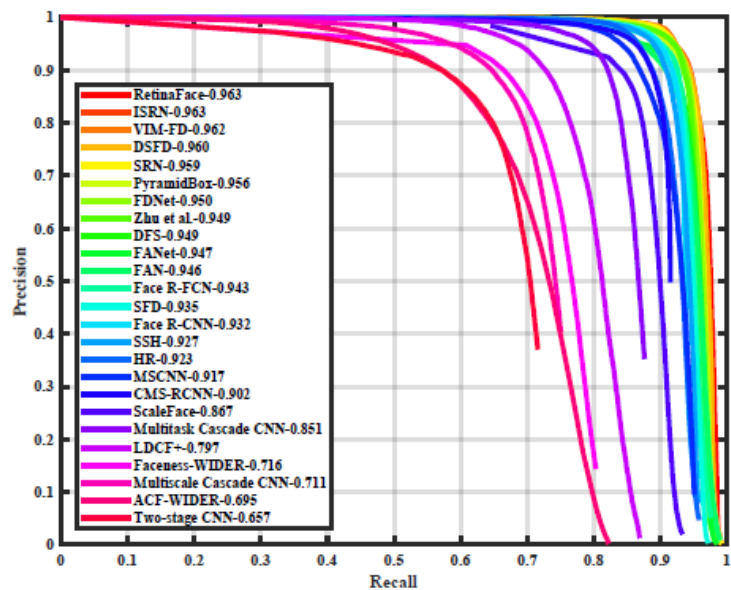
<sup>3</sup>Middlesex University London

<sup>4</sup>FaceSoft

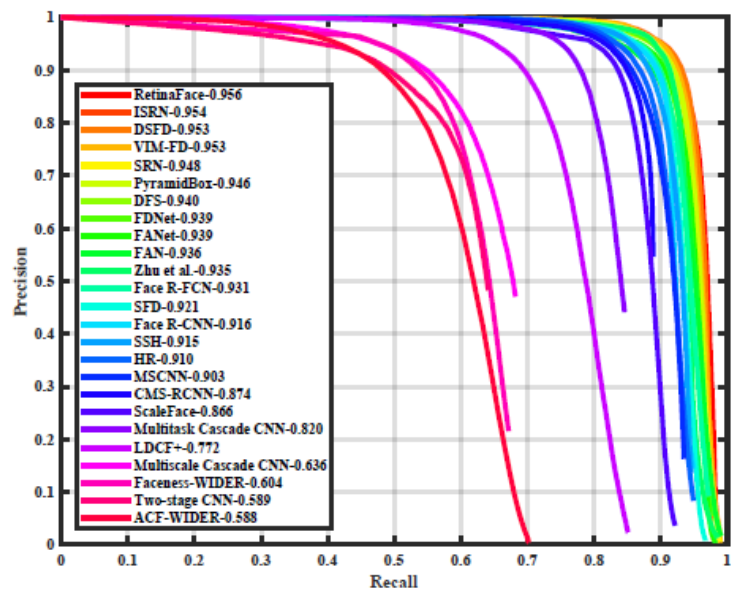




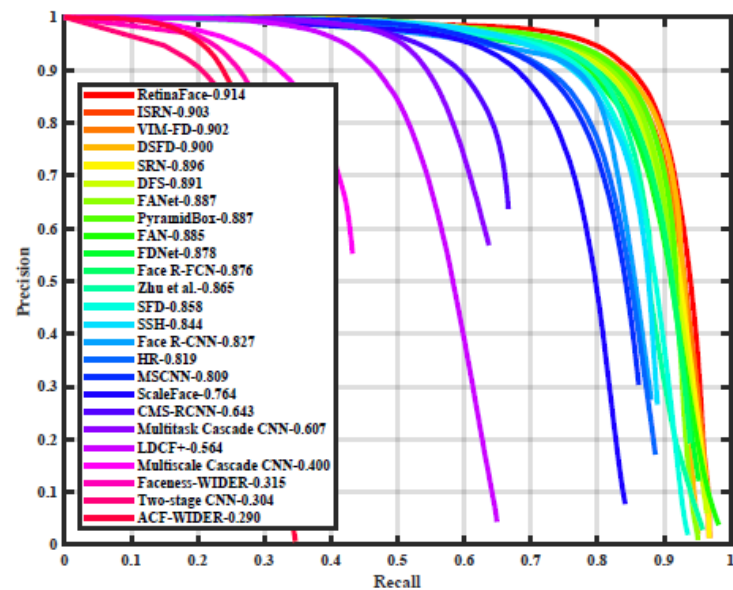
Perfect



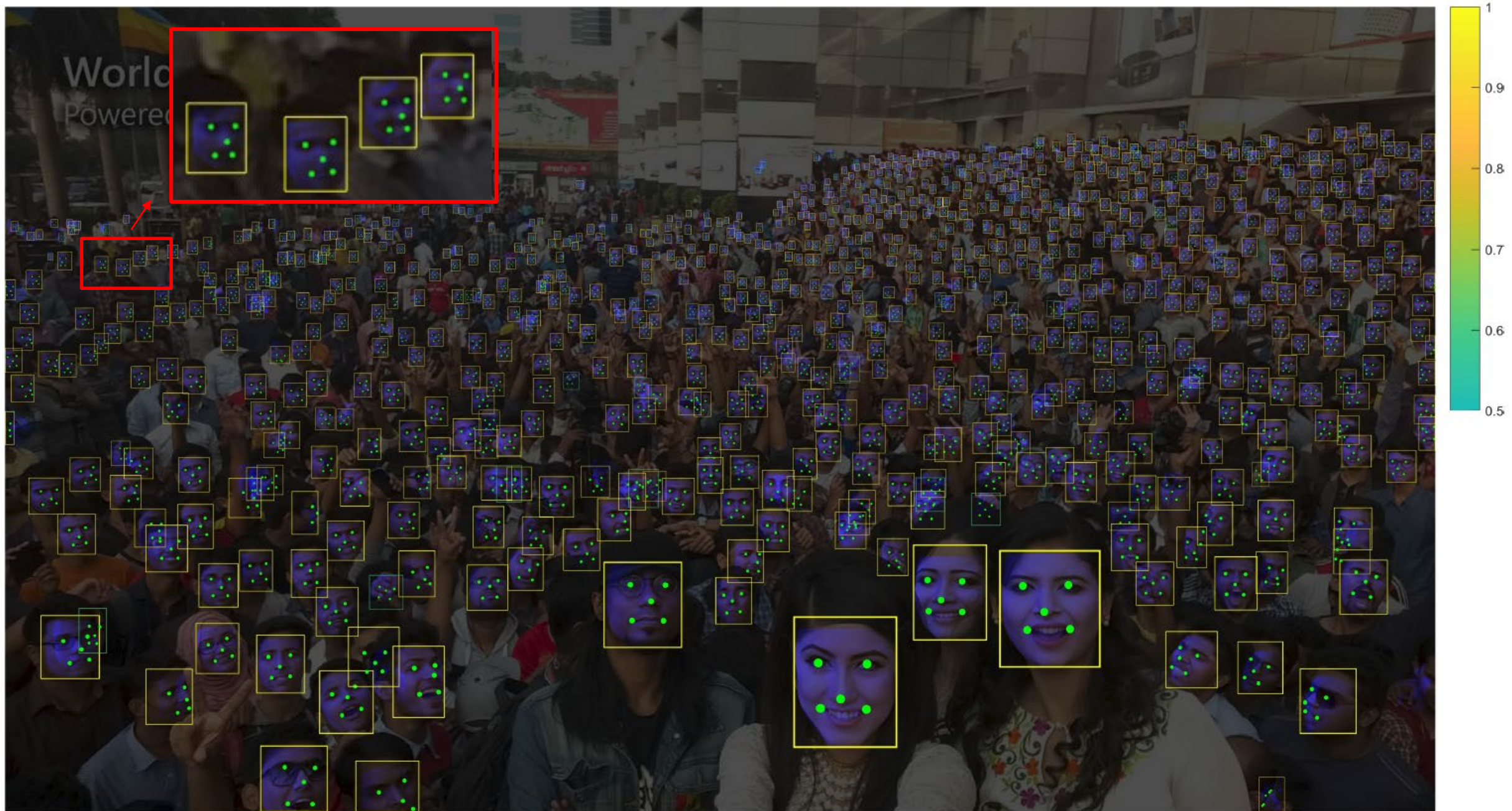
(d) Test: Easy



(e) Test: Medium



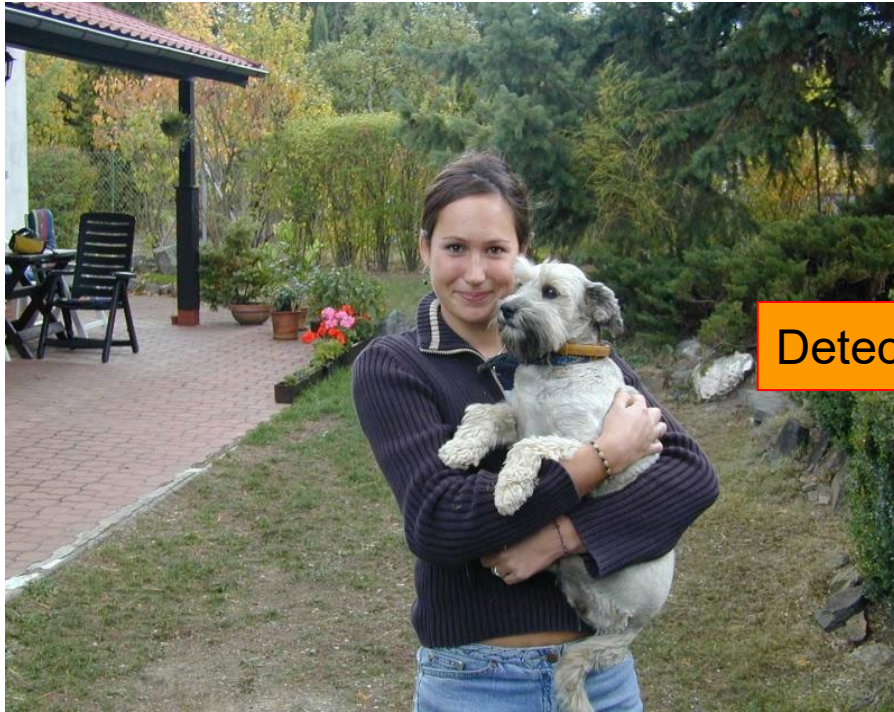
(f) Test: Hard



RetinaFace can find around 900 faces (threshold at 0.5) out of the reported 1151 people



# Face recognition



Detection



Recognition

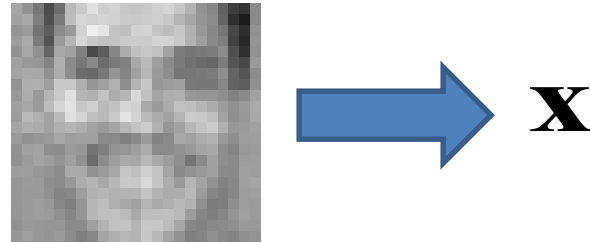
“Sally”

# Face recognition

- 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)

# Simple technique

1. Treat pixels as a vector



2. Recognize face by nearest neighbor



$$k = \underset{k}{\operatorname{argmin}} \left\| \mathbf{y}_k^T - \mathbf{x} \right\|$$



# 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

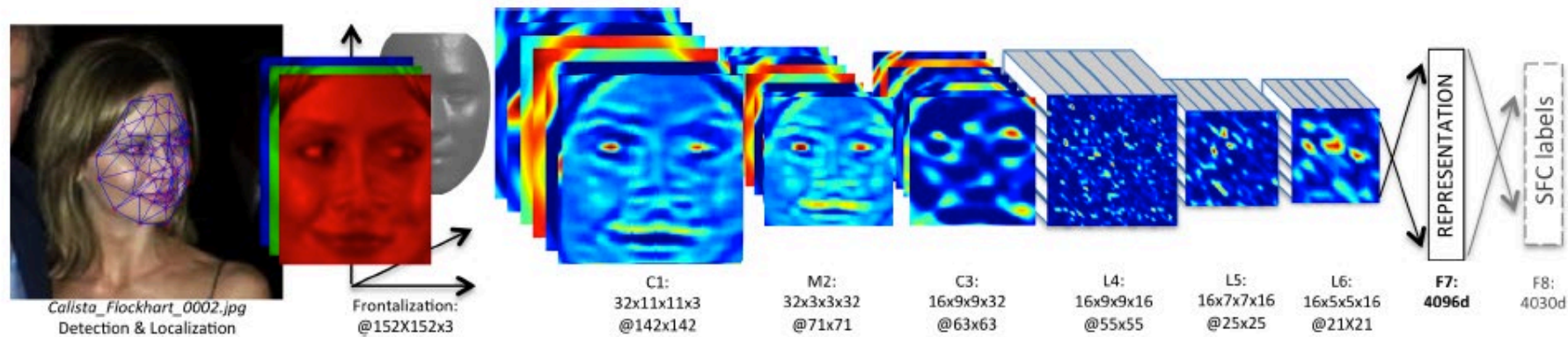
Menlo Park, CA, USA

{yaniv, mingyang, ranzato}@fb.com

Tel Aviv University

Tel Aviv, Israel

wolf@cs.tau.ac.il



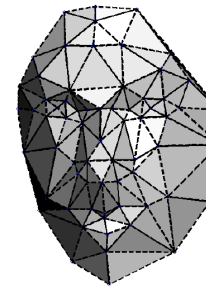
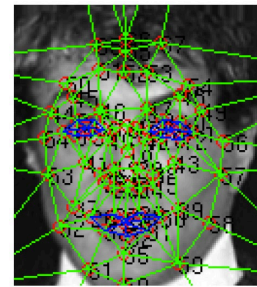
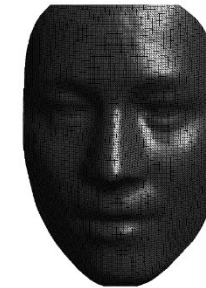
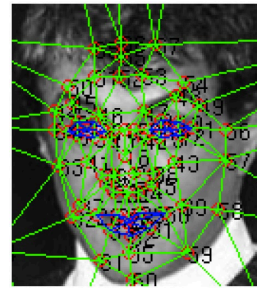
DeepFace: Closing the Gap to Human-Level Performance in Face Verification

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

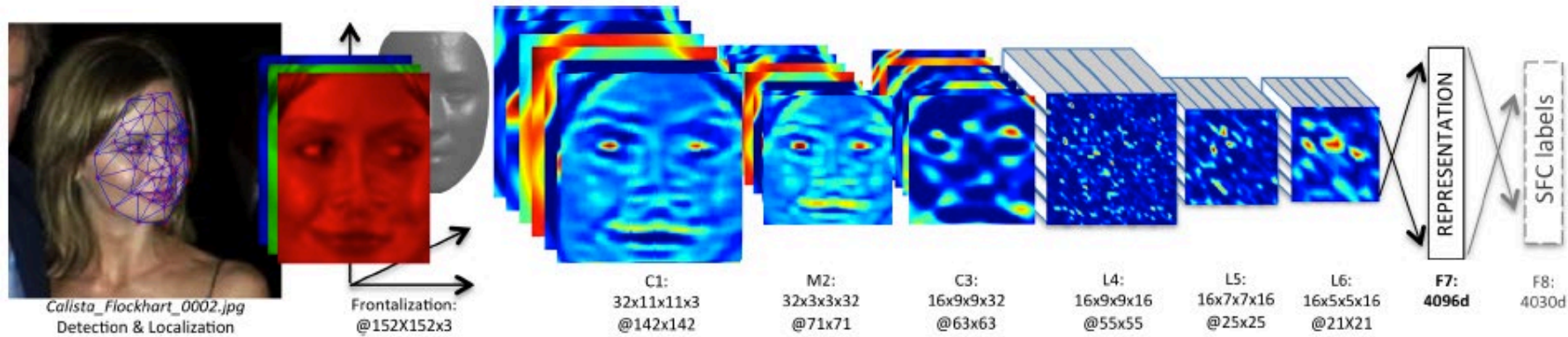
Following slides adapted from Daphne Tsatsoulis

# 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 projection and use it to frontally warp the face



# Train DNN classifier on aligned faces



Architecture (deep neural network classifier)

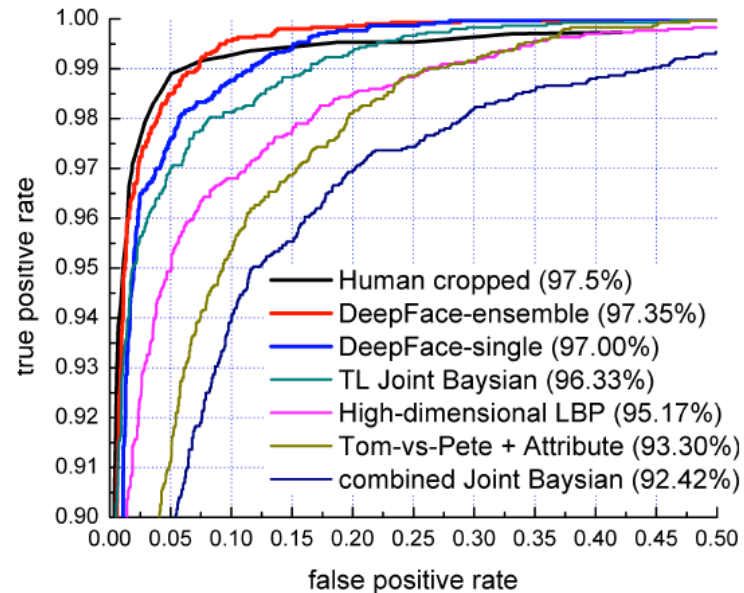
- Two convolutional layers (with one pooling layer)
- 3 locally connected and 2 fully connected layers
- > 120 million parameters

Train on dataset with 4400 individuals, ~1000 images each

- Train to identify face among set of possible people

Face matching (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)



Transforming faces

# Figure-centric averages

- Need to Align
  - Position
  - Scale
  - Orientation



Antonio Torralba & Aude Oliva (2002)

**Averages:** Hundreds of images containing a person are averaged to reveal regularities in the intensity patterns across all the images.

# How do we average faces?



<http://www2.imm.dtu.dk/~aam/datasets/datasets.html>

# Morphing

image #1



image #2

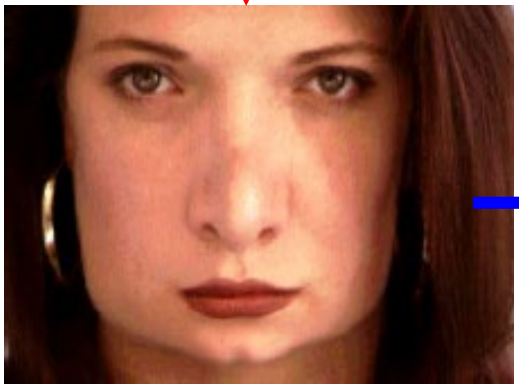


warp

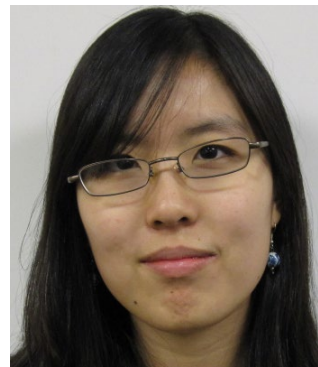


morphing

warp



# Cross-Dissolve vs. Morphing



Average of  
Appearance Vectors



Average of  
Shape Vectors



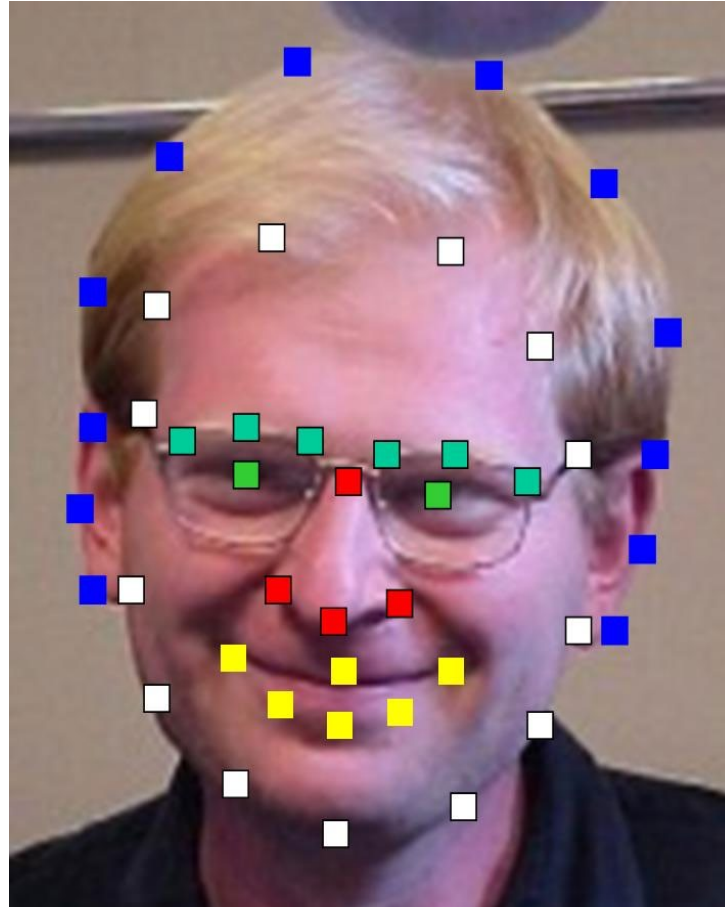
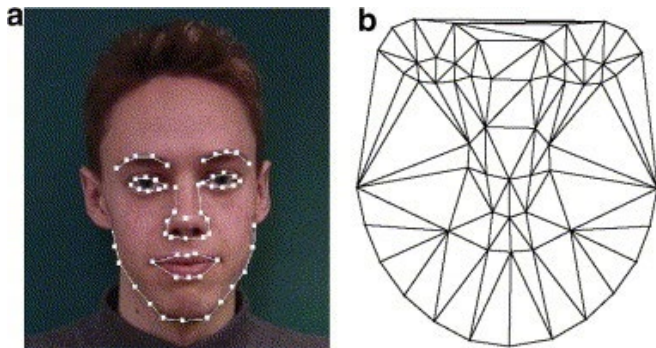
[http://www.faceresearch.org/  
demos/vector](http://www.faceresearch.org/demos/vector)

Images from James Hays



# Aligning Faces

- Need to Align
  - Position
  - Scale
  - Orientation
  - **Key-points**
- The more key-points, the finer alignment



# Appearance Vectors vs. Shape Vectors

Appearance  
Vector

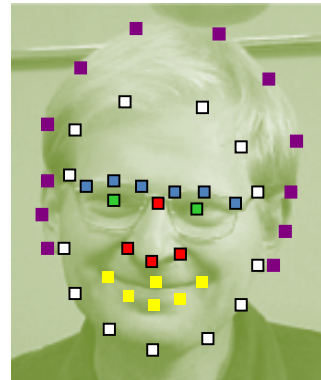


200\*150 pixels (RGB)



Vector of  
200\*150\*3  
Dimensions

Shape  
Vector



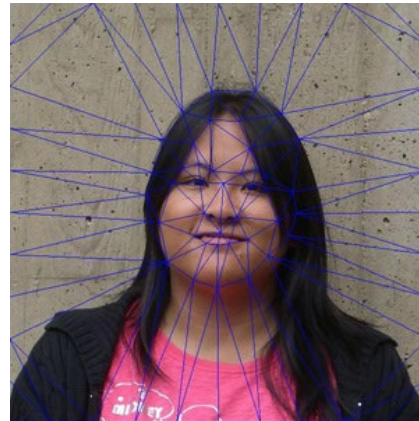
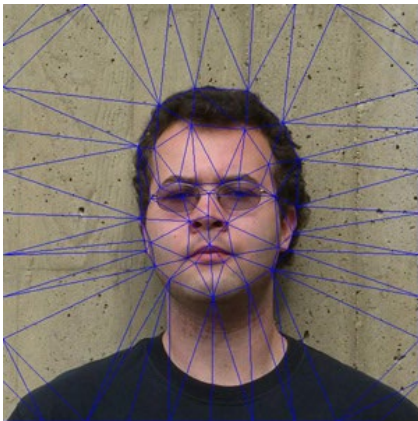
43 coordinates (x,y)



Vector of  
43\*2  
Dimensions

# Average of two Faces

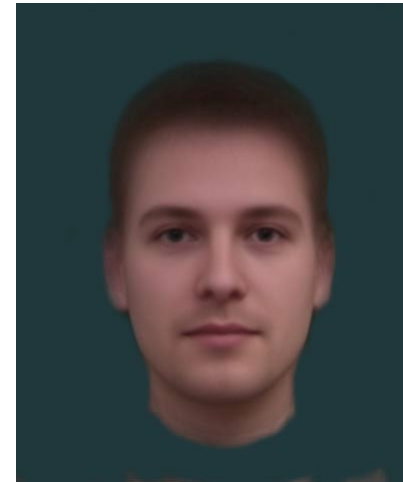
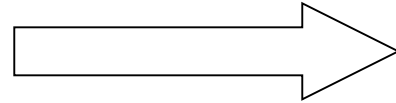
1. Input face keypoints
2. Pairwise average keypoint coordinates
3. Triangulate the faces
4. Warp: transform every face triangle
5. Average the pixels



# Average of multiple faces



1. Warp to mean shape
2. Average pixels





# Average Men of the world



AUSTRIA



AFGHANISTAN



ARGENTINA



BURMA (MYANMAR)



GERMANY



GREECE



CAMBODIA



ENGLAND



ETHIOPIA



FRANCE



IRAQ



IRELAND



MONGOLIA



PERU



POLAND



PUERTO RICO



UZBEKISTAN



AFRICAN AMERICAN



# Average Women of the world



Central African

Burmese

Cambodian

English

Ethiopian

Filipino



Greek

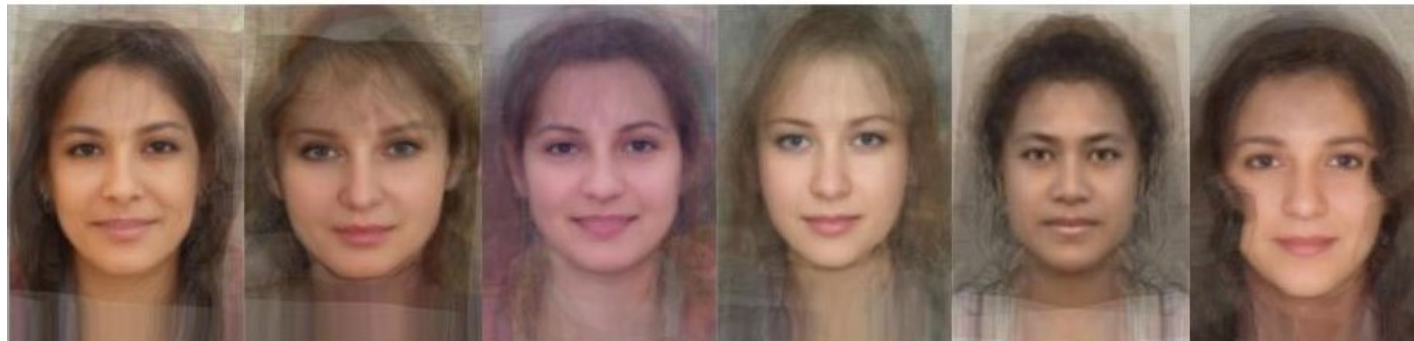
Indian

Iranian

Irish

Israeli

Italian



Peruvian

Polish

Romanian

Russian

Samoan

South African

# Subpopulation means

## Other Examples:

- Average Kids
- Happy Males
- Etc.



Average kid



Average happy male



Average female

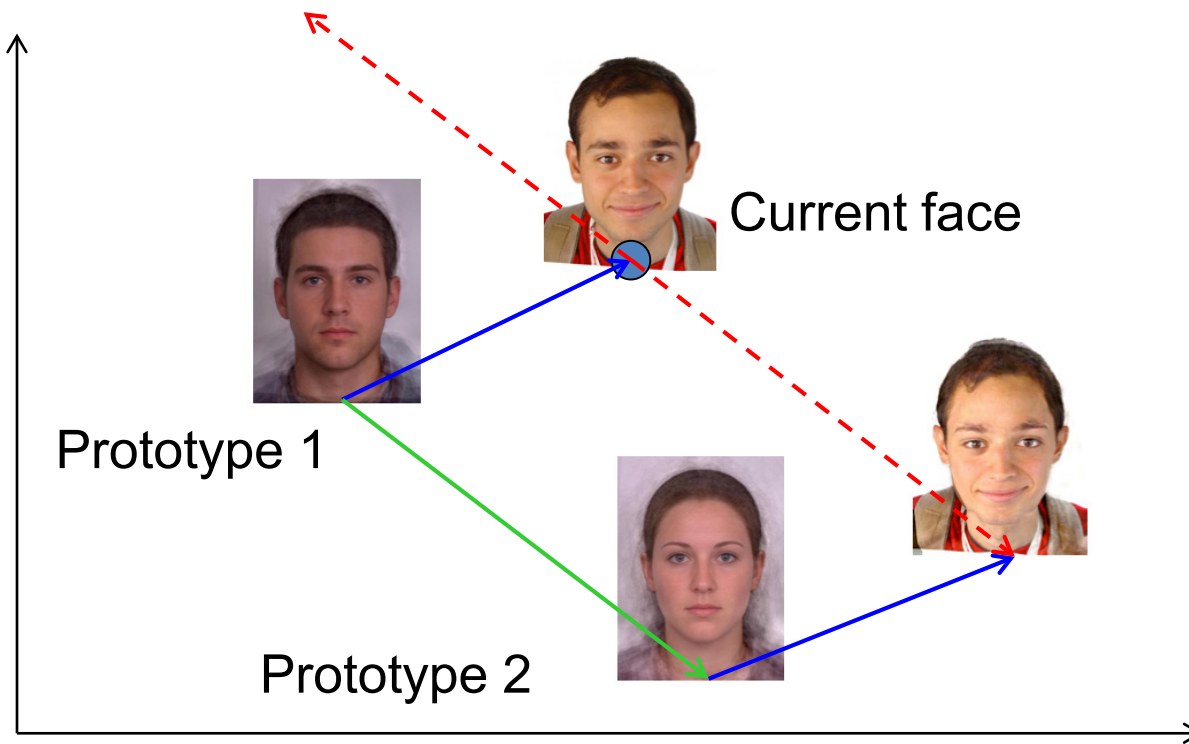


Average male

# Manipulating faces

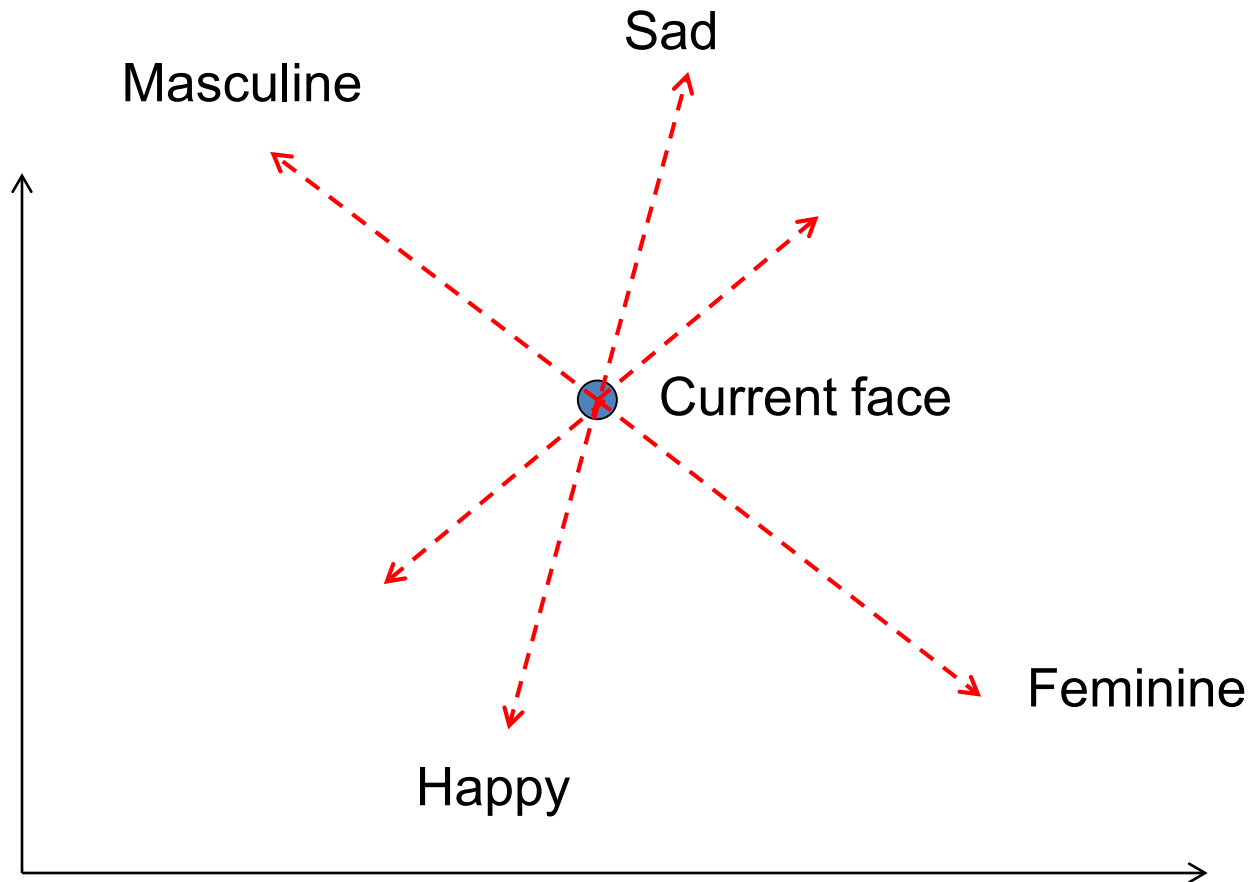
How can we make a face look more female/male, young/old, happy/sad, etc.?

With shape/texture analogies!



# Manipulating faces

We can imagine various meaningful directions



# Averaging and transformation demos

<http://www.faceresearch.org/demos>



# State-of-the-art in Face Faking

## A Style-Based Generator Architecture for Generative Adversarial Networks

Tero Karras  
NVIDIA

tkarras@nvidia.com

Samuli Laine  
NVIDIA

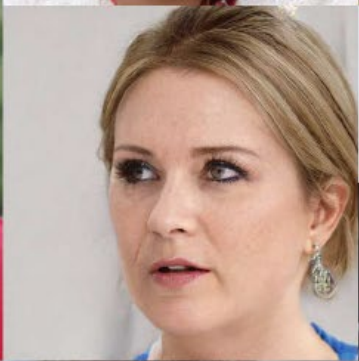
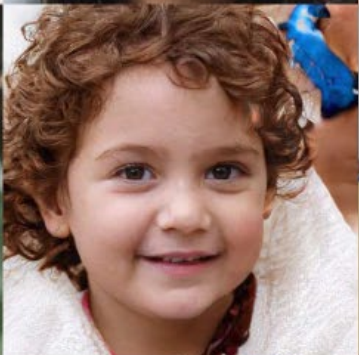
slaine@nvidia.com

Timo Aila  
NVIDIA

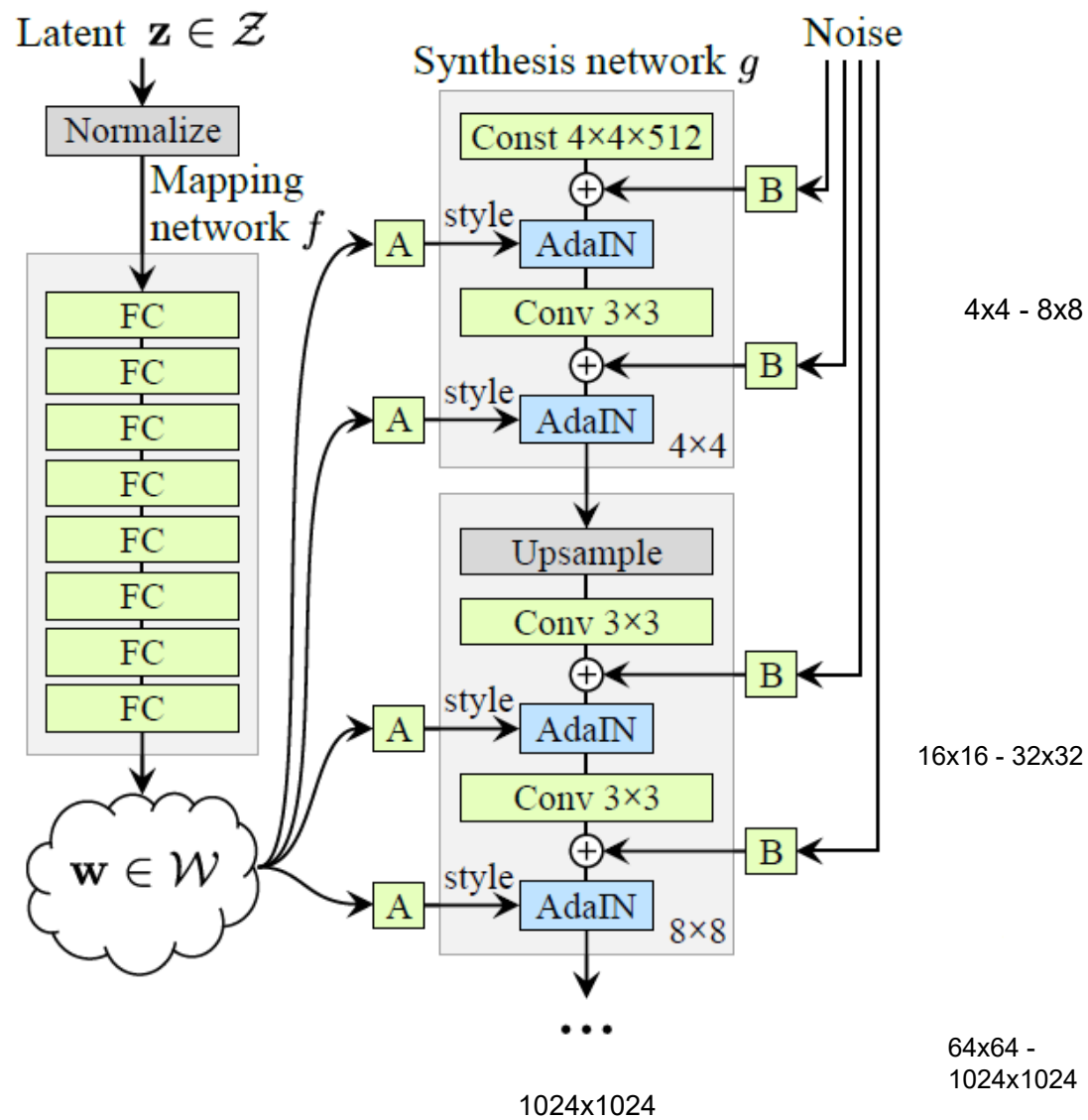
taila@nvidia.com



CVPR 2019 (Best Paper Honorable Mention)



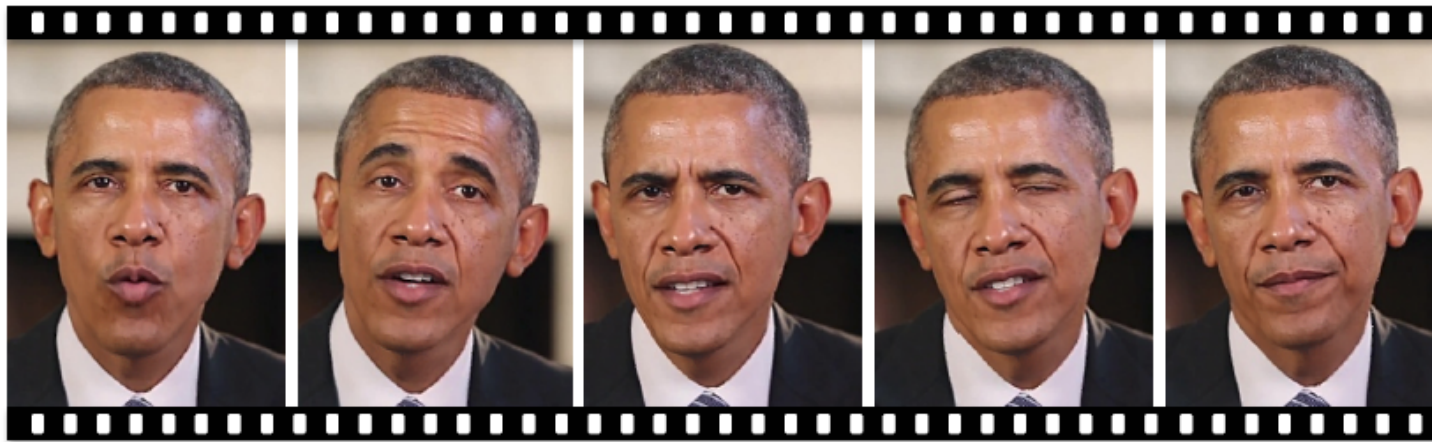




# Making people say what you want

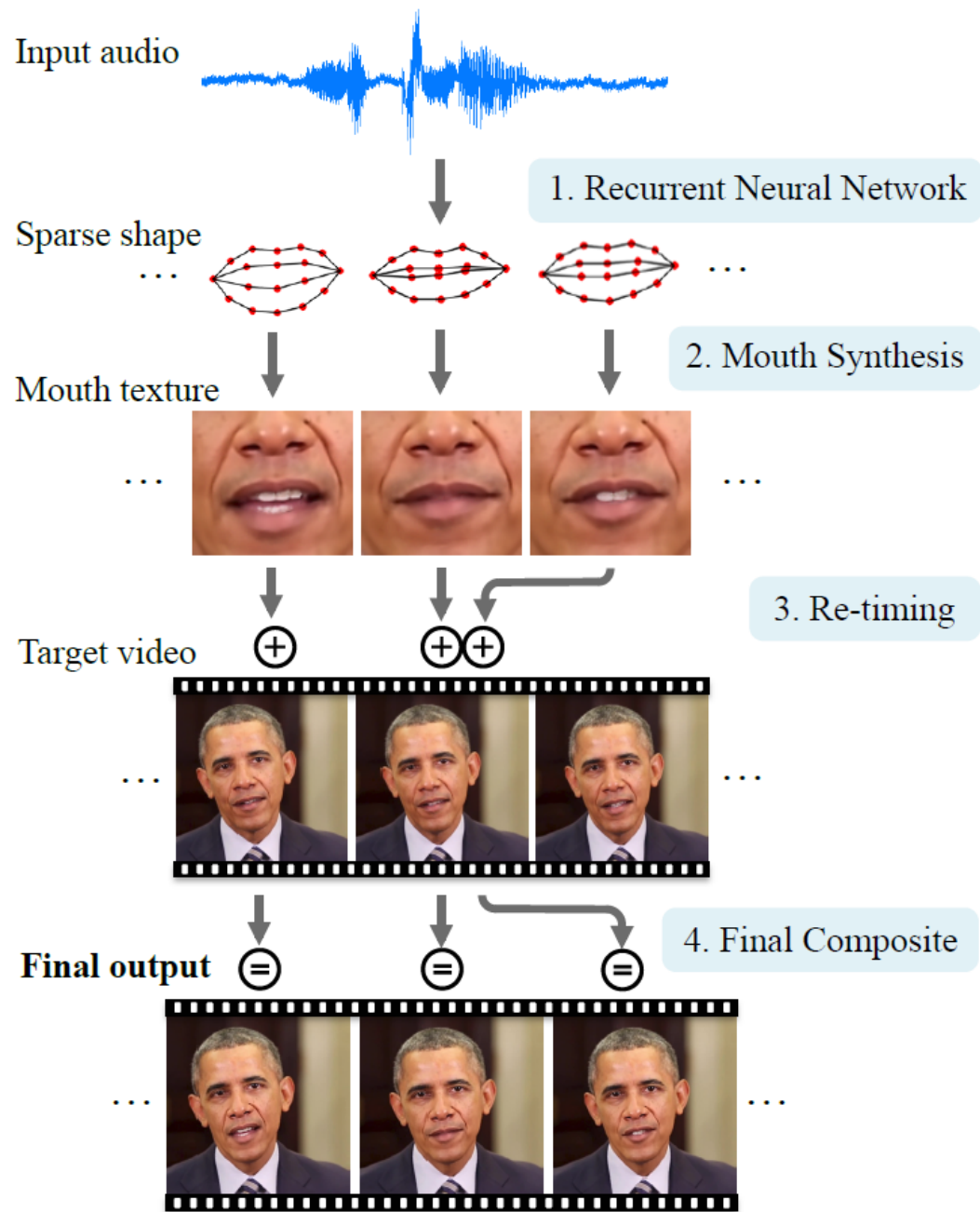
## Synthesizing Obama: Learning Lip Sync from Audio

SUPASORN SUWAJANAKORN, STEVEN M. SEITZ, and IRA KEMELMACHER-SHLIZERMAN, University of Washington



Output Obama Video

Fig. 1. Given input Obama audio and a reference video, we synthesize photorealistic, lip-synced video of Obama speaking those words.



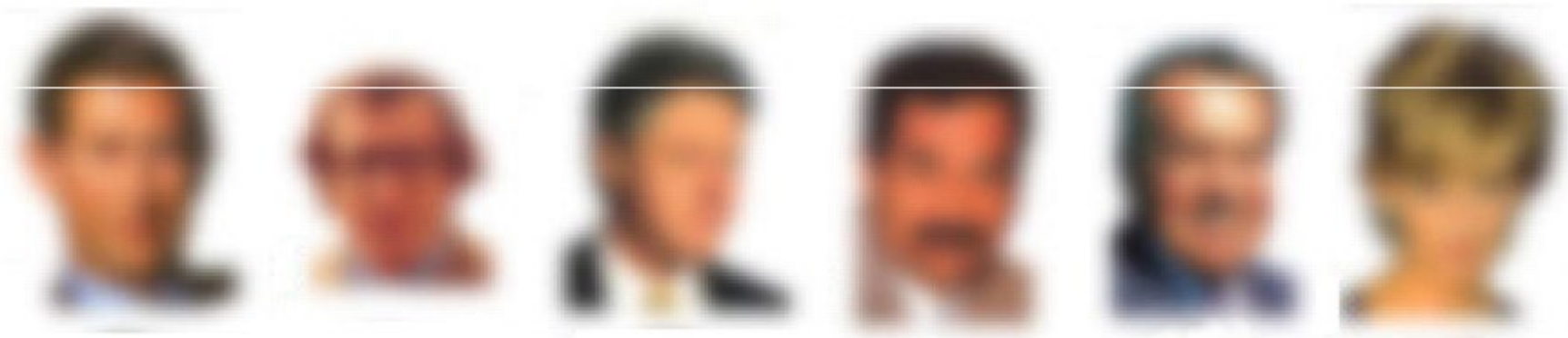


# Human Perception

## Result 1

---

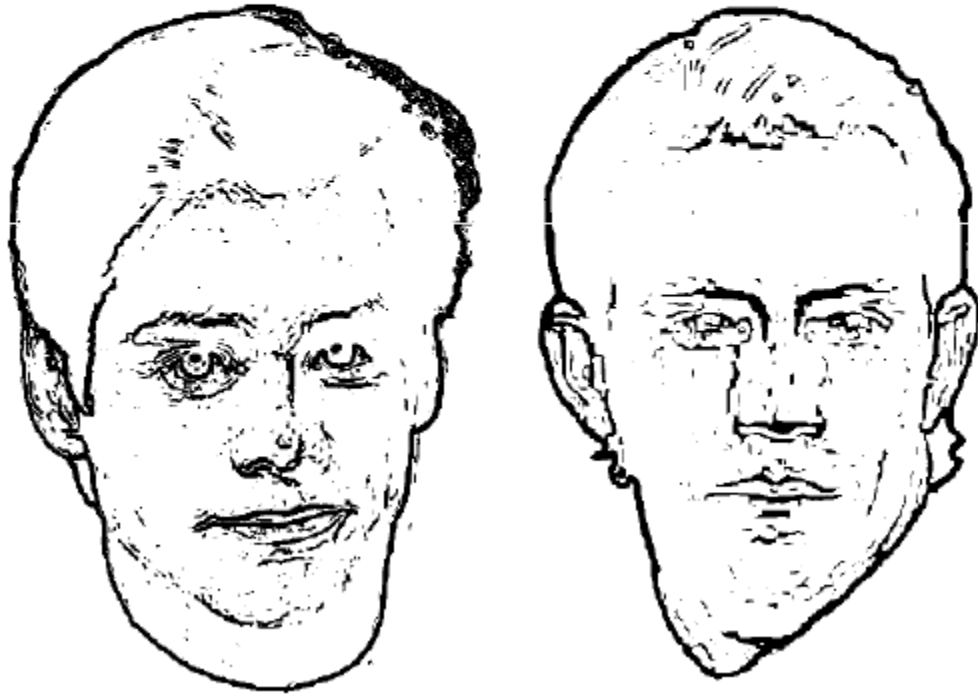
- ▶ Humans can recognize faces in extremely low resolution images.



## 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 8

---

- ▶ Vertical inversion dramatically reduces recognition performance

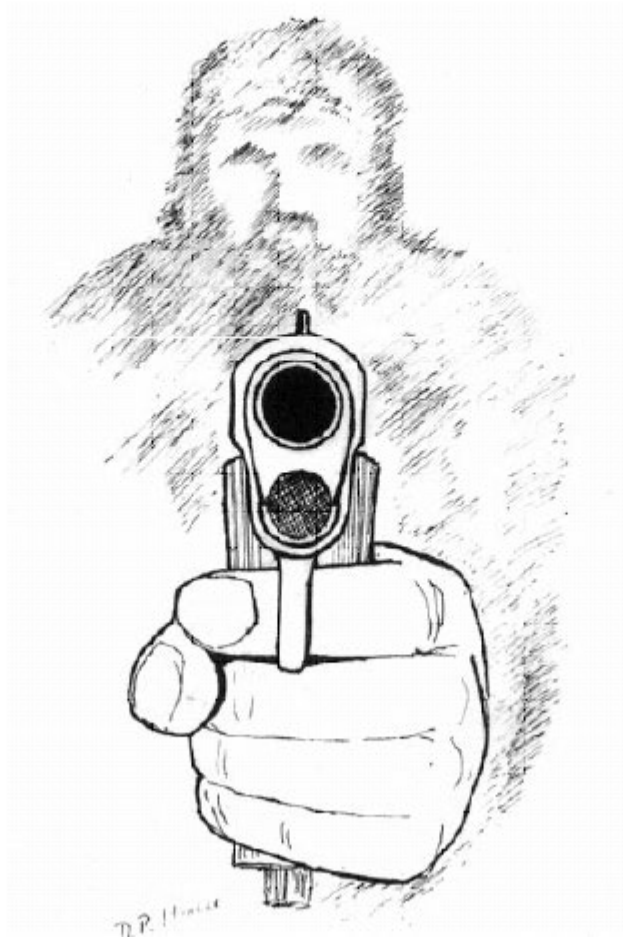




## Result 20

---

- ▶ Human memory for briefly seen faces is rather poor



# Things to remember

- Face Detection: train face vs. non-face model and scan over multi-scale image
- Face Recognition: detect, align, compute features, and compute similarity
- Represent faces with an appearance vector and a shape vector
- Can transform faces by moving shape vector in a given direction and warping
- Deep network methods enable more flexible mixing and generation

# Next lectures

- Motion magnification
- Cutting edge and ICES forms

Old slides

# How to represent variations?

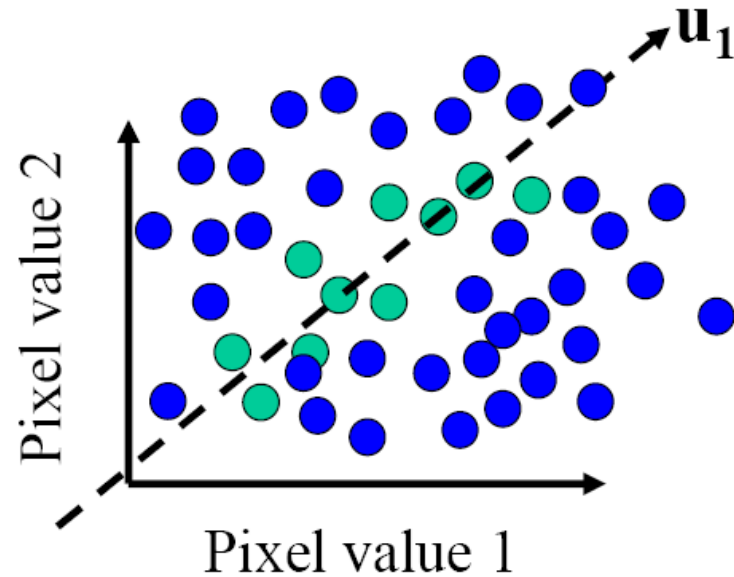
- Training images
- $\mathbf{x}_1, \dots, \mathbf{x}_N$





# 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

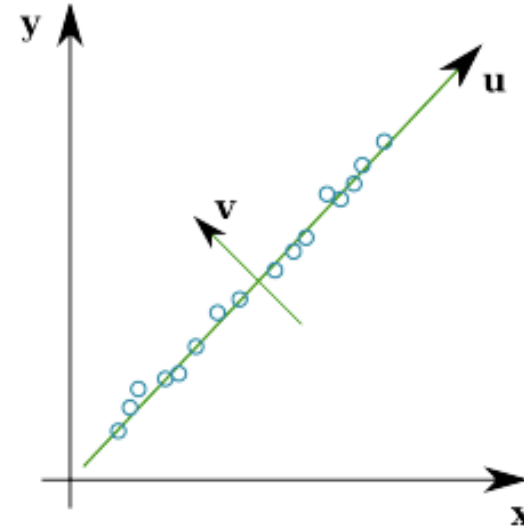
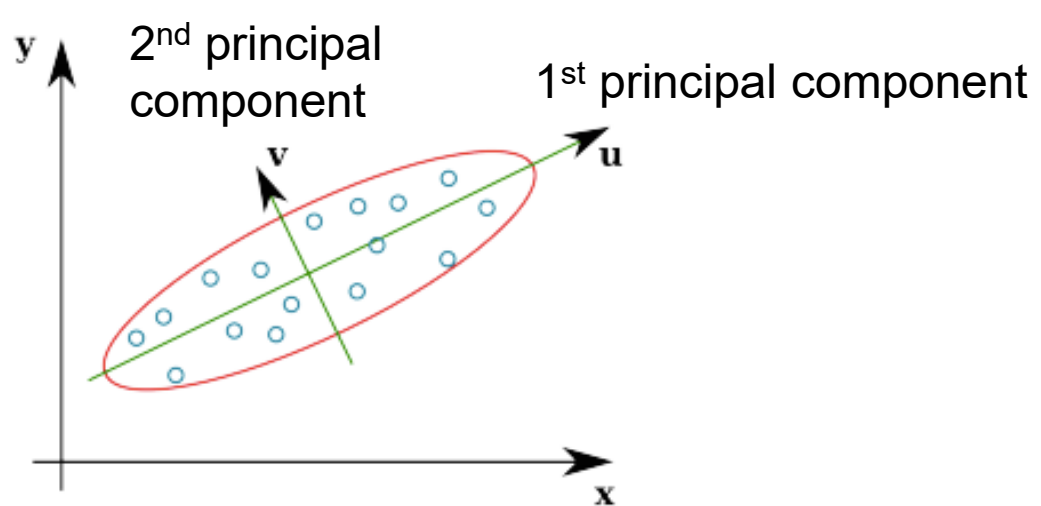
● A (non-face) image

# PCA

- General dimensionality reduction technique
  - Finds major directions of variation
- Preserves most of variance with a much more compact representation
  - Lower storage requirements (eigenvectors + a few numbers per face)
  - Faster matching/retrieval

# Principal Component Analysis

- Given a point set  $\{\vec{p}_j\}_{j=1\dots P}$ , in an  $M$ -dim space, PCA finds a basis such that
  - The most variation is in the first basis vector
  - The second most, in the second vector that is orthogonal to the first vector
  - The third...



# 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:

$$\begin{aligned} \text{Maximize} \quad & \frac{1}{N} \sum_{i=1}^N \underbrace{\mathbf{u}^T (\mathbf{x}_i - \mu)}_{\text{Projection of data point}} (\mathbf{u}^T (\mathbf{x}_i - \mu))^T \quad \text{subject to } \|\mathbf{u}\|=1 \\ & = \mathbf{u}^T \left[ \underbrace{\frac{1}{N} \sum_{i=1}^N (\mathbf{x}_i - \mu) (\mathbf{x}_i - \mu)^T}_{\text{Covariance matrix of data}} \right] \mathbf{u} \\ & = \mathbf{u}^T \Sigma \mathbf{u} \end{aligned}$$

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)

# PCA in MATLAB

```
x=rand(3,10);%10 3D examples
```

```
mu=mean(x,2);
```

```
x_norm = x-repmat(mu,[1 n]);
```

```
x_covariance = x_norm*x_norm';
```

```
[U, E] = eig(x_covariance)
```

U =

```
0.74 0.07 -0.66  
0.65 0.10 0.74  
-0.12 0.99 -0.02
```

E =

```
0.27 0 0  
0 0.63 0  
0 0 0.94
```

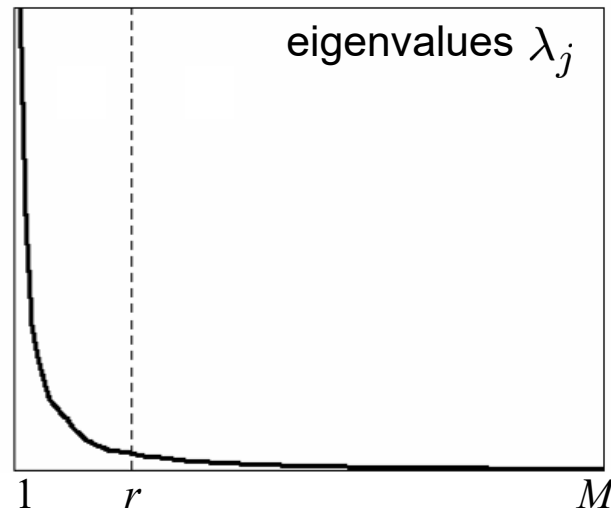


# Principal Component Analysis

First  $r < M$  basis vectors provide an approximate basis that minimizes the mean-squared-error (MSE) of reconstructing the original points

Choosing subspace dimension  $r$ :

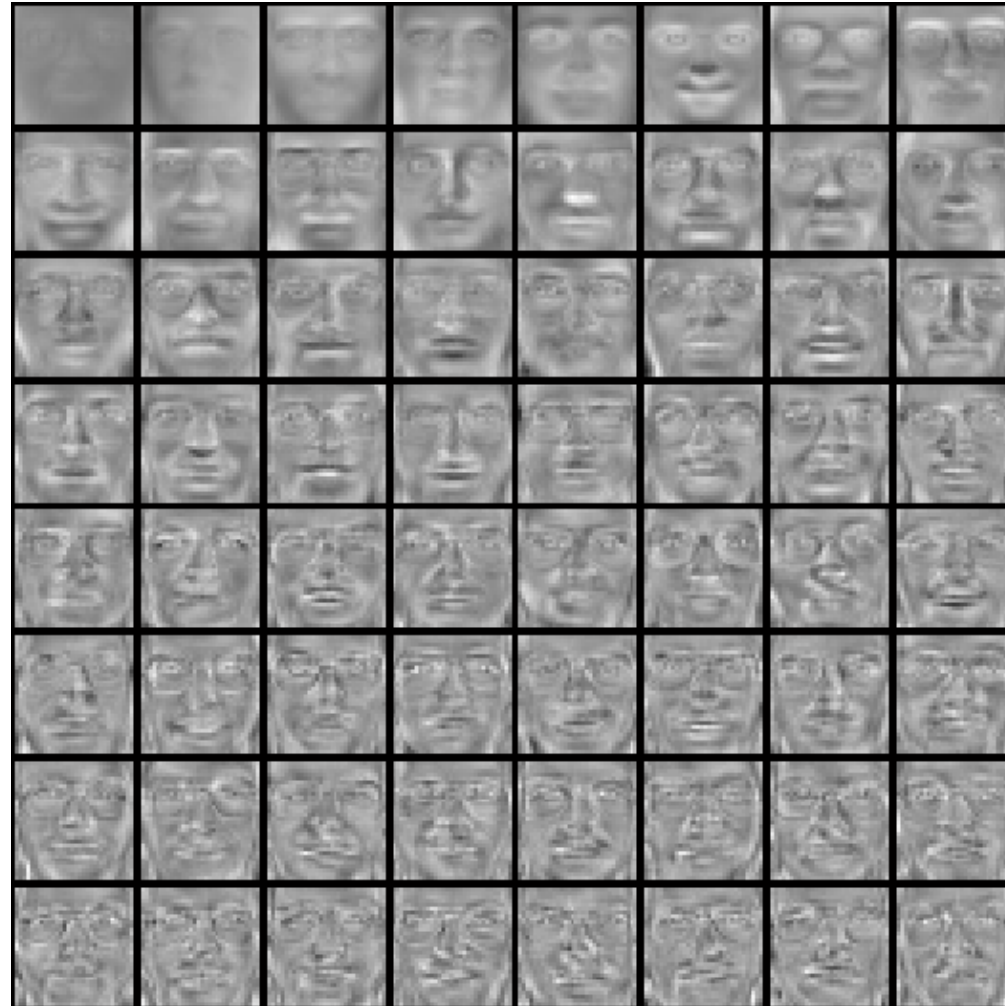
- look at decay of the eigenvalues as a function of  $r$
- Larger  $r$  means lower expected error in the subspace data approximation



# Eigenfaces example (PCA of face images)

Top eigenvectors:  $u_1, \dots, u_k$

Mean:  $\mu$



# Visualization of eigenfaces (appearance variation)

Principal component (eigenvector)  $u_k$



$\mu + 3\sigma_k u_k$



$\mu - 3\sigma_k u_k$



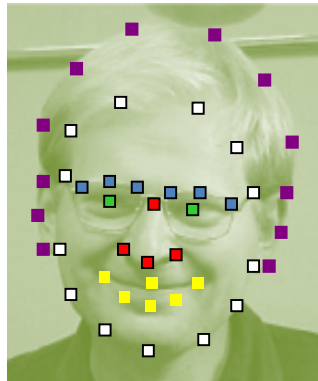
# Can represent face in appearance or shape space

Appearance  
Vector



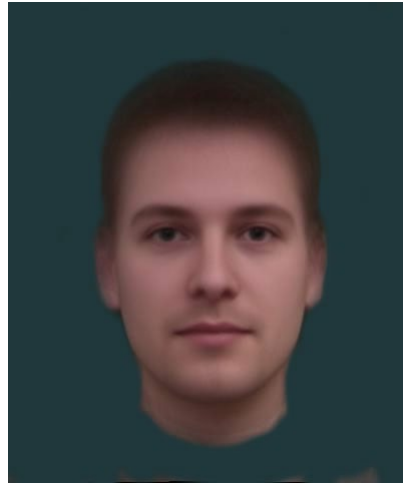
200\*150 pixels (RGB)

Shape  
Vector



43 coordinates (x,y)

# First 3 Shape Bases with PCA



Mean appearance

