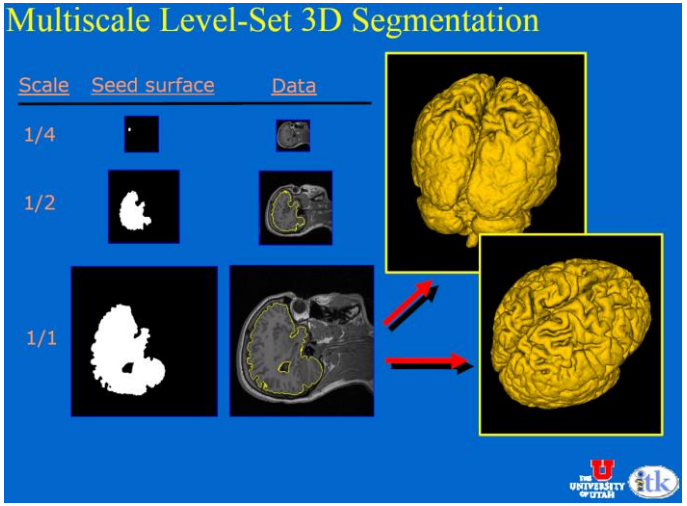
Medical Image Analysis
Lecture 10

Statistical Prior Based Segmentation

Martin Urschler

Model Based Segmentation

- High-Level Deformable Models (AC, LS)
  - „Weak“ Model Knowledge:
  - Segmentation is connected
  - Contour/Surface is „smooth“



- Strong Model Knowledge:
  - Explicitly train on prior instances!
  - Subspace Models of Appearance & Shape

Model Based Segmentation - Idea

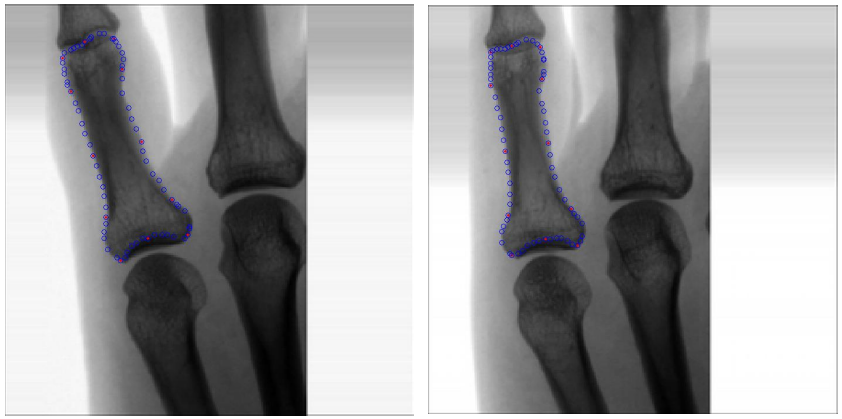
Sample Objects (Training Instances)

Generate Parameterized Model

Training Instance — $\mathbf{x} = \bar{\mathbf{g}} + \mathbf{Pb}$ — Parameters

Model Based Segmentation - Idea

Training Instances



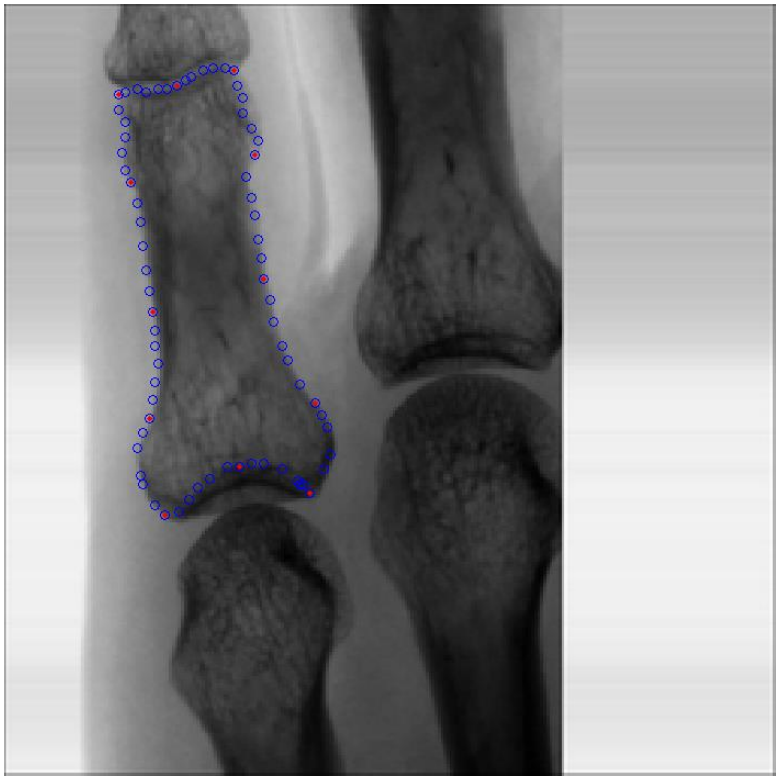
Parameterized Model

$$\mathbf{x} = \bar{\mathbf{g}} + \mathbf{Pb}$$

Modify Model Parameters

Synthetic Object

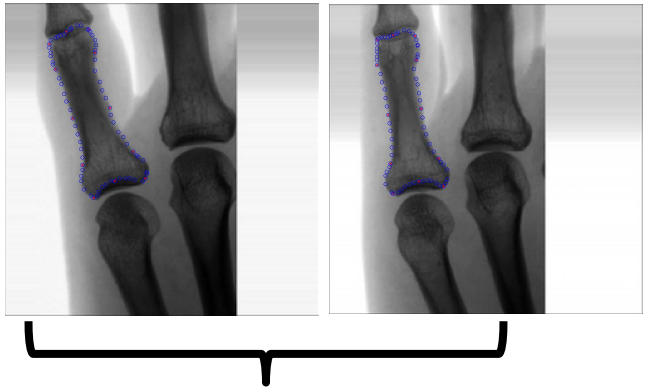
New Unseen Image



Active Shape Models

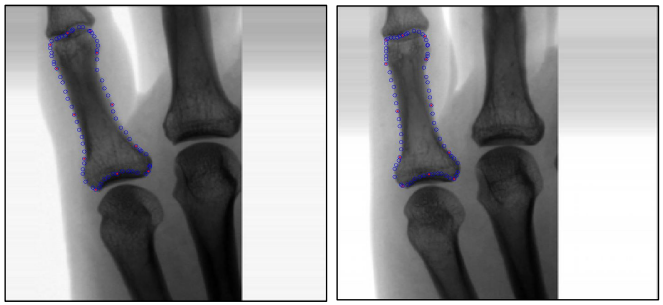
Active Shape Model (ASM)

- Starting from statistical shape model
  - Trained from (sufficient number of) examples

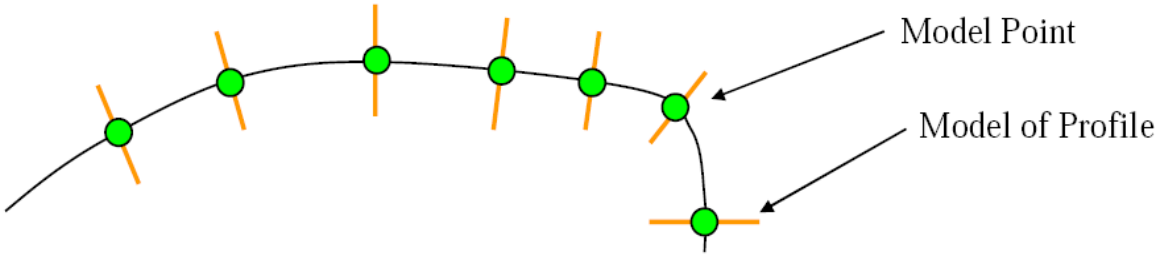


- How do we use it to interpret new images?
  - Use an “Active Shape Model” (Cootes 1995), an iterative method for matching a trained model to an unseen image

ASM Workflow



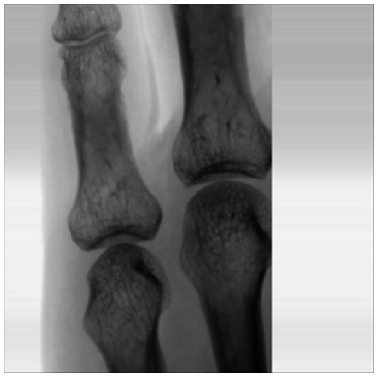
Parameterized Model



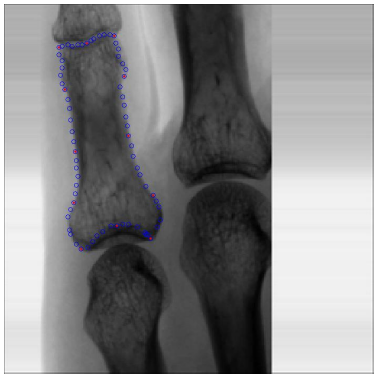
Local Model of Intensity, e.g. along normals (profiles) to shape. Also derived from training set.

Statistical Model of Shape

Unseen Image



Iterative Matching



Segmented Image

ASM Search Overview

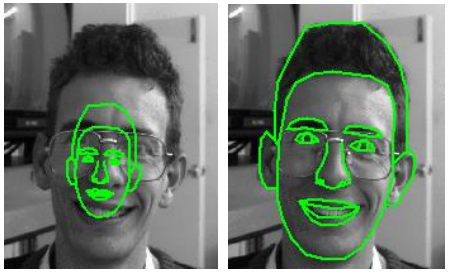
- Iterative two-stage optimization algorithm
  1. Look along normals through each model point to find the best local match for the model at that point
  2. Update the pose and model shape parameters to best fit the model instance to the found points
  3. Repeat until convergence

Initial pos

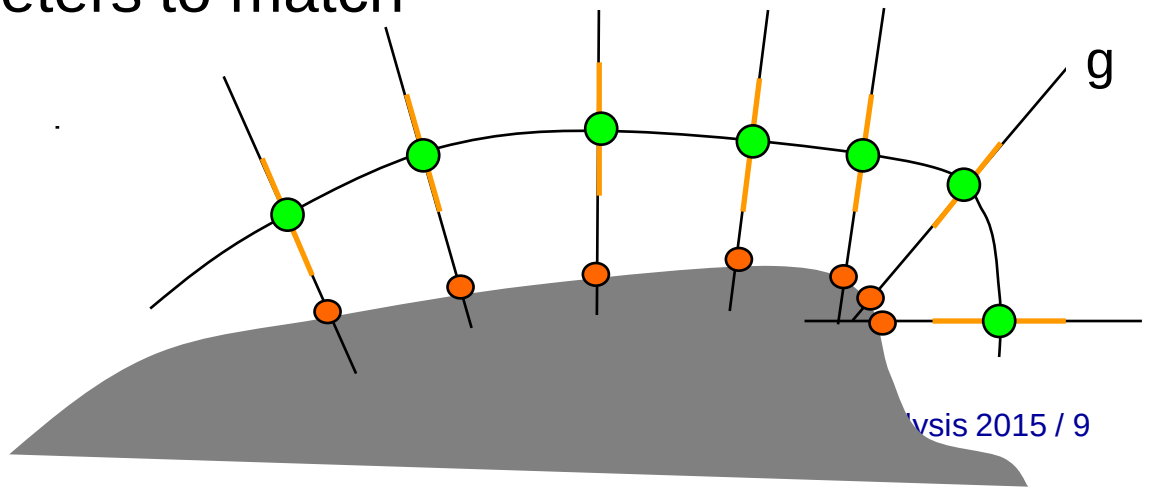
5th iteration

convergence

ASM Search Overview



- Iterative, Two-Step Optimization
  - Nonlinear problem, not convex -> reach local optimum -> we need to have good initialization!
- Formally:
  1. Start from $\mathbf{X^0}$, search along profiles g for best match $\mathbf{X}$
  2. Update parameters to match $\mathbf{X}$ as good as possible

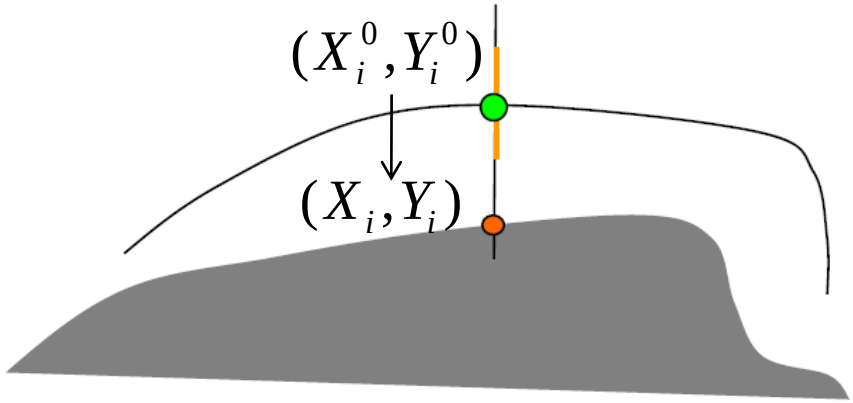


- (X_i^0, Y_i^0)
- (X_i, Y_i)

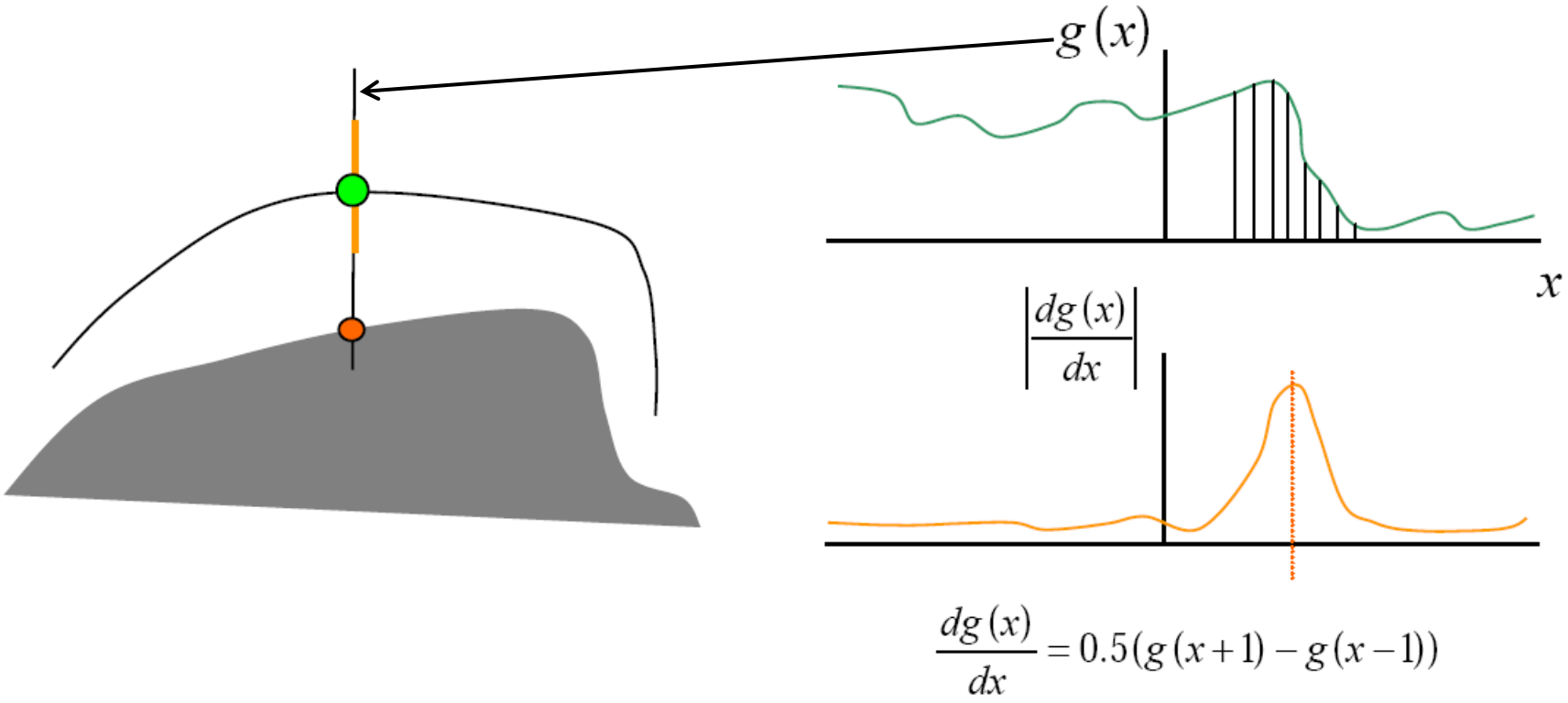
Local Structure Models

- Step 1: Need to search for local match for each point along profile

- Possible Model Assumptions:
  - Strongest edge
  - Statistical model of profile (PCA)



Searching for Strong Edges

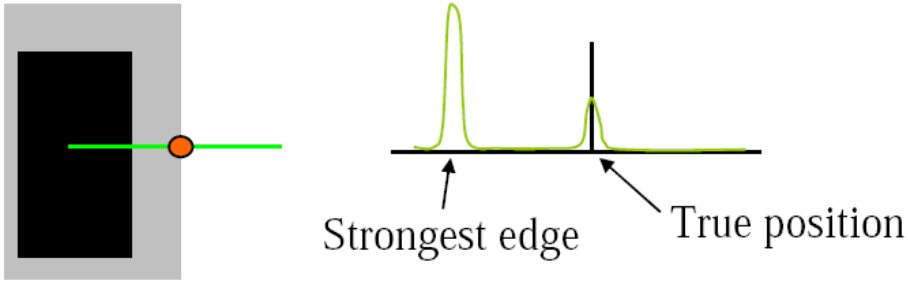


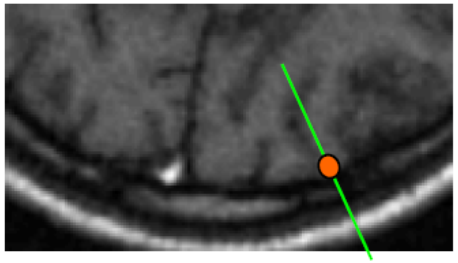
$$\frac{dg(x)}{dx} = 0.5(g(x+1) - g(x-1))$$

Select point along profile at strongest edge

Profile Models

- Sometimes the true point is not on the strongest edge!

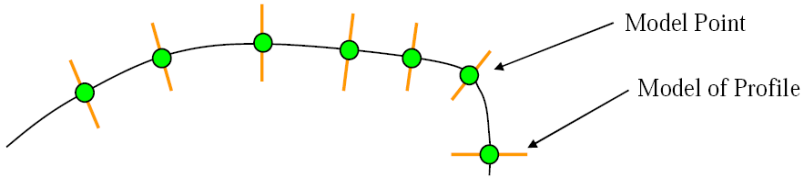




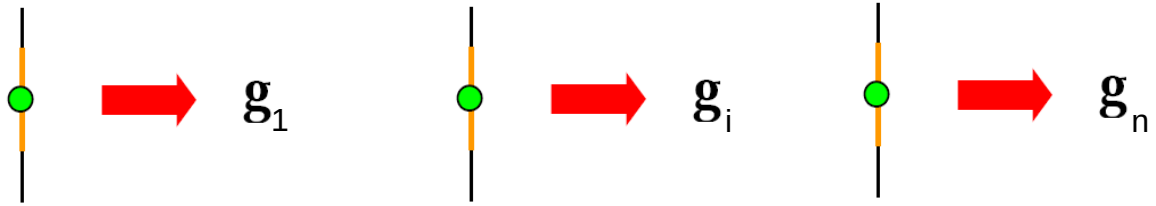
- Model local structure to help locate the point

How to model? -> Statistical Appearance Model along 1D profiles (see EigenPatch Approach, just 1D)

Statistical Profile Models



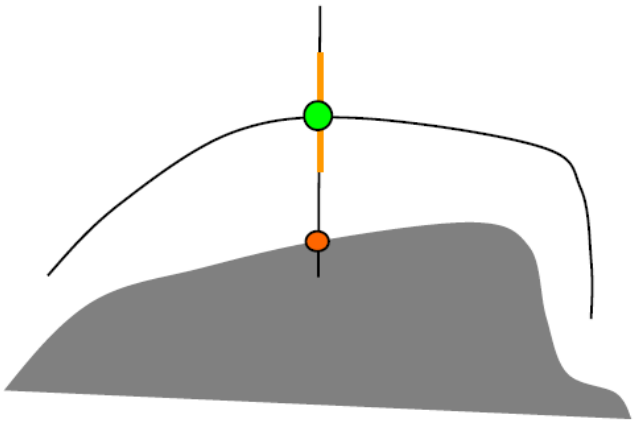
- Collect Sample Profiles from Training Set

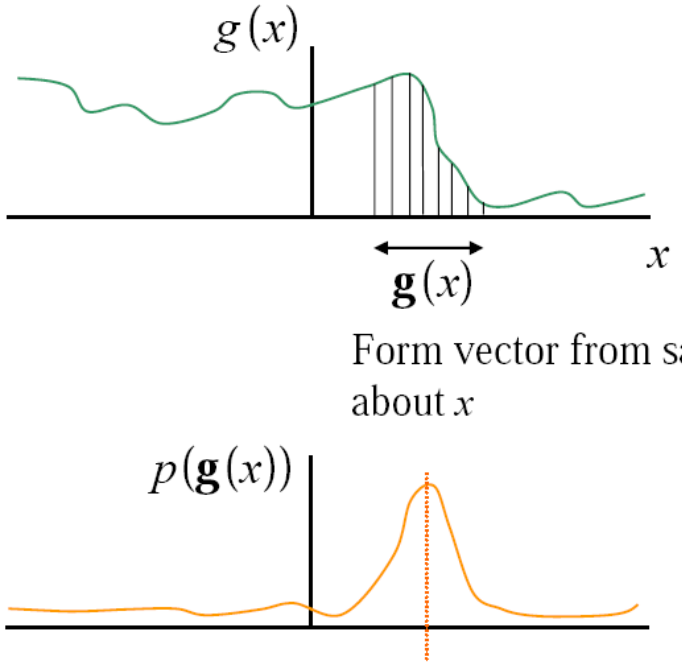


- Normalize intensity profiles to allow for global lighting variations
- From training set learn generative PCA model (Gauss assumption) of 1D lines p(g)

Searching Along Profiles

- During search we look along a normal for the best match for each profile

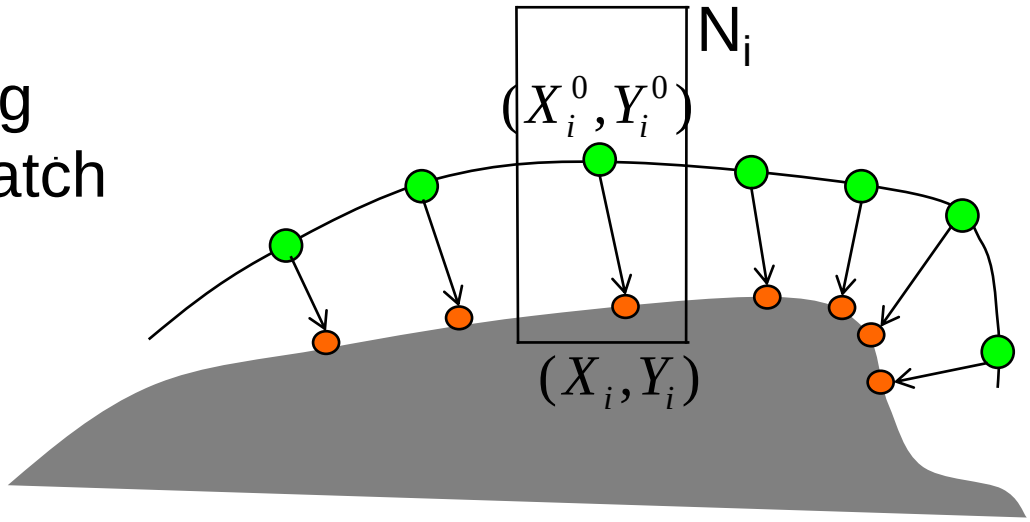




-> “object” detection along profile!

Searching in Local Neighborhood

- Obvious improvement: instead of searching only in normal direction, search 2D intensity patches in a neighborhood N around point!
- Models:
  - Template Matching (e.g. with Mean Patch from Training Set)
  - PCA Patch Model (2D)



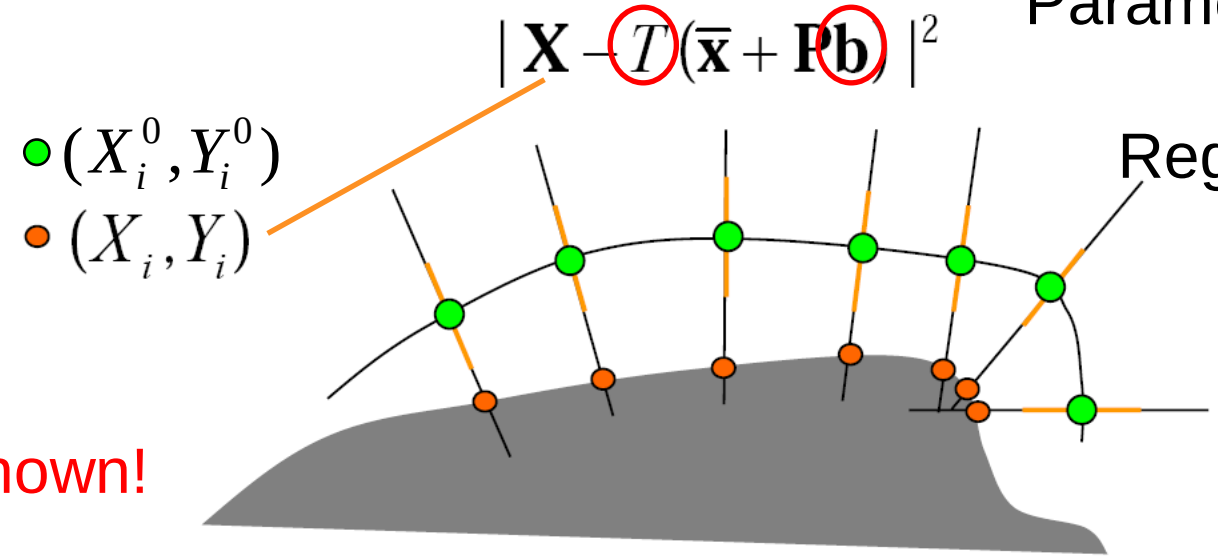
Iterative Two-Stage Search Algorithm, continued

Step 1: Search along profile for a candidate **X**

Step 2: Update similarity transformation *T* and parameters ***b*** to minimize

$$|\mathbf{X} - T(\bar{\mathbf{x}} + \mathbf{P}\mathbf{b})|^2$$

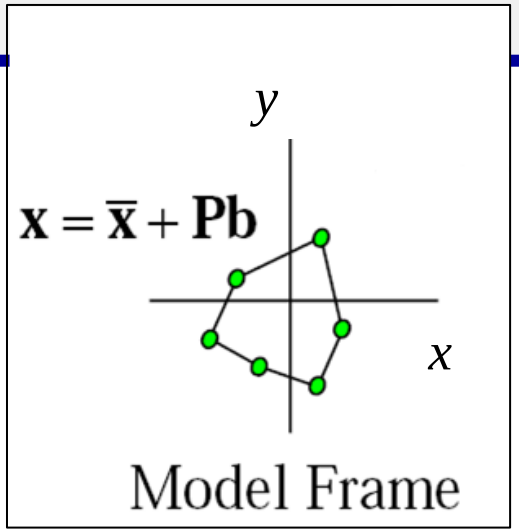
Parameter Update b = Regularization Step!

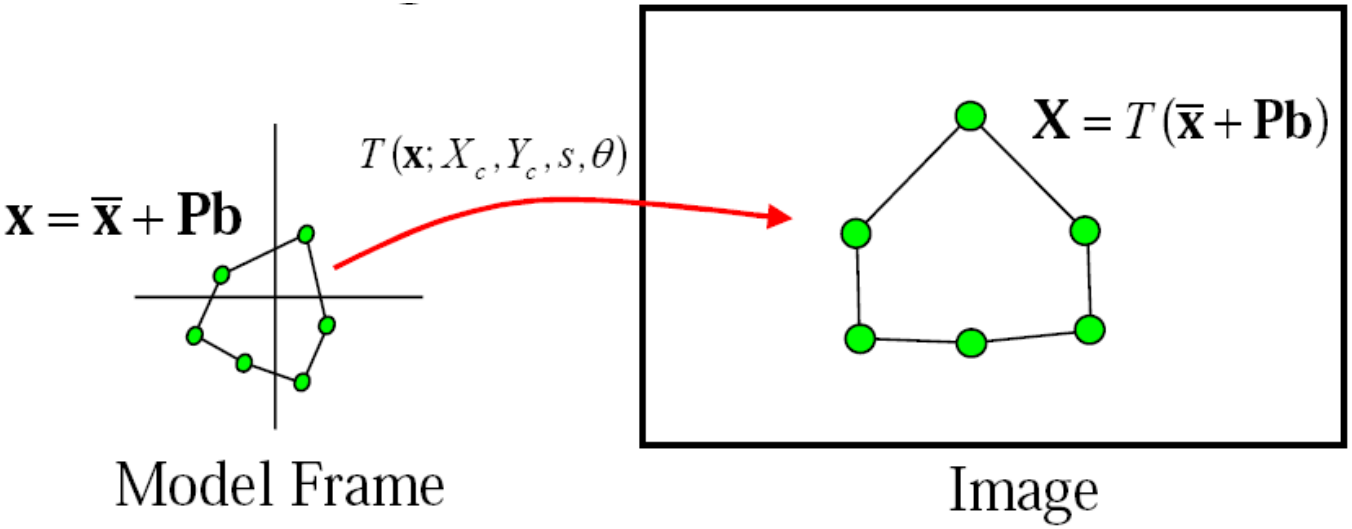


Unknown!

How to place model in an image?

- The model points are defined in a model coordinate frame
- Must apply global similarity transformation T to place the model into an image





$$T(\mathbf{x}) = \begin{pmatrix} s\cos\theta & \sin\theta & X_c \\ -\sin\theta & s\cos\theta & Y_c \\ 0 & 0 & 1 \end{pmatrix}$$

Step 2: Updating Parameters

- Find shape model (b) and pose parameters to minimize cost S

Pose:

$$\Theta := (X_c, Y_c, s, \theta)$$

$$S(\mathbf{b};\Theta) = \left|\mathbf{X} - T(\bar{\mathbf{x}} + \mathbf{P}\mathbf{b};\Theta)\right|^2$$

- Nonlinear, either
  - (Put into general optimizer)
  - Use iterative approach, switching between global trans-formation (pose) and shape model updates

Updating Parameters Alternately

$$S(\mathbf{b};\Theta) = \left|\mathbf{X} - T(\bar{\mathbf{x}} + \mathbf{P}\mathbf{b};\Theta)\right|^2$$

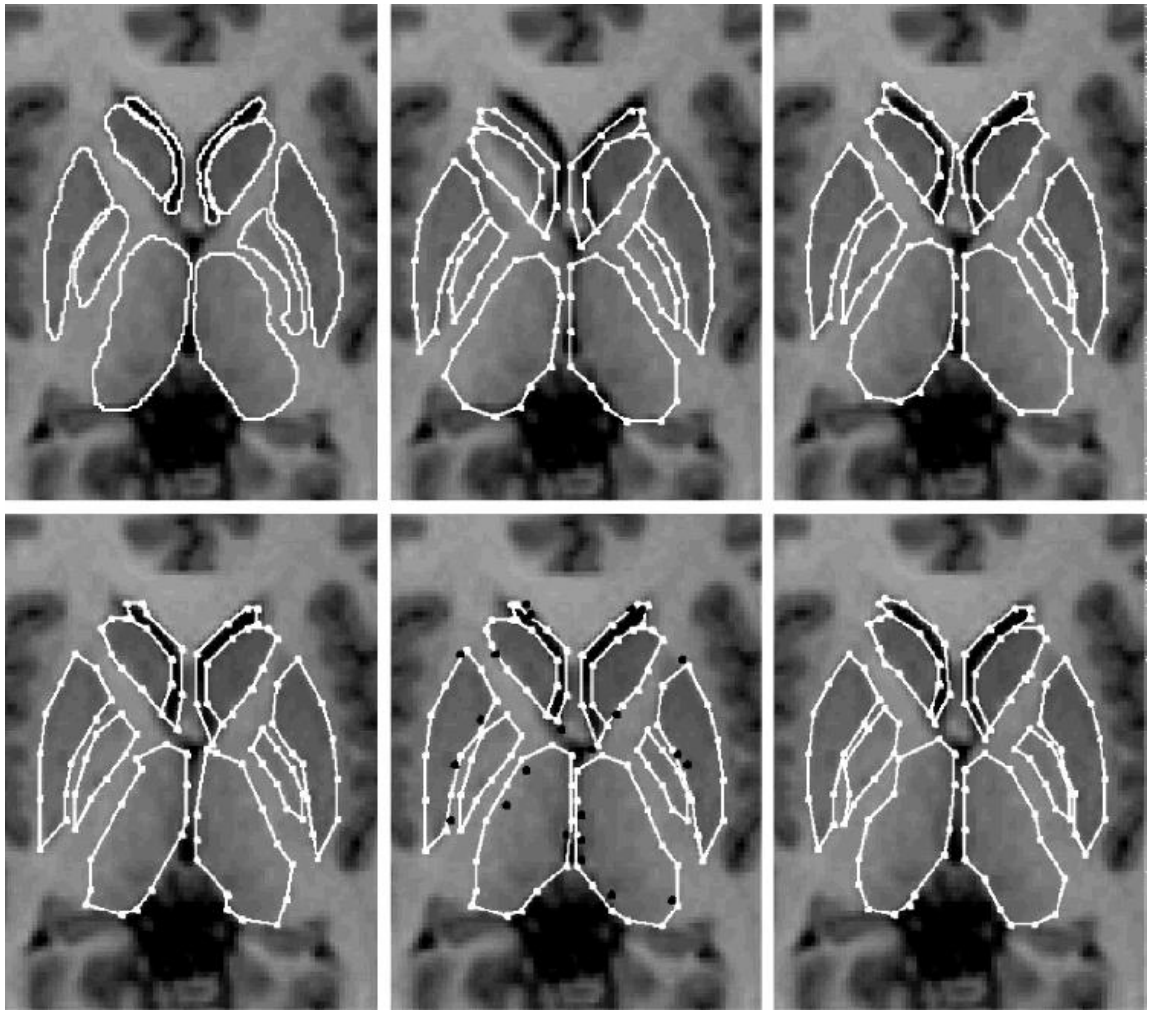
- Repeat until convergence:
  - Fix $\boldsymbol{b}$ giving model instance $\mathbf{X}_b$ and find Θ minimizing $S(\Theta) = \left|\mathbf{X} - T(\mathbf{X}_b;\Theta)\right|^2$
  - Analytic solution (Procrustes) exists: Θ'
  - Given fixed Θ' find $\boldsymbol{b}$ which minimizes

$$S(\mathbf{b}) = \left|\mathbf{X} - T(\Theta';\bar{\mathbf{x}} + \mathbf{P}\mathbf{b})\right|^2 \longrightarrow d\mathbf{b} = \mathbf{P}^T d\mathbf{x}$$

Example – ASM Matching

Manual

ASM



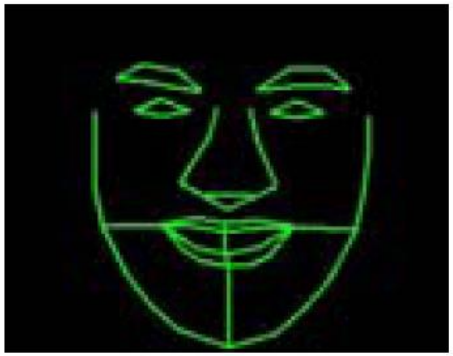
Summary ASM

- Practical Implementation in a Multi-Resolution Manner
- Advantages
  - Fast, simple, accurate
  - Efficient to extend to 3D
- Disadvantages
  - No full use of image information
  - Treats local models as independent

Appearance Models

Appearance Models

- Shape models represent shape variation
- Eigen-models can represent texture variation
- Combined appearance models represent both



Building Appearance Models

- For each example, extract shape vector **x**

Shape, $\mathbf{x} = (x_1, y_1, \ldots, x_n, y_n)^T$

- Build statistical shape model, $\mathbf{x} = \bar{\mathbf{x}}_s + \mathbf{P}_s \mathbf{b}_s$

Building Appearance Models

- For each example, extract texture vector **g**

Shape, $\mathbf{x} = (x_1, y_1, \ldots, x_n, y_n)^T$

Texture, **g**

Important:
Warp g to
mean shape!

Texture intensities always
sampled at same locations!

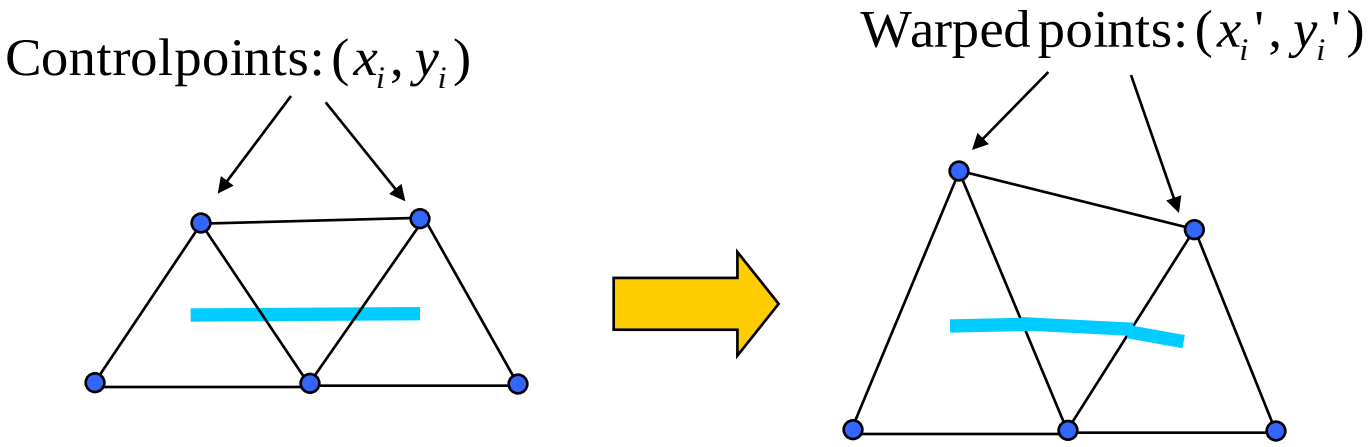
Warping texture

mean shape

- Problem:
  Given corresponding points in two images, how do we warp one image into the other?

- Two common solutions
  1. Piece-wise linear warp using triangles
  2. Thin-plate spline interpolation

Interpolation using Triangles



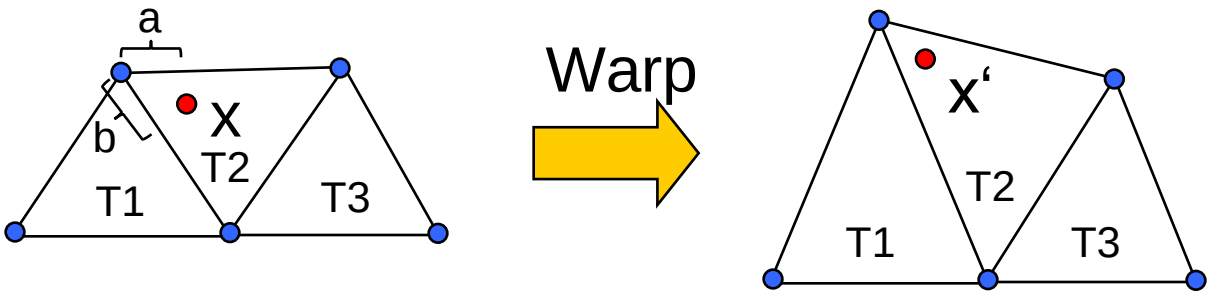
Region of interest enclosed by triangles.

Moving nodes changes each triangle

Just need to map regions between two triangles (piecewise affine mapping)

This is exactly what graphics hardware is optimized for!

Interpolation using Triangles

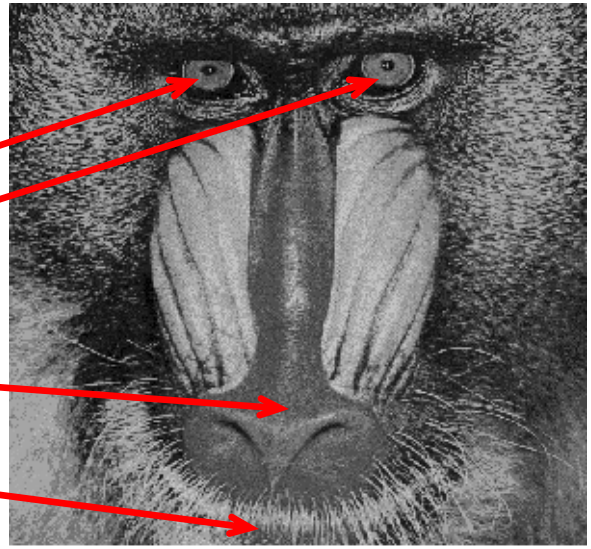


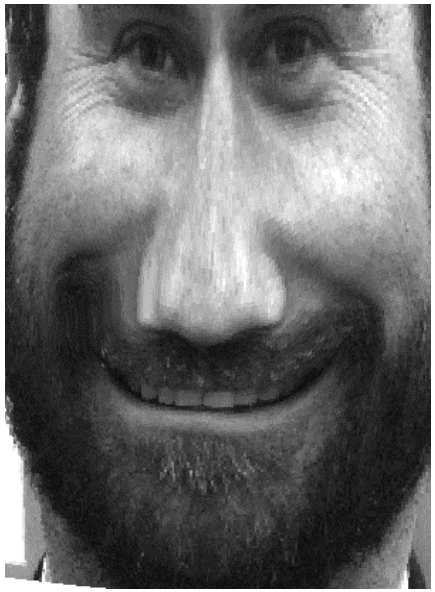
Mean Shape Triangle Set

Training Shape Triangle Set

- To resample training shape under mean shape:
  - Go over pixels under shape (mean shape coord. syst.)
  - Determine mean shape triangle (corresponds to certain training shape triangle)
  - Compute barycentric co-ordinates of x -> a,b
  - Interpolate intensity for corresponding triangle at a,b -> x’

Interpolation using Thin-Plate Splines (TPS)



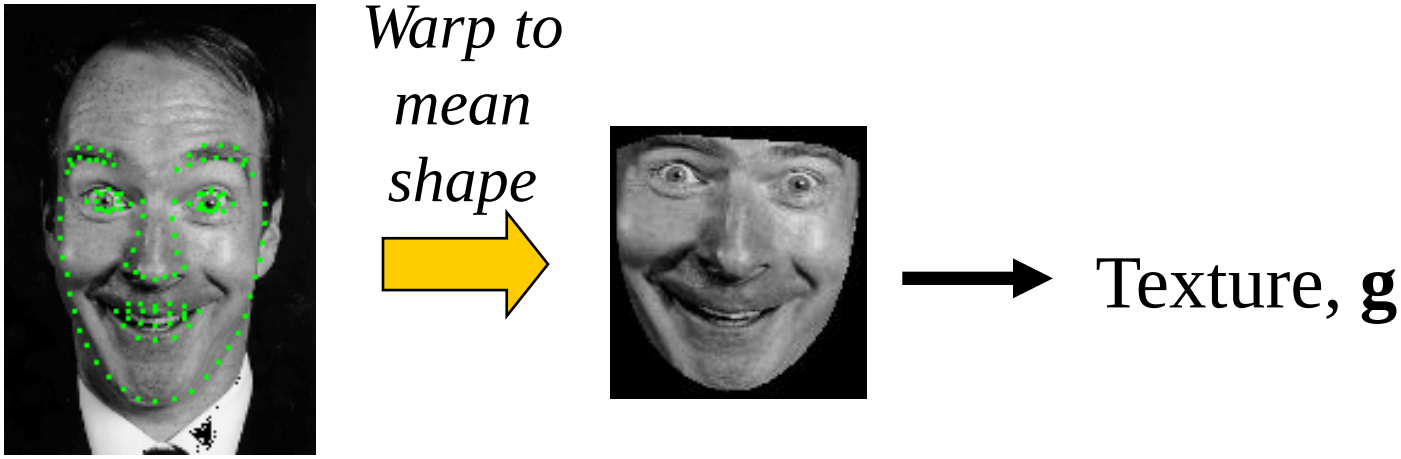


Professor **Baboon** **Warped Professor**

Sparse set of correspondences defines nonlinear deformation field (warping)

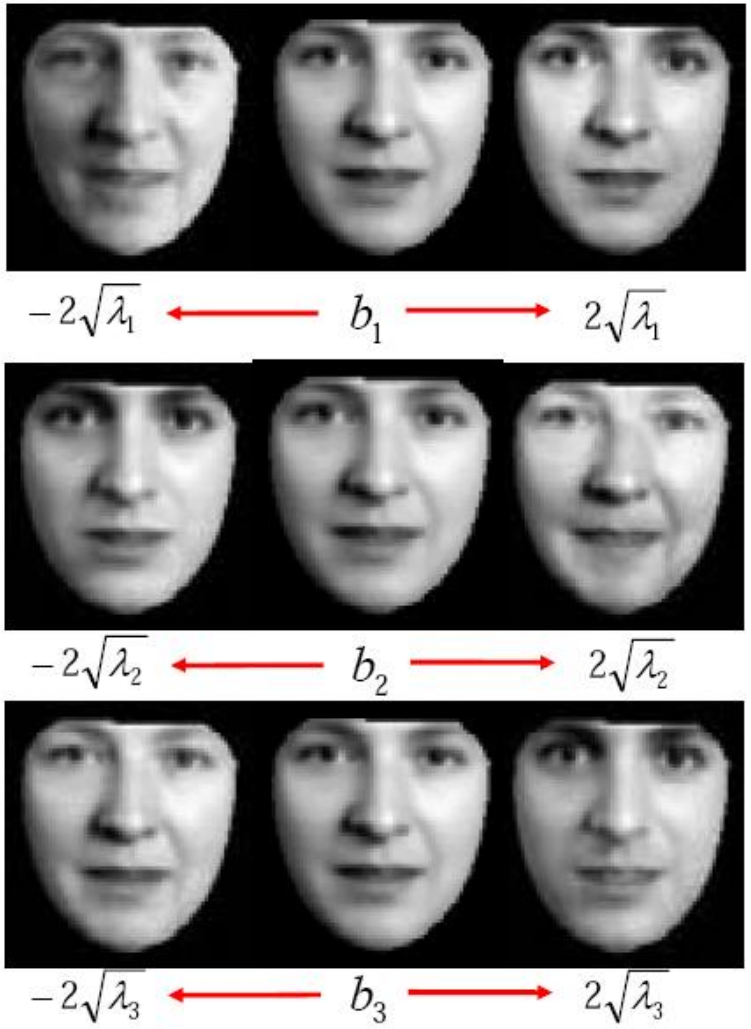
Building Texture Models

- For each example, extract texture vector



- Normalise vectors (as for eigen-faces)
- Build eigen-model $\mathbf{g} = \bar{\mathbf{g}}_g + \mathbf{P}_g \mathbf{b}_g$

Face Texture Model



Textured Shape Modes

- Generate position of control points

$$\mathbf{X} = T(\bar{\mathbf{x}}_s + \mathbf{P}_s \mathbf{b}_s)$$

- Warp mean texture image to control points

(Mean points go to new points X)

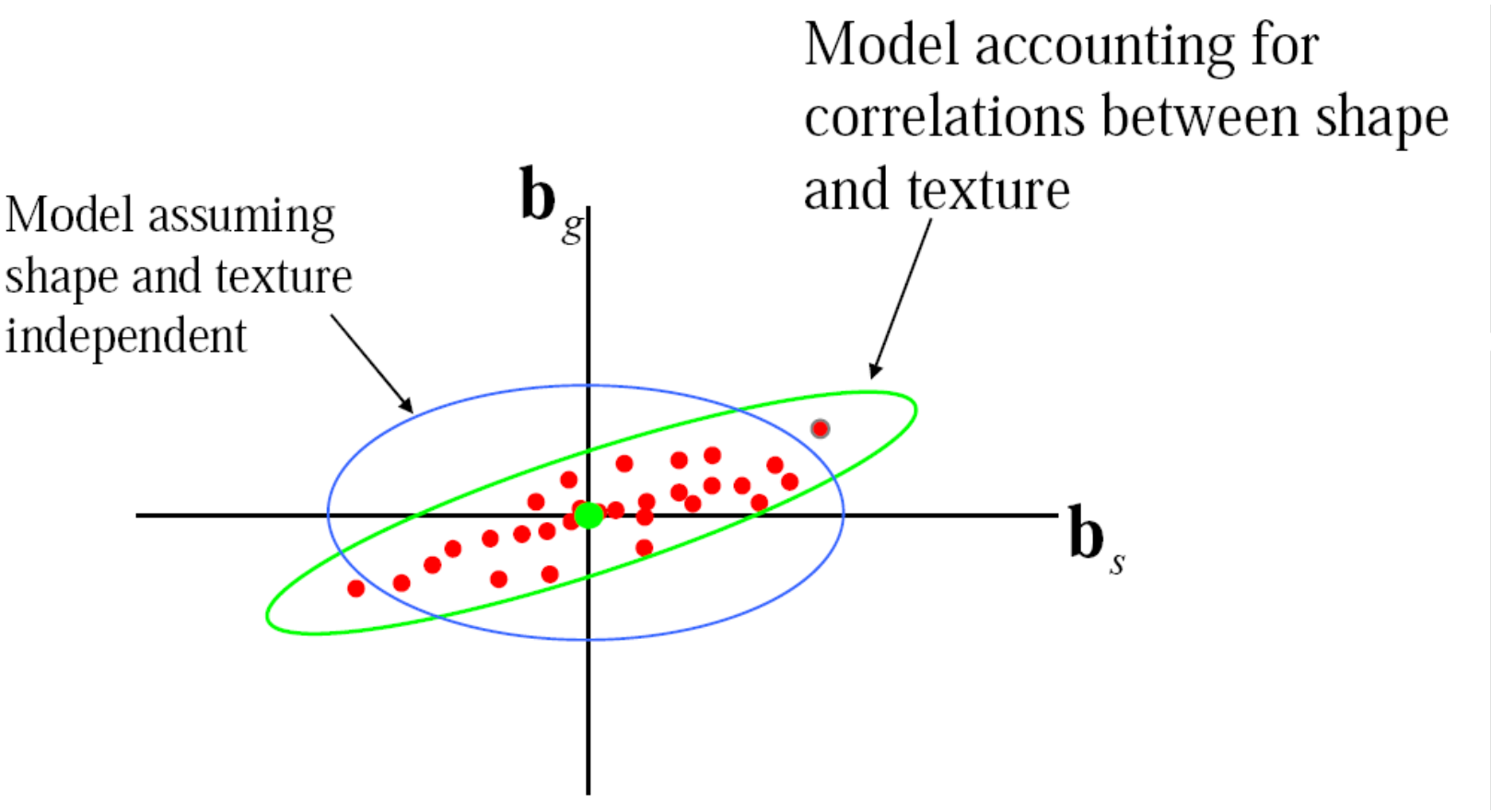
Shape variation (texture fixed)

Visualizes Mean Texture According To Shape Variations

Combined Models

- Shape and texture changes are often correlated!
  - When smile, shadows change (texture) and shape changes
- Learning this correlation leads to a more compact (and specific) model
  - model uses combined model parameters b_c derived from b_g and b_s

Learning Correlations



Learning Correlations

- For each image in the training set we have best fitting shape and texture parameters $\boldsymbol{b_s}$, $\boldsymbol{b_g}$ (both zero mean → $\boldsymbol{b_c}$ zero mean)
- Construct new vector $$\mathbf{b}_c = \begin{pmatrix} \mathbf{W}\mathbf{b}_s \\ \mathbf{b}_g \end{pmatrix}$$
- W relates shape and appearance (units!)
- Apply PCA (mean + eigenvec. of covar.)

$$\mathbf{b}_c = \mathbf{P}_c\mathbf{c} = \begin{pmatrix} \mathbf{P}_{cs} \\ \mathbf{P}_{cg} \end{pmatrix}\mathbf{c}$$

Combined Appearance Models

$$\mathbf{b}_c = \begin{pmatrix} \mathbf{W}\mathbf{b}_s \\ \mathbf{b}_g \end{pmatrix} = \mathbf{P}_c\mathbf{c} = \begin{pmatrix} \mathbf{P}_{cs} \\ \mathbf{P}_{cg} \end{pmatrix}\mathbf{c}$$

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{Q}_s\mathbf{c}$$
$$\mathbf{g} = \bar{\mathbf{g}} + \mathbf{Q}_g\mathbf{c}$$

Varying **c** changes both shape and texture!

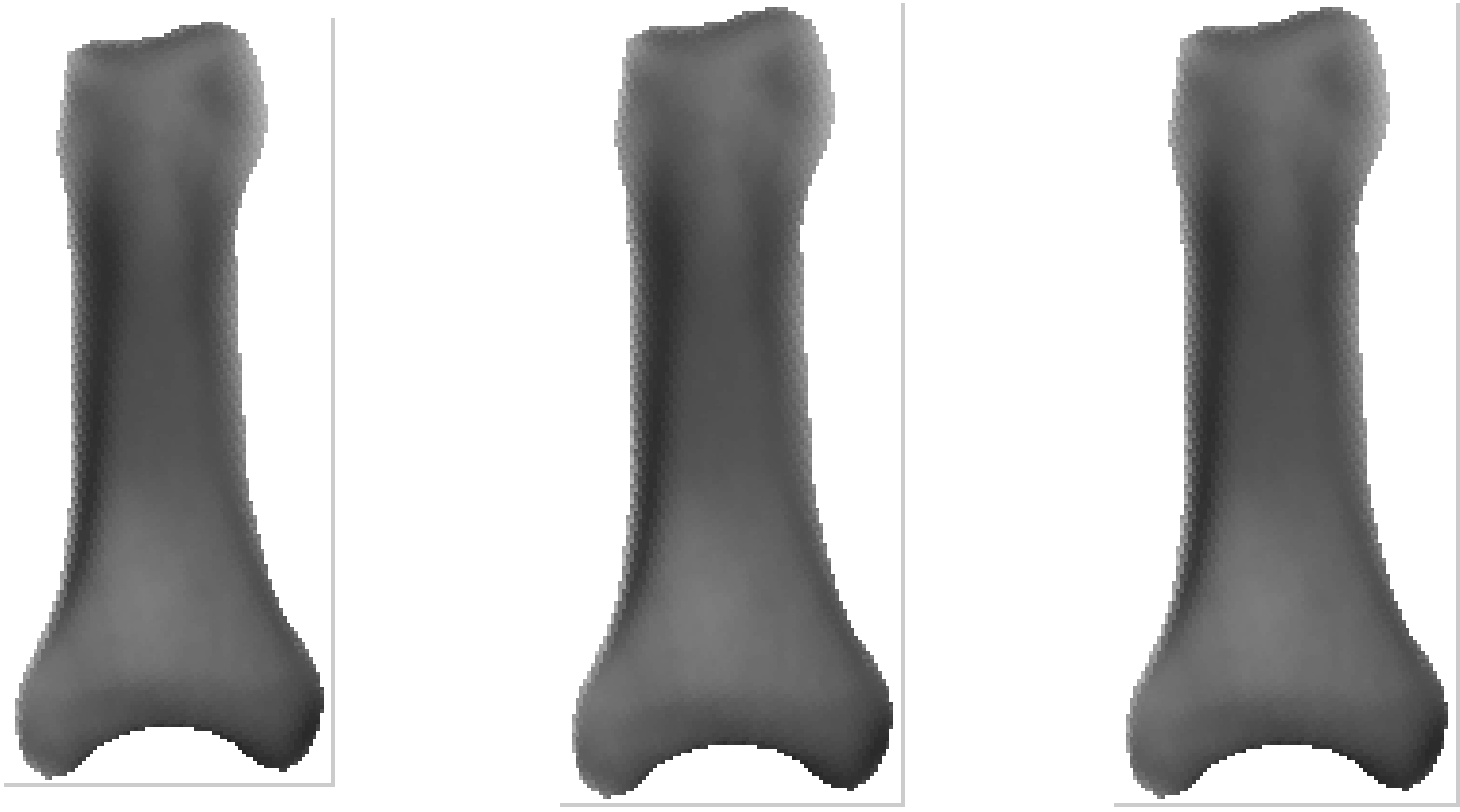
$$\mathbf{Q}_s = \mathbf{P}_s\mathbf{W}_s^{-1}\mathbf{P}_{cs}$$
$$\mathbf{Q}_g = \mathbf{P}_g\mathbf{P}_{cg}$$

Choice of W:
e.g. W = rI
with $r^2 =$ $\frac{\text{total intensity variation}}{\text{total shape variation}}$

Combined Appearance Model

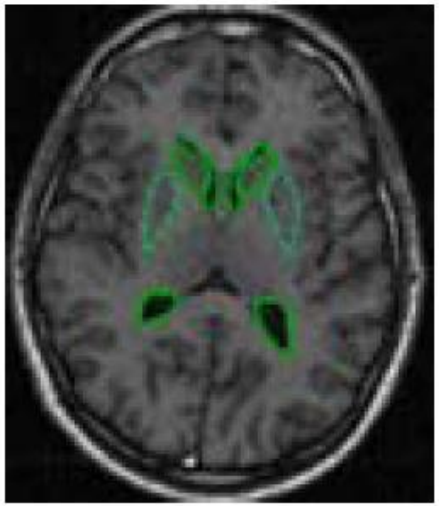
- Generate shape X and texture g
- Warp texture so mean control points lie on new X

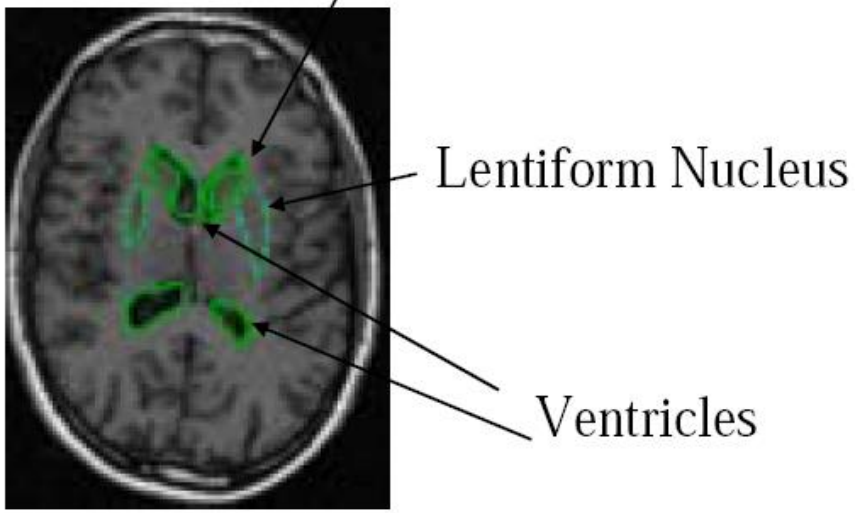
AAM Variations



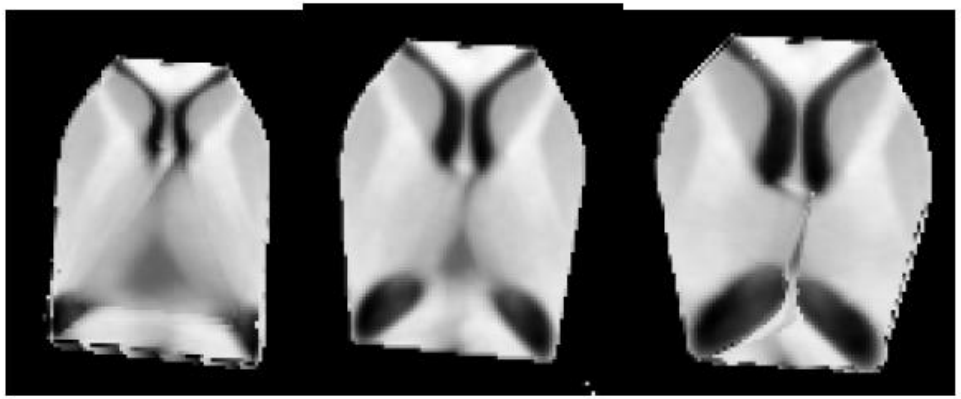
Sub-cortical Structures

- 72 examples
- 123 points
- 5000 pixel model

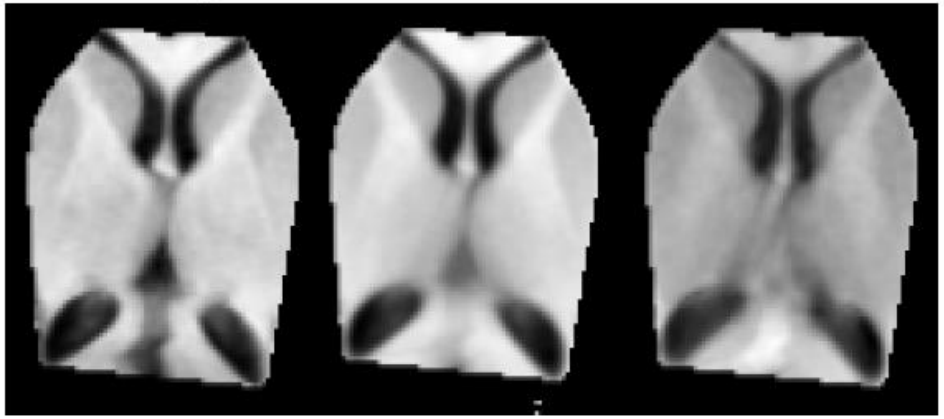




Shape and Texture Modes



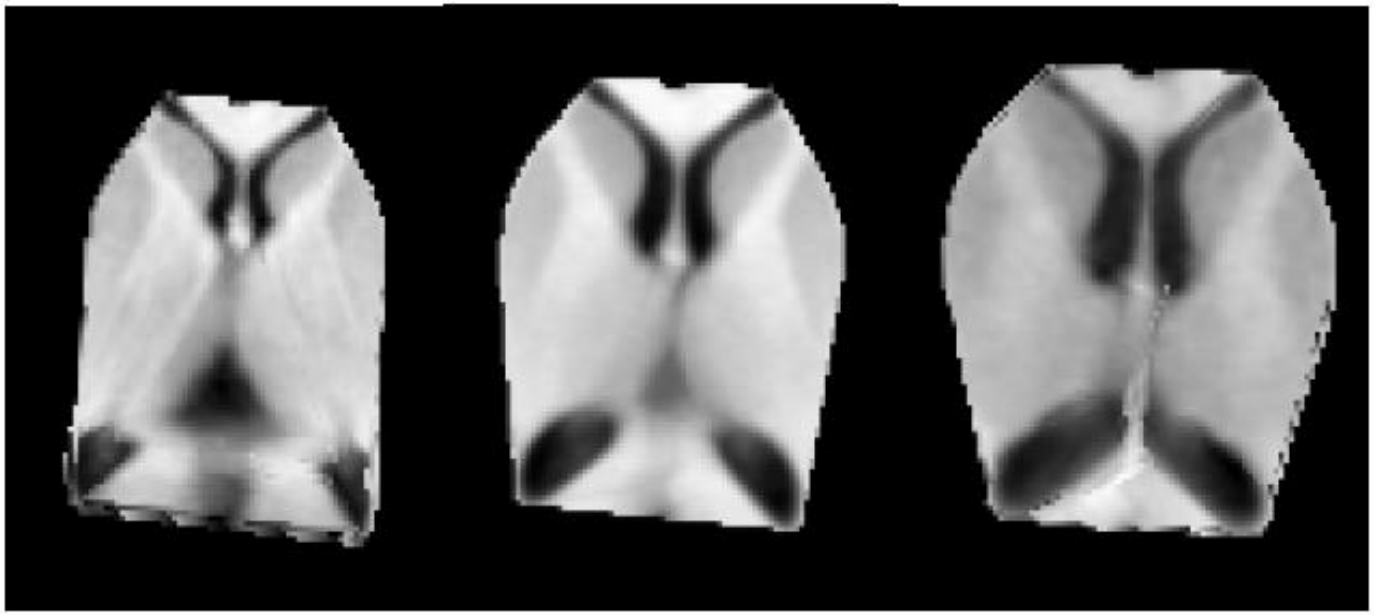
Shape variation (texture fixed)



Texture variation (shape fixed)

Combined Appearance Model

- Shape and texture correlated



Summary

- Now we have components ready to train a model with flexibility in both shape and appearance
- A compact representation is derived by using the combined formulation (i.e. model correlations of shape and texture)

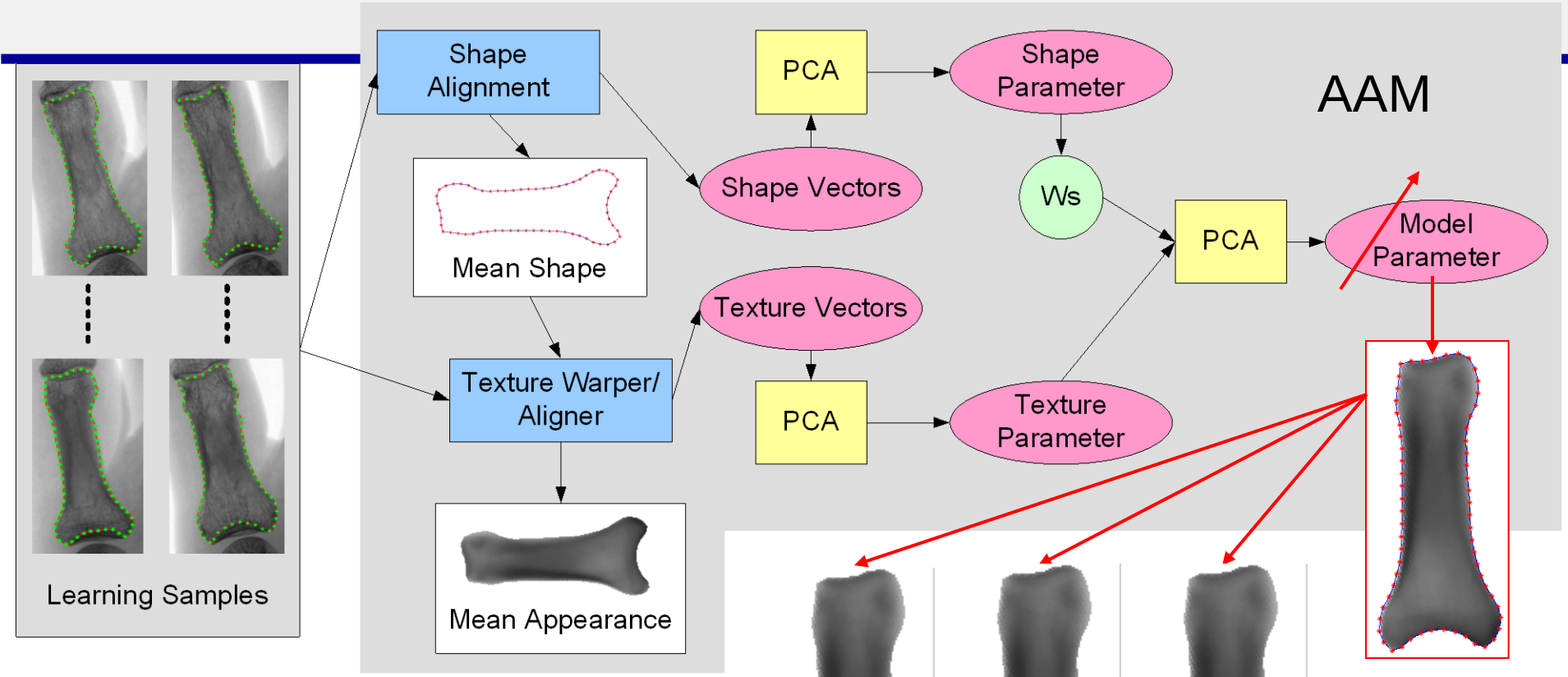
- Next: fit an Active Appearance Model to a given image

Active Appearance Models (AAM)

„How to fit a statistical shape & appearance model to unseen images“ [+]

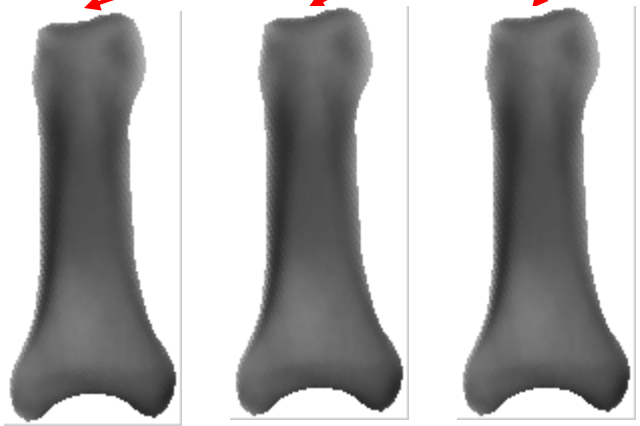
[+] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *Proc. European Conference on Computer Vision*, volume 2, pages 484–498. Springer, 1998.

Active Appearance Models (AAMs)



Proximal Phalanx
X-Ray Images

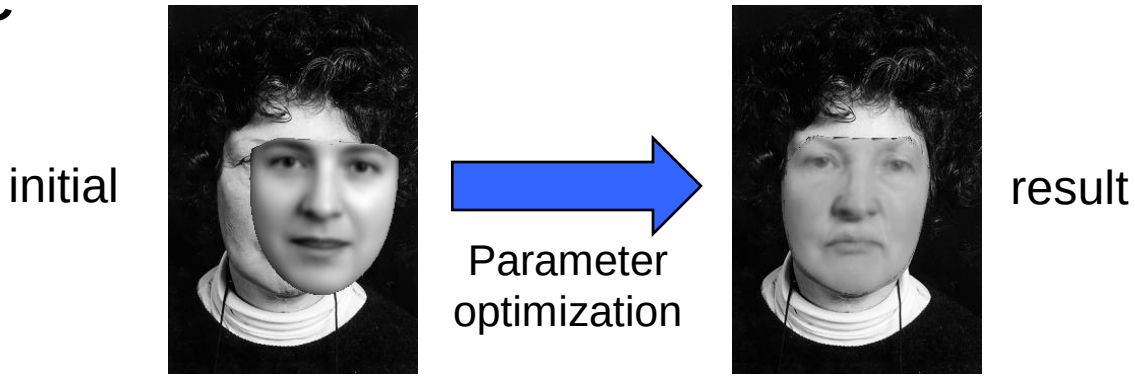
Variations 1, 2, 3



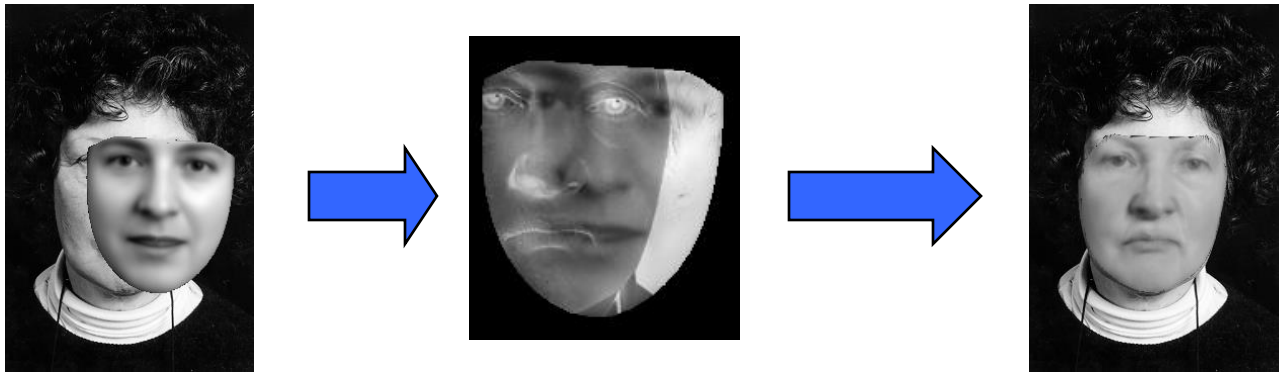
Average

Active Appearance Model Fitting

- Start from generative statistical shape & appearance model
- Nonlinear optimization technique that iteratively matches initial and current synthetic model instances to the given image



Interpreting Images



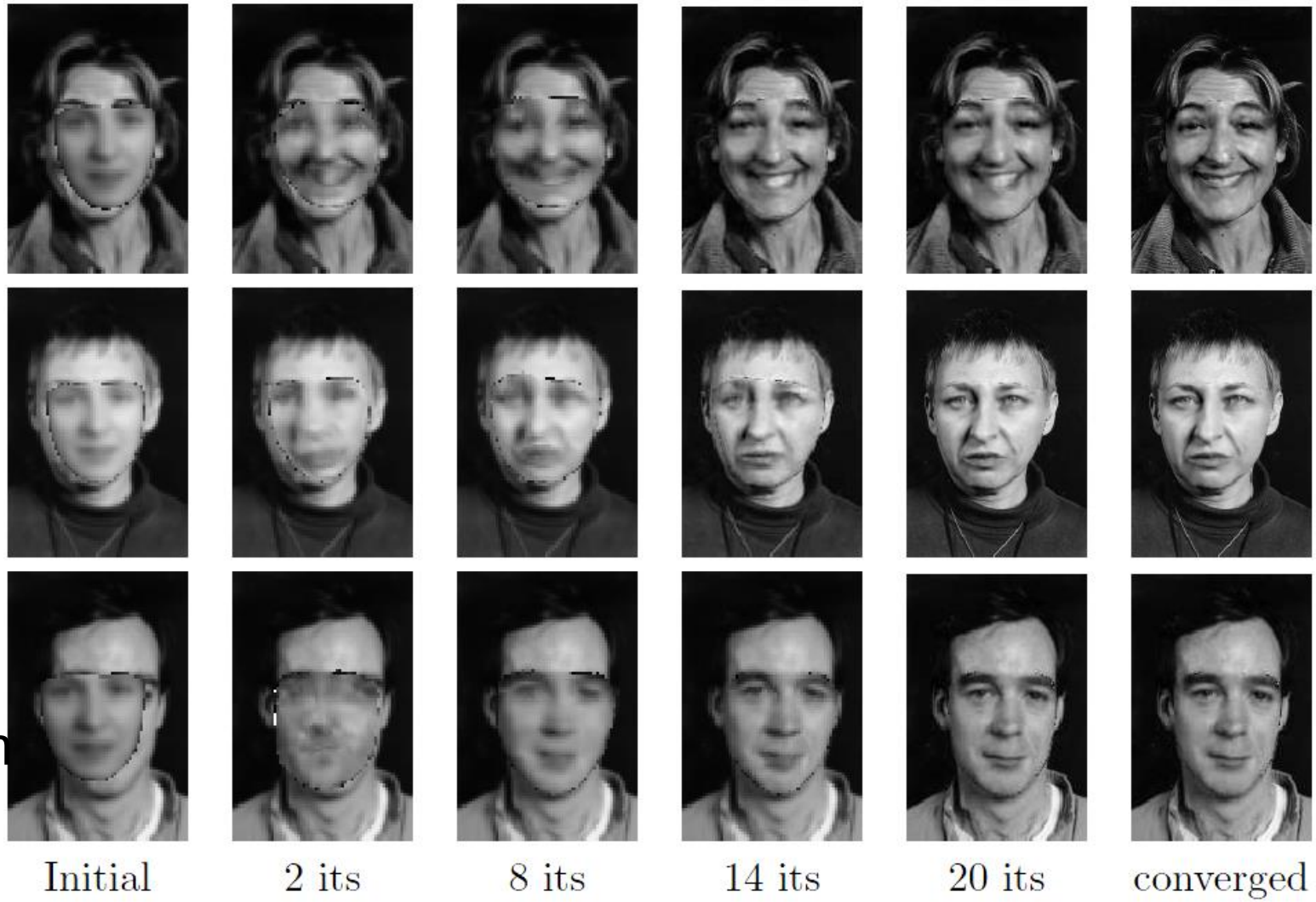
Place model in image

Measure Image Difference

Update Model Parameters

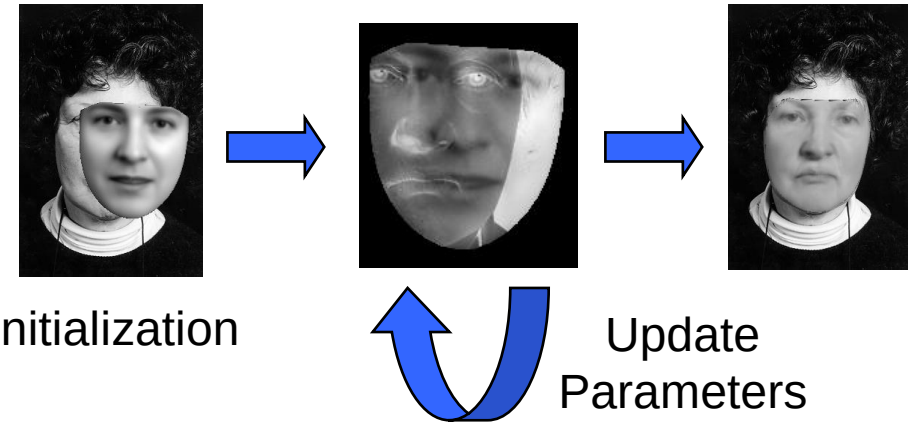
Iterate over unknown pose and model parameters

Interpreting Images



Multi-Resolution

Image Interpretation Workflow



Requirements:

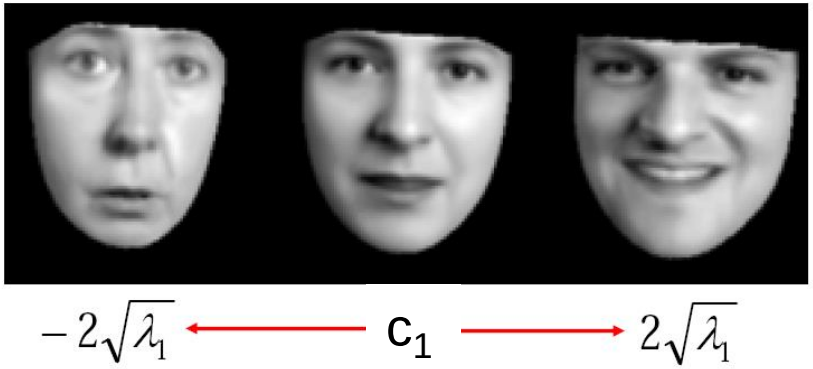
- Initial Solution (e.g. object detection)
- Error Function E over unknown pose and model parameters

$$E(\mathbf{c};\Theta)$$

- Image synthetization from $\mathbf{c};\Theta$
- Optimization strategy that minimizes E

Image Synthetization

- Given: Parameters $\mathbf{c};\Theta$
- Generate Image $\mathbf{I}(\mathbf{c};\Theta)$



- Combined AAM parameters $\mathbf{c}$
  - Shape parameters
  - Texture model parameters

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{Q}_s\mathbf{c}$$

$$\mathbf{g} = \bar{\mathbf{g}} + \mathbf{Q}_g\mathbf{c}$$

- Pose parameters
  - Translation, scale, rotation

$$\Theta = (\mathbf{X}_c, \mathbf{Y}_c, s, \theta)$$

- Define $\mathbf{p} := (\mathbf{c};\Theta)$

Error Function

- Residual difference:

$$\mathbf{r}(\mathbf{p}) = \mathbf{I}_{\text{model}}(\mathbf{p}) - \mathbf{I}_{\text{image}}$$

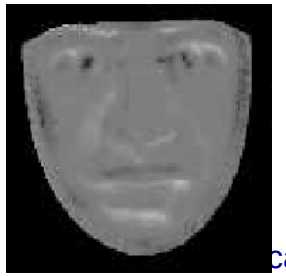
Residual Difference Image

- We need to find the parameters which minimize the sum of squares of the residual

$$E(\mathbf{p}) = \left\|\mathbf{r}(\mathbf{p})\right\|^2$$

→ Cost function optimization

- Reach local optimum of E, minimal residual difference



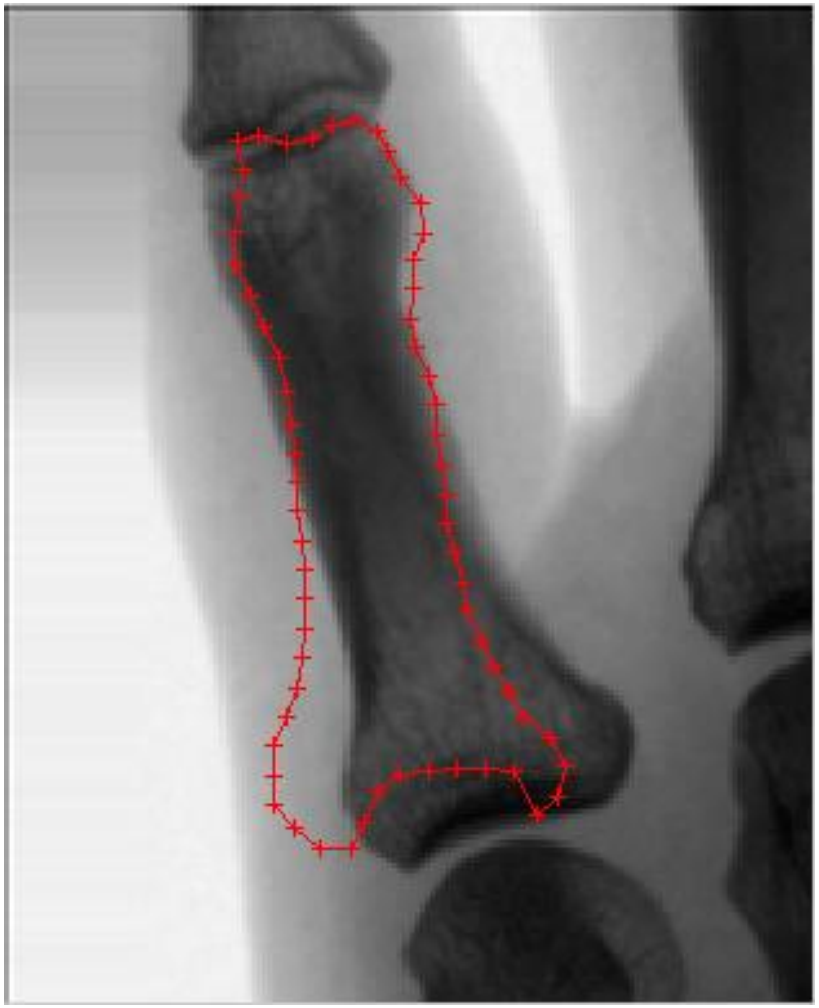
Optimizing the Match

- We must find $\mathbf{p}$ to minimize $E(\mathbf{p})$
- $\mathbf{p}$ may have many (50's-100's) of dimensions
- We could use a standard multi-dimensional nonlinear optimizer
  - Steepest Descent, Levenberg-Marquardt
  - Problem: many dimensions of p -> gradients of E are costly to compute because gradient for each dimension leads to a synthesized image!
- We can find an approximation to find an acceptable solution rapidly
  - see (Cootes, Technical Report, p. 45-46) for details

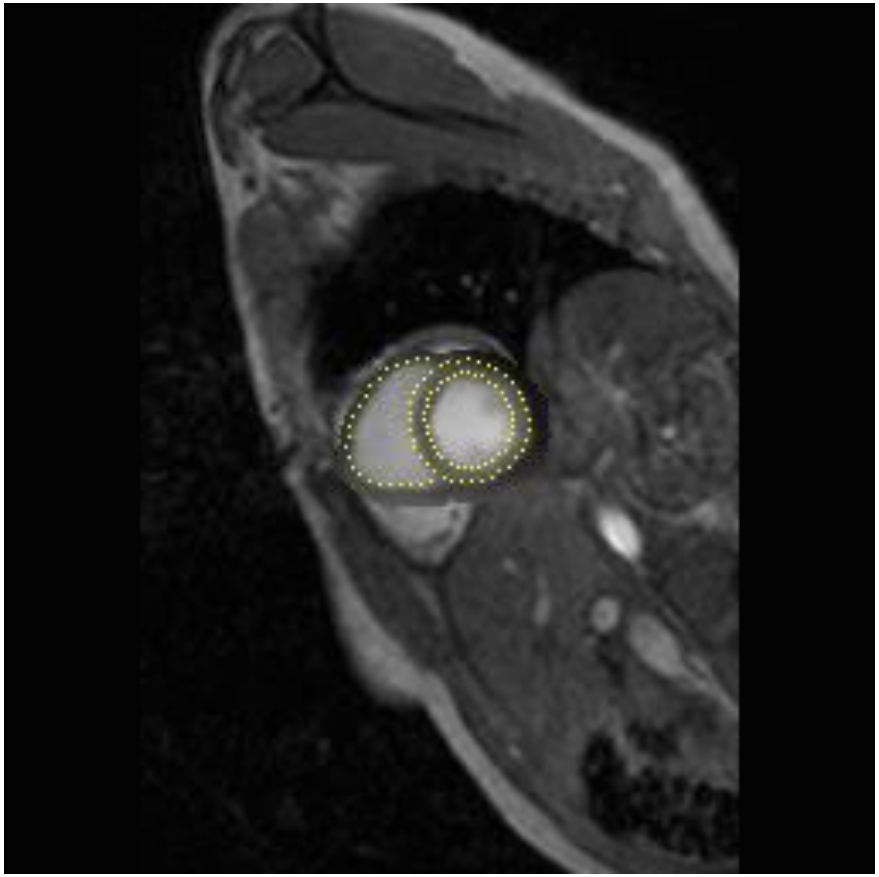
AAM Algorithm

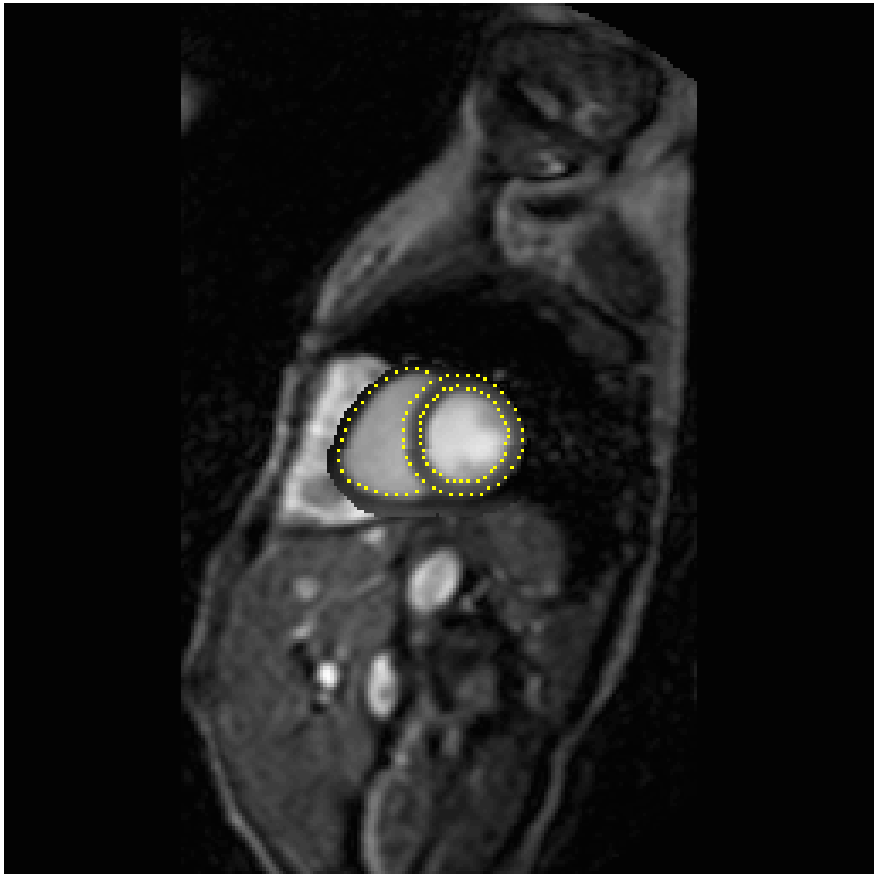
- Initial estimate model and pose parameters, calculate $\mathbf{I}_{model}(\mathbf{p})$
- Iterate:
  - Measure residual error, $\mathbf{r}(\mathbf{p})$
  - predict correction $\delta\mathbf{p}$ (see TR of Cootes)
  - $\mathbf{p} \Rightarrow \mathbf{p} + \delta\mathbf{p}$
  - repeat to convergence
- Looks simple, but hard to implement!

Example – AAM Matching



Example – AAM Matching





END

Thank you for your attention!