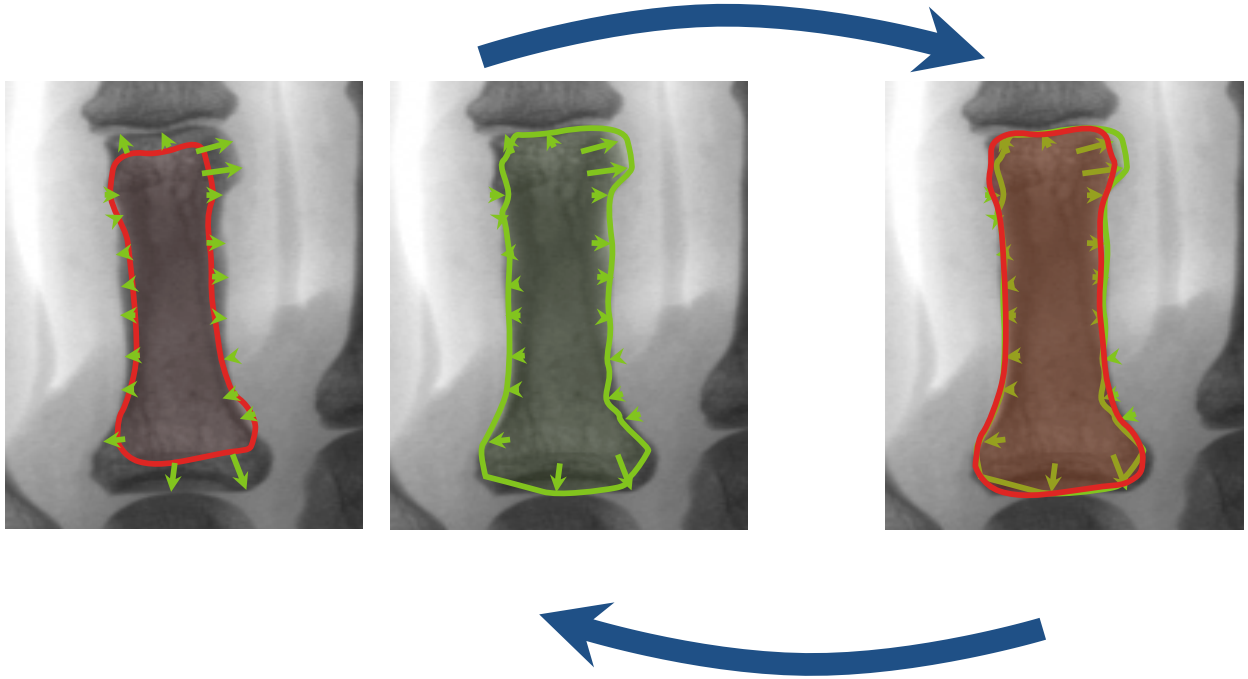
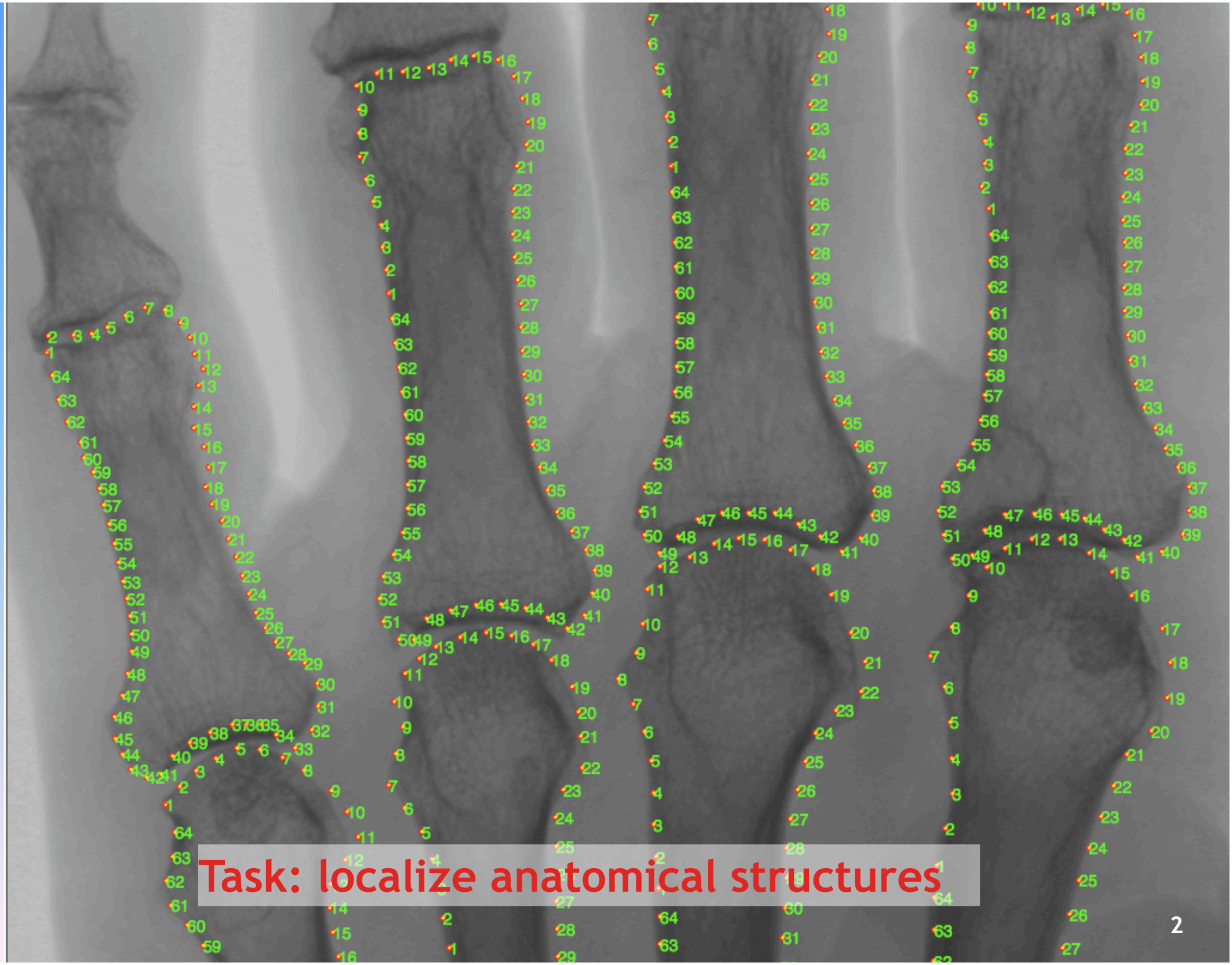
This lecture: models of shape and texture

- **Active Shape Models**
- Eigenfaces
- Active Appearance Models



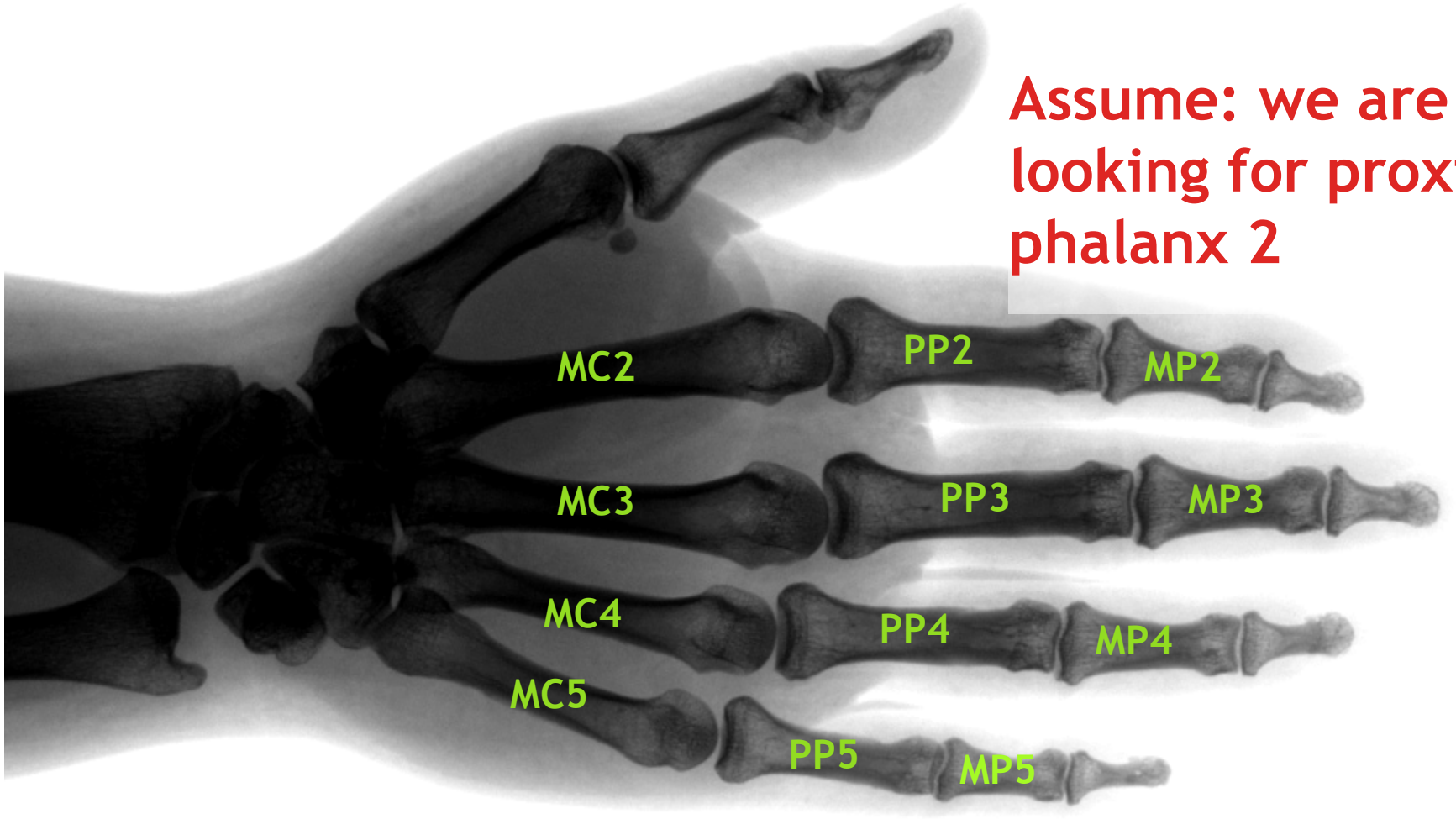
1



Task: localize anatomical structures

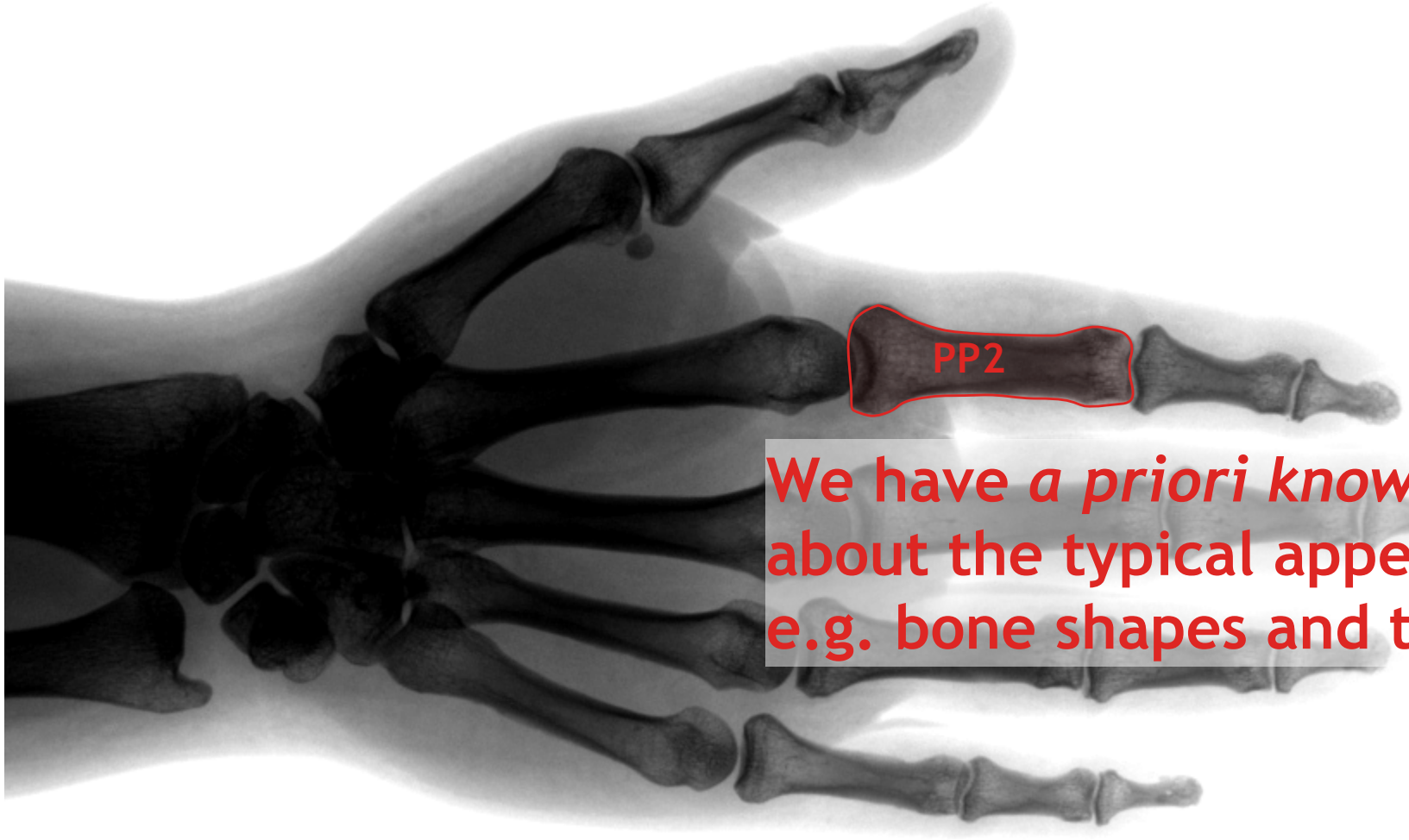
Task: Analyze a hand radiograph

Task: Analyze a hand radiograph



Assume: we are looking for proximal phalanx 2

Analyzing a hand radiograph

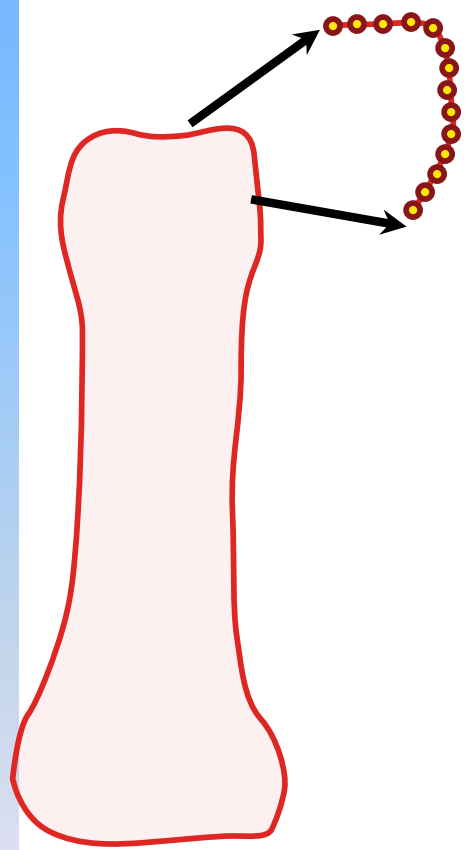


We have *a priori knowledge* about the typical appearance: e.g. bone shapes and texture

How can we represent this knowledge?

How can we exploit it?

How to capture *a priori knowledge*?



Simple shape representation: shape consists of a set of points defined by their individual coordinates

- very powerful
- very general
- very unspecific

Alternative:

- capture common properties of the bone
- find a representation that is restricted to plausible bones.

Prior knowledge: first works

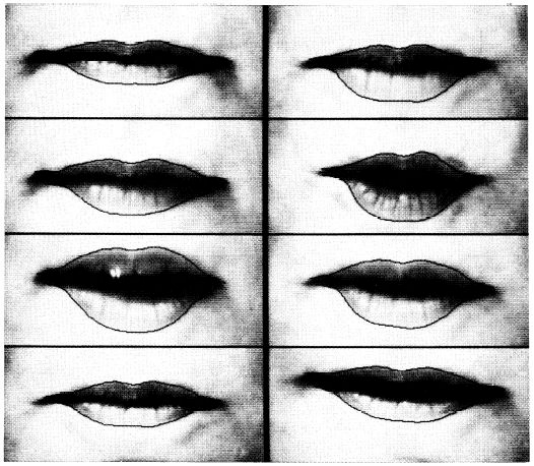
- Snakes

M. Kass, A. Witkin and D. Terzopoulos, `Snakes', ICCV 1987, IJCV 88

$$E^*_{\text{snake}} = \int_0^1 E_{\text{snake}}(\mathbf{v}(s))\, ds$$
$$= \int_0^1 E_{\text{int}}(\mathbf{v}(s)) + E_{\text{image}}(\mathbf{v}(s))$$
$$+ E_{\text{con}}(\mathbf{v}(s))\, ds$$

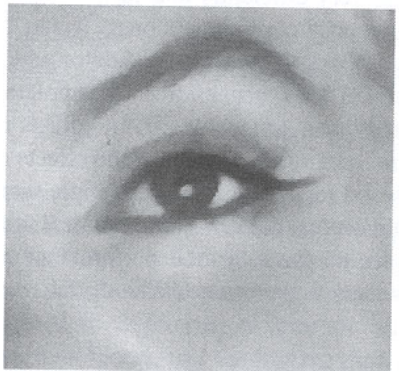
Data-driven

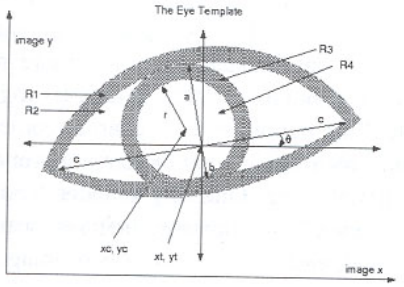
Prior knowledge



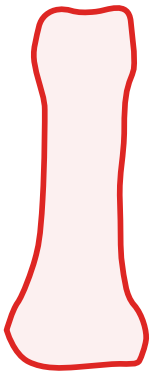
- Deformable templates

A.L.Yuille, D. Cohen, and P.Hallinan, "Feature extraction from faces using deformable templates", CVPR 89, IJCV 92.





Current work: Statistical Shape Models



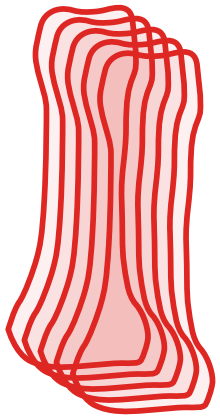
$$\mathbf{x}_i = \begin{pmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ x_m \\ y_m \end{pmatrix}$$

Each example is represented by a vector containing the coordinates of the landmarks.

Learning: Model Acquisition
Inference: Model Fitting

Model learning

- How can we learn what a bone looks like?



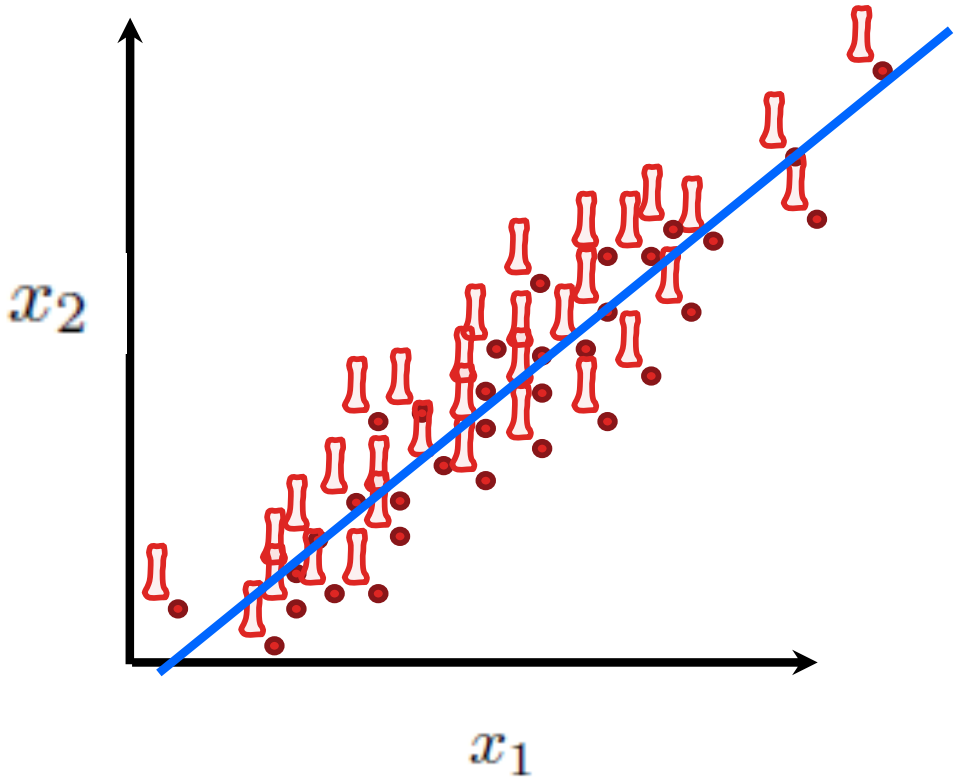
$\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ To build a statistical model of the data we need many examples.

- Objectives:
  - Expressiveness (model can account for all bones)
  - Compactness (as simple as possible)

- Method: Principal Components Analysis (PCA)

The space of all bone shapes

- Bone shapes: vectors in R^{2m}

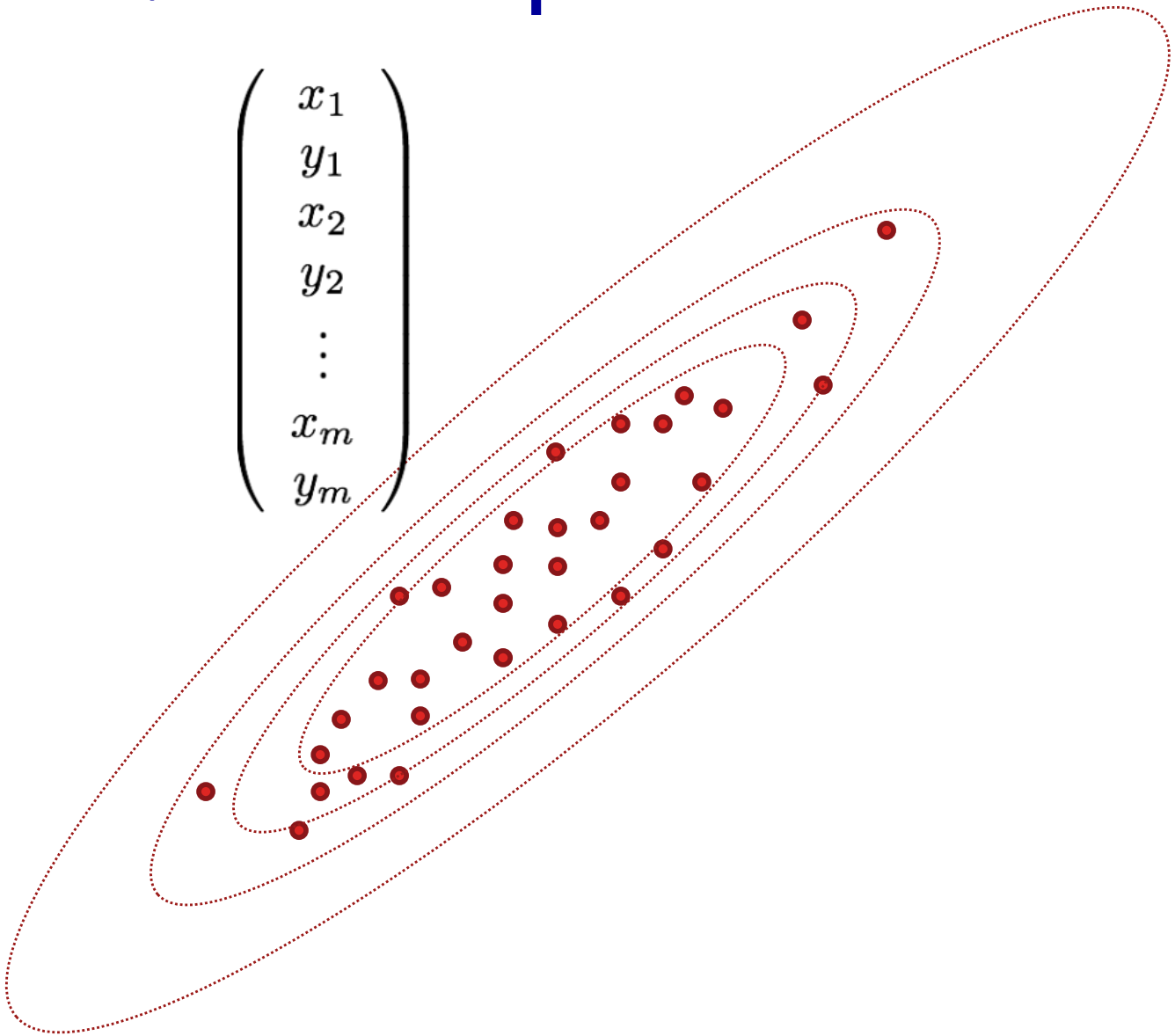


$$\mathbf{x}_i = \begin{pmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ x_m \\ y_m \end{pmatrix}$$

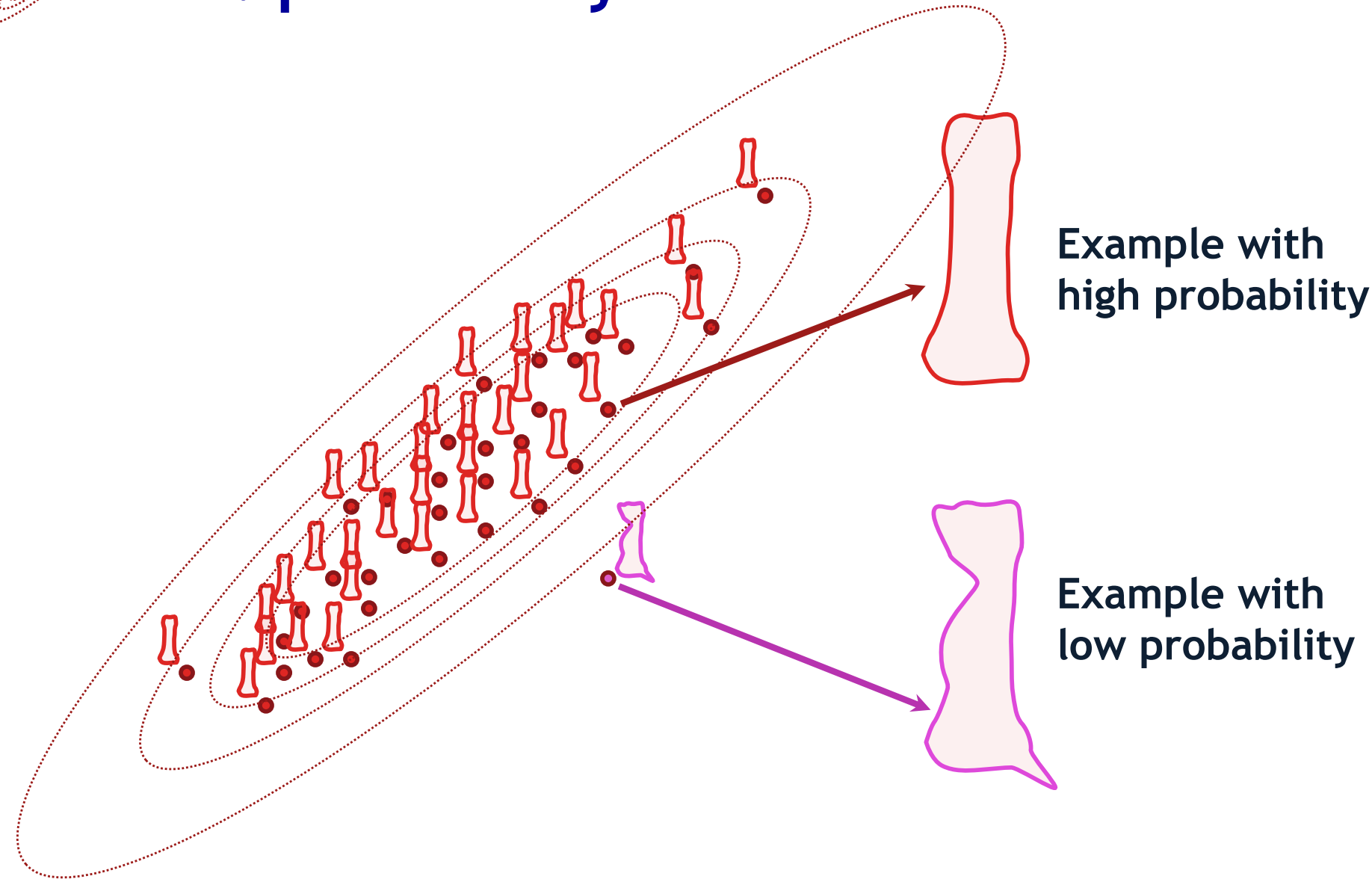
- Goal: project data onto a low-dimensional linear subspace that best explains their variation.

PCA: feature space

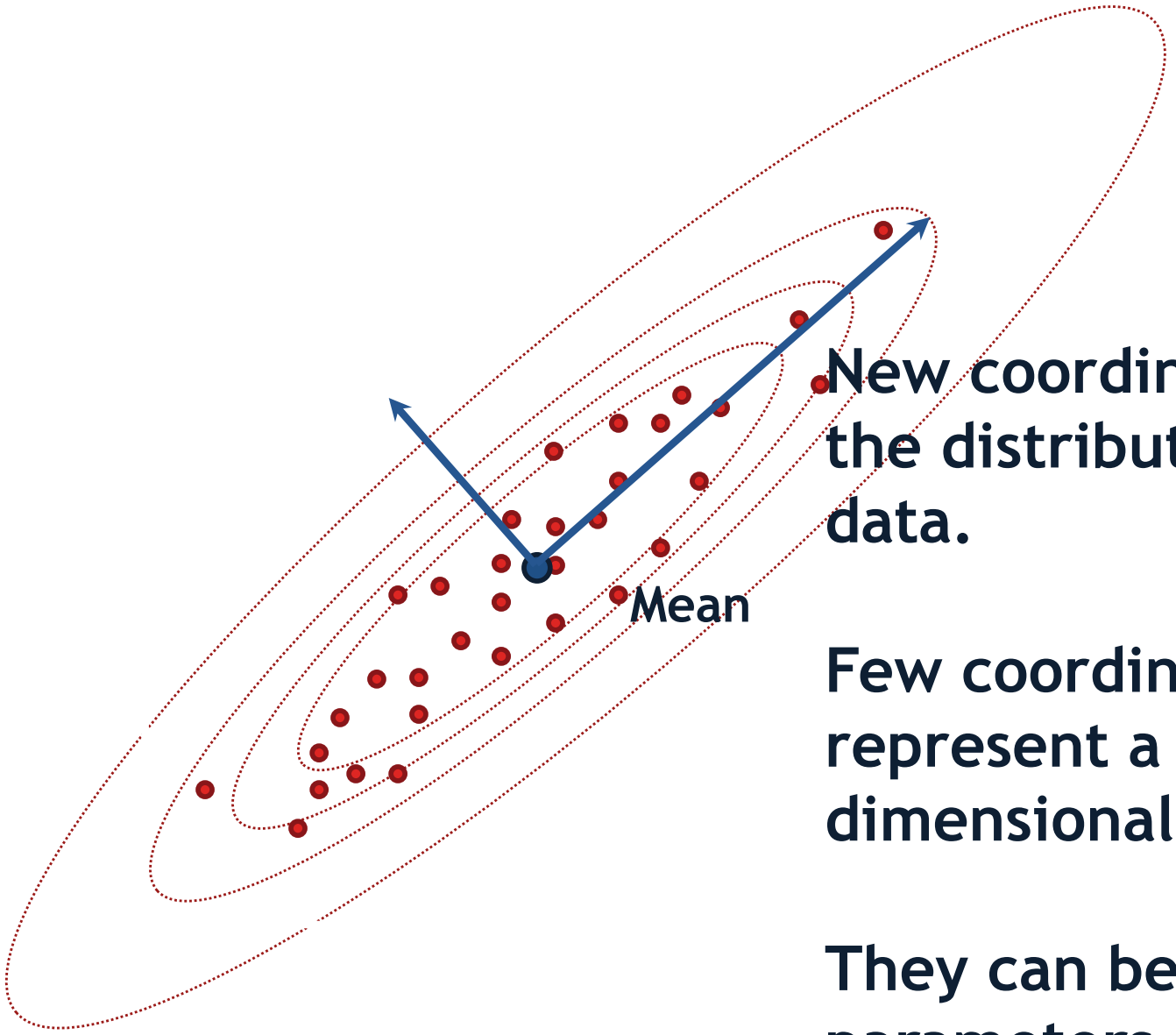
$$\begin{pmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ x_m \\ y_m \end{pmatrix}$$



PCA: probability distribution



New subspace: `better' coordinate system



New coordinates reflect the distribution of the data.

Few coordinates suffice to represent a high dimensional vector

They can be viewed as parameters of a model

Principal Component Analysis

- Given: N data points $x_1, \ldots, x_N$ in R^d
- We want to find a new set of features that are linear combinations of original ones:

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

(μ: mean of data points)

- What unit vector u in R^d captures the most variance of the data?

Principal Component Analysis

- **Variance of projection on u:**

$$var(u) = \frac{1}{N}\sum_{i=1}^{N} \mathbf{u}^{\mathrm{T}}(\mathbf{x}_i - \mu)(\mathbf{u}^{\mathrm{T}}(\mathbf{x}_i - \mu))^{\mathrm{T}}$$

Projection of data point

$$= \mathbf{u}^{\mathrm{T}}\left[\sum_{i=1}^{N}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^{\mathrm{T}}\right]\mathbf{u}$$

Covariance matrix of data

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

Direction: Unit norm vector

The direction that maximizes the variance: the eigenvector associated with the largest eigenvalue of Σ

Principal component analysis

- The direction that captures the maximum covariance of the data is the eigenvector corresponding to the largest eigenvalue of the data covariance matrix

- Furthermore, the top *k* orthogonal directions that capture the most variance of the data are the *k* eigenvectors corresponding to the *k* largest eigenvalues

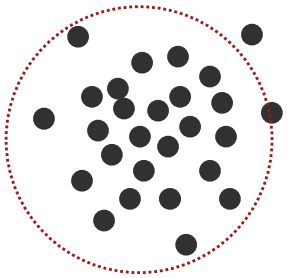
Covariance matrix reminder

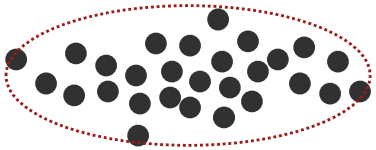
- **Covariance matrix:**

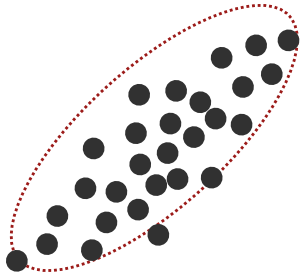
$$\Sigma_{i,j} = E\left((x_i - E(x_i))(x_j - E(x_j))\right)$$

$$\hat{\Sigma}_{i,j} = \frac{1}{N-1}\left[\sum_{i=1}^{N}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T\right] \qquad \mu = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$$

- **Uncorrelated coordinates: diagonal covariance**







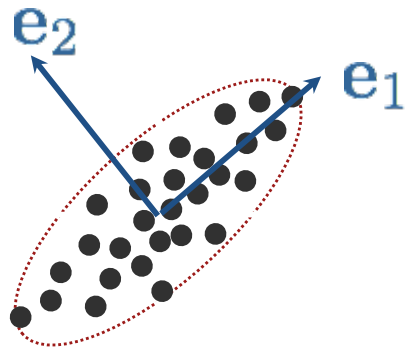
$$\Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 4 & 0 \\ 0 & 0.5 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 2.45 & 1.2 \\ 1.2 & 2.1 \end{pmatrix}$$

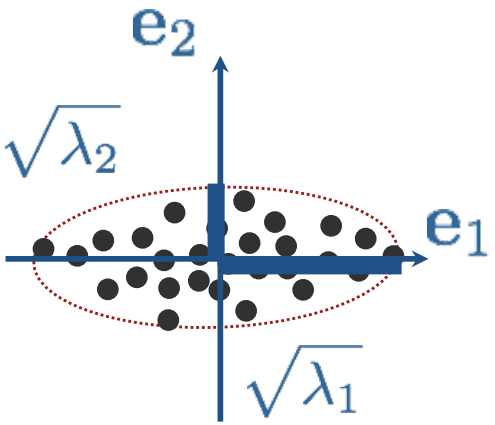
Height, Income

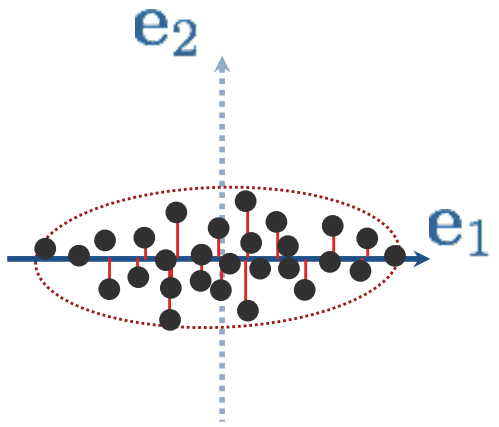
Height, Weight

PCA: decorrelation of random variables

- **PCA: projection onto eigenvectors of covariance matrix**







$$\Sigma = \begin{pmatrix} 2.45 & 1.2 \\ 1.2 & 2.1 \end{pmatrix} \qquad \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

Dimensionality reduction by using only leading eigenvectors

PCA: step by step

1. Compute the empirical mean and subtract it from the data, and compute empirical covariance matrix
2. Compute eigenvalue decomposition of the covariance matrix resulting in

$$\Lambda, E \quad \Sigma = E\Lambda E^T$$

3. Retain only the k eigenvectors with highest corresponding eigenvalues.

PCA: step by step

4. Given a new input vector

$$x_{new} \in R^m$$

5. project it into the eigenspace

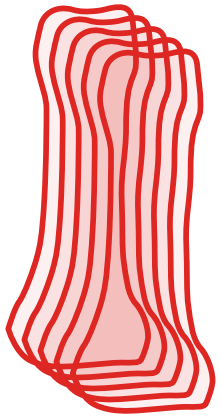
$$b_{new} = E_{[k]}^T (x_{new} - \mu) \in R^k$$

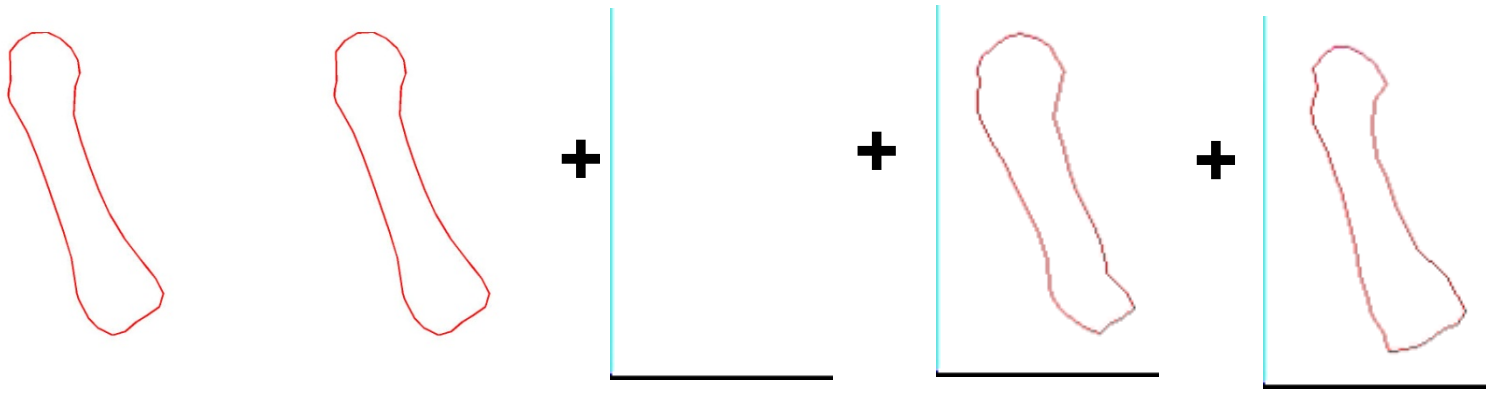
Only first k eigenvectors

6. resulting in the coefficient vector or parameter vector

$$b_{new}$$

Using PCA to model shape

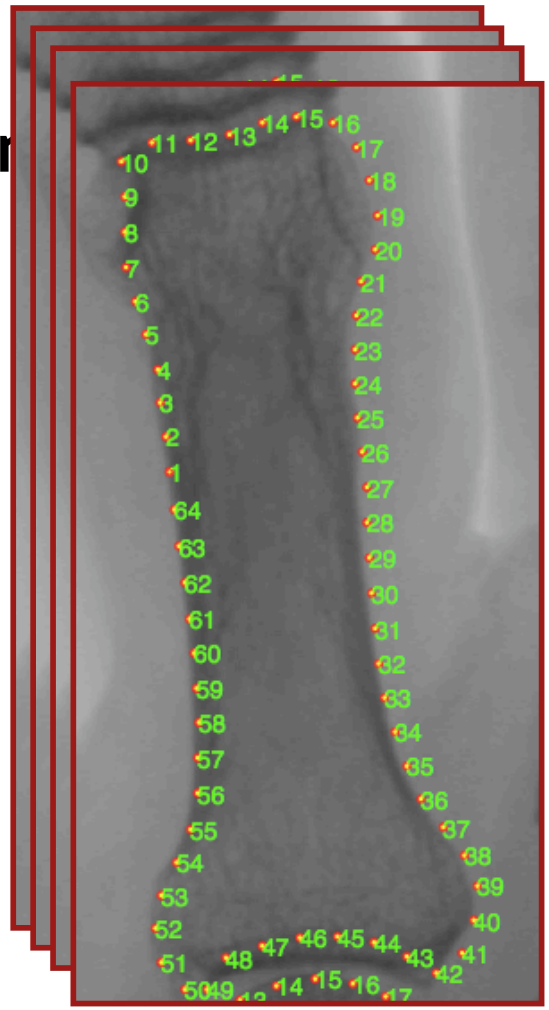




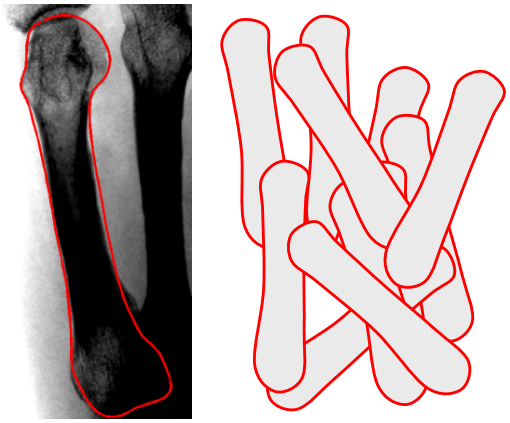
$$\mathbf{x}_{new} = \hat{\mathbf{m}} + b_1\mathbf{e}_1 + b_2\mathbf{e}_2 + b_3\mathbf{e}_3$$

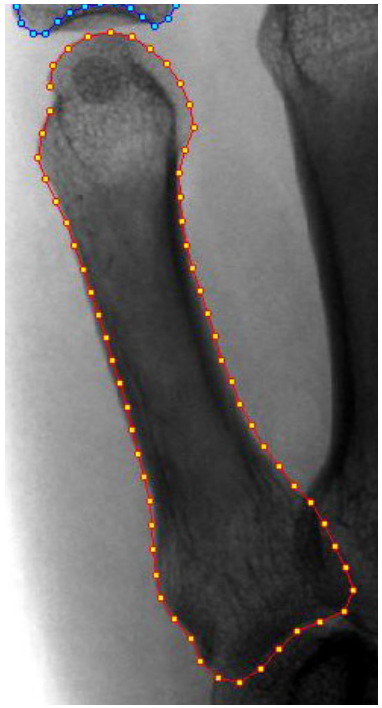
Active shape models (ASM)

- A set of training examples (images)
- A set of landmarks, that are present on all images
- Build a statistical model of shape variation (PCA)
- Build a statistical model of the local texture (PCA)
- Use the model for the search in a new image



ASM: model building



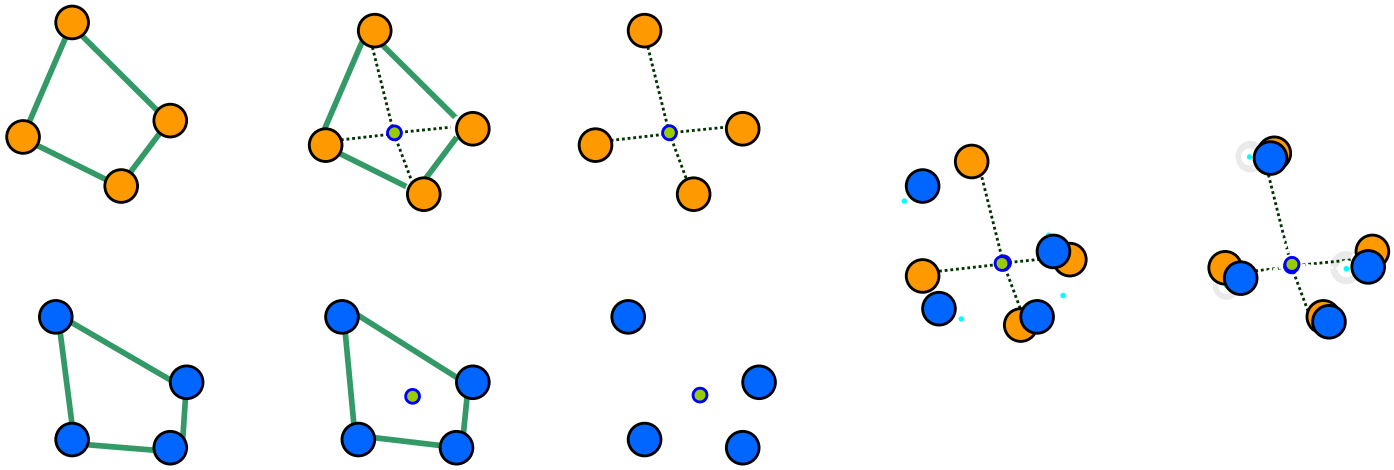


- shape
- texture
- orientation
- scale
- position

pose

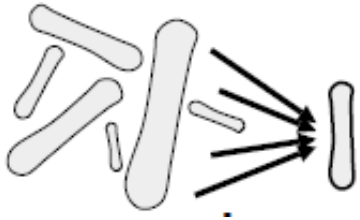
$$\mathbf{x}_i \in \mathbb{R}^{2m}$$

ASM: alignment of examples



- The alignment removes rotation, scale and translation differences
- Minimizes Procrustes distance

ASM: alignment of examples



- Center all training shapes $\mathbf{x}_i$
- Choose one example $\mathbf{x}_j$ as initial mean shape $\mathbf{x}_{mean}$ and scale $|\mathbf{x}_{mean}| = 1$

- Align all shapes $\mathbf{x}_i$ to $\mathbf{x}_{mean}$ by minimizing the Procrustes distance
- Recalculate the mean shape $\mathbf{x}_{mean}$ from the aligned set.
- Iterate until the mean shape converges

ASM: shape model

- Given: a set of aligned shapes $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- Perform PCA on the set of shapes, resulting in a set of Eigenvectors and corresponding Eigenvalues

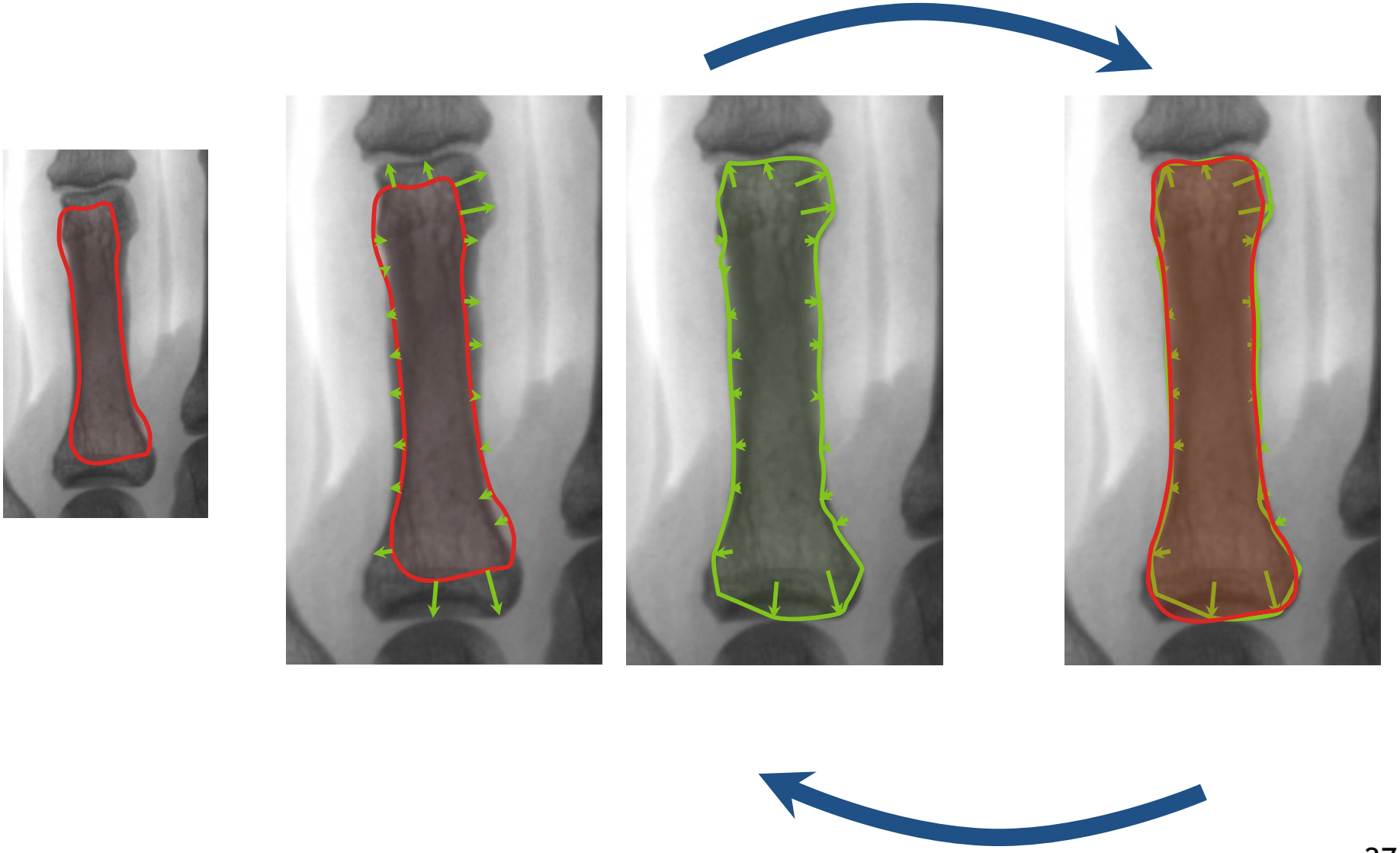
$$\mathbf{E} = (\mathbf{e}_1, \ldots, \mathbf{e}_m)$$

$$\mathbf{\Lambda} = (\lambda_1, \ldots, \lambda_m)$$

- Each shape can be reconstructed by

$$\mathbf{x} = \mathbf{E}\mathbf{b} + \mathbf{m}_{mean}$$

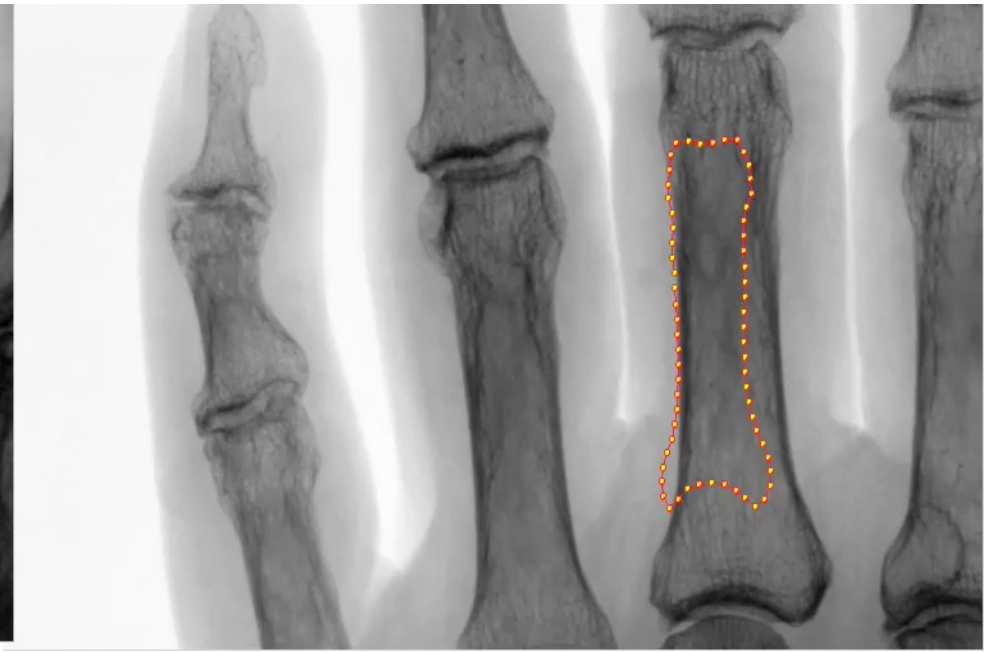
ASM search



ASM search: shape

- Update the instance according to shape model and pose
- Initialize $\mathbf{b} = 0$
- Generate model pose instance $\mathbf{x}' = \mathbf{E}\mathbf{b} + \mathbf{m}_{mean}$
- Find pose parameters to map $\mathbf{x}'$ to $\mathbf{x}$
- Project $\mathbf{x}$ into the model coordinates
- Update model parameters $\mathbf{b}$ to match $\mathbf{x}$
- Apply constraints $|b_i| < 3\sqrt{\lambda_i}$

ASM search



Modelling the appearance

- To peform a search with the shape model, we have to capture the appearance, too.
  - Local appearance at the landmark positions:

    Active shape model (ASM)
  - Appearance of the enclosed texture

    Active appearance model (AAM)

This lecture: models of shape and texture

- Active Shape Models
- **Eigenfaces**
- Active Appearance Models
- Class Projects

$$\hat{x} = \mu + w_1 u_1 + w_2 u_2 + w_3 u_3 + w_4 u_4 + \ldots$$

Face Recognition Problem

Face Recognition Problem

[Sinha and Poggio 2002]

Challenges: Image Variability

Short Term

Expression

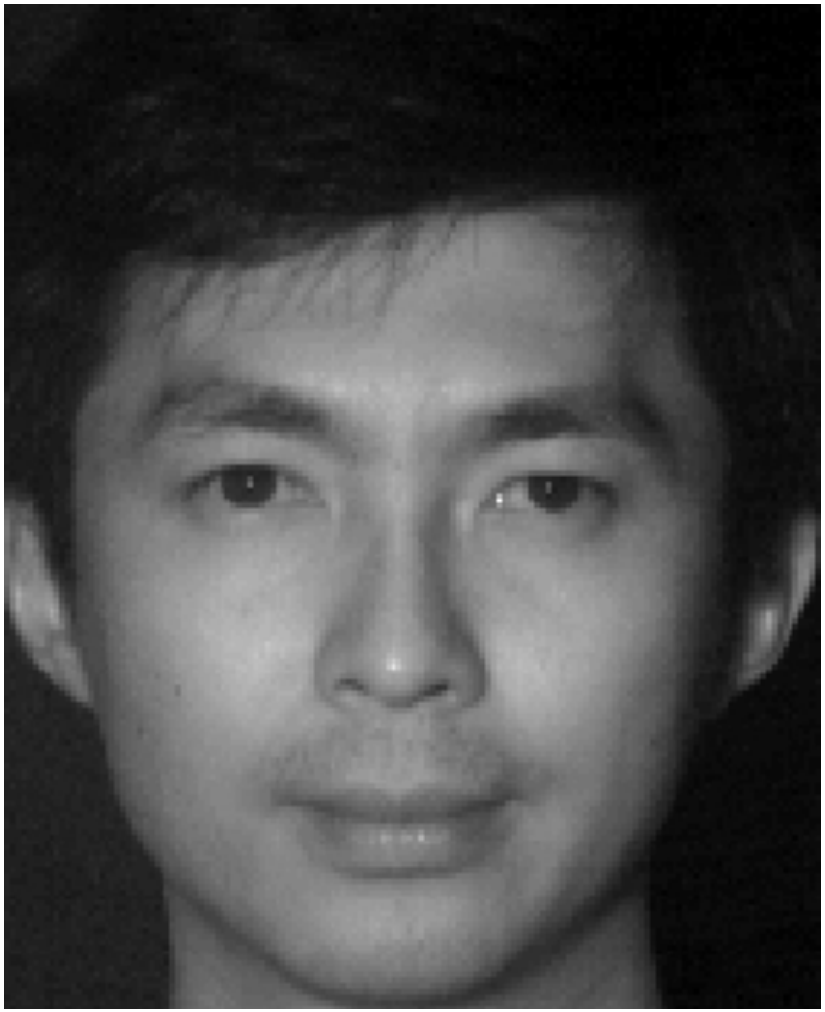
Pose

Illumination

Long Term

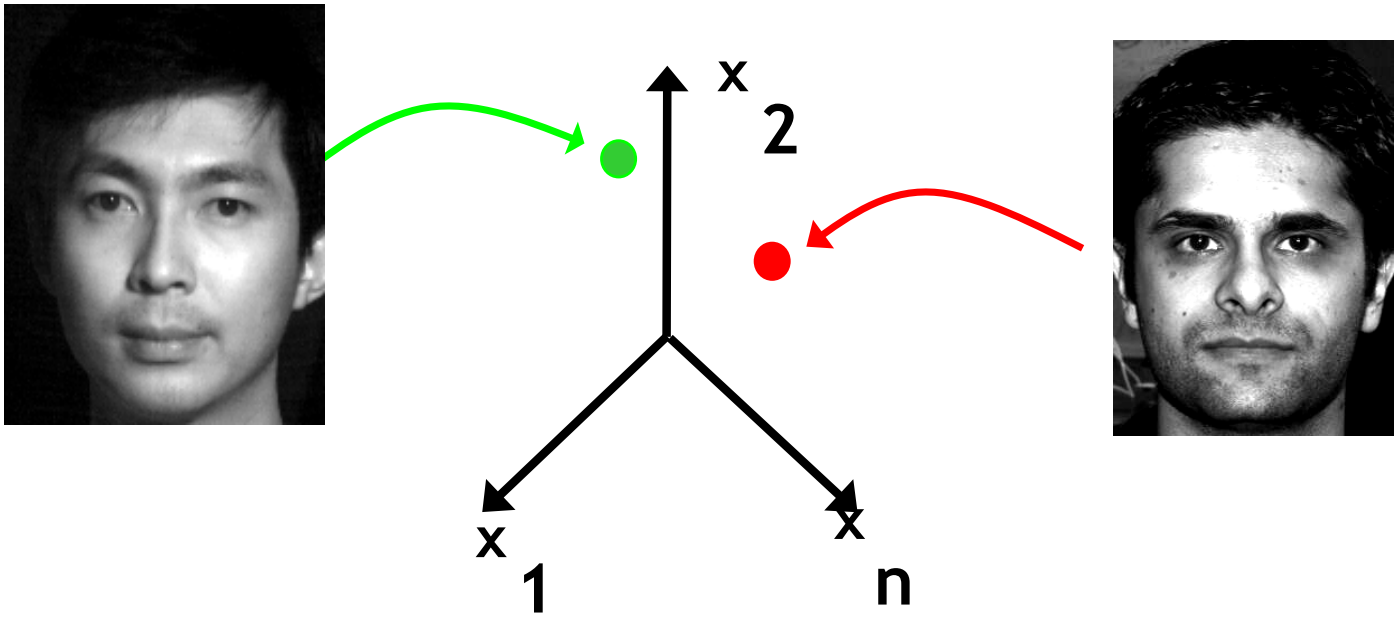
- Facial Hair
- Makeup
- Eyewear
- Hairstyle
- Piercings
- Aging

Invariance Problem



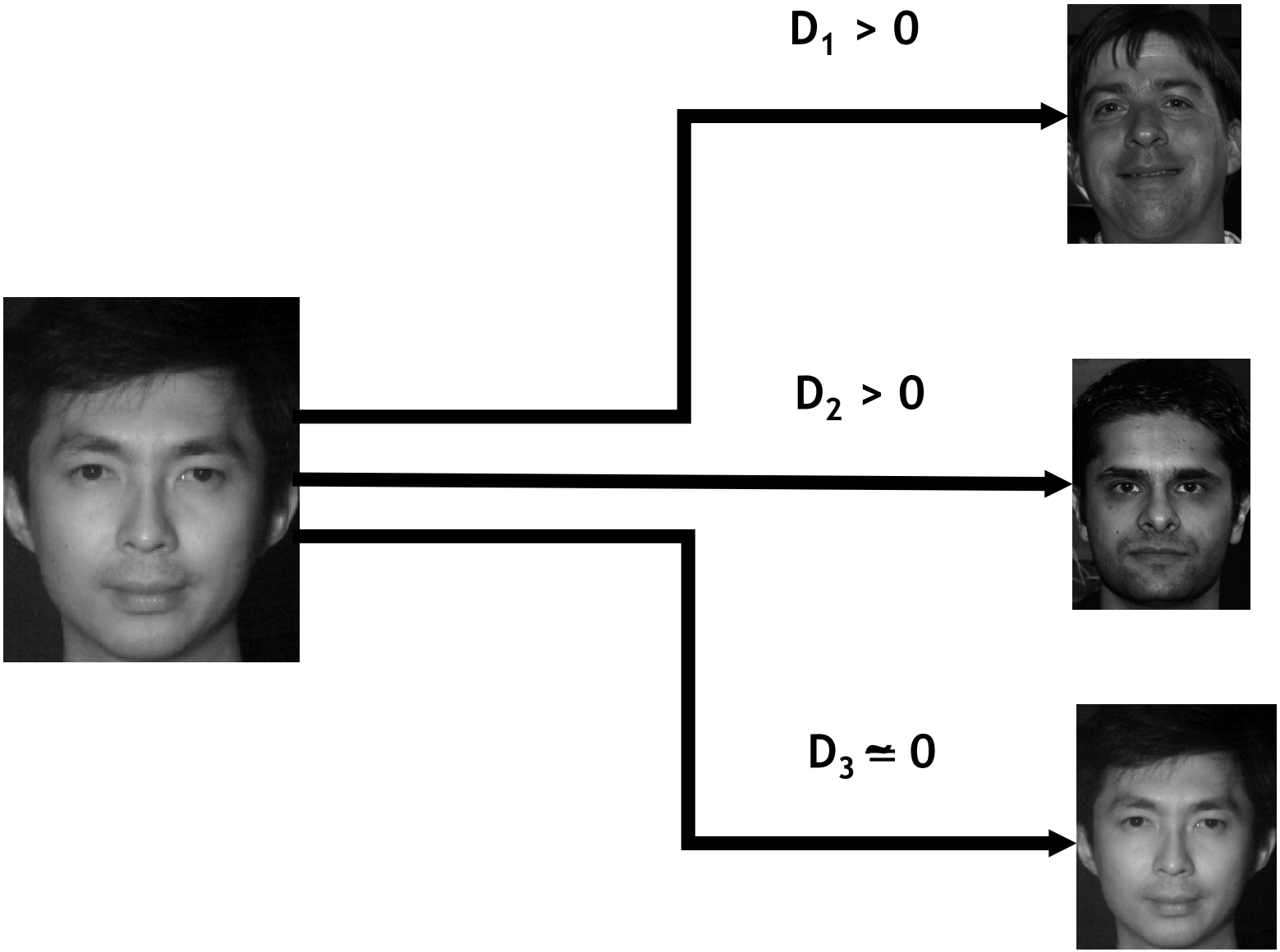
Invariance Problem

Images as Points in Euclidean Space

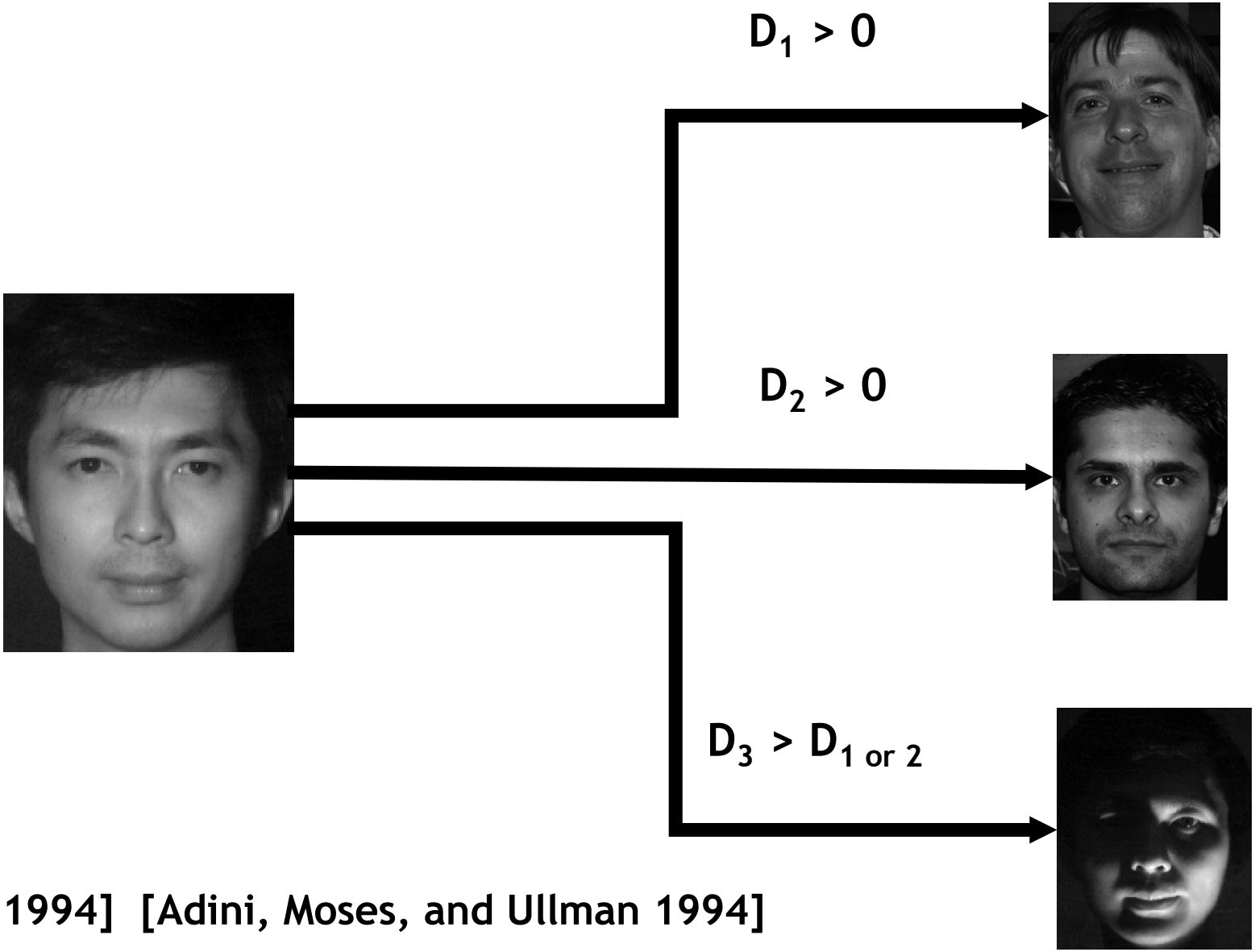


- Let an n-pixel image to be a point in an n-D space, $x \in R^n$.
- Each pixel value is a coordinate of x.

Face Recognition: Euclidean Distances



Face Recognition: Euclidean Distances

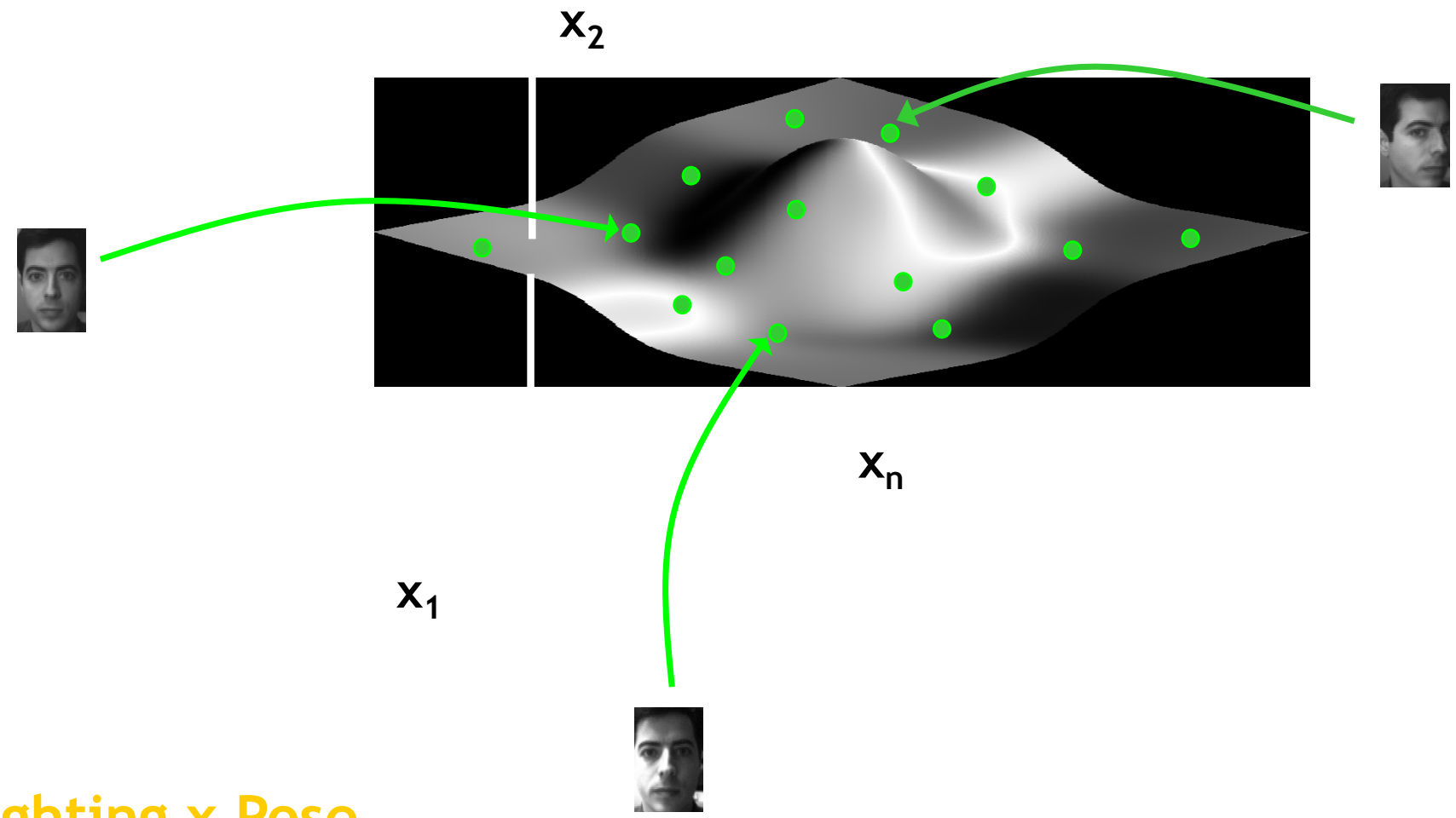


[Hallinan 1994] [Adini, Moses, and Ullman 1994]

Nearest Neighbor Classifier

- 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.
- Some solutions:
  - Bayesian classification
  - Dimensionality reduction

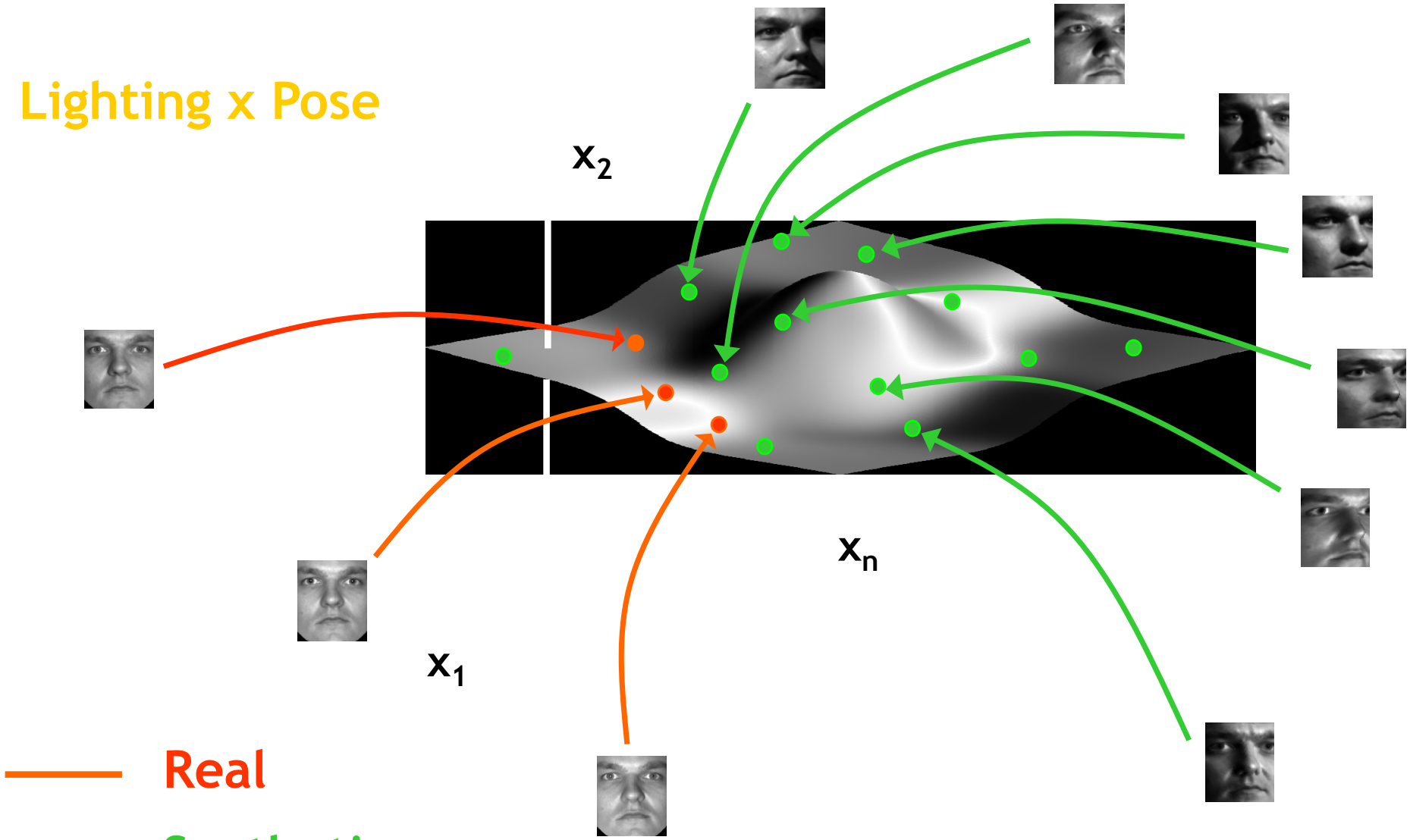
Image Variability: Appearance Manifolds



Lighting x Pose

[Murase and Nayar 1993]

Image Variability: From Few to Many



[Georghiades, Belhumeur, and Kriegman 1999]

Modeling Image Variability

Can we model illumination and pose variability in images of a face?

Yes, if we can determine the shape and texture of the face. But how?

Appearance modelling for faces

- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
- Very few vectors correspond to valid face images

- Original coordinates are not revealing about face properti
- We want to model the subspace of face images
- Same story as shape modelling….

Eigenfaces: Key idea

- Assume that most face images lie on a low-dimensional subspace determined by the first k ($k<d$) directions of maximum variance

- Use PCA to determine the vectors or "eigenfaces" $u_1,\ldots u_k$ that span that subspace

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

- Same idea as shapes

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

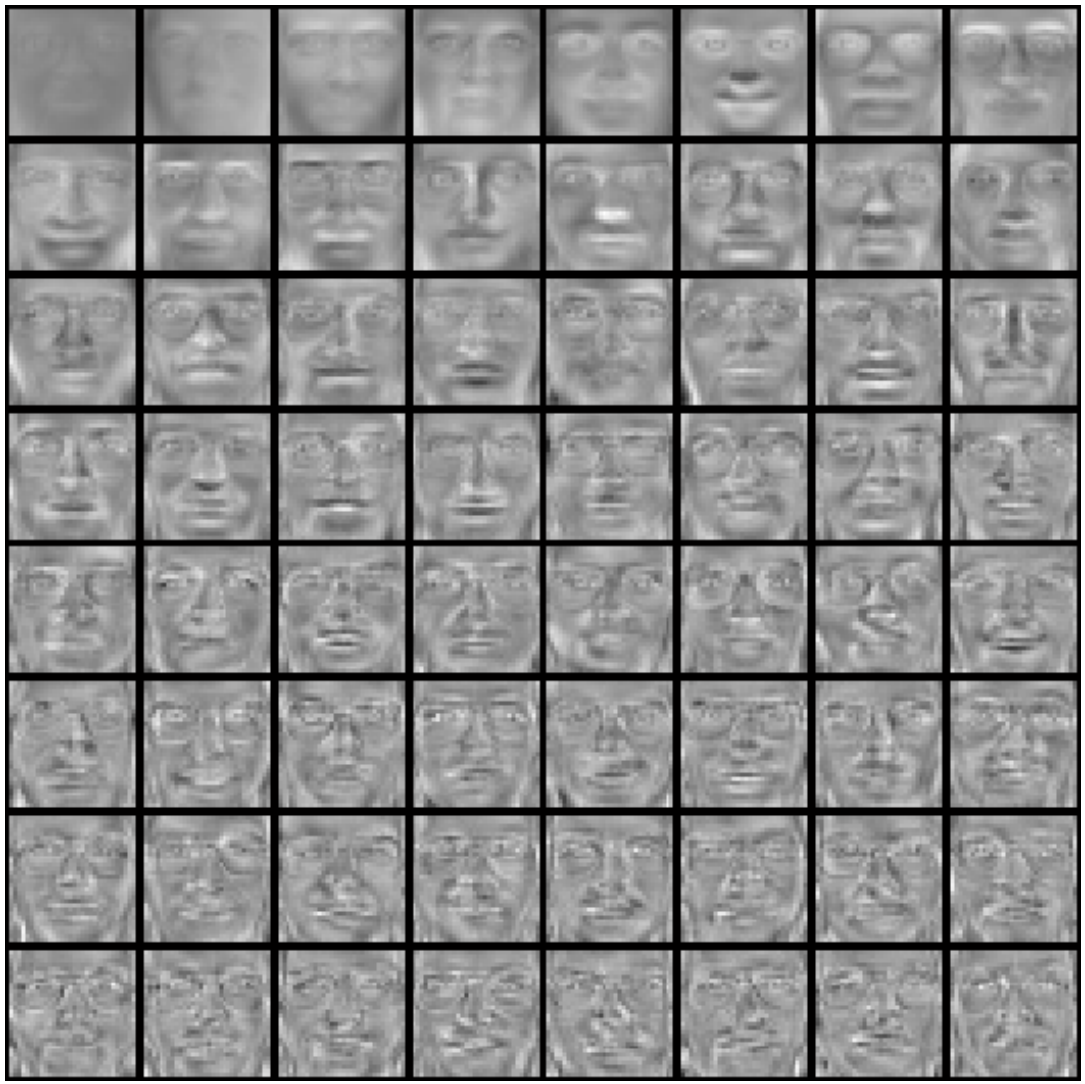
Eigenfaces example

- Training images
- $x_1,\ldots,x_N$

Eigenfaces example

Top eigenvectors: $\mathbf{u}_1,\ldots\mathbf{u}_k$

Mean: $\boldsymbol{\mu}$



Eigenfaces example

Principal component (eigenvector) u_k

$\mu + 3\sigma_k u_k$

$\mu - 3\sigma_k u_k$

Eigenfaces example

- Face x in “face space” coordinates:

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

Eigenfaces example

- Face x in “face space” coordinates:

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

- Reconstruction:

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

Recognition with eigenfaces

- **Process labeled training images:**
  - Find mean μ and covariance matrix Σ
  - Find k principal components (eigenvectors of Σ) $u_1,\ldots u_k$
  - Project each training image x_i onto subspace spanned by principal components:
    $(w_{i1},\ldots,w_{ik}) = (u_1^T(x_i - \mu), \ldots , u_k^T(x_i - \mu))$

- **Given novel image x:**
  - Project onto subspace:
    $(w_1,\ldots,w_k) = (u_1^T(x - \mu), \ldots , u_k^T(x - \mu))$
  - Check reconstruction error $x - \hat{x}$ to determine whether image is really a face
  - Classify as closest training face in k-dimensional subspace

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

Challenges addressed by Eigenfaces

Short Term

Expression

Pose

Illumination

Long Term

- Facial Hair
- Makeup
- Eyewear
- Hairstyle
- Piercings
- Aging

Limitations

- Global appearance method: not robust to misalignment, background variation



Offline Training Phase:
Input a set I of M labeled training images and produce a basis set B and a vector of coefficents for each image.

$I = \{I_1, I_2, \ldots, I_M\}$ is the set of training images. (input)
$B = \{F_1, F_2, \ldots, F_m\}$ is the set of basis vectors. (output)
$A_j = [a_{j1}, a_{j2}, \ldots, a_{jm}]$ is the vector of coefficients for image I_j. (output)

1. $I_{mean} = mean(I)$.

2. $\Phi = \{\Phi_i | \Phi_i = I_i - I_{mean}\}$, the set of difference images

3. Σ_Φ = the covariance matrix obtained from Φ.

4. Use the principal components method to compute eigenvectors and eigenvalues of Σ_Φ. (see text)

5. Construct the vector **B** as the basis set by selecting the most significant m eigenvectors; start from the largest eigenvalue and continue in decreasing order of the eigenvalues to select the corresponding eigenvectors.

6. Represent each training image I_j by a linear combination of the basis vectors: $I_j^m = a_{j1}F_1 + a_{j2}F_2 + \ldots + a_{jm}F_m$

Online Recogniton Phase:
Input the set of basis vectors B, the database of coefficient sets $\{A_j\}$, and a test image I_u. Output the class label of I_u.

1. Compute vector of coefficients $A_u = [a_{u1}, a_{u2}, \ldots, a_{um}]$ for I_u;

2. Find the h nearest neighbors of vector A_u in the set $\{A_j\}$;

3. Decide the class of I_u from the labels of the h nearest neighbors (possibly reject in case neighbors are far or inconsistent in labels);

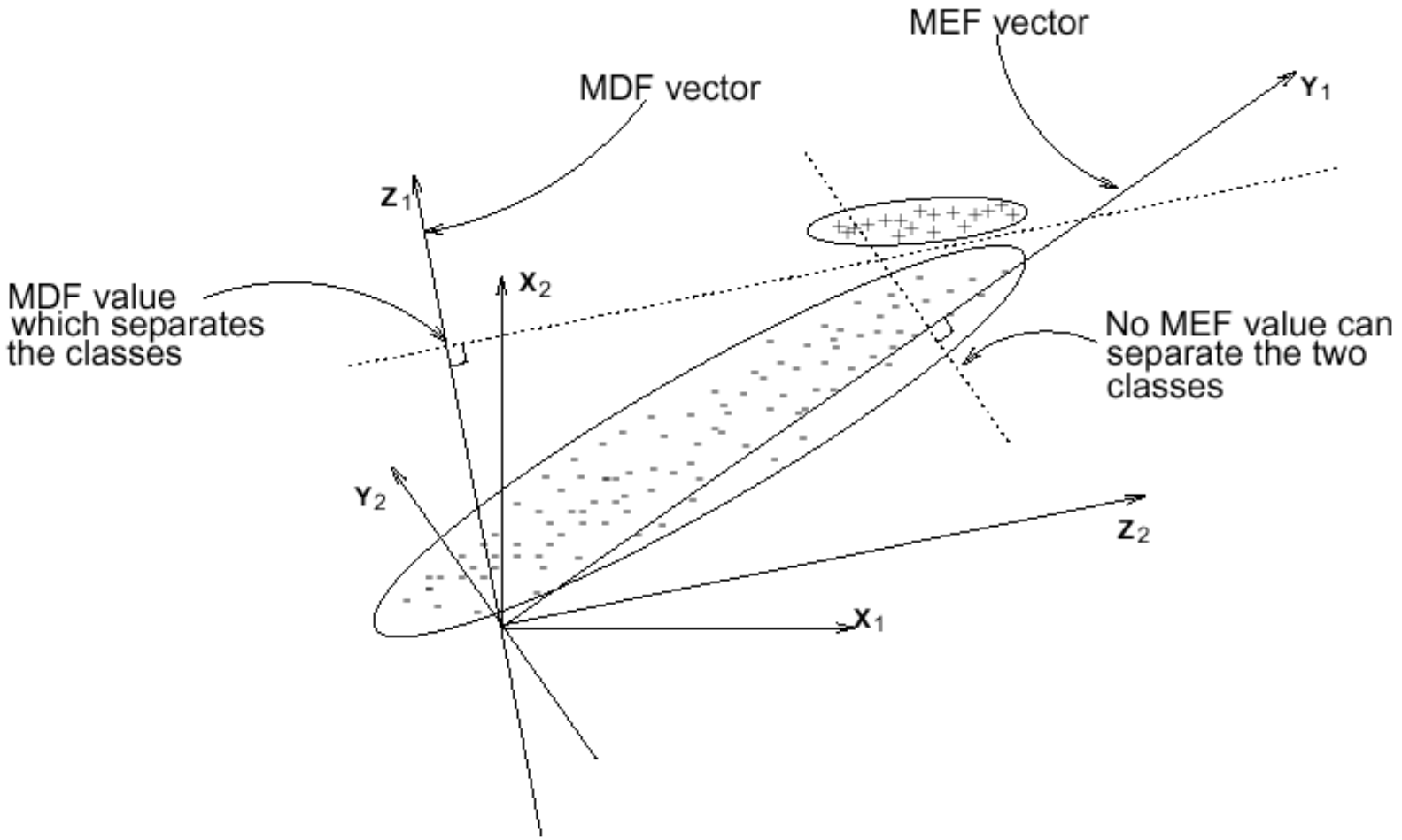


Figure 14.40: The most expressive features determined by the eignevectors of the scatter matrix represent the data well, but may not represent the differences between classes well. Discriminant analysis can be used to find subspaces that emphasize differences between classes. (Figure contributed by J. Swets and J. Weng.)

Linear Discriminant Analysis

- **We wish to choose linear functions of the features that allow good discrimination.**
  - **Assume class-conditional covariances are the same**
  - **Want linear feature that maximises the spread of class means for a fixed within-class variance**

Algorithm 22.6: Canonical variates identifies a collection of linear features that separating the classes as well as possible.

Assume that we have a set of data items of g different classes. There are n_k items in each class, and a data item from the k'th class is $\boldsymbol{x}_{k,i}$, for $i \in \{1, \ldots, n_k\}$. The j'th class has mean $\boldsymbol{\mu}_j$. We assume that there are p features (i.e. that the $\boldsymbol{x}_i$ are p-dimensional vectors).

Write $\overline{\boldsymbol{\mu}}$ for the mean of the class means, i.e.

$$\overline{\boldsymbol{\mu}} = \frac{1}{g}\sum_{j=1}^{g} \boldsymbol{\mu}_j$$

Write

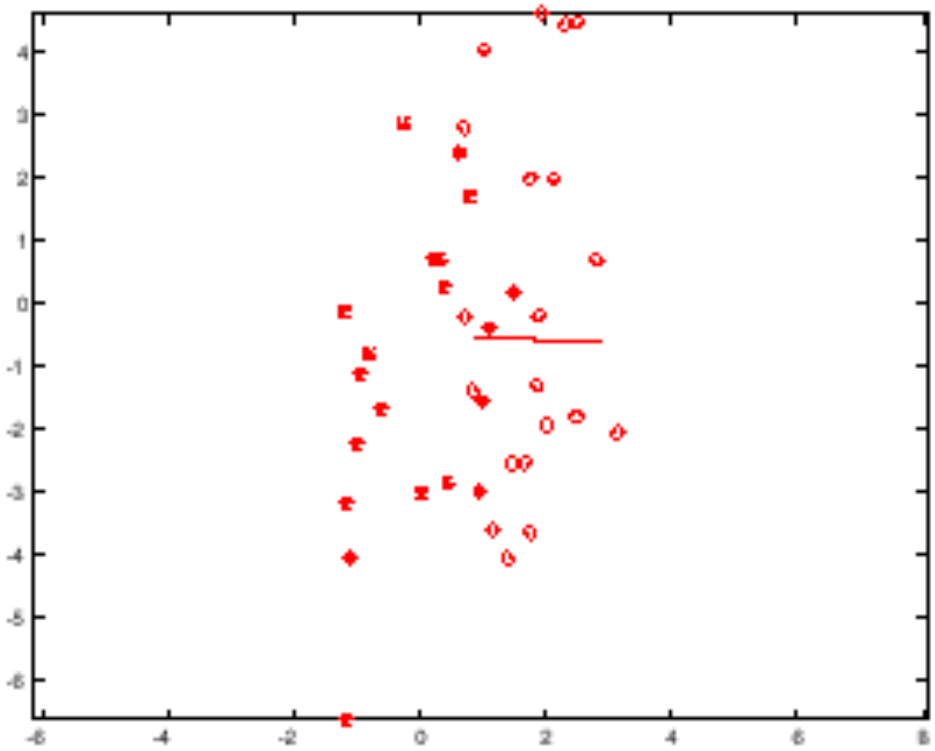
$$\mathcal{B} = \frac{1}{g-1}\sum_{j=1}^{g} (\boldsymbol{\mu}_j - \overline{\boldsymbol{\mu}})(\boldsymbol{\mu}_j - \overline{\boldsymbol{\mu}})^T$$

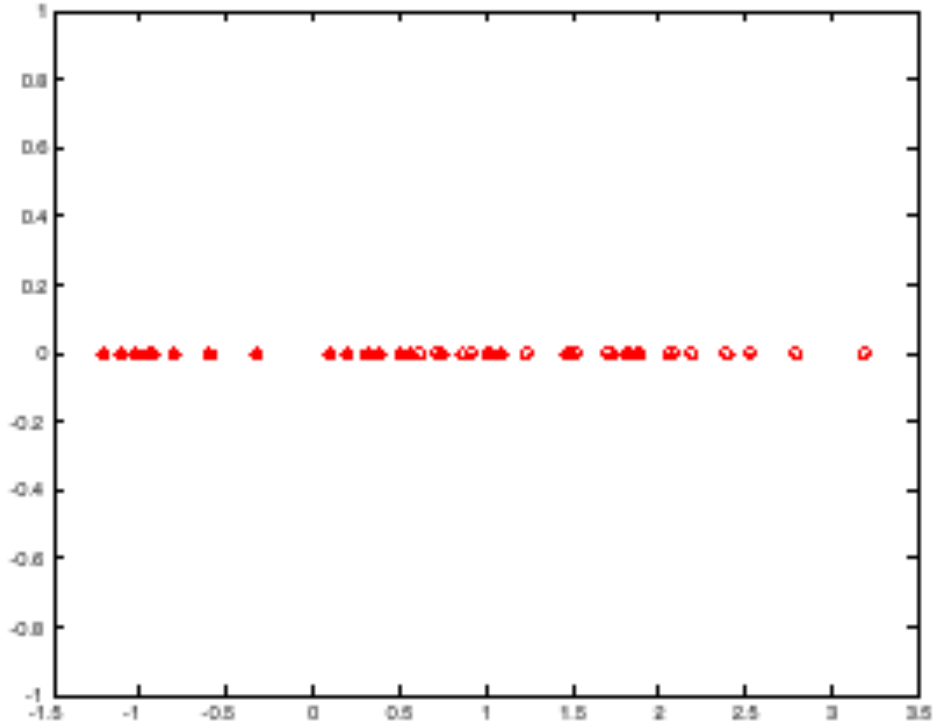
Assume that each class has the same covariance Σ, which is either known or estimated as

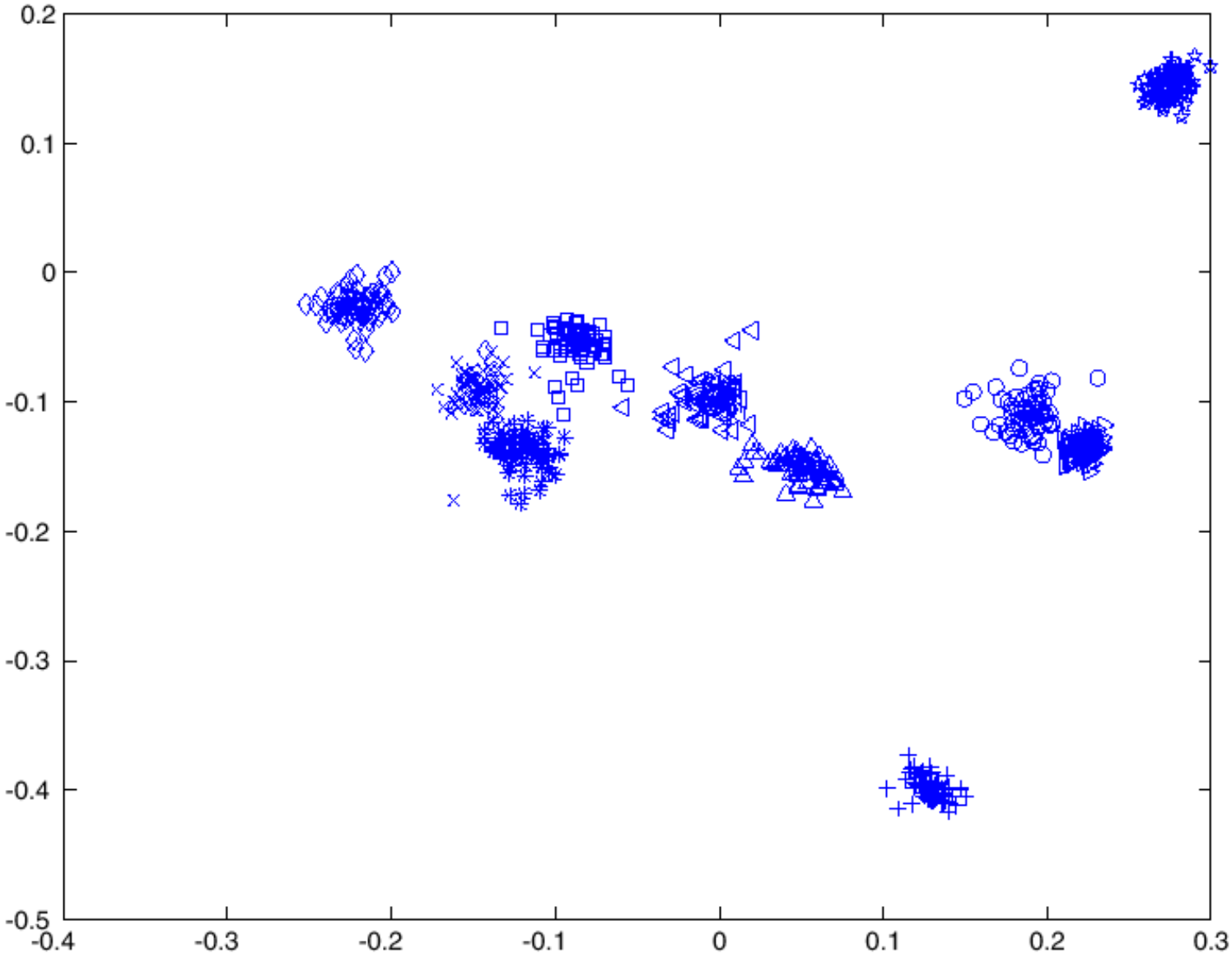
$$\Sigma = \frac{1}{N-1}\sum_{c=1}^{g} \left\{ \sum_{i=1}^{n_c} (\boldsymbol{x}_{c,i} - \boldsymbol{\mu}_c)(\boldsymbol{x}_{c,i} - \boldsymbol{\mu}_c)^T \right\}$$

The unit eigenvectors of $\Sigma^{-1}\mathcal{B}$ — which we write as $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_d$, where the order is given by the size of the eigenvalue and $\boldsymbol{v}_1$ has the largest eigenvalue — give a set of features with the following property:

- Projection onto the basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ gives the k-dimensional set of linear features that best separates the class means.


4
3
2
1
0
-1
-2
-3
-4
-5
-6
-6
-4
-2
0
2
4
6
8


1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
3
3.5


0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3

This lecture: models of shape and texture

- Active Shape Models
- Eigenfaces
- **Active Appearance Models**
- Class Projects

Challenges: Image Variability

Short Term

Expression

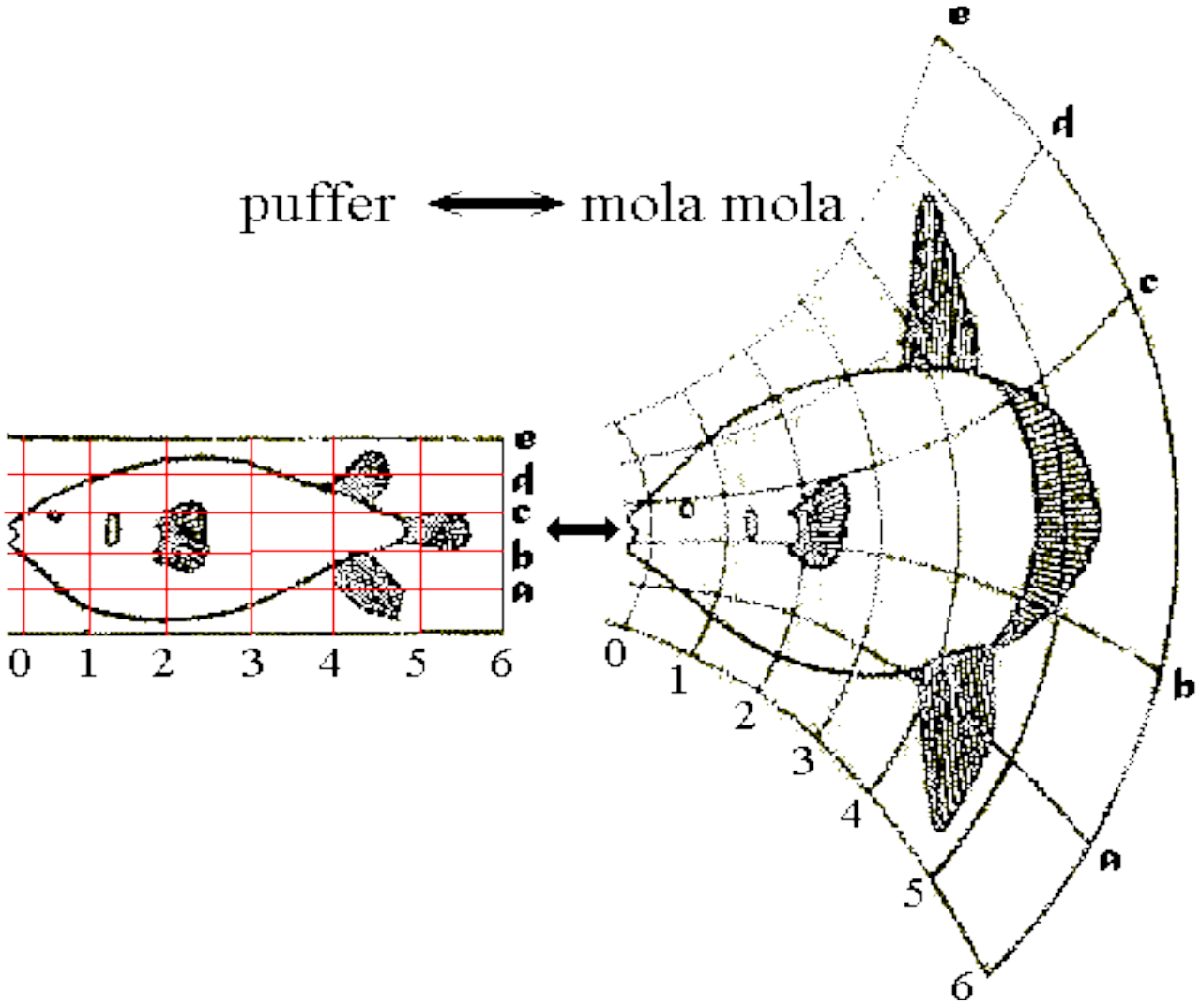
Pose

Illumination

Long Term

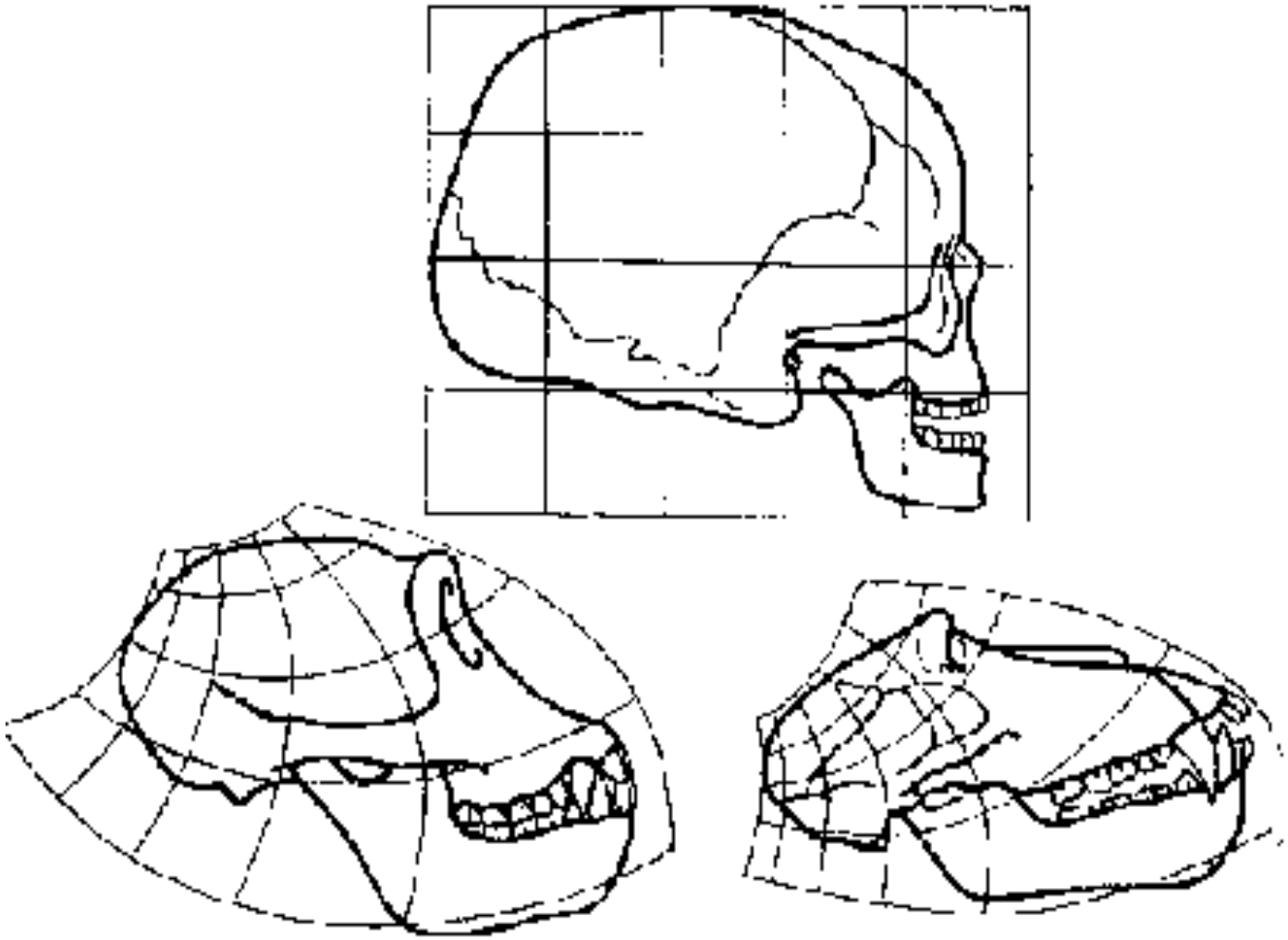
- Facial Hair
- Makeup
- Eyewear
- Hairstyle
- Piercings
- Aging

Modelling Shape Variability via Deformations



D' Arcy Thompson, "On Growth and Form" (1917)

Modelling Shape Variability via Deformations



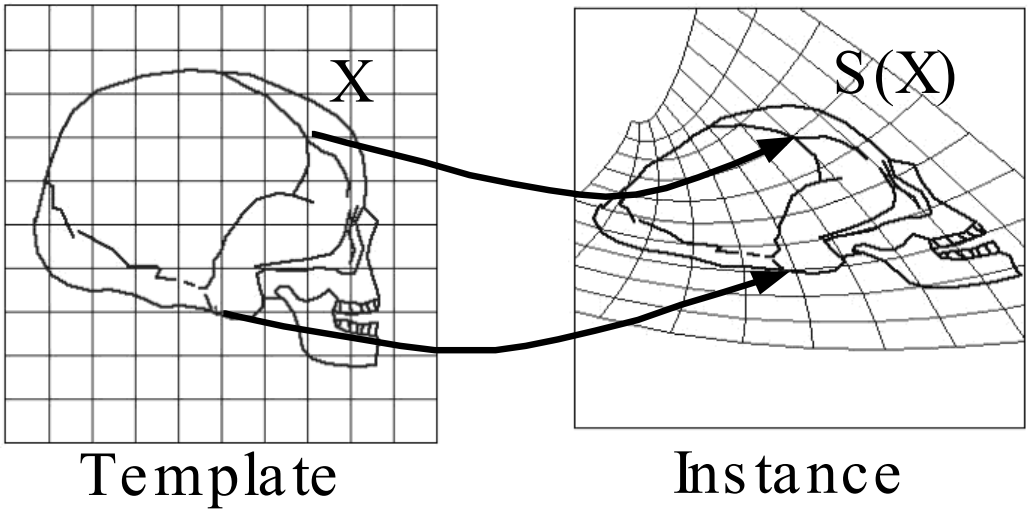
Skulls of a human, a chimpanzee and a baboon
and transformations between them

D' Arcy Thompson, "On Growth and Form" (1917)

Active Appearance Models

- *Appearance* = *Shape* + *Texture*
- *Shape*: deformation from `template' domain to instance domain

$$I(S(x)) = T(x)$$

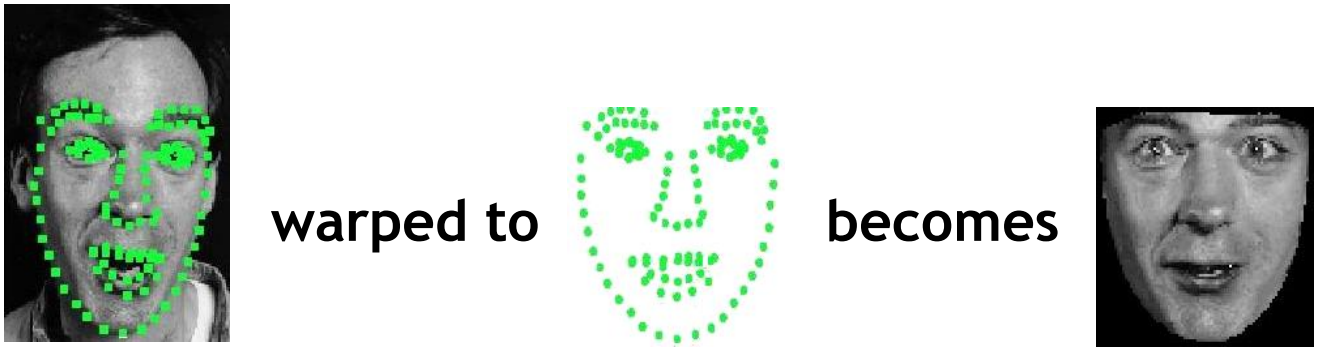


- *Texture*: intensity (or color) patch of an image in the template domain.

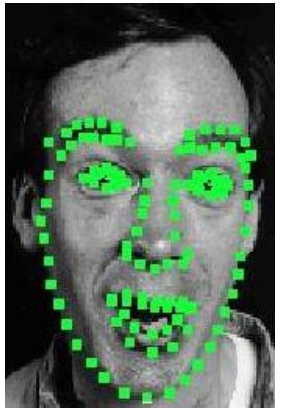
Shape-free texture

- An attempt to eliminate texture variation due to shape
- Given instance I and a reference "template" T we warp our image so that points of *I* move into the corresponding points of *T*

I *T*



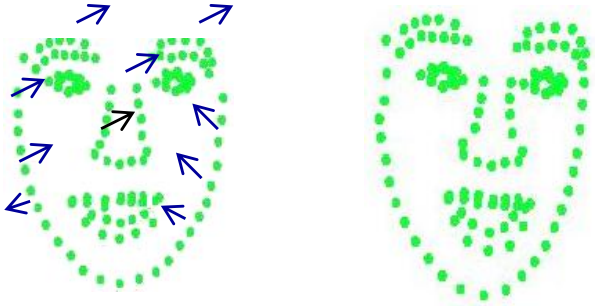
AAM training



- Training set: annotated grayscale images

Shape: displacements from mean shape

Texture (shape-free)



PCA

Optional:
Interpolate to recover dense deformation field

PCA

Model of shape

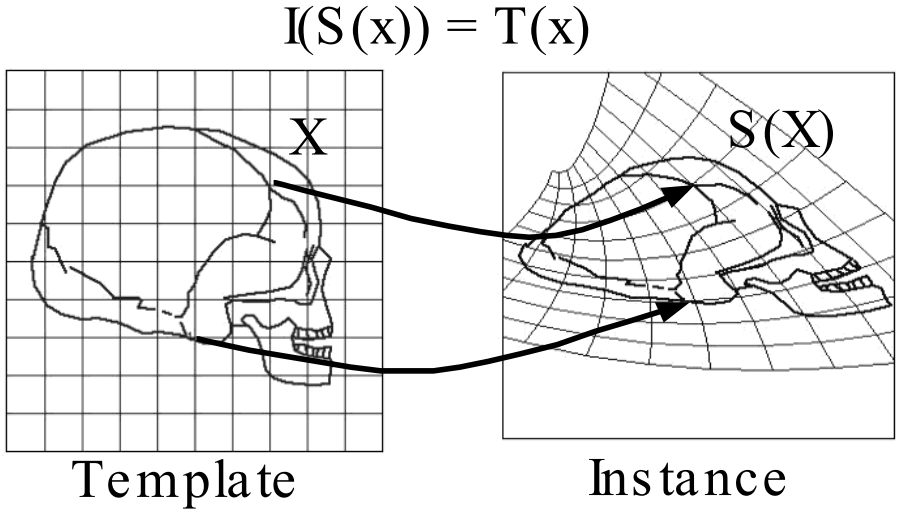
Model of texture

Active Appearance Models

Shape: $$\mathcal{S}(\mathbf{x};\mathbf{s}) = \sum_i \mathbf{s}_i S_i(\mathbf{x}),$$

Appearance: $$\mathcal{T}(\mathbf{x};\mathbf{t}) = \sum_i \mathbf{t}_i T_i(\mathbf{x})$$

Synthesis: $$I(\mathcal{S}(\mathbf{x};\mathbf{s})) \simeq \mathcal{T}(\mathbf{x};\mathbf{t})$$

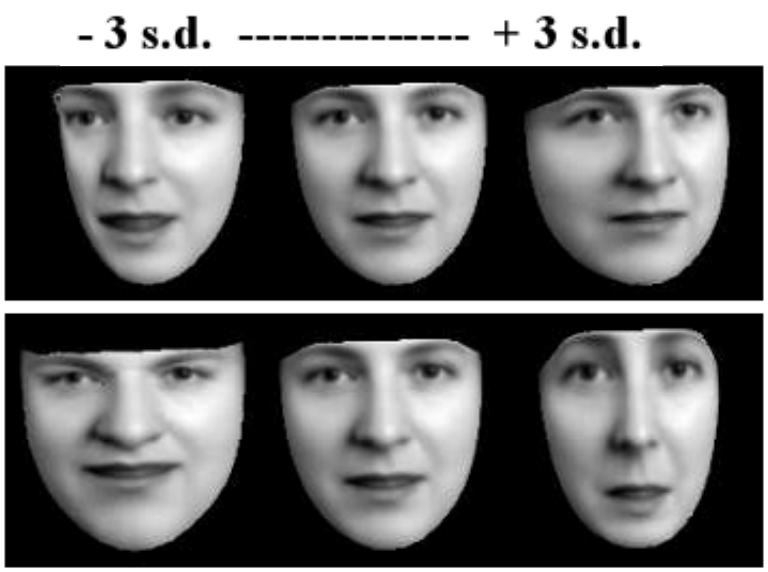


AAM model

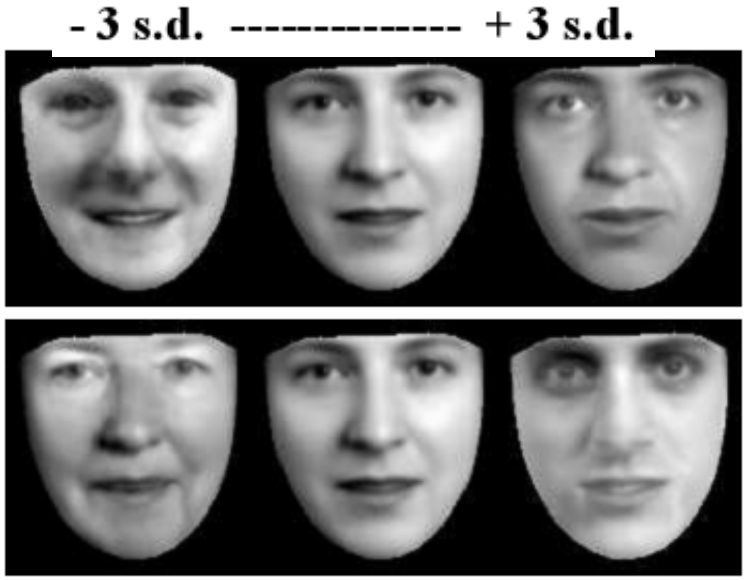
- Shape Eigenvectors

- Texture Eigenvectors

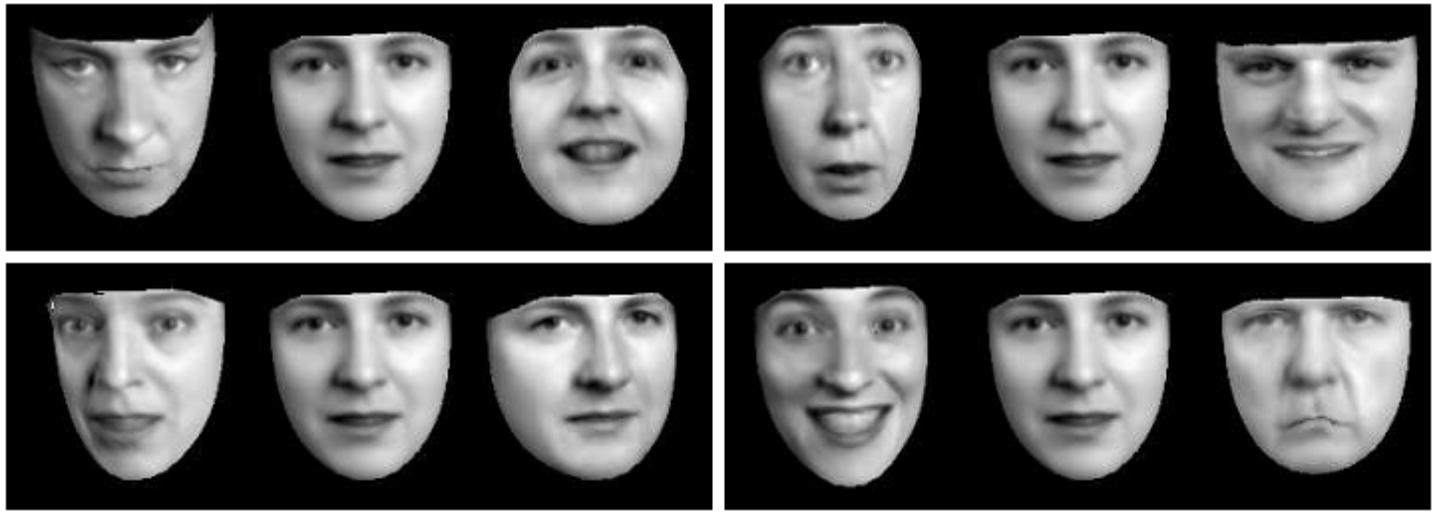
Playing with the Parameters



First two modes of shape variation



First two modes of gray-level variation



First four modes of appearance variation

Active Appearance Model fitting

Given:

1) an appearance model,
2) a new image,
3) a starting approximation

Find:

the best matching synthetic image

- **Minimize reconstruction error**

$$E(\mathbf{s},\mathbf{t}) = \sum_{\mathbf{x}} \left[I(\mathcal{S}(\mathbf{x};\mathbf{s})) - \mathcal{T}(\mathbf{x};\mathbf{t}) \right]^2$$

- **How?**
  - **Alternate between estimating s and t**
- **For t: projection of deformed image $I(S(\mathbf{x};\mathbf{s}))$ onto PCA basis**
- **For s?**

AAM parameter estimation: shape

- **Iterative scheme (Newton-Raphson minimization)**

$$E(\mathbf{s},\mathbf{t}) = \sum_{\mathbf{x}} \left[I(S(\mathbf{x};\mathbf{s})) - \mathcal{T}(\mathbf{x};\mathbf{t}) \right]^2$$

$$E(\mathbf{s}+\boldsymbol{\Delta}\mathbf{s},\mathbf{t}) = E(\mathbf{s},\mathbf{t}) + \mathcal{J}\boldsymbol{\Delta}\mathbf{s} + \frac{1}{2}\boldsymbol{\Delta}\mathbf{s}^T\mathcal{H}\boldsymbol{\Delta}\mathbf{s}$$

$$\Delta\mathbf{s}^* = -\mathcal{J}\mathcal{H}^{-1} \qquad \mathbf{s}' = \mathbf{s} - \mathcal{J}\mathcal{H}^{-1}$$

$$I(S(\mathbf{x};\mathbf{s}+\Delta\mathbf{s})) \simeq I(S(\mathbf{x};\mathbf{s})) + \sum_{i=1}^{Ns} \frac{dI}{d\mathbf{s}_i}(\mathbf{x};\mathbf{s})\Delta\mathbf{s}_i$$

$$\frac{dI}{d\mathbf{s}_i}(\mathbf{x};\mathbf{s}) = \frac{\partial I(S(\mathbf{x};\mathbf{s}))}{\partial x}\frac{\partial S_x}{\partial \mathbf{s}_i} + \frac{\partial I(S(\mathbf{x};\mathbf{s}))}{\partial y}\frac{\partial S_y}{\partial \mathbf{s}_i},$$

$$\mathcal{J}_i = \sum_x \left[I(S(\mathbf{x},\mathbf{s})) - T(\mathbf{x},\mathbf{t}) \right] \frac{\mathbf{dI}}{\mathbf{ds_i}}(x) \qquad \mathcal{H}_{i,j} = \sum_x \frac{\mathbf{dI}}{\mathbf{ds_i}}(x)\frac{\mathbf{dI}}{\mathbf{ds_j}}(x)$$

Active Appearance Model Search (Results)

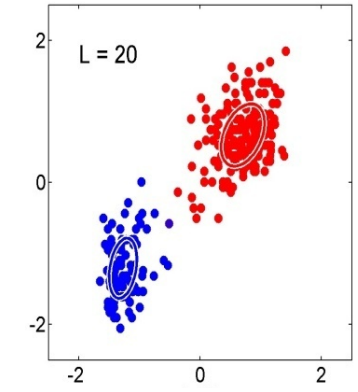
Model fitting + Image segmentation

- AAM fitting

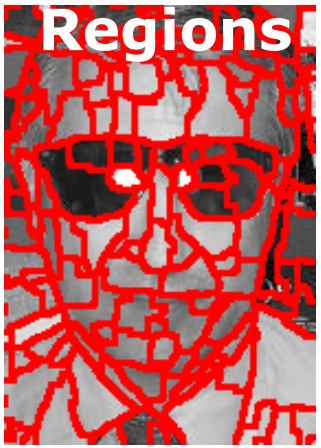
- Variational Segmentation

Model fitting + Image segmentation

- **EM formulation**
  - **E-step: Assign data to models**
  - **M-step: Estimate model paramete**



- **Joint segmentation & recognition**
  - **E-step: segmentation**
  - **M-step: deformable model fitting**



Challenges: Image Variability

Short Term

Expression

Pose

Illumination

Long Term

- Facial Hair
- Makeup
- Eyewear
- Hairstyle
- Piercings
- Aging

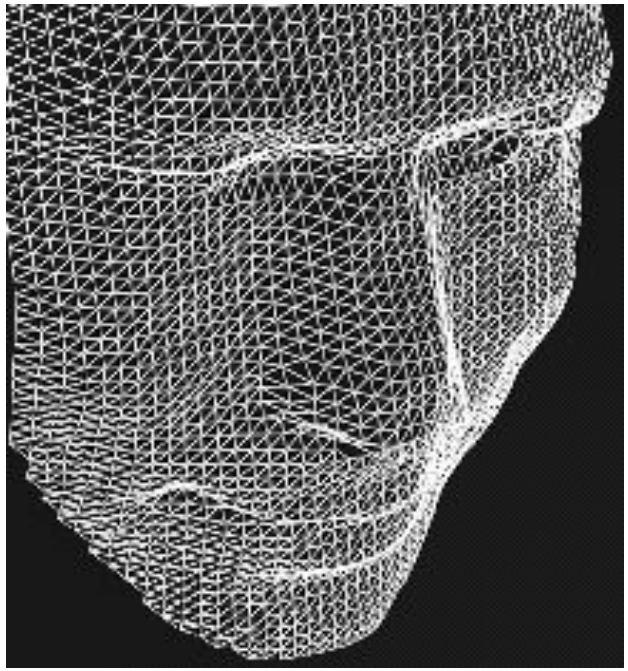
Modeling Image Variability: 3-D Faces

Laser Range Scanners

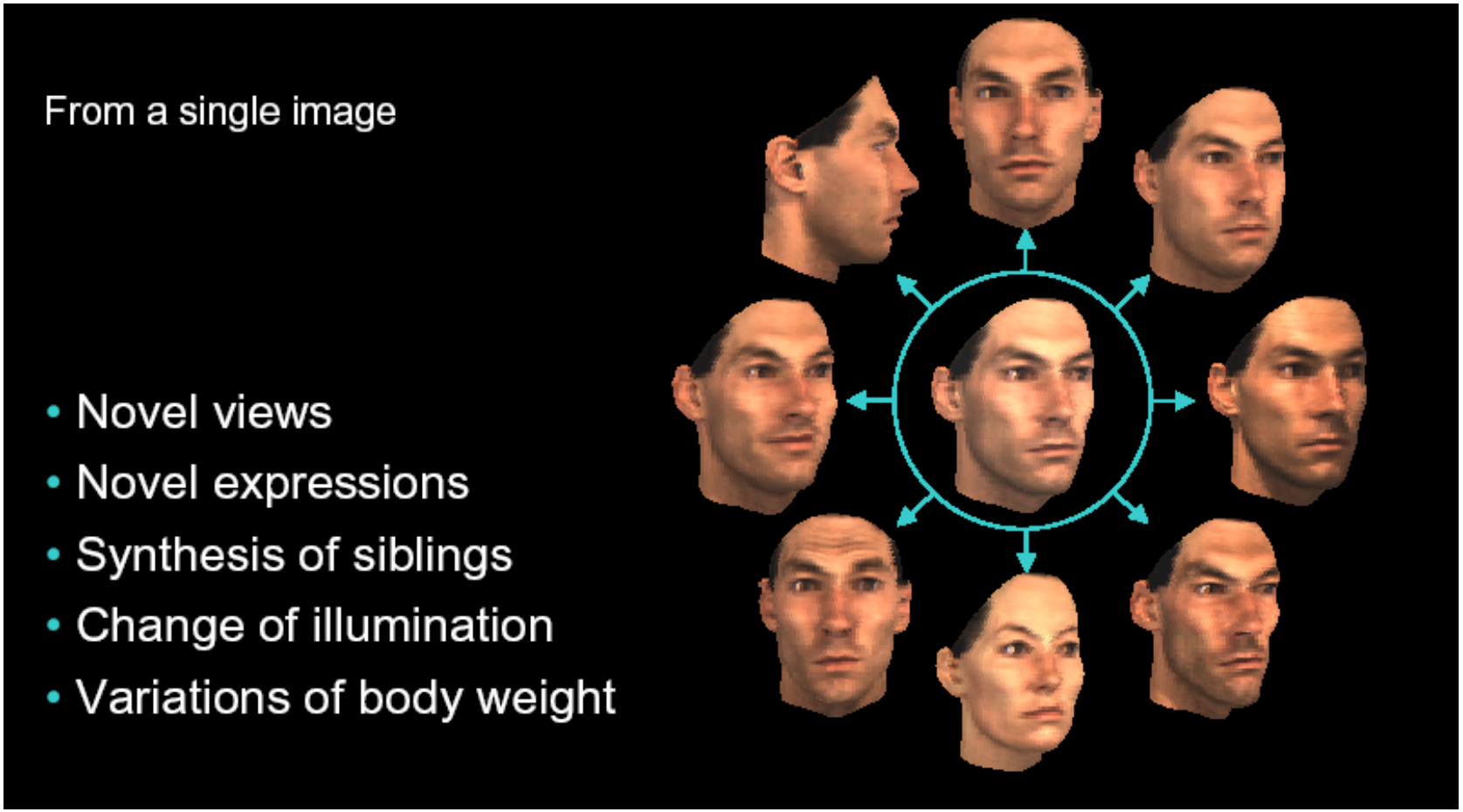
Stereo Cameras

Structured Light

Photometric Stereo

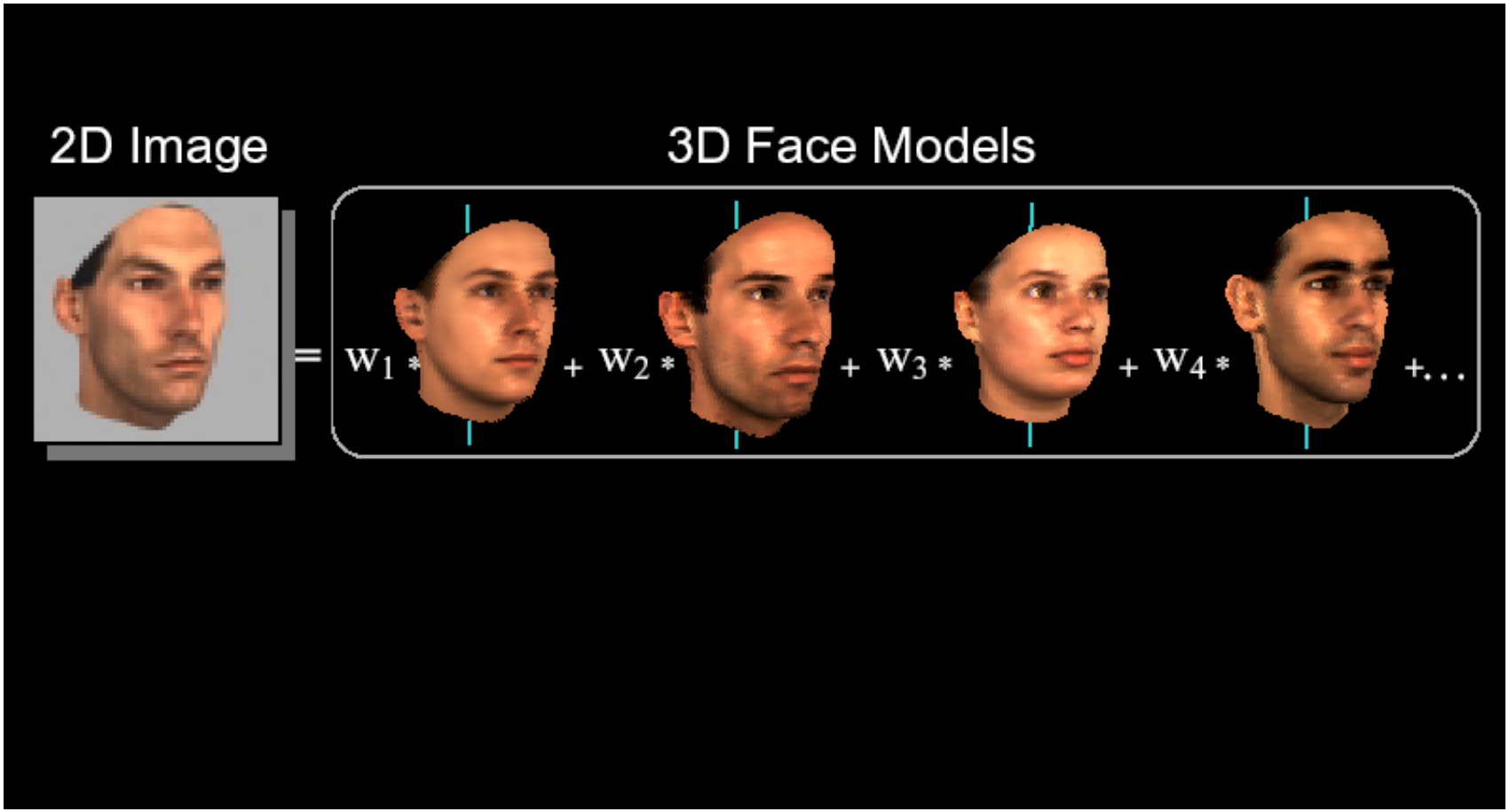


What can we do with 3d shape?



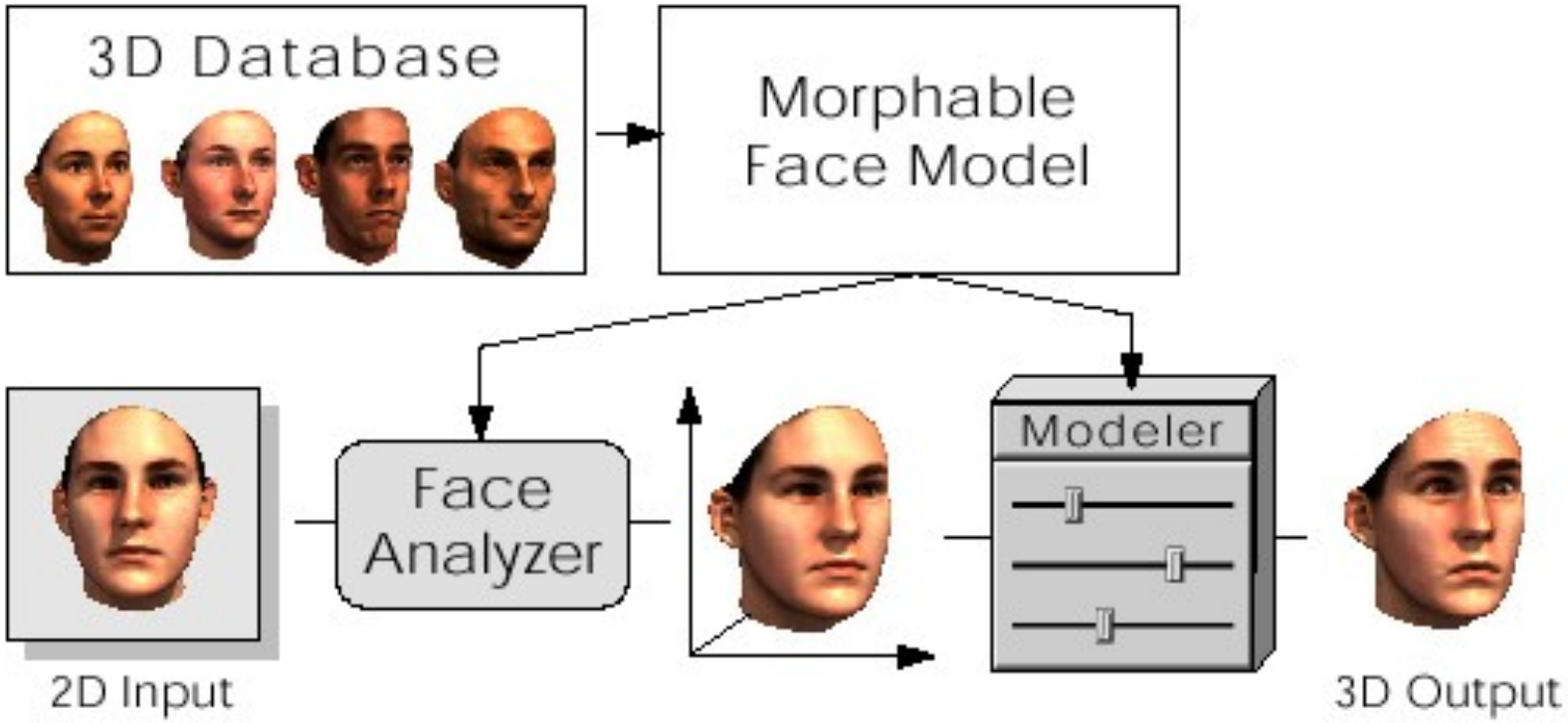
[Blanz and Vetter 1999, 2003]

Building a Morphable Face Model



[Blanz and Vetter 1999, 2003]

3-D Morphable Models



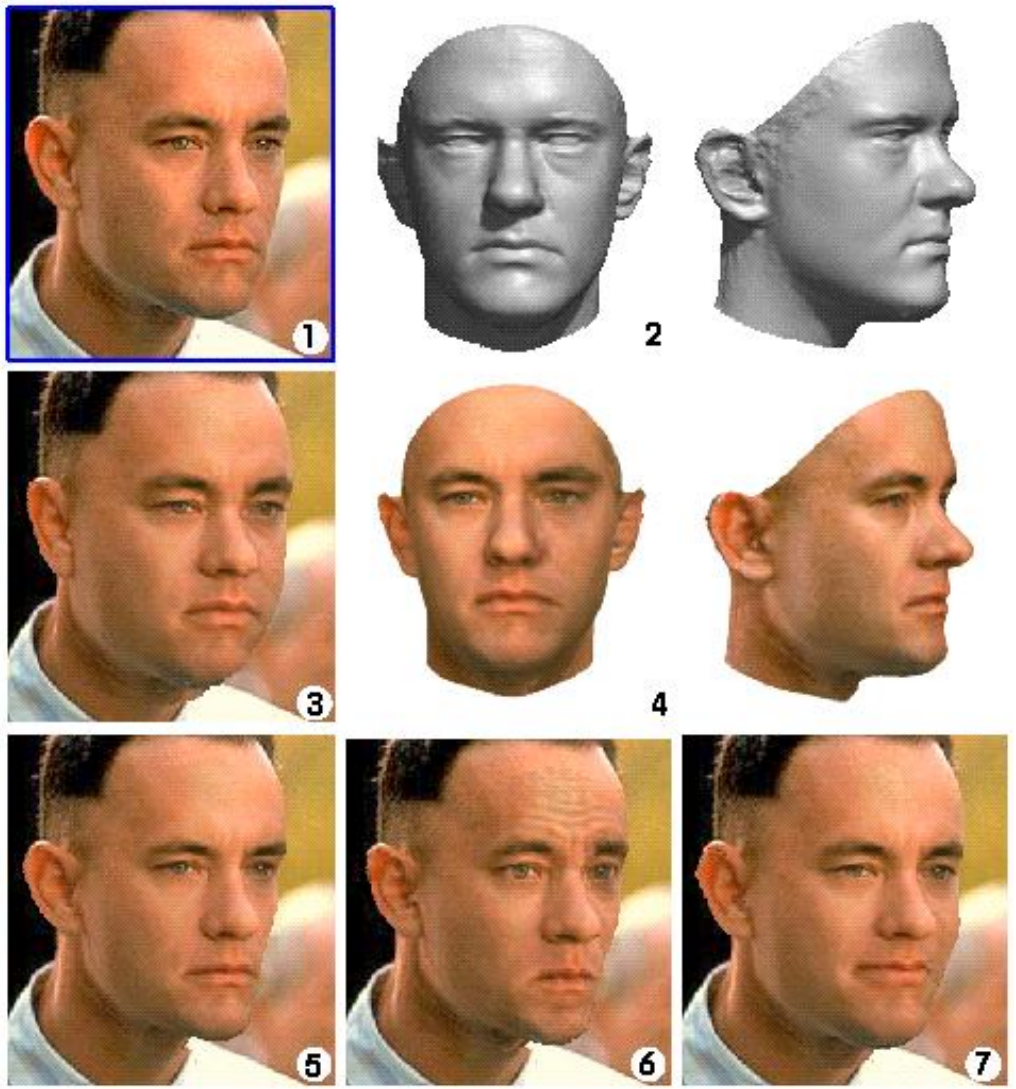
[Blanz and Vetter 1999, 2003]

3D Morphable models

Recover Shape

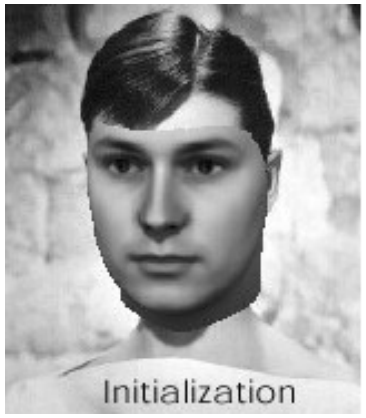
Synthesize new views

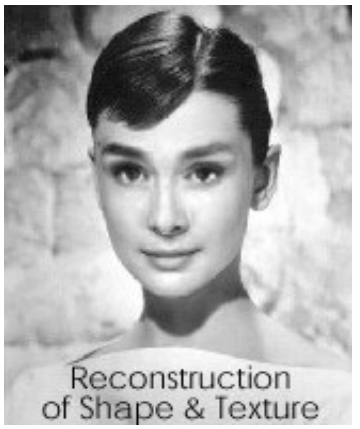
Synthesize new expressions



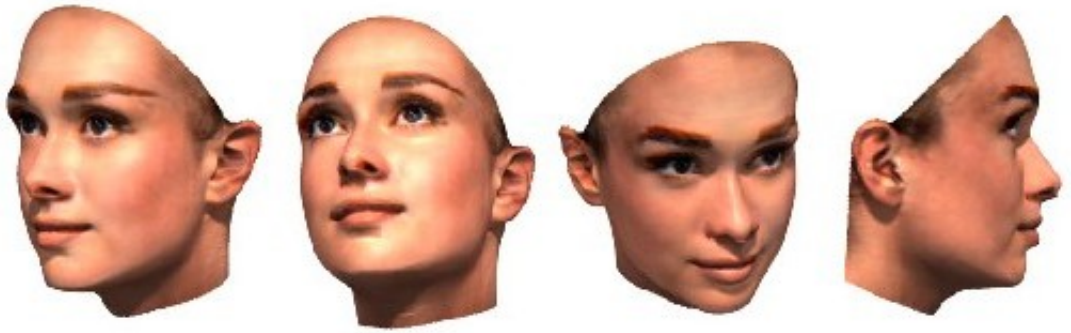
3-D Morphable Models fitting

- **Rough manual initialization**
- **Gradient descent to minimize reconstruction error functional**

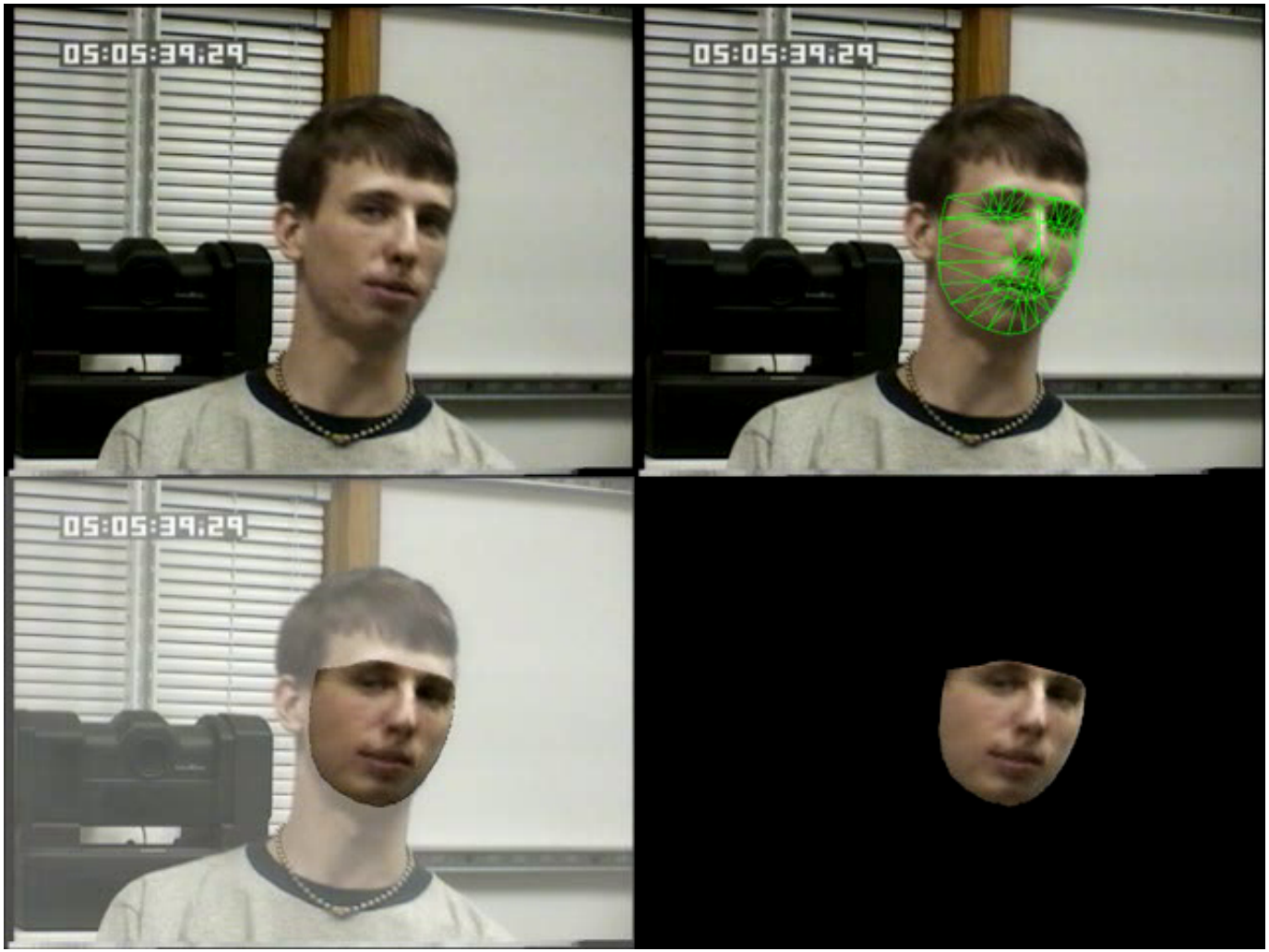




- **And then**

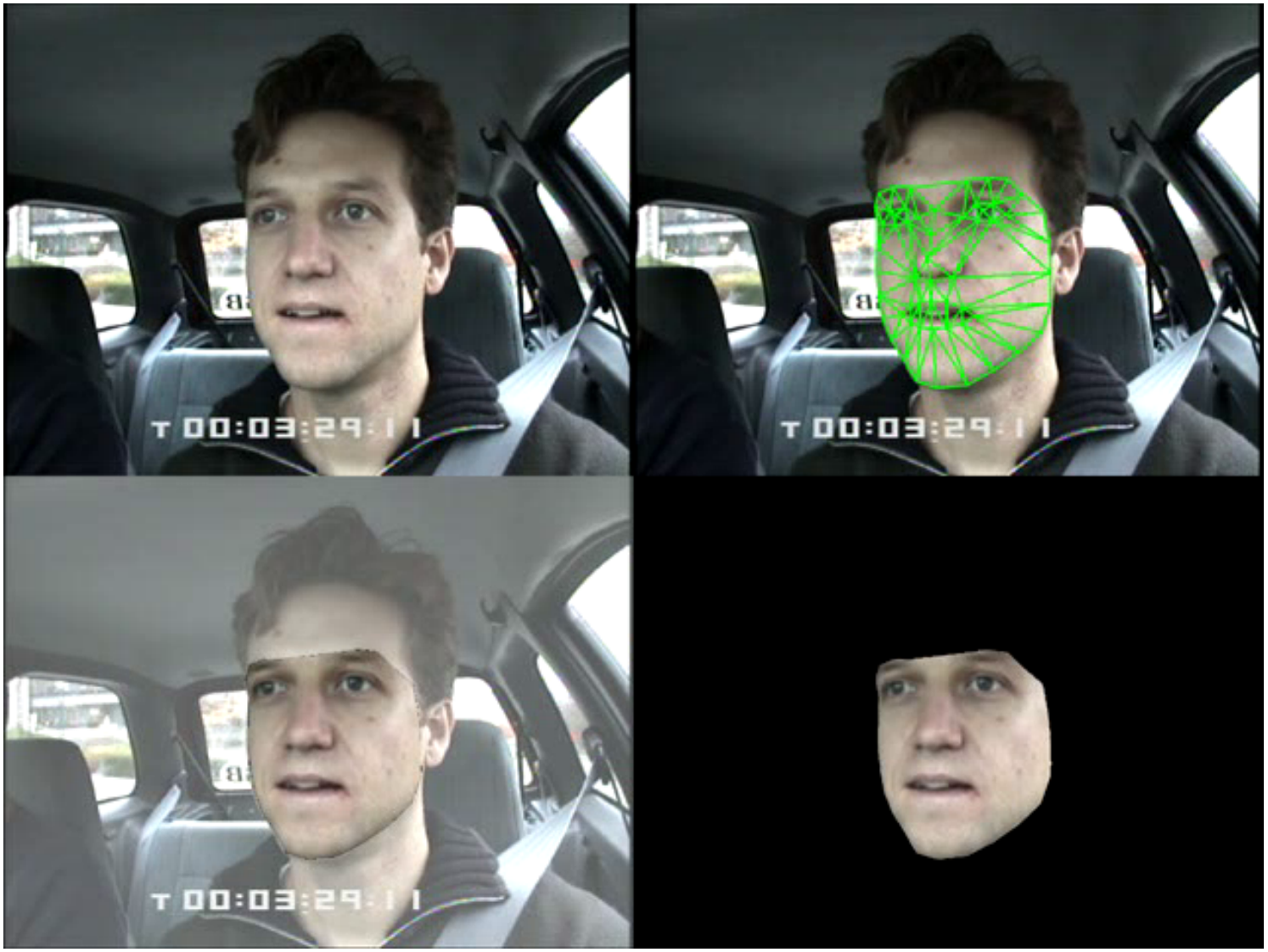


3D AAM for face tracking



CMU group: I. Matthews, S. Baker, R. Gross
(230 Frames per second)

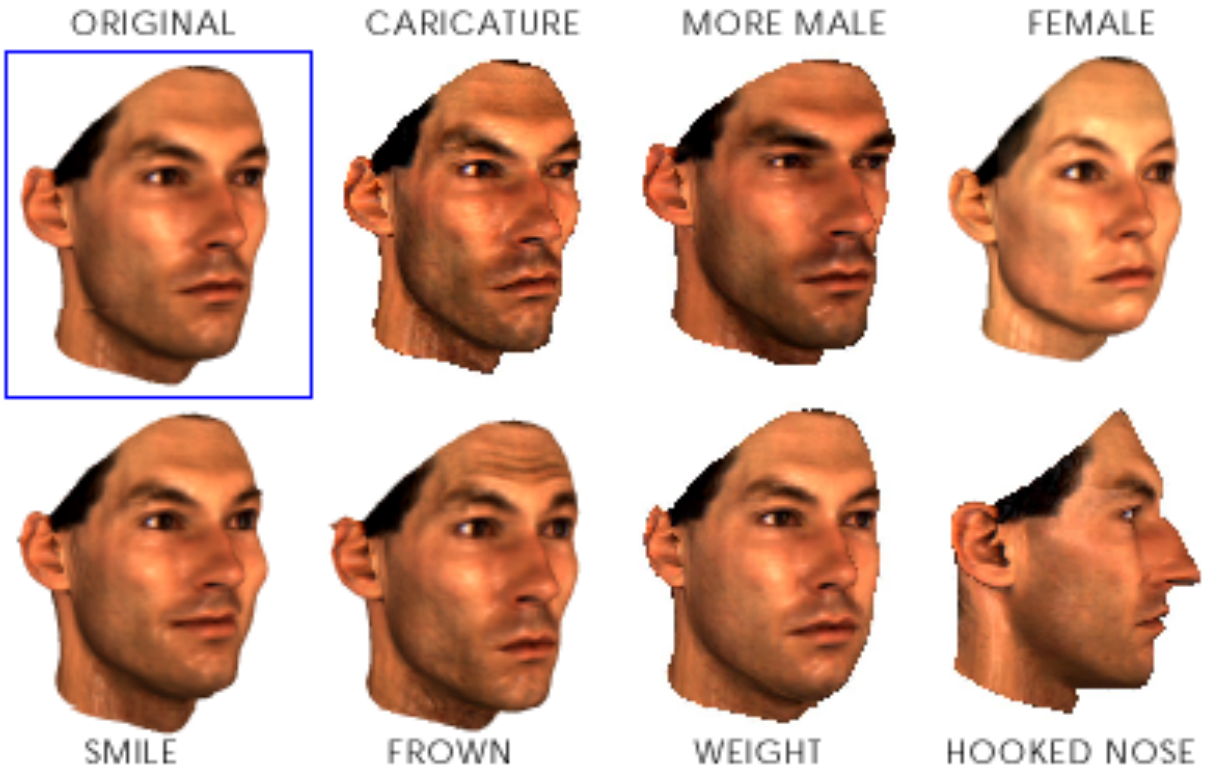
3D AAM for face tracking



Playing with Facial Attributes

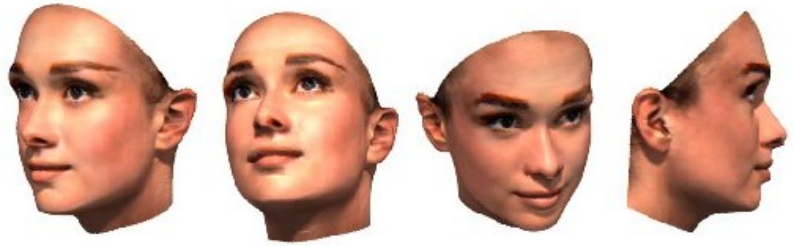
Several classes of attributes are modeled:

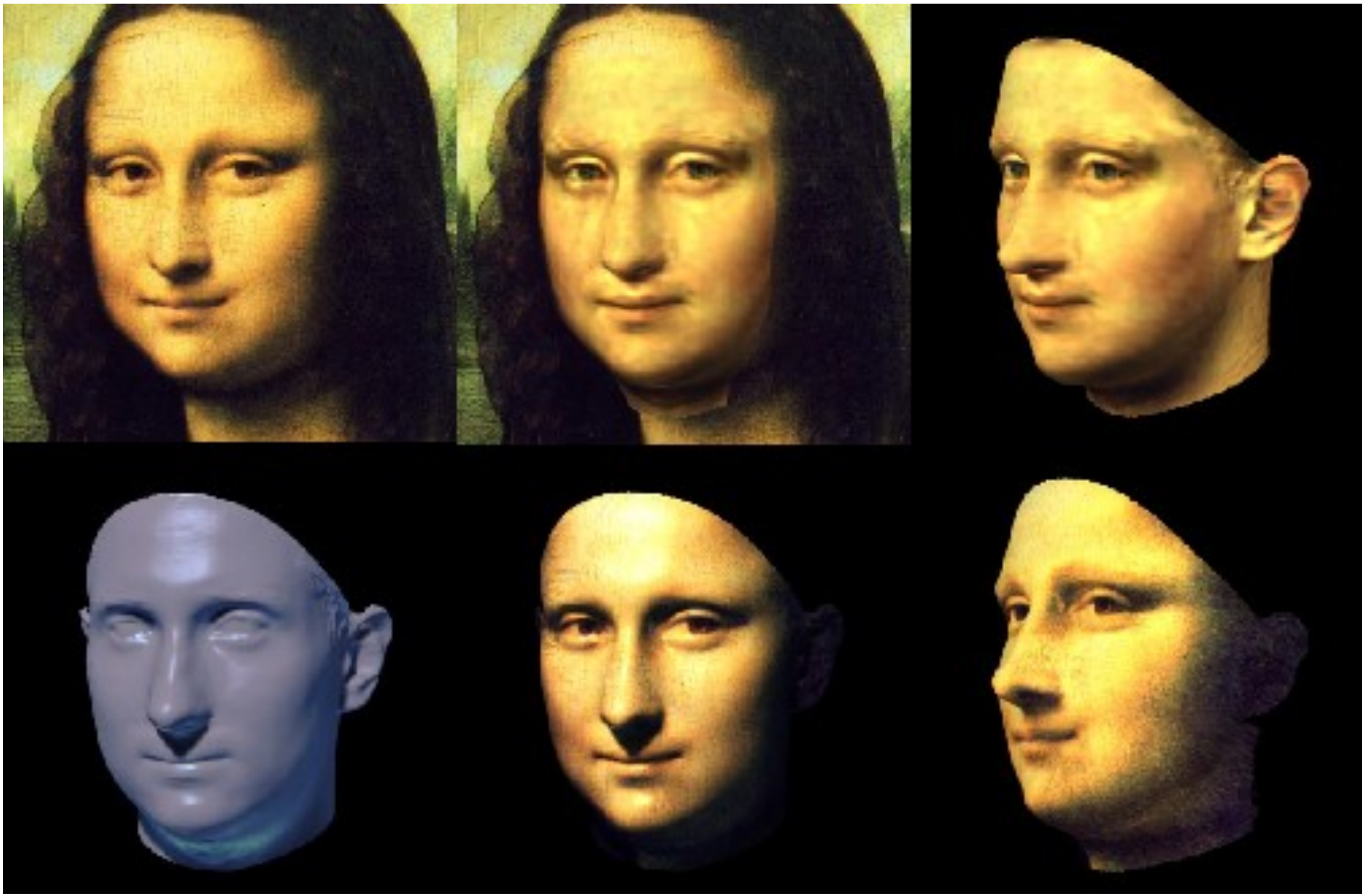
- **Facial expressions (smile, frown)**
- **Individual characteristics (double chin, hooked nose, ‘maleness’)**
- **Distinctiveness**

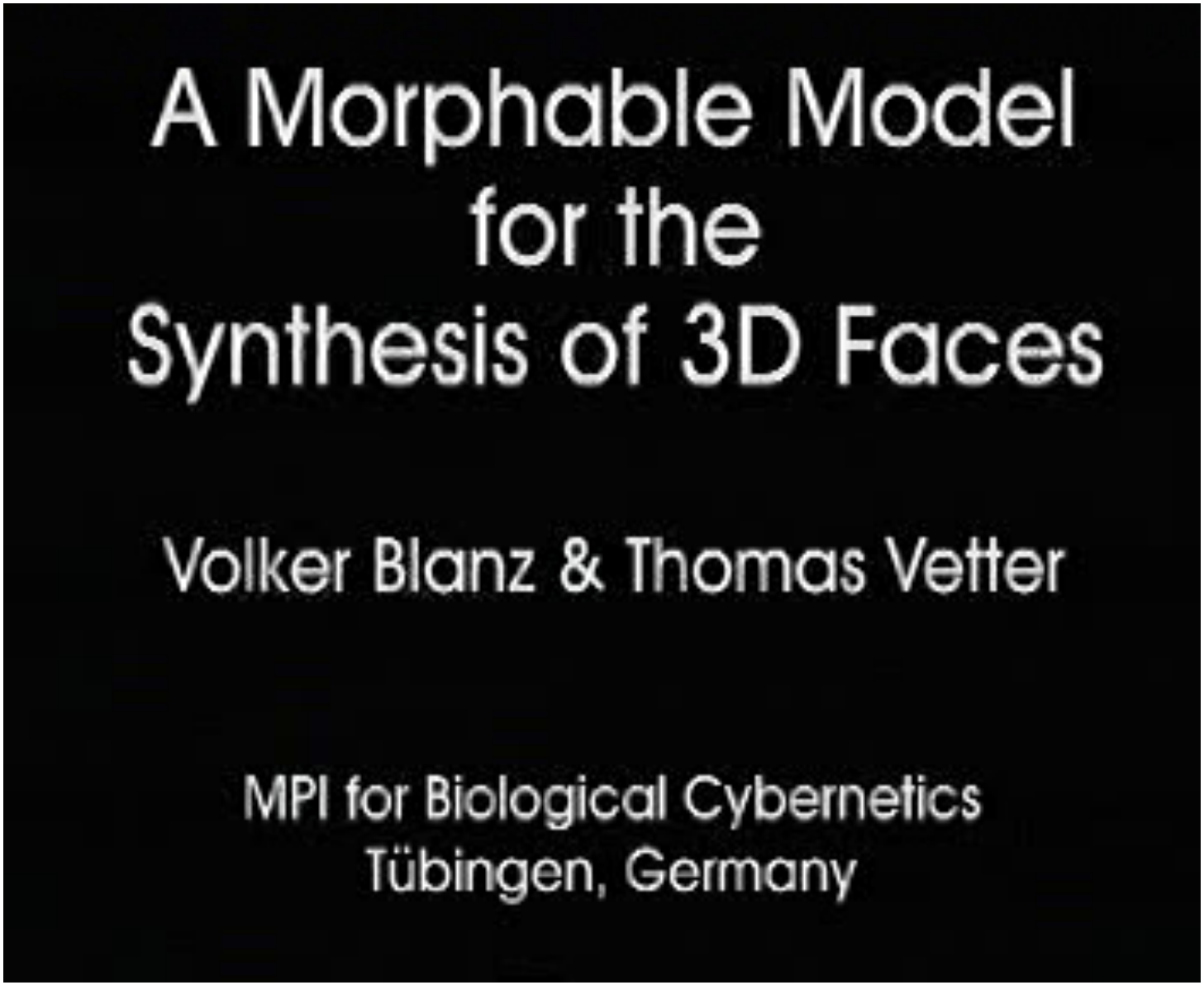


Manipulating Facial Attributes via Deformations

- For each face in the database, two scans are recorded: $S_{neutral}$, and $S_{expression}$.
- The difference vector $\Delta S = S_{expression} - S_{neutral}$ is saved and later on simply added to the 3D reconstruction of the input image.






A Morphable Model
for the
Synthesis of 3D Faces
Volker Blanz & Thomas Vetter
MPI for Biological Cybernetics
Tübingen, Germany