Active Appearance Models
Learning as an Analysis Tool

Georg Langs
georg.langs@ecp.fr

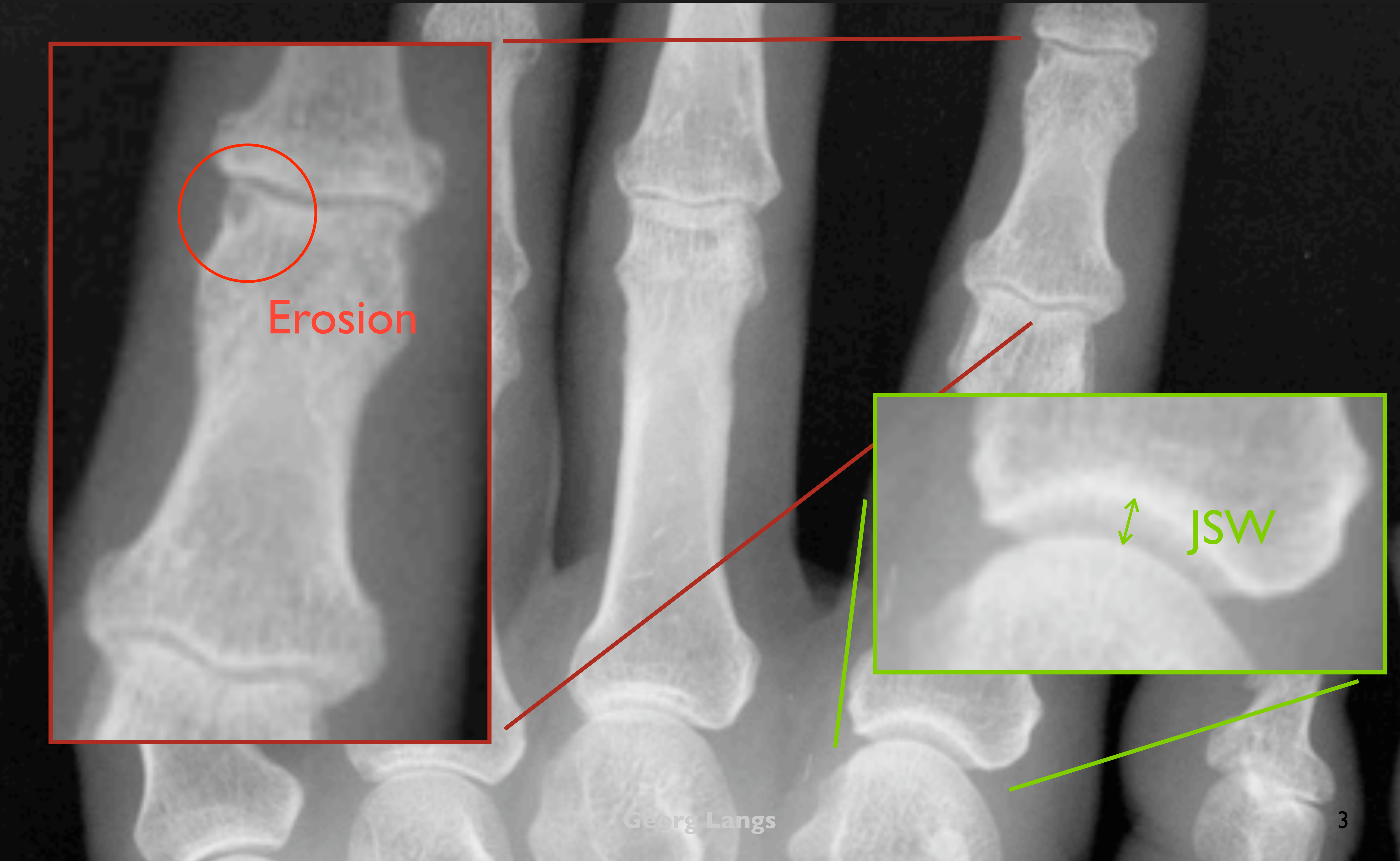
MAS Lab - Ecole Centrale Paris, France

GALEN Group - INRIA Saclay, France

Vast amount of data

- Structure the data and make use of it
- Localize and analyze anatomical structures
- **Build models of anatomical structures**
  - They should by able to find structures in new data
  - We want to learn them supervised ...
  - ... or even better: un-supervised

Example



Grasping bone contours

We have to localize anatomical structures

Modeling bone contours

Outline

1. Active Appearance Models (AAMs)
2. Autonomous Model Building
3. Structuring the Model: Shape Maps

1. Active Appearance Models

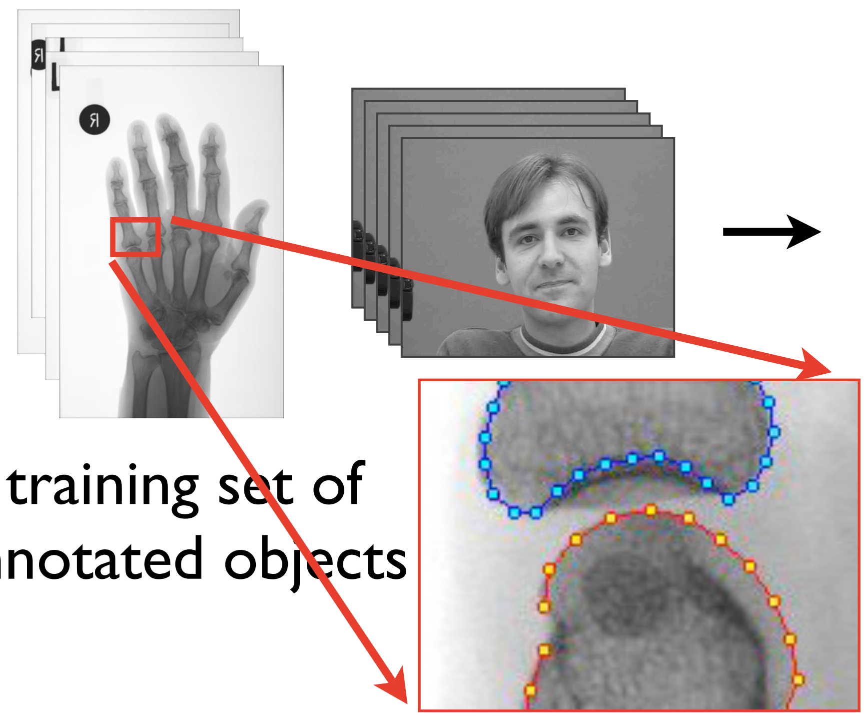
Active appearance models

- Idea is to build a model of shape and appearance
- **Statistical model of shape variation**
- **Statistical model of entire texture *living* within the shape**
- Build the model based on a training set
- Search in new images by fitting the model to the image content

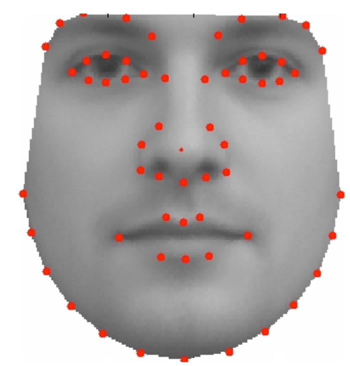
[Cootes et al. PAMI 2002]

AAM Concept

Building



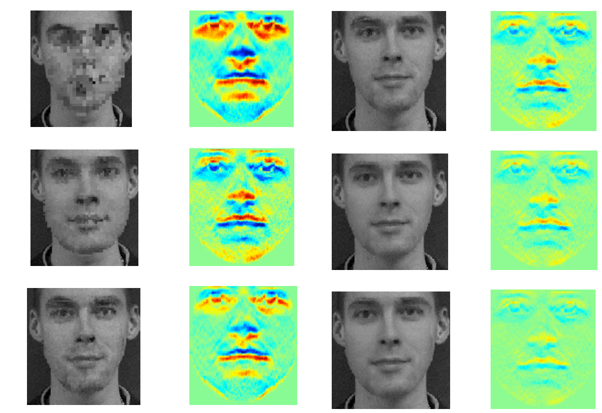
A training set of annotated objects



Shape model

Texture model

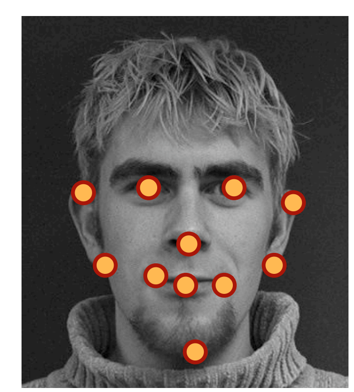
Training



Train search algorithm with artificial displacements of model paramaters. Model correlation between ∂p and residual.

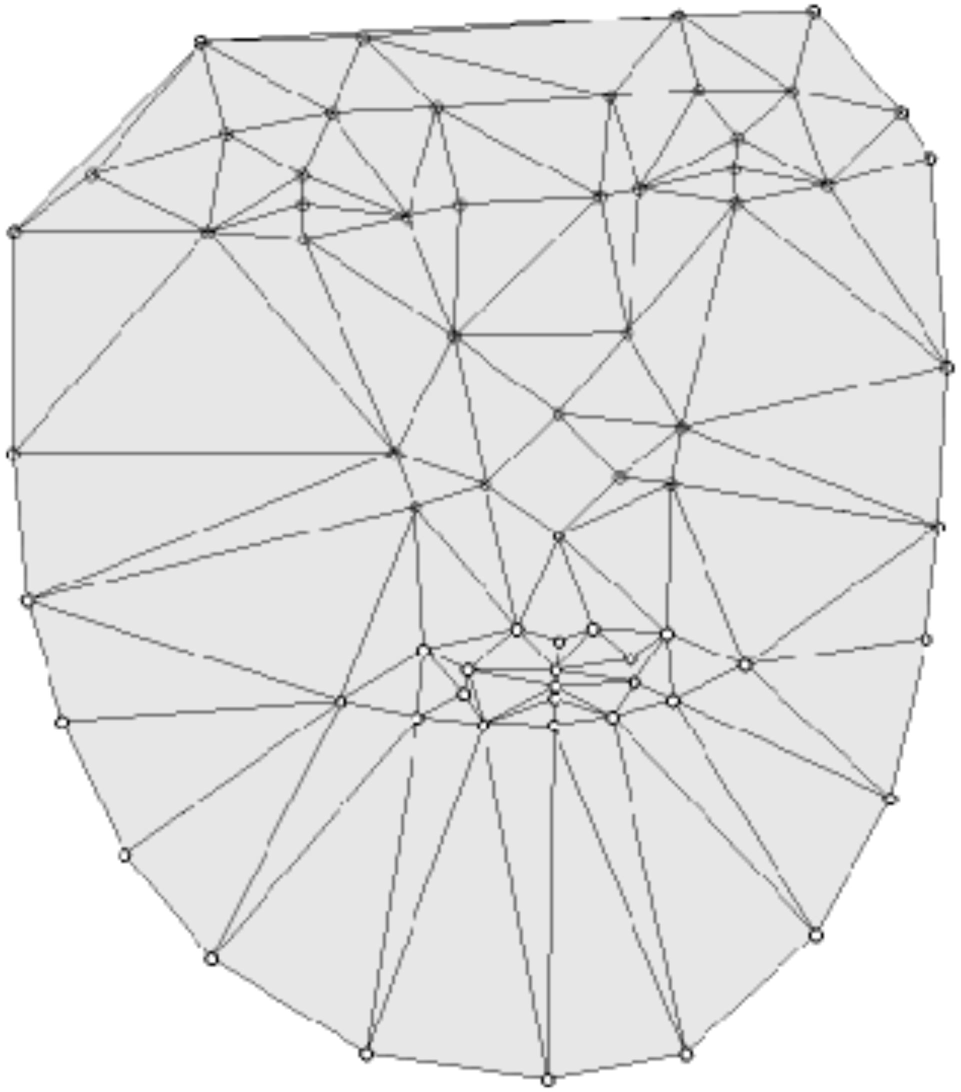
Search

Iteratively calculate residual, ∂p and update model fit.



[PAMI06, ICPR06]

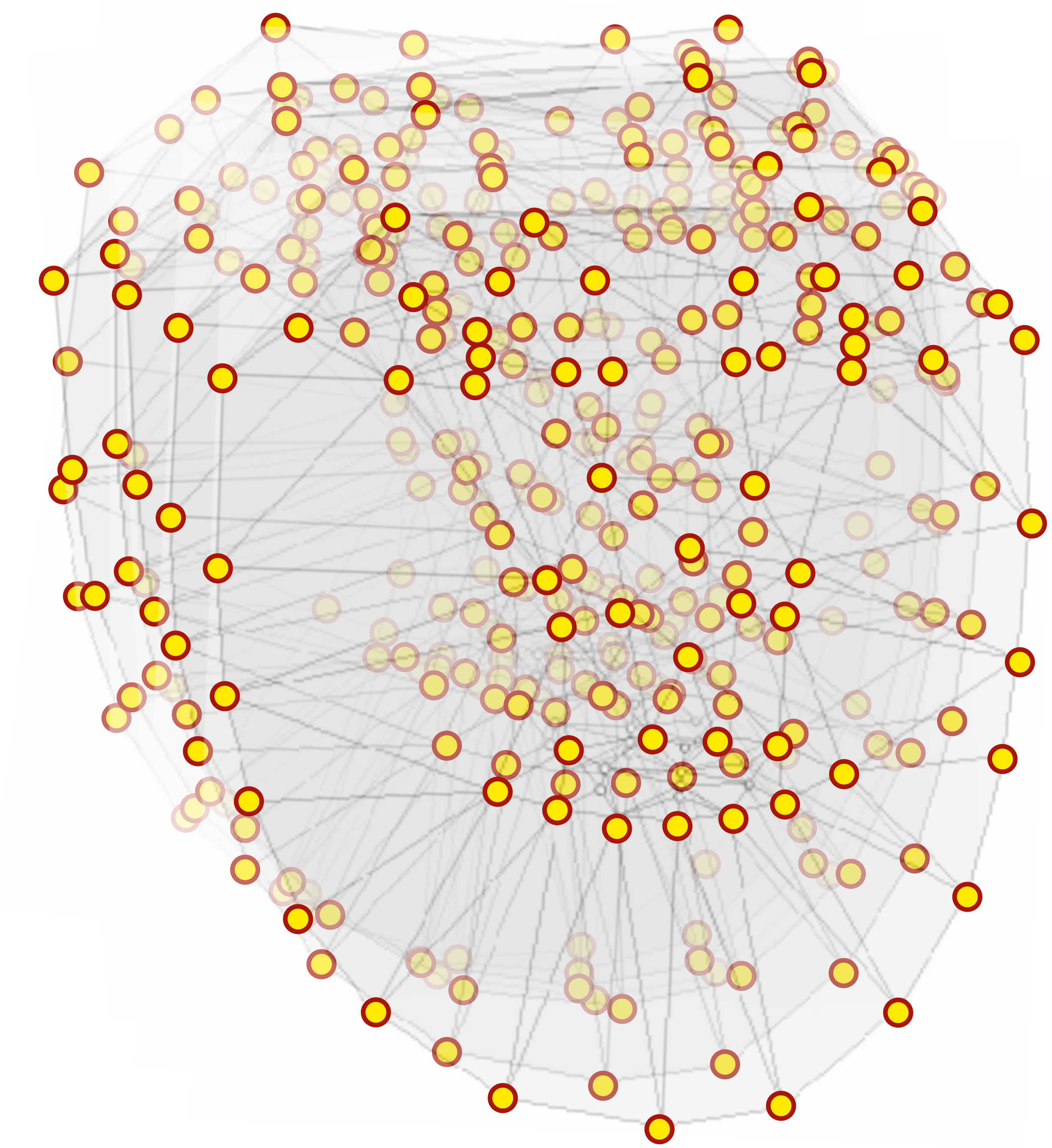
1. Shape model



- For a set of landmarks on the training images

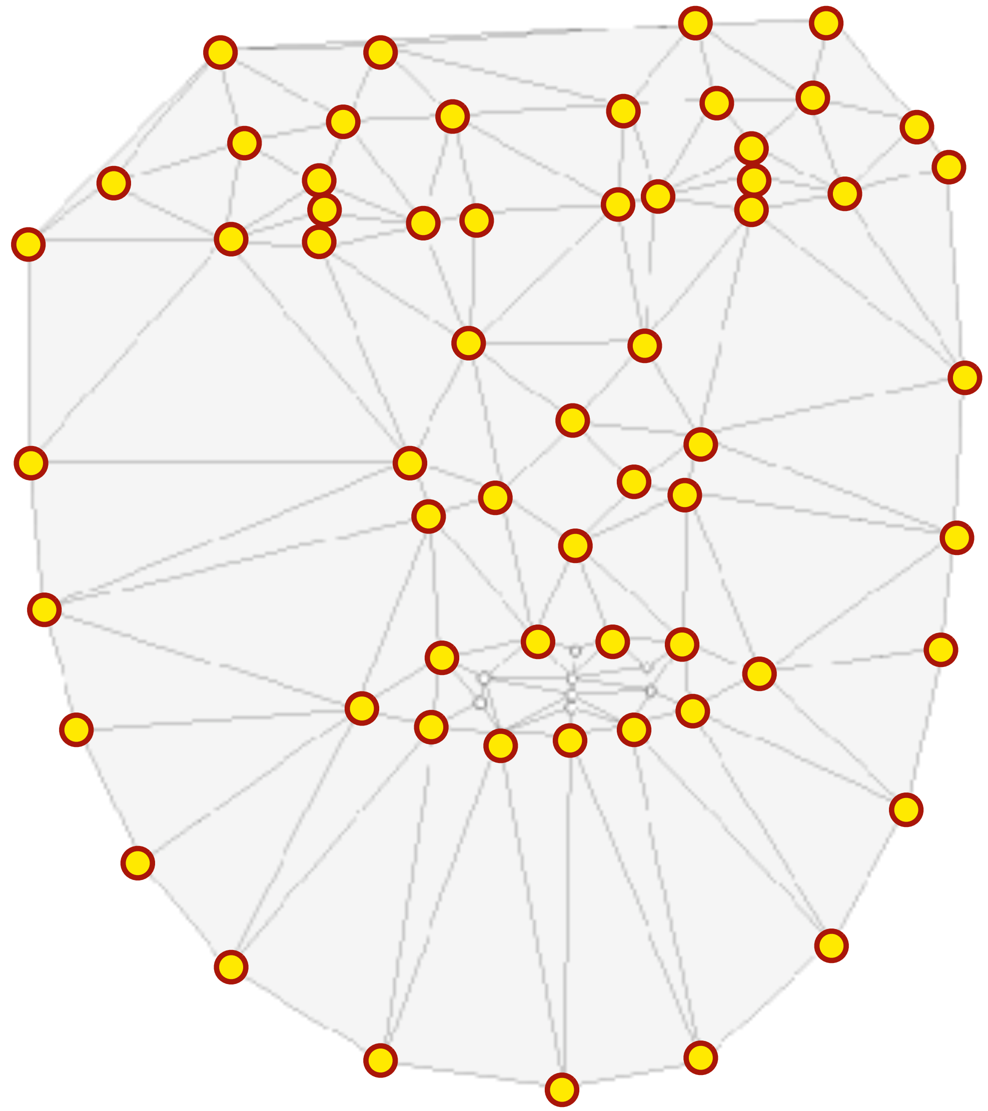
Shape representation

- AAM represent shape based on **landmarks**
- For a set of landmarks, positions are known on each training image - correspondences

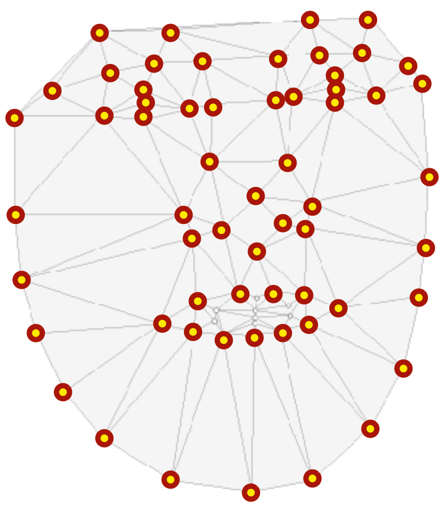


Shape representation

- The sets of landmarks are aligned to exclude rotation, translation, and scaling variation
- Then PCA is performed on the shape vectors

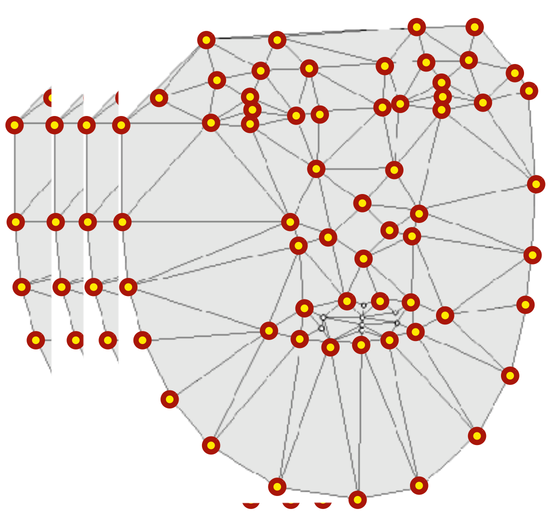


PCA on the shape vectors



$$\mathbf{x}_i = \begin{pmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ x_{m/2} \\ y_{m/2} \end{pmatrix}$$

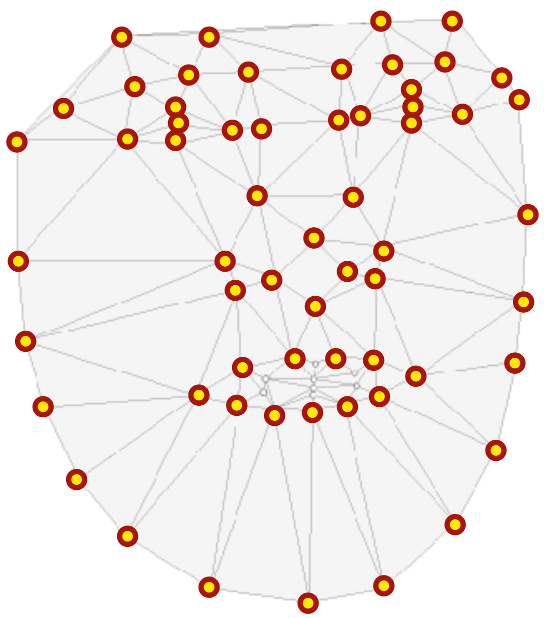
Each example is represented by a vector encompassing the coordinates of the landmarks.



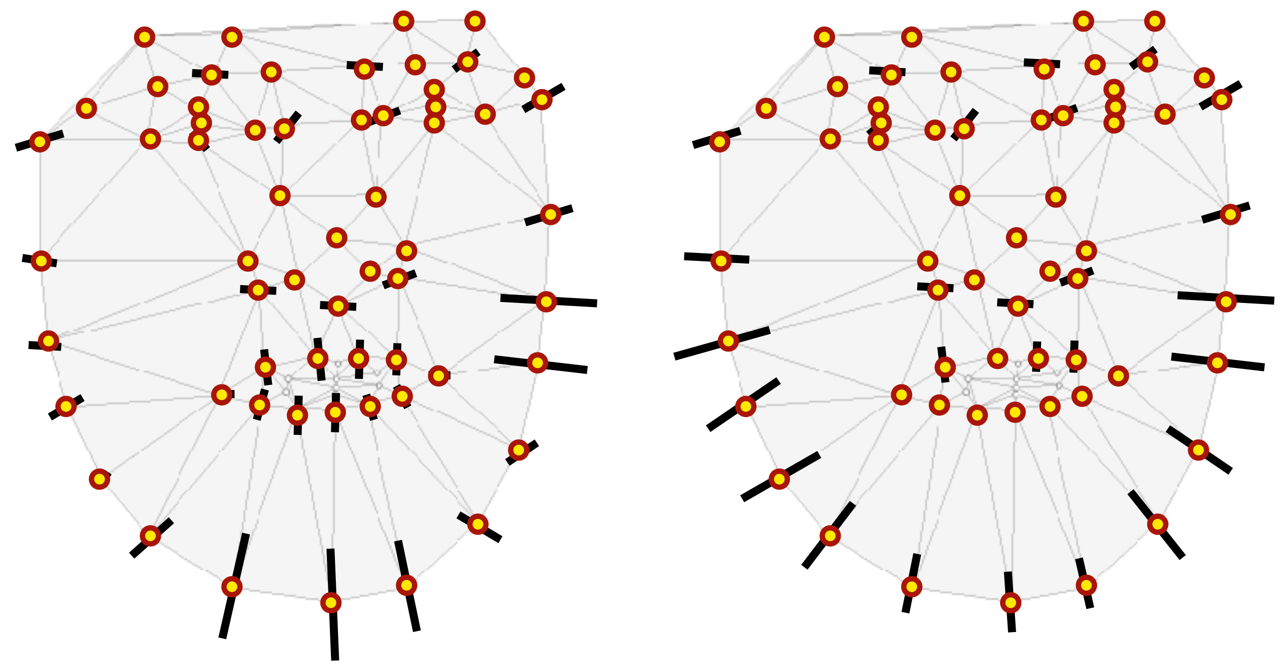
$$\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$$

After alignment the set of training examples is used to build a statistical model of shape variation.

Shape model



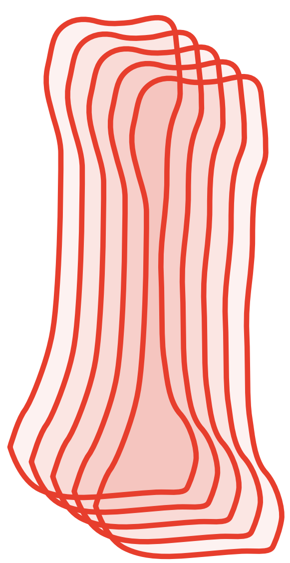
Mean shape

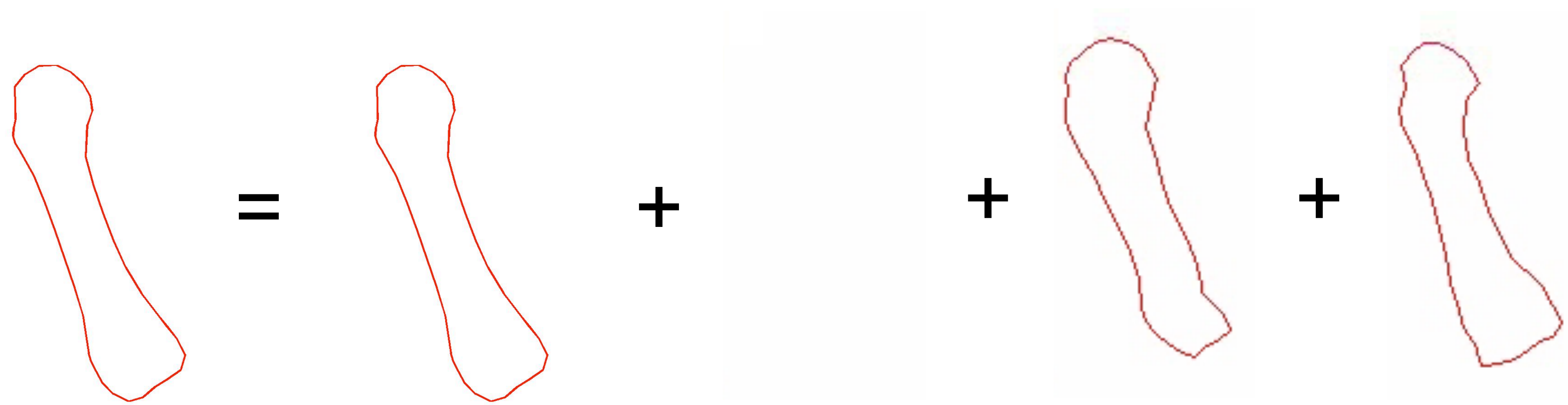


Modes of shape variation

- The PCA results in a statistical shape model, comprising mean shape and a set of modes - the eigenvectors of the covariance matrix which are plausible deformations of the shape.

Using PCA to model shape

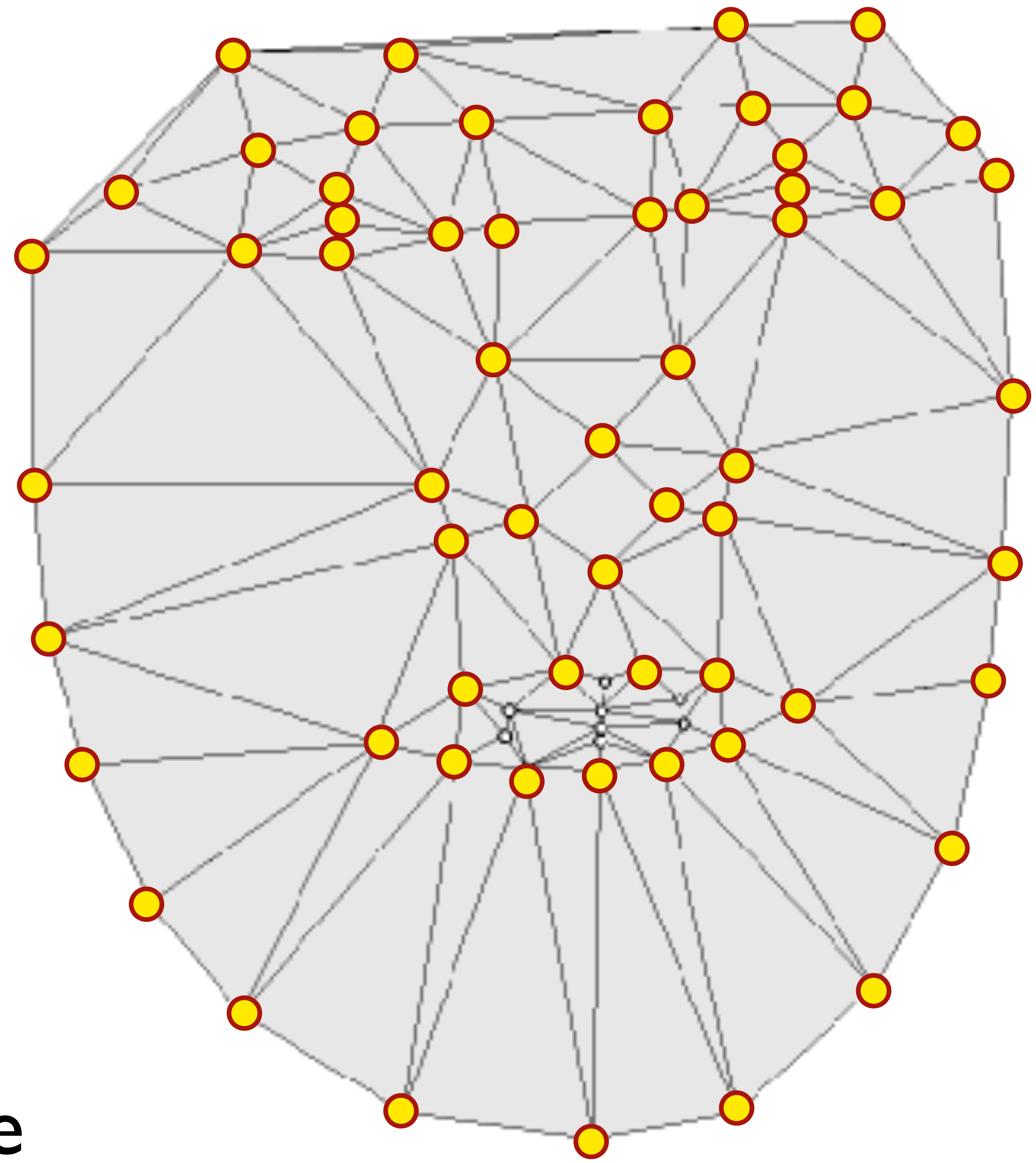




$$\mathbf{x}_{new} = \hat{\mathbf{m}} + b_1\mathbf{e}_1 + b_2\mathbf{e}_2 + b_3\mathbf{e}_3$$

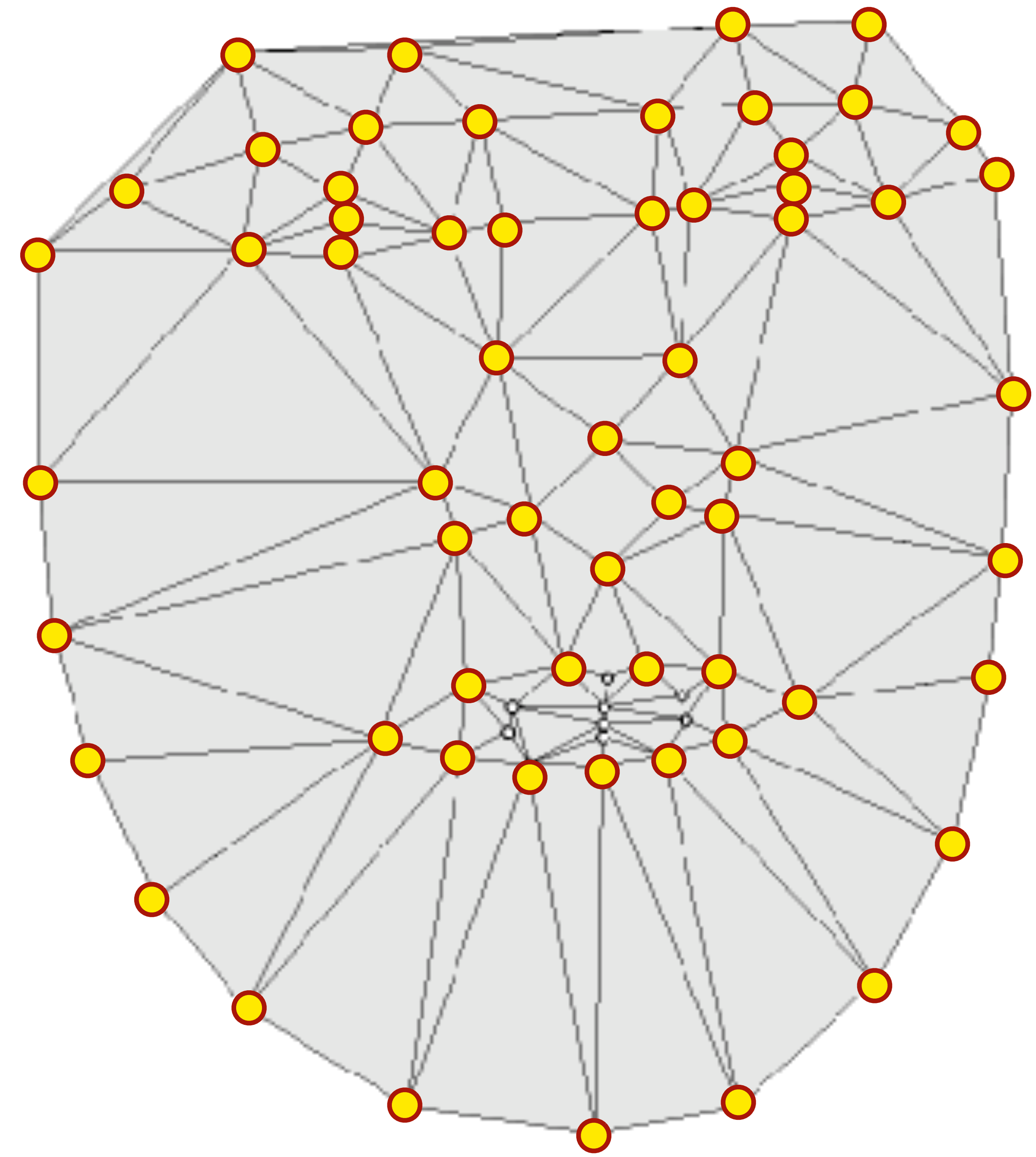
Texture model

- Unlike as ASMs, AAMs represent the entire texture enclosed by the landmarks
- The area that is covered is defined by
  - The convex hull of the landmarks, or a more restrictive hull, or
  - By the hull spanned by additional automatic landmarks surrounding the shape



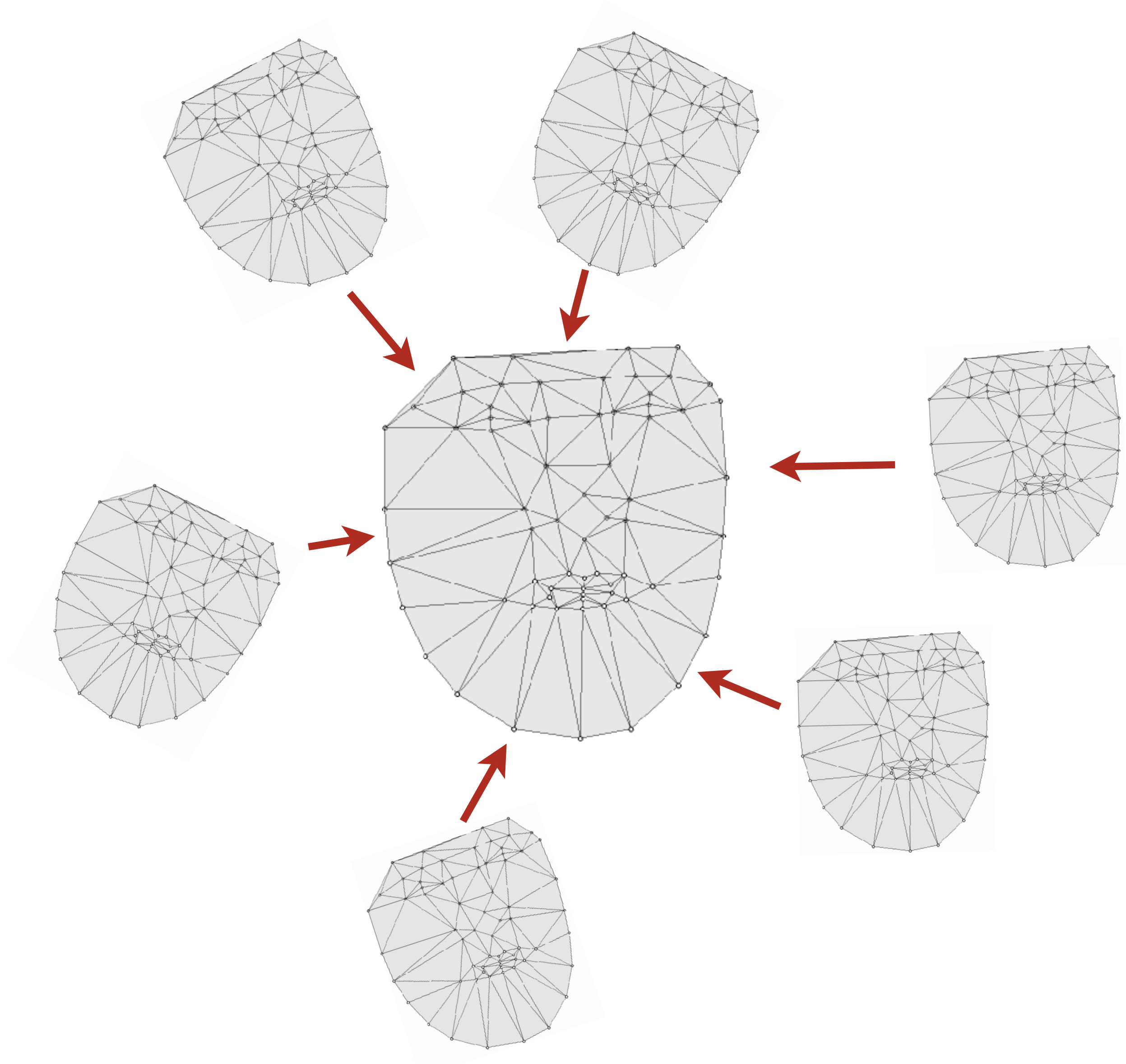
How to capture the texture?

- Triangulate the mean shape
- Propagate this triangulation to all training shapes
- Model the texture mapped onto the mean shape by warping all training shape triangles onto the mean shape triangles

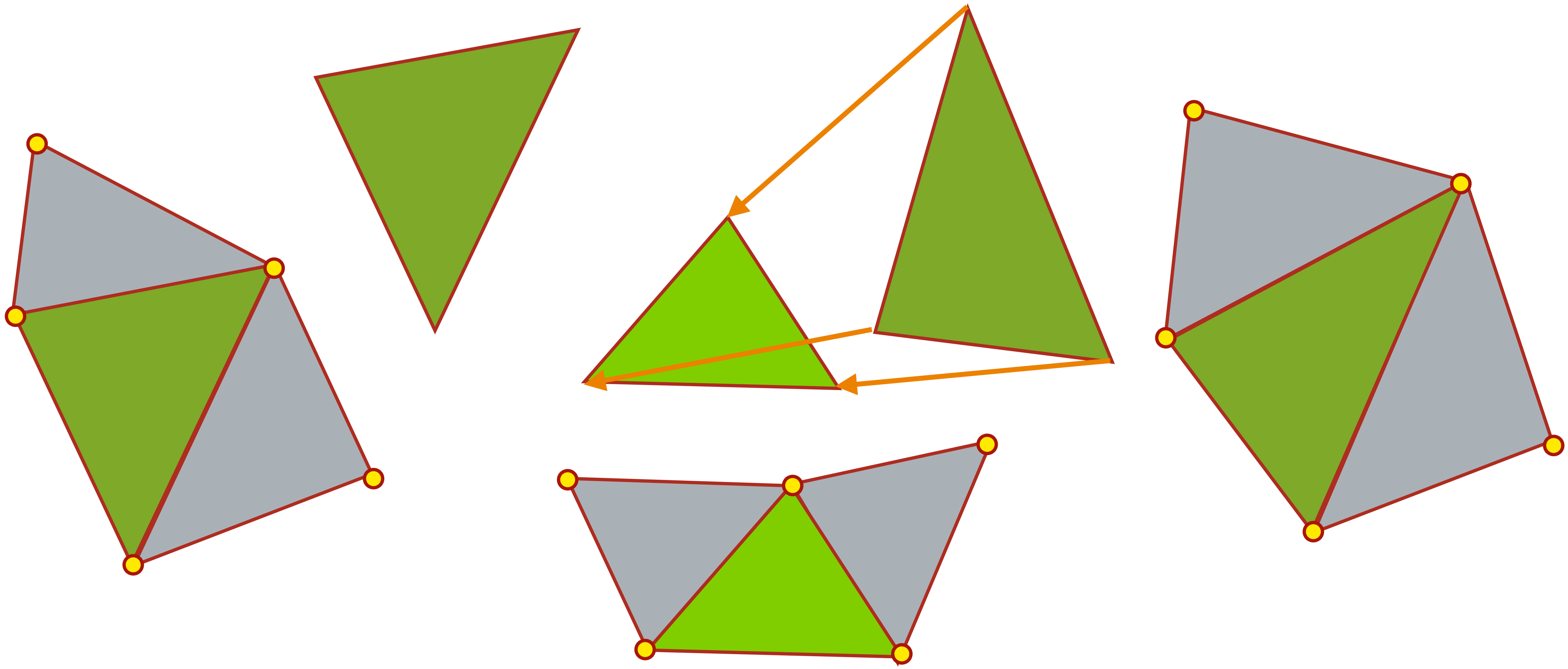


Texture representation

- The texture is represented by a normalization with respect to the shape
- All examples are mapped to the mean shape
- Then the texture model is built



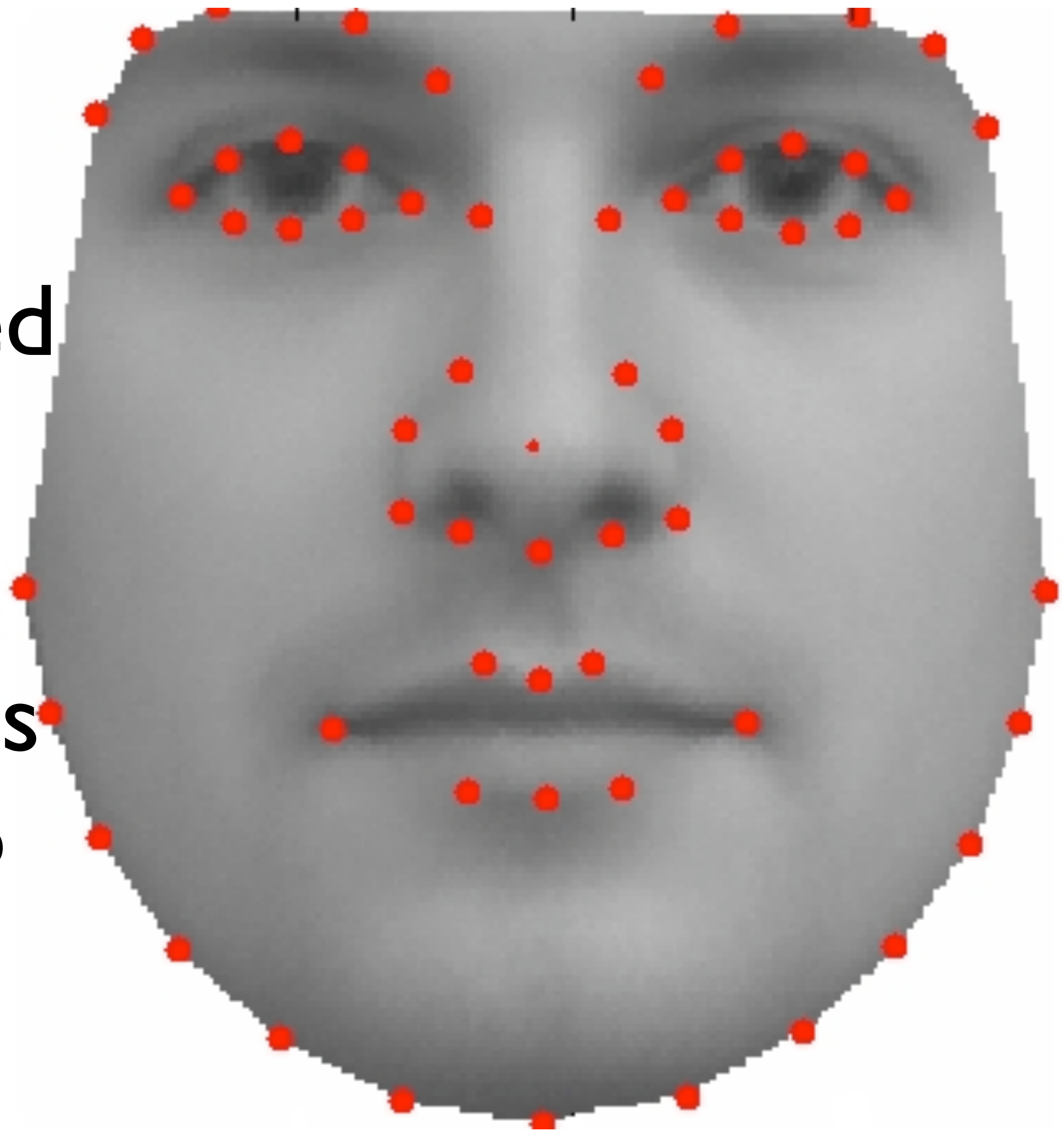
Texture mapping



- The triangles of the training examples are mapped onto the triangles of the mean shape

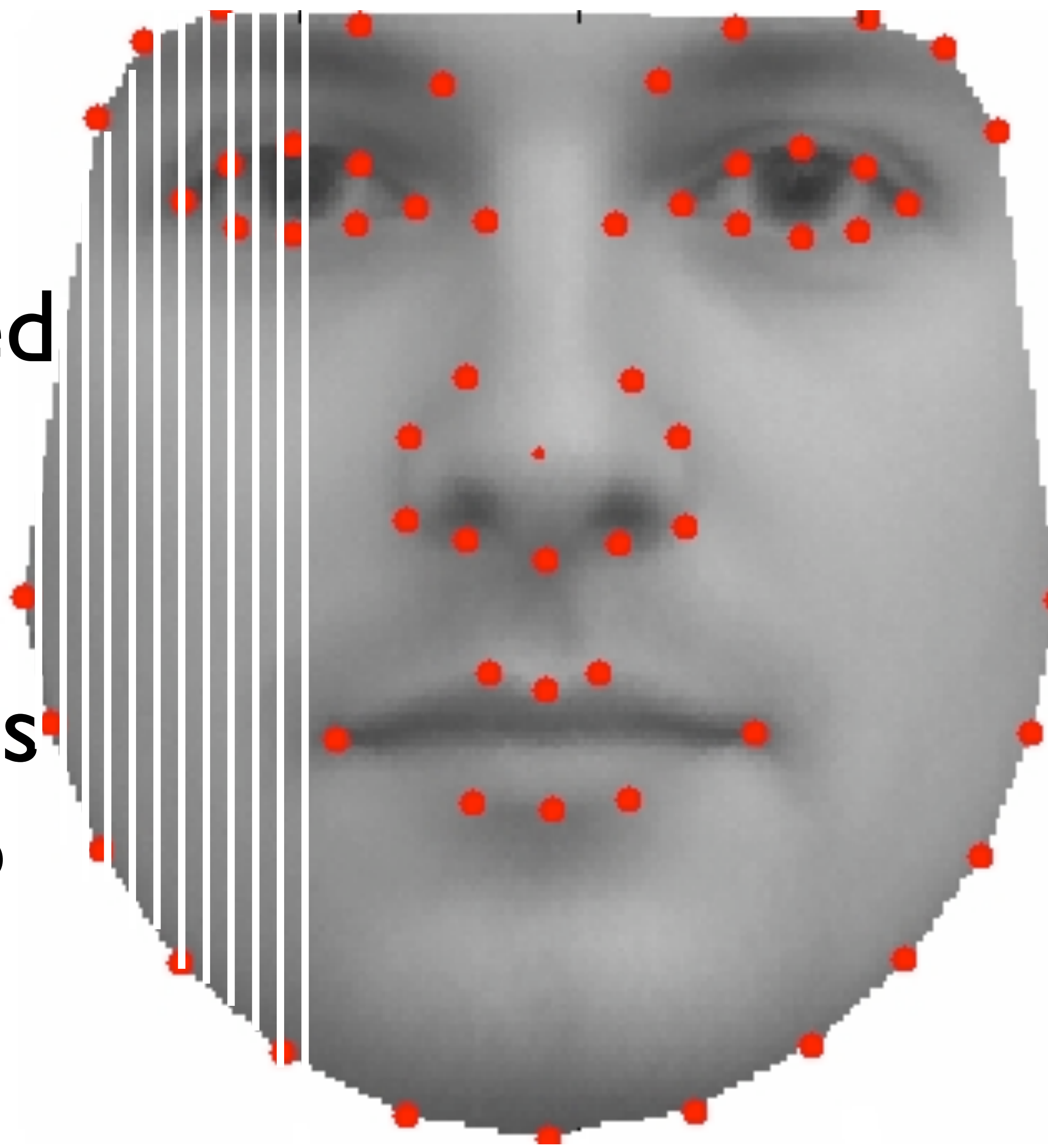
Texture model

- After the shape normalization a mean texture can be calculated
- Analogous to the shape vectors the gray values of the mapped texture is read out columnwise to form texture vectors



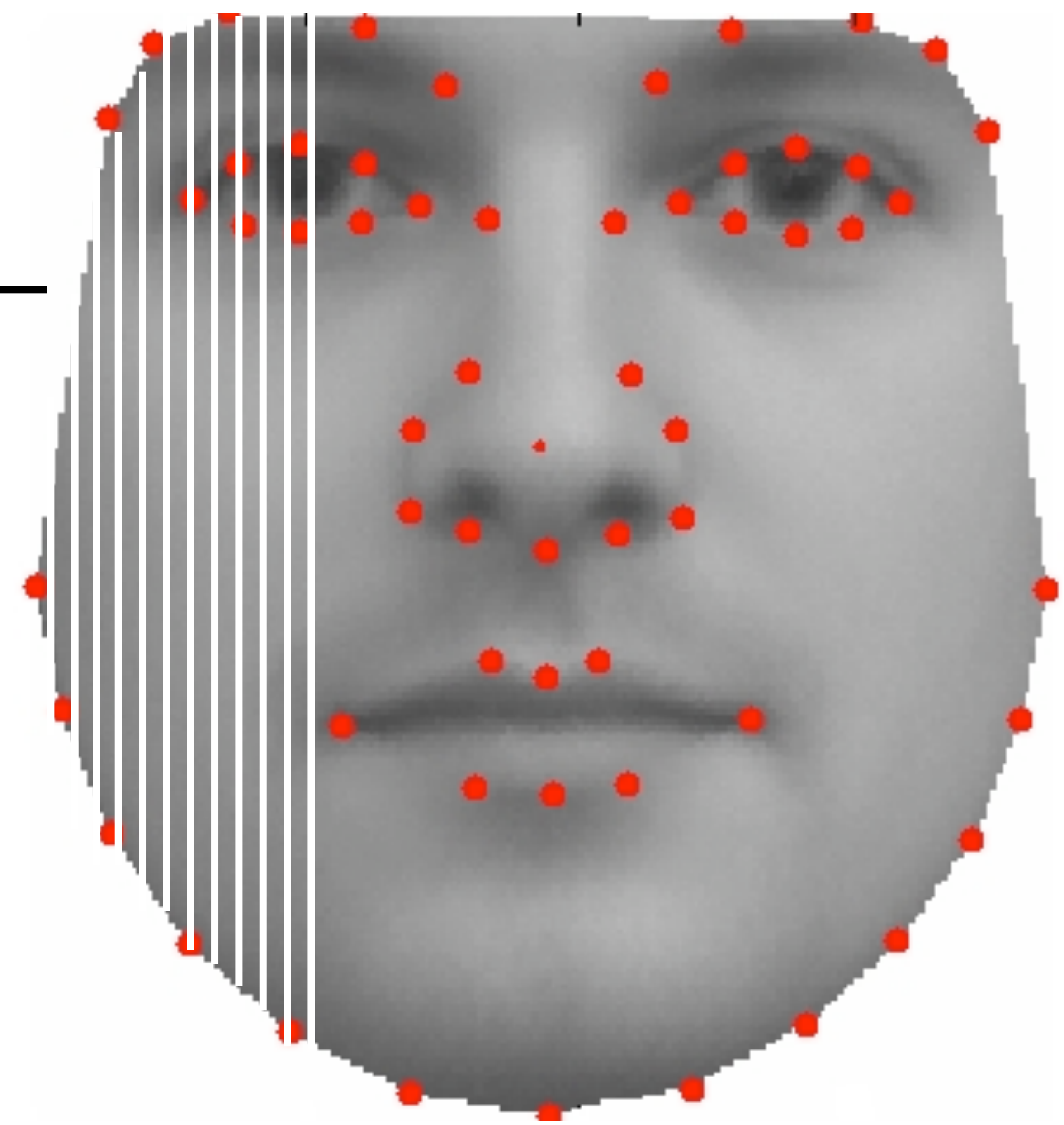
Texture representation

- After the shape normalization a mean texture can be calculated
- Analogous to the shape vectors the gray values of the mapped texture is read out columnwise to form texture vectors



Texture representation

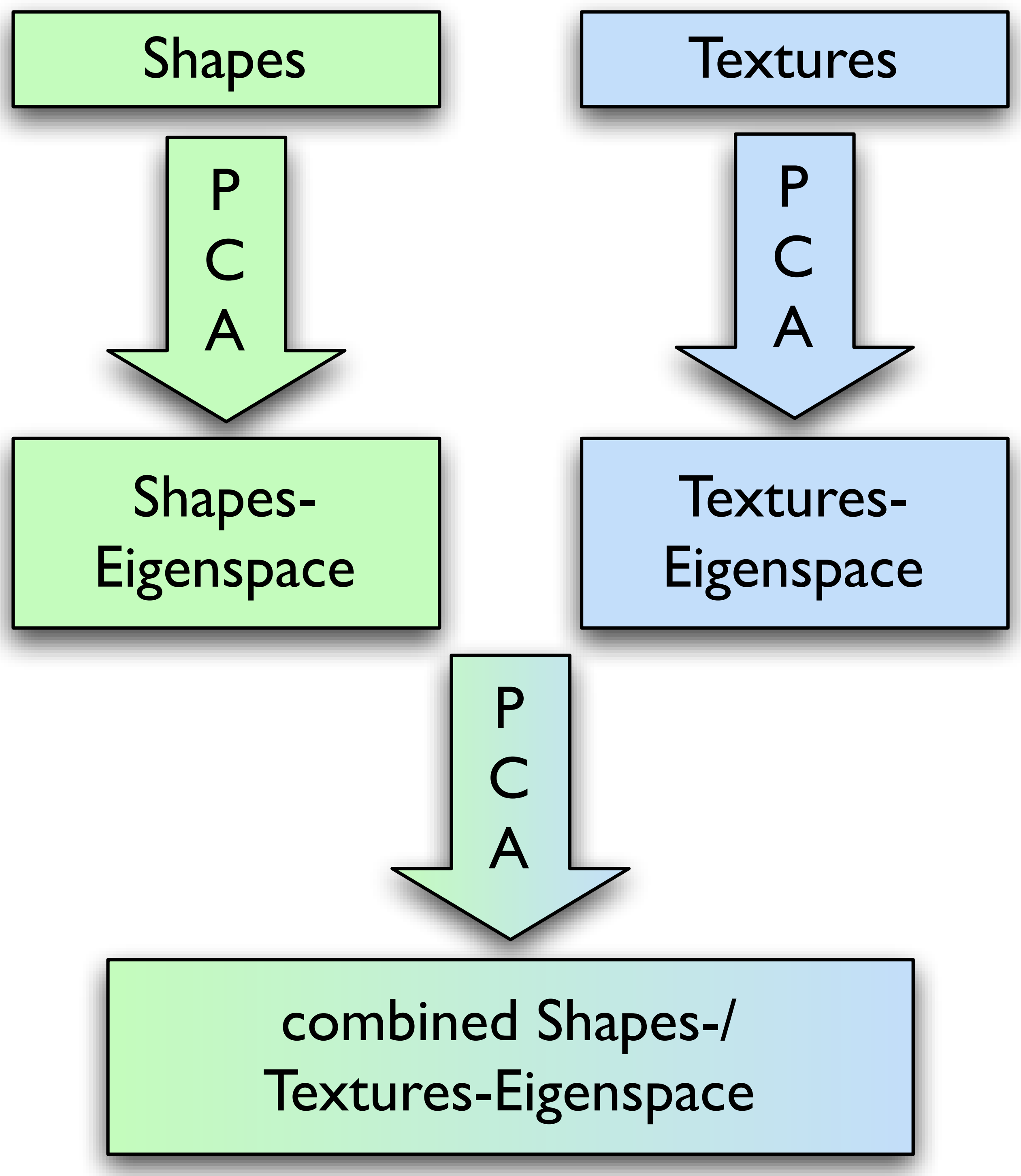
Texture vector: $\mathbf{g}_i$ ⟵

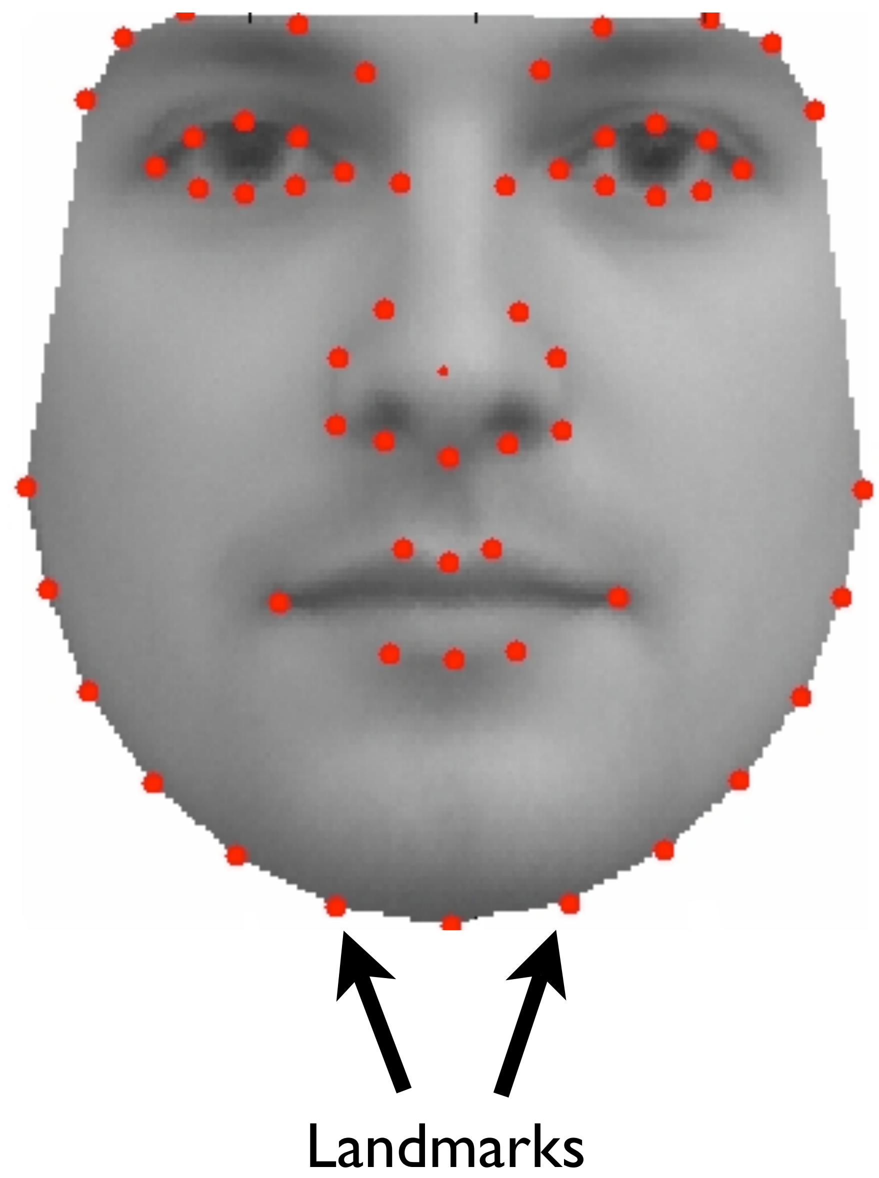


Set of texture vectors representing the textures in the training images:

$$\{\mathbf{g}_1, \dots, \mathbf{g}_n\}$$

Combined model





Combined models

- Shape and texture variation are represented by a single model
- It exploits correlations between texture and shape variation
- Provides a compact representation of the variation in the trianing set

Model search?

- AAM is a **generative model** i.e. it is able to generate instances of the object class in the trianing set.
- But, how to use this combined model to
  - search for landmarks in images?
  - Match the model texture onto images?

Training necessary

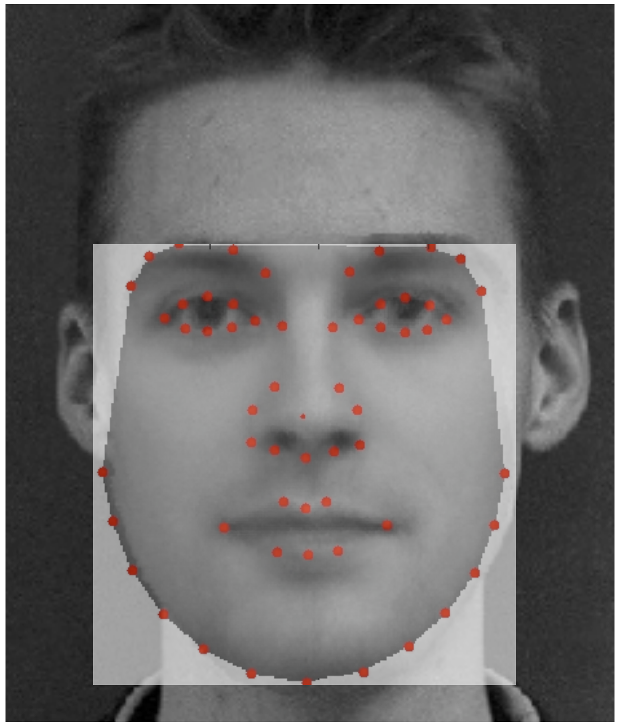
- For AAMs first a training is necessary to enable a fast search
- Relations between a mismatch of the model, and the difference between generated image and observed images are learned
- They can be used to update the model parameters during search.

AAM training

- Capture the relation between model mismatch and model parameter displacement by regression:
  - We know the correct parameters
  - We change them by a known difference vector
  - We observe the resulting model match difference

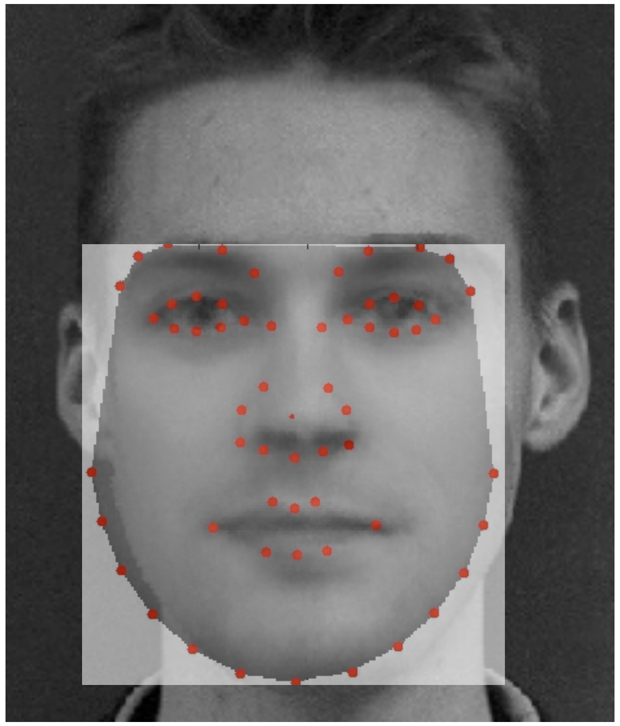
AAM training

- Capture the relation between model mismatch and model parameter displacement by regression:
  - We know the correct parameters
  - We change them by a known difference vector
  - We observe the resulting model match difference



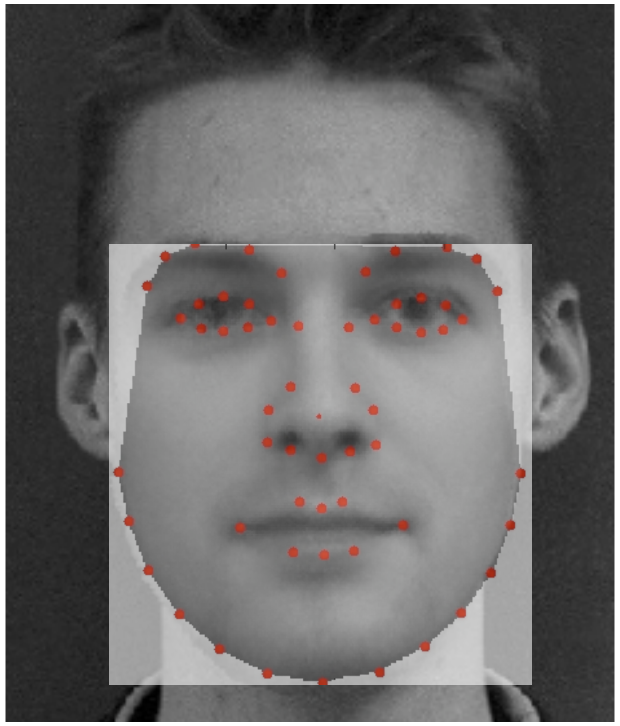
AAM training

- Capture the relation between model mismatch and model parameter displacement by regression:
  - We know the correct parameters
  - We change them by a known difference vector
  - We observe the resulting model match difference



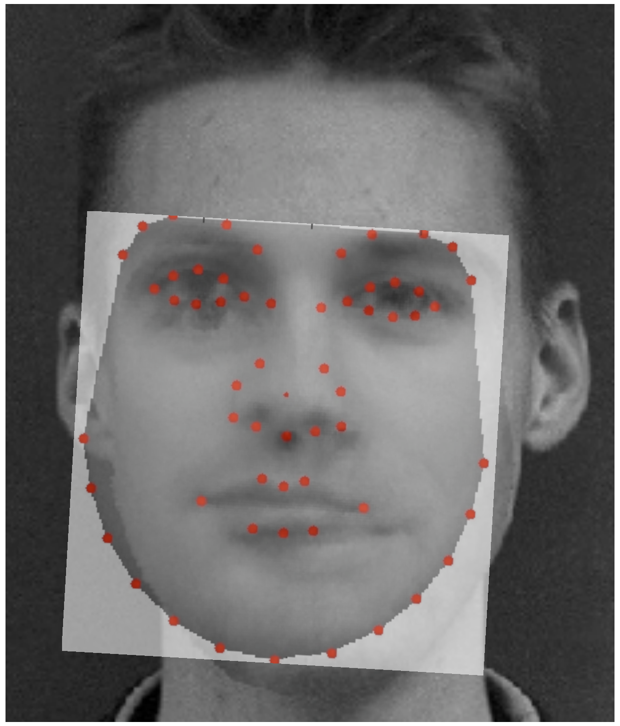
AAM training

- Capture the relation between model mismatch and model parameter displacement by regression:
  - We know the correct parameters
  - We change them by a known difference vector
  - We observe the resulting model match difference



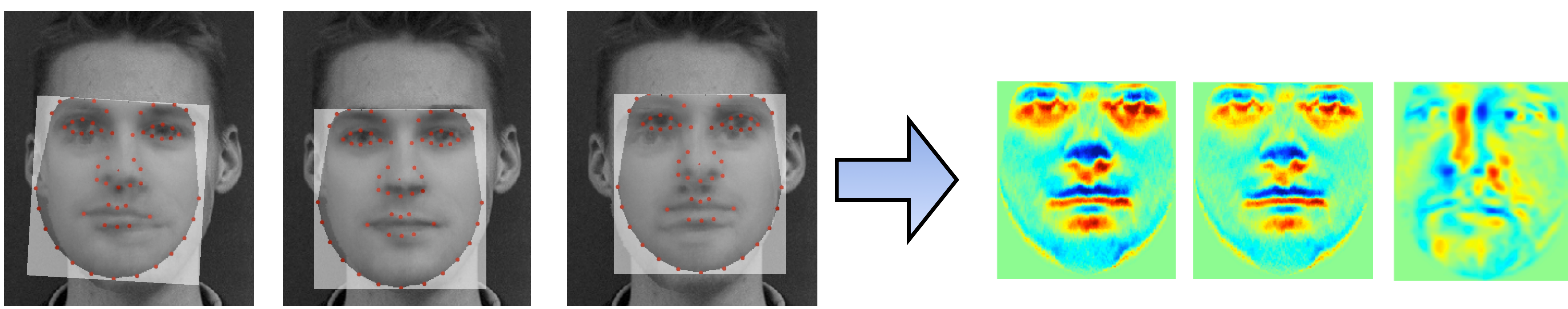
AAM training

- Capture the relation between model mismatch and model parameter displacement by regression:
  - We know the correct parameters
  - We change them by a known difference vector
  - We observe the resulting model match difference



Training how to fit the model

- The relation between residuals and parameter vectors can be learned by either
  - Linear regression

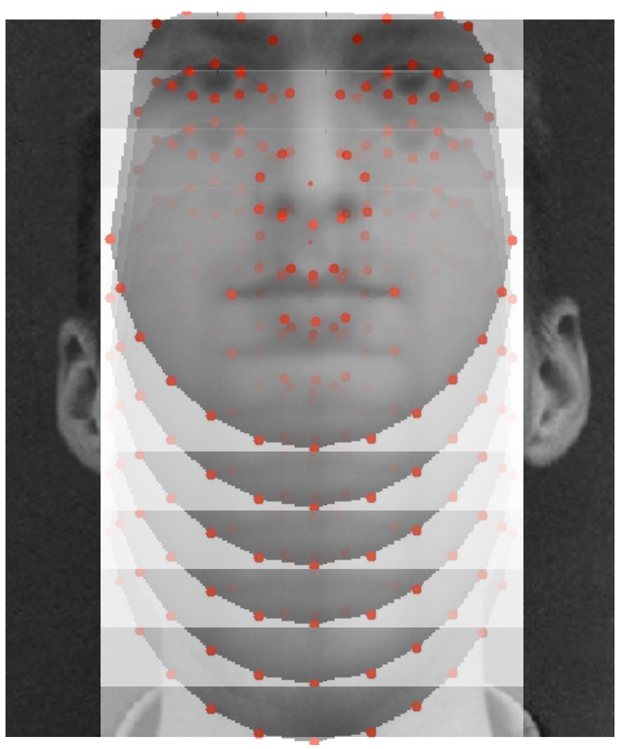


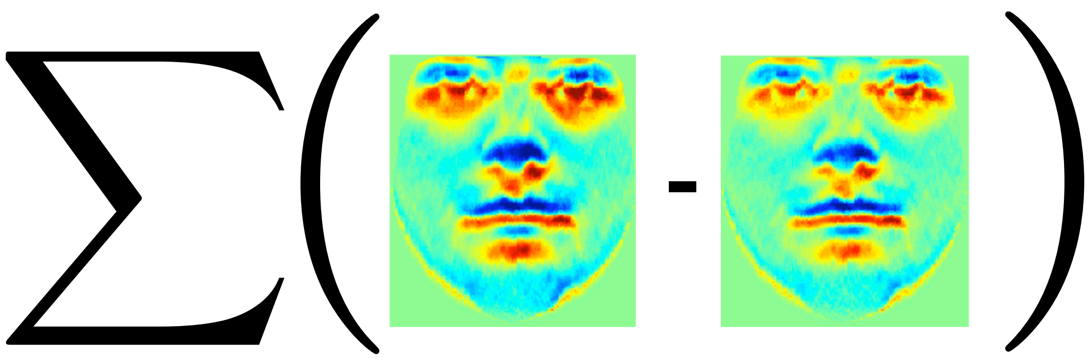
Displaced model

Residual error

Numerical differentiation

- For each mode:
  - Vary the parameter within a certain range
  - And perform numerical differentiation with regard to the parameter

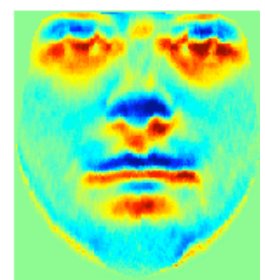
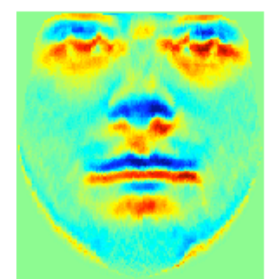




Regression matrix

How to use during search?

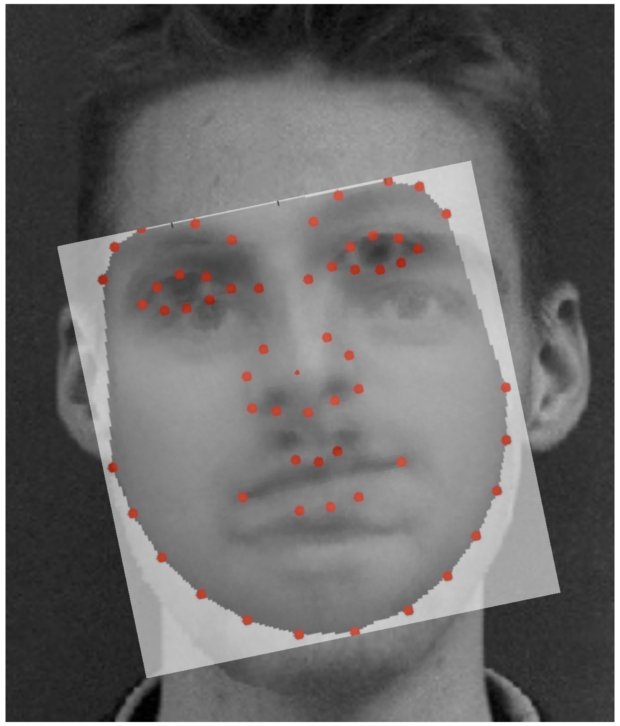
- For each mode we can calculate the correction of the paremeter by simply projecting the current difference image onto the one in the regression matrix

$\delta p_i =$ ( , )

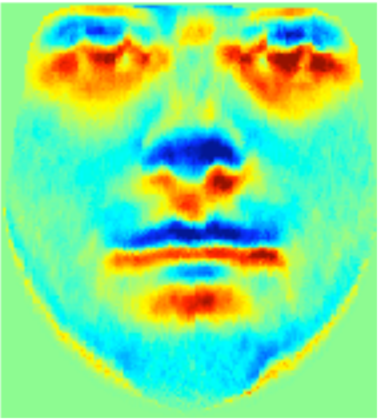
Current observation during search

Taken from one line in the regression matrix

AAM search



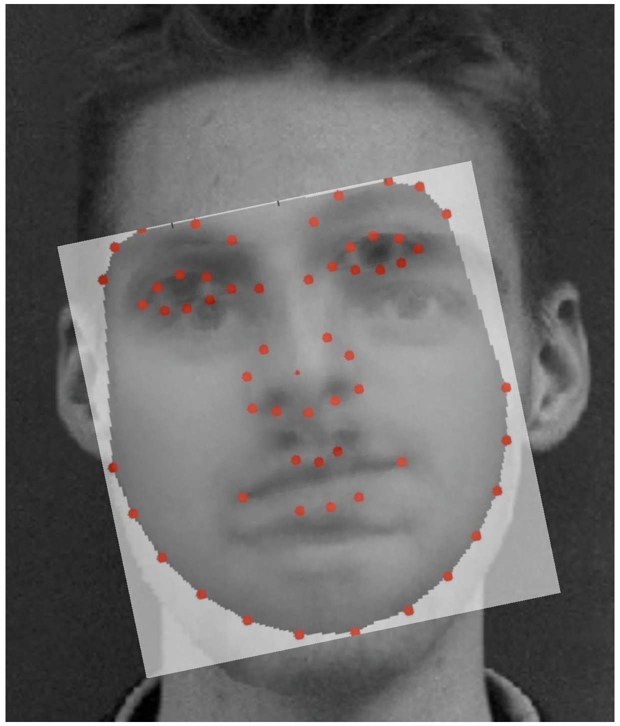
Initialization



Difference image

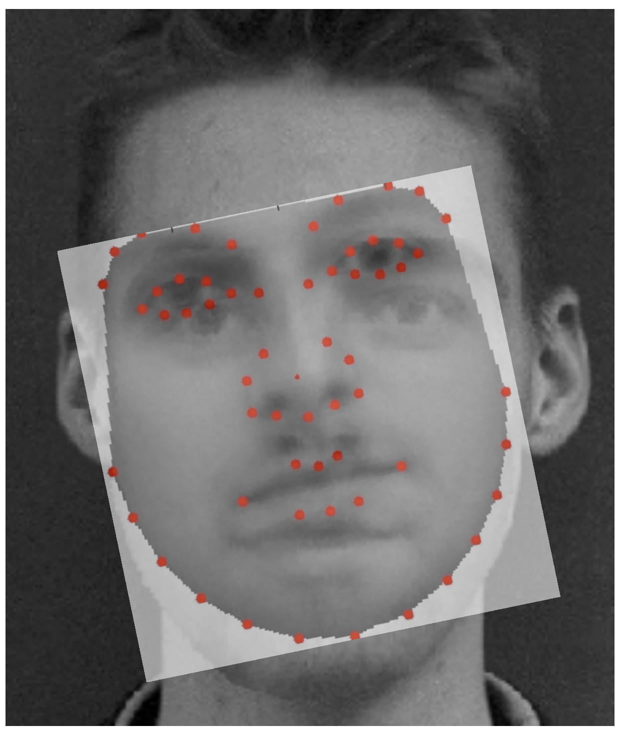
$$\delta \mathbf{p}$$

Parameter update

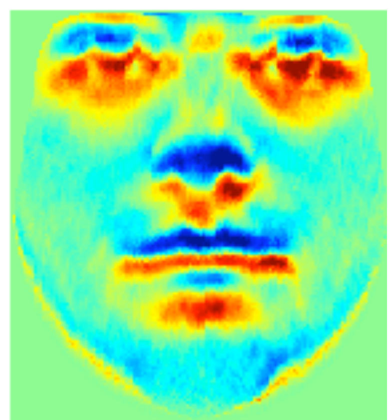


Model update

AAM search



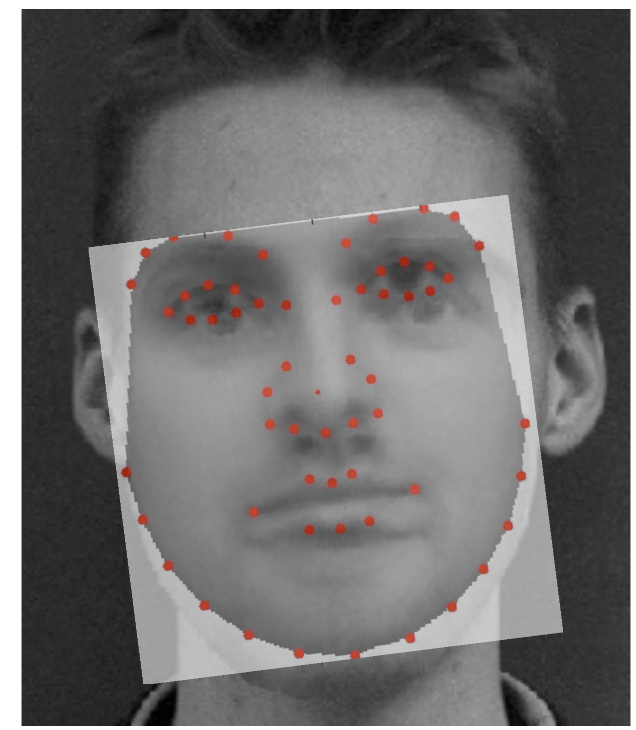
Initialization



Difference image

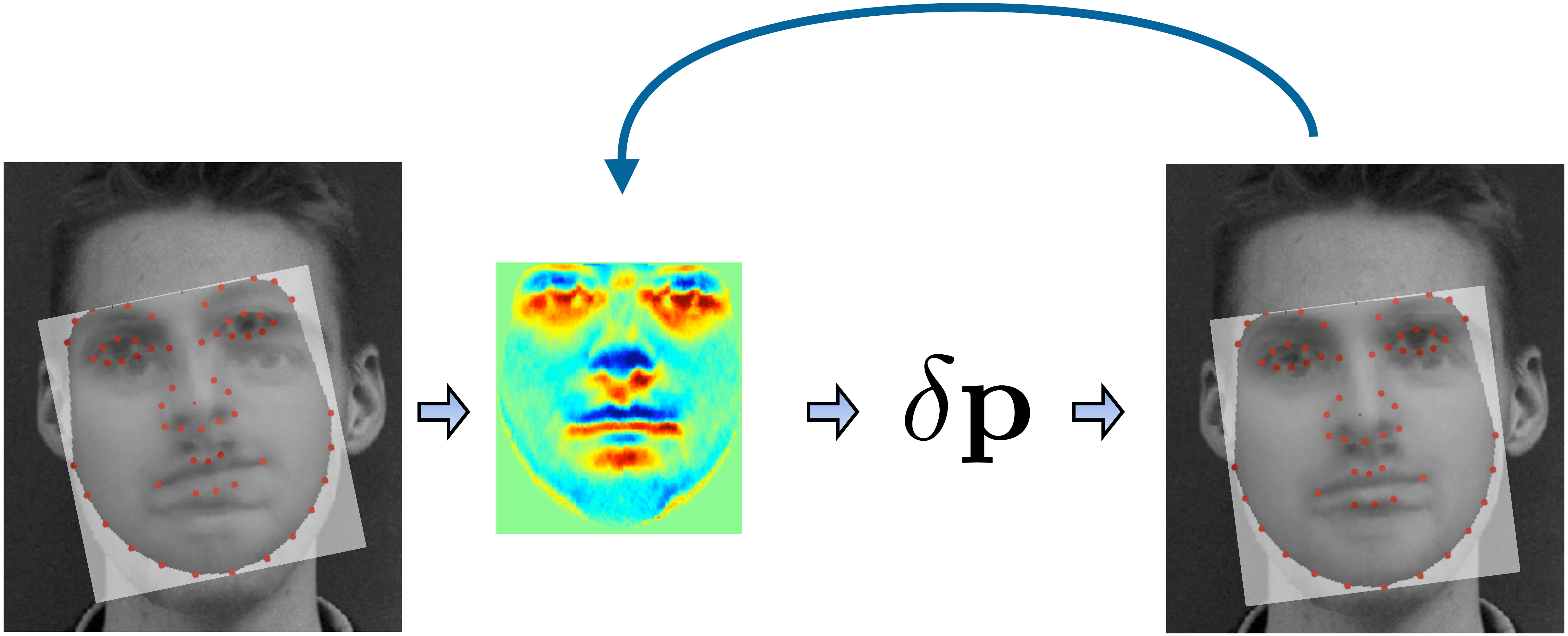
$\delta\mathbf{p}$

Parameter update



Model update

AAM search



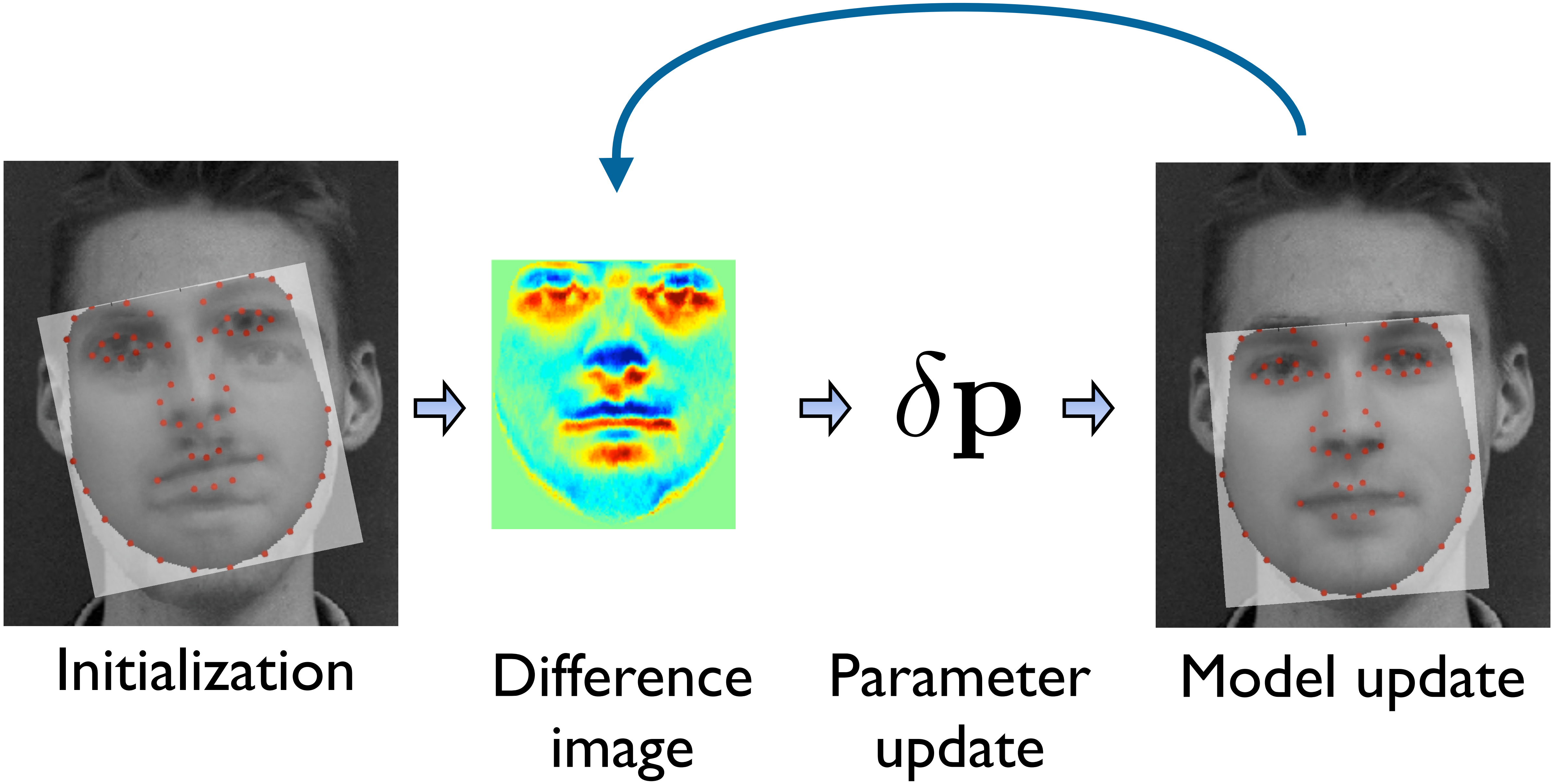
Initialization

Difference image

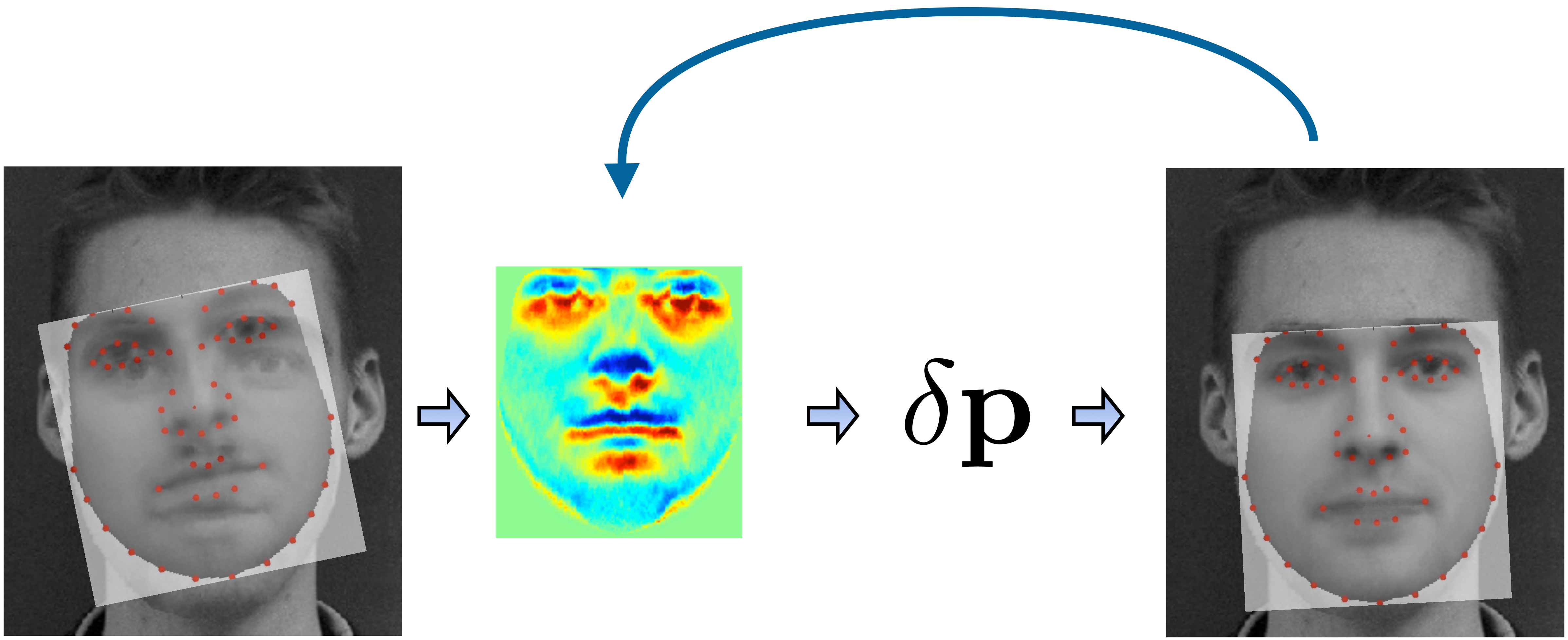
Parameter update

Model update

AAM search


Initialization
Difference image
$\delta \mathbf{p}$
Parameter update
Model update

AAM search



Initialization

Difference image

Parameter update

Model update

AAM Search

Wrap up

- AAMs represent shape and texture variation
- They are generative (i.e., they can generate model instances)
- To be able to perform search with an AAM the relation between parameter displacements and residual difference image has to be learned
- Search is performed by initializing the model and updating the parameters according to training until the search converges

Extensions to more dimensions

- The concept of AAMs can be extended to more dimensions.
- 3D landmarks
- Instead of a triangulation and a patch based texture representation ...
- ... the volume enclosed by the landmarks can be warped to the mean shape
- It is represented analogously to the 2D case by vectorization

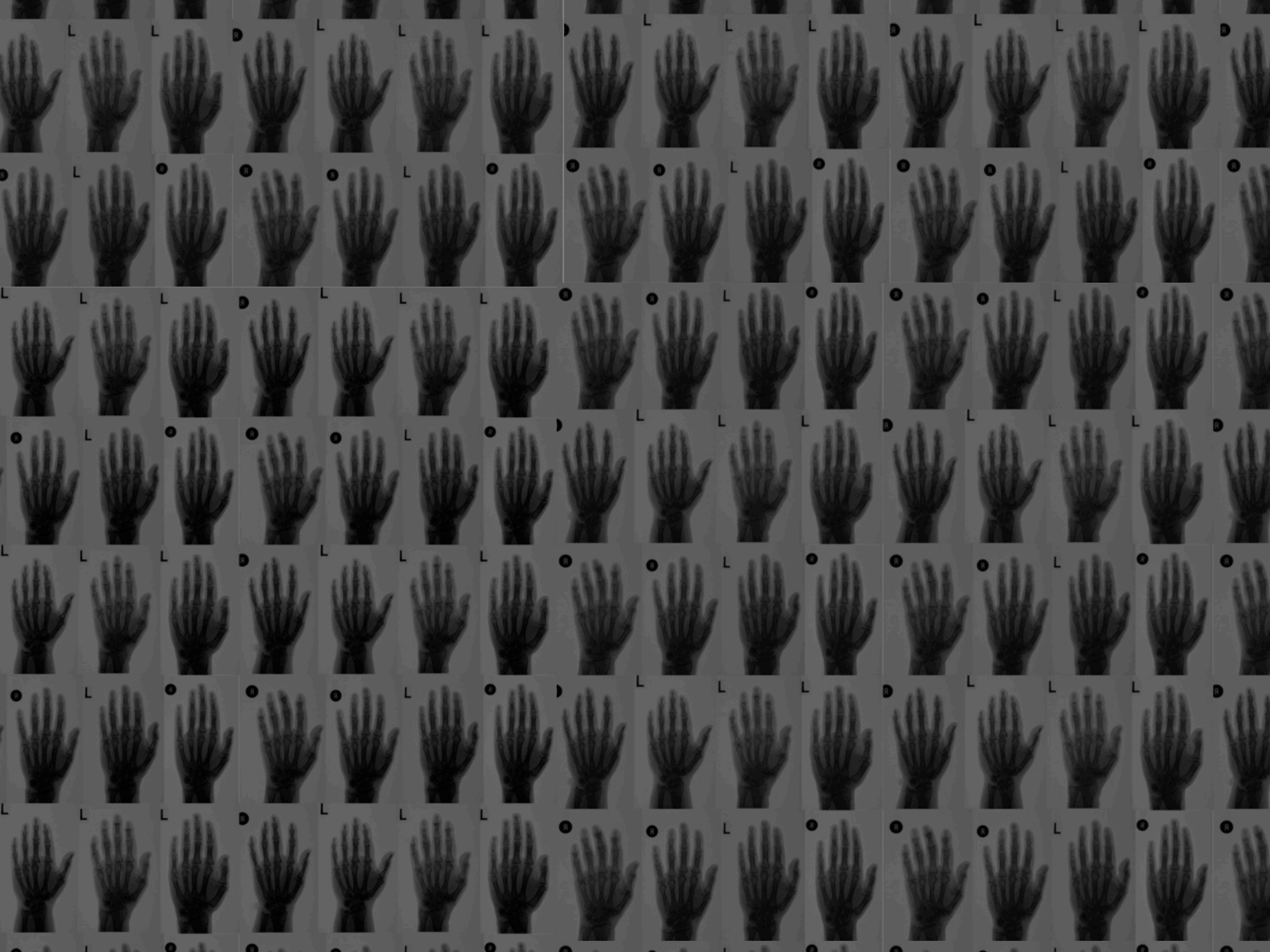
Pitfalls ...

- 3D or 4D models suffer from the very high dimensionality of the examples
- Compared to that the number of training examples is very low

- Alternatives?
  - **Use active shape models** in 3D data
  - **Use only selected texture parts** close to the landmarks

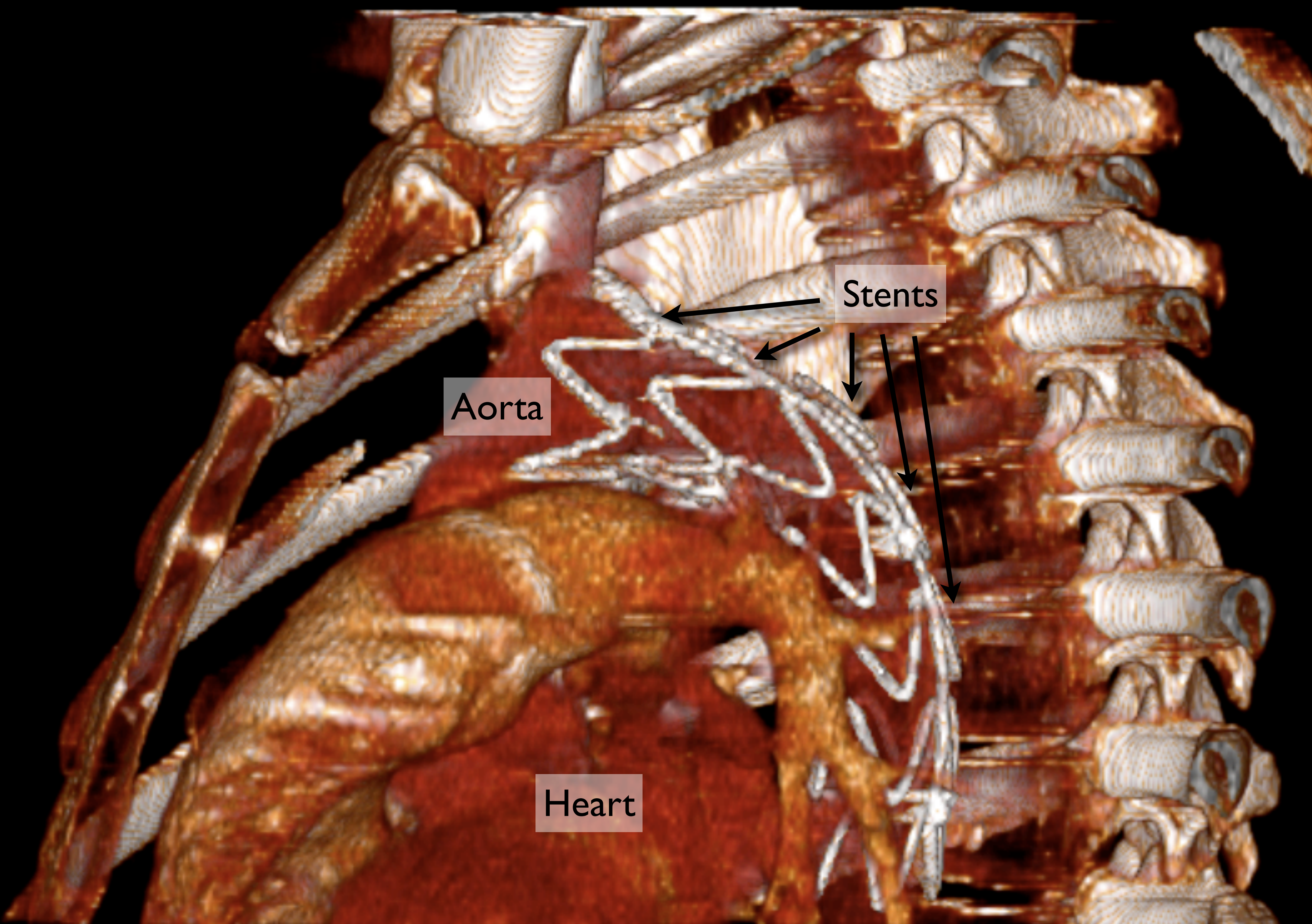
Summary

- AAMs capture
  - **Shape** variation
  - **Texture** variation of the texture enclosed by the landmarks
- AAMs are **generative models** i.e. the can generate instances of the learned object class
- AAMs can be used to **search for the objects** in images
- AAM search minimizes the residual error between generated model instance and observation to fit the model to an image

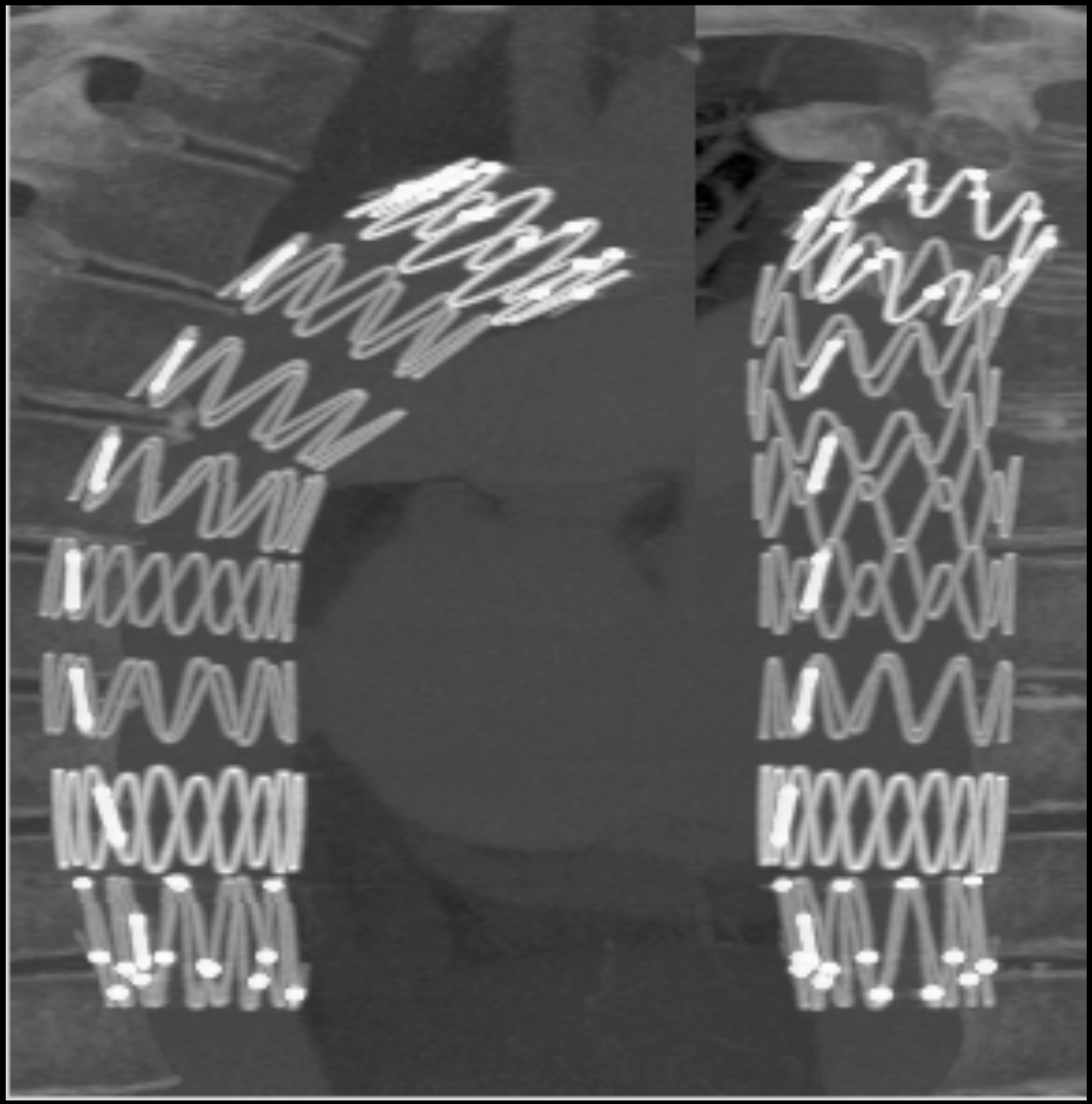


1. Learning Models

Stent Grafts

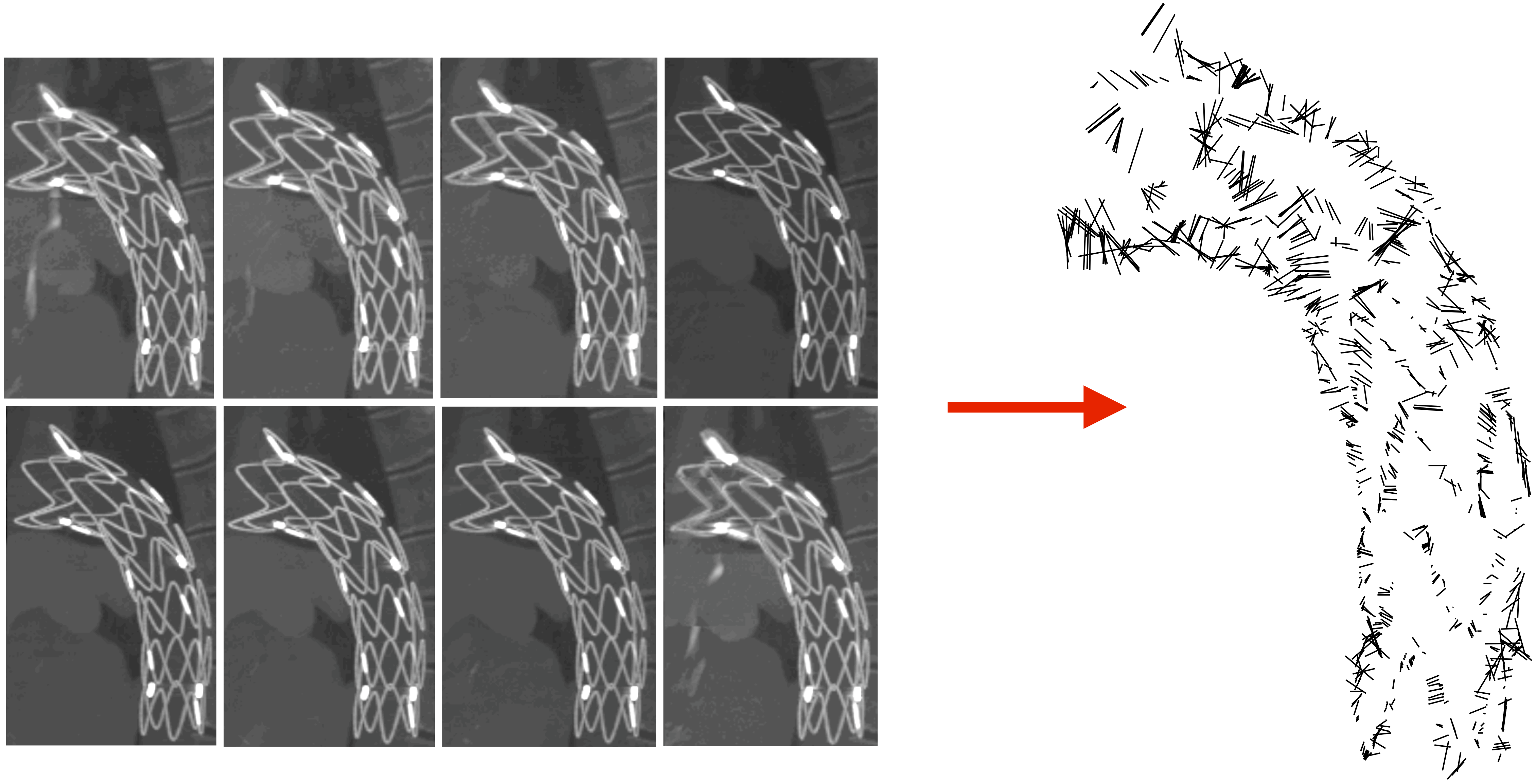


Gated CT sequences



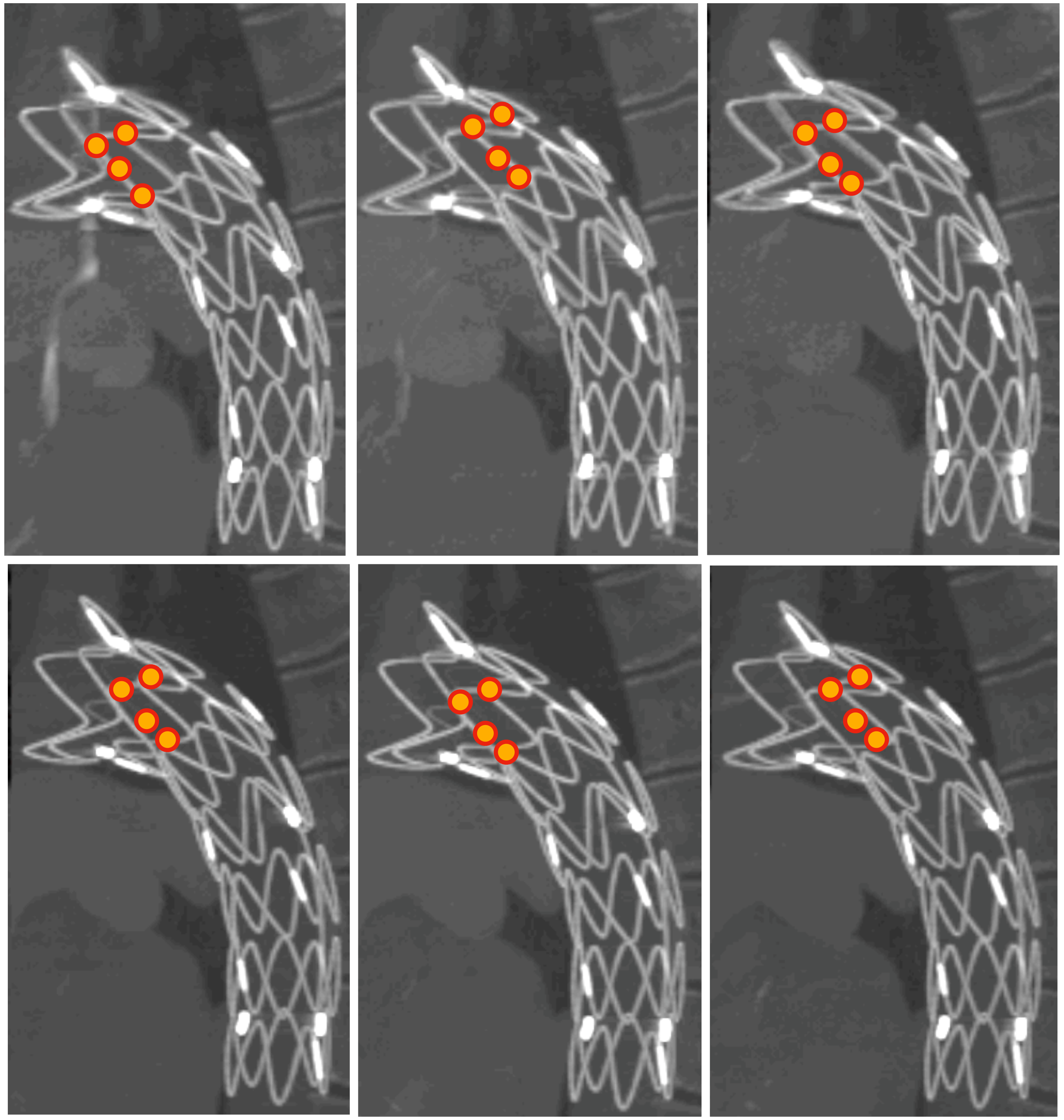
Georg Langs

Learning a model



The manual way ...

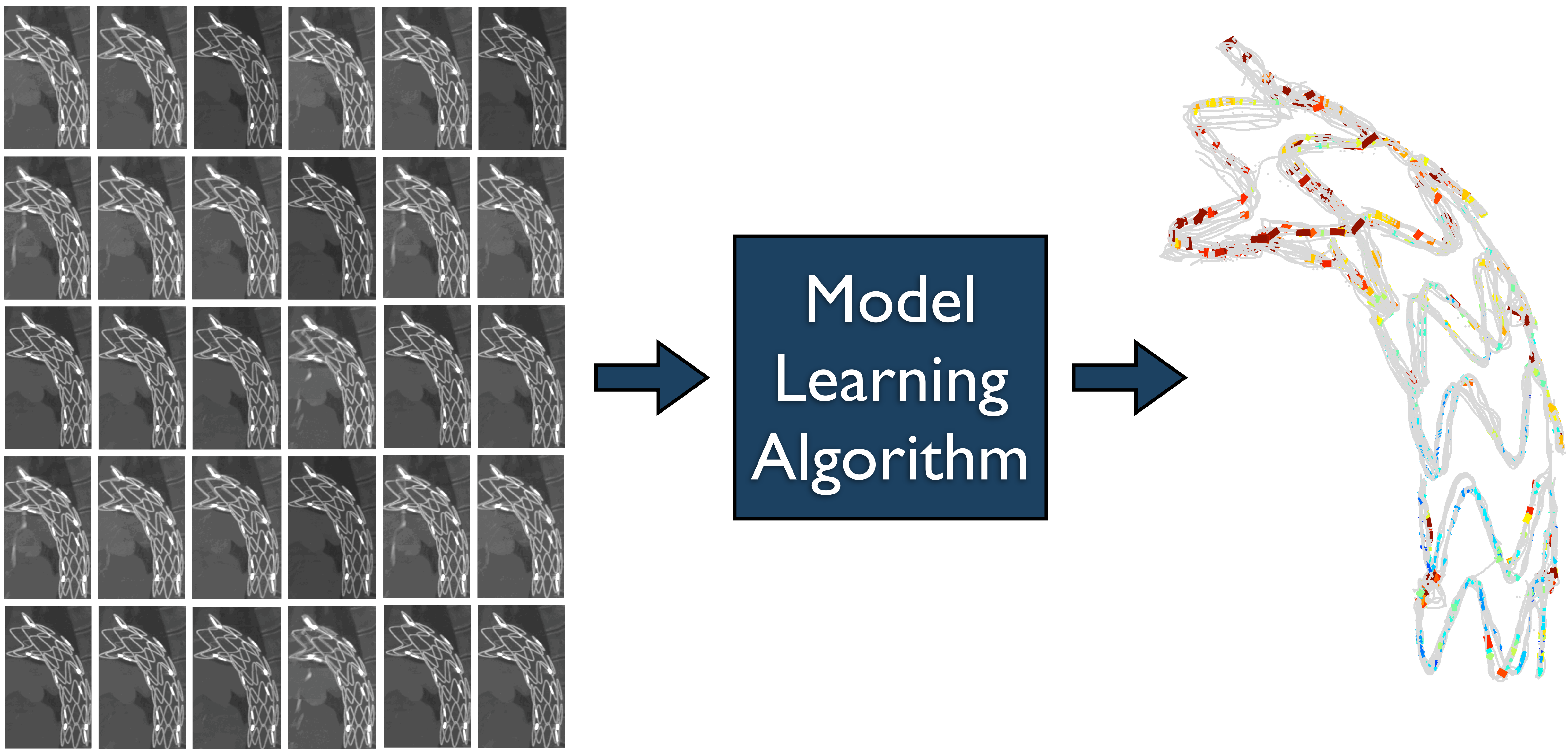
- Find an expert, who has time
- Let the expert manually annotated lots of examples



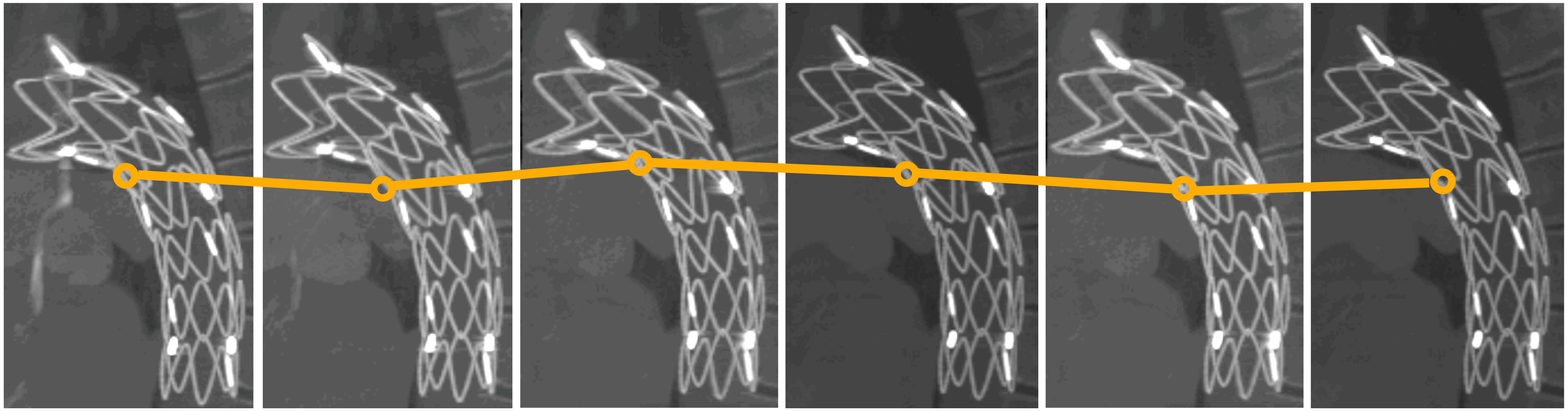
Learning and Analysis

- It is no longer feasible to perform supervised learning and then apply the algorithm to new data
- **Unsupervised and weakly supervised learning approaches**
- ... One more step: learning becomes a part of the analysis process, as we learn the variability and nature of the data, we acquire knowledge about its structure

What we want

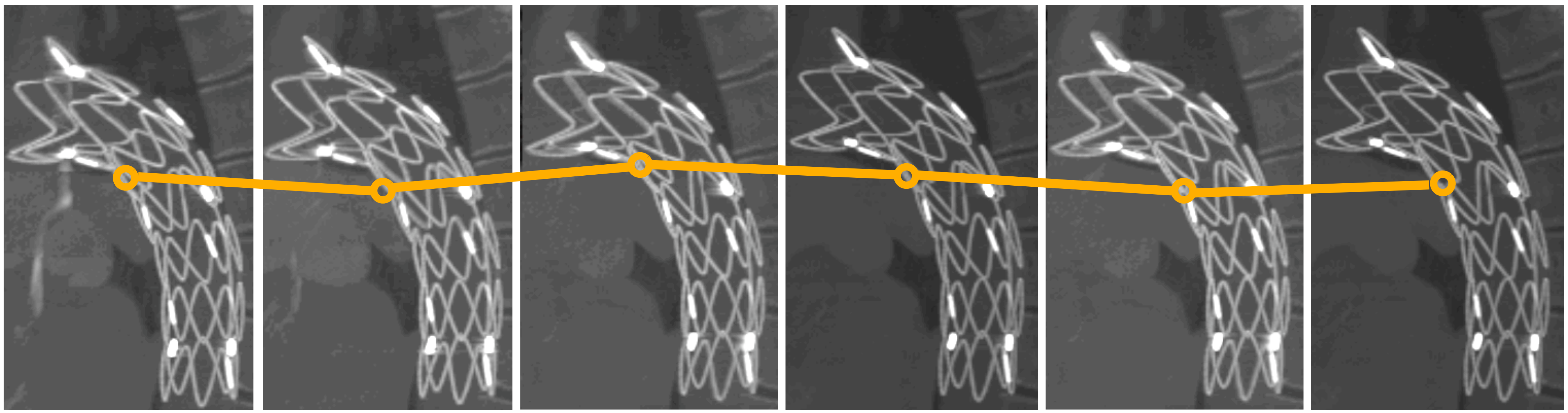


Group-wise registration

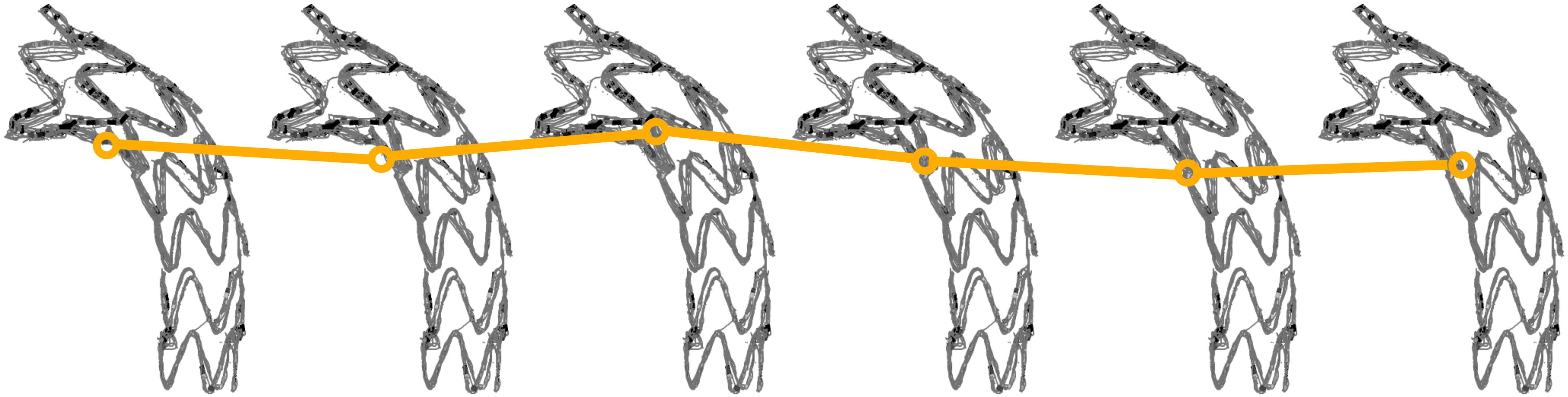


This can be tricky

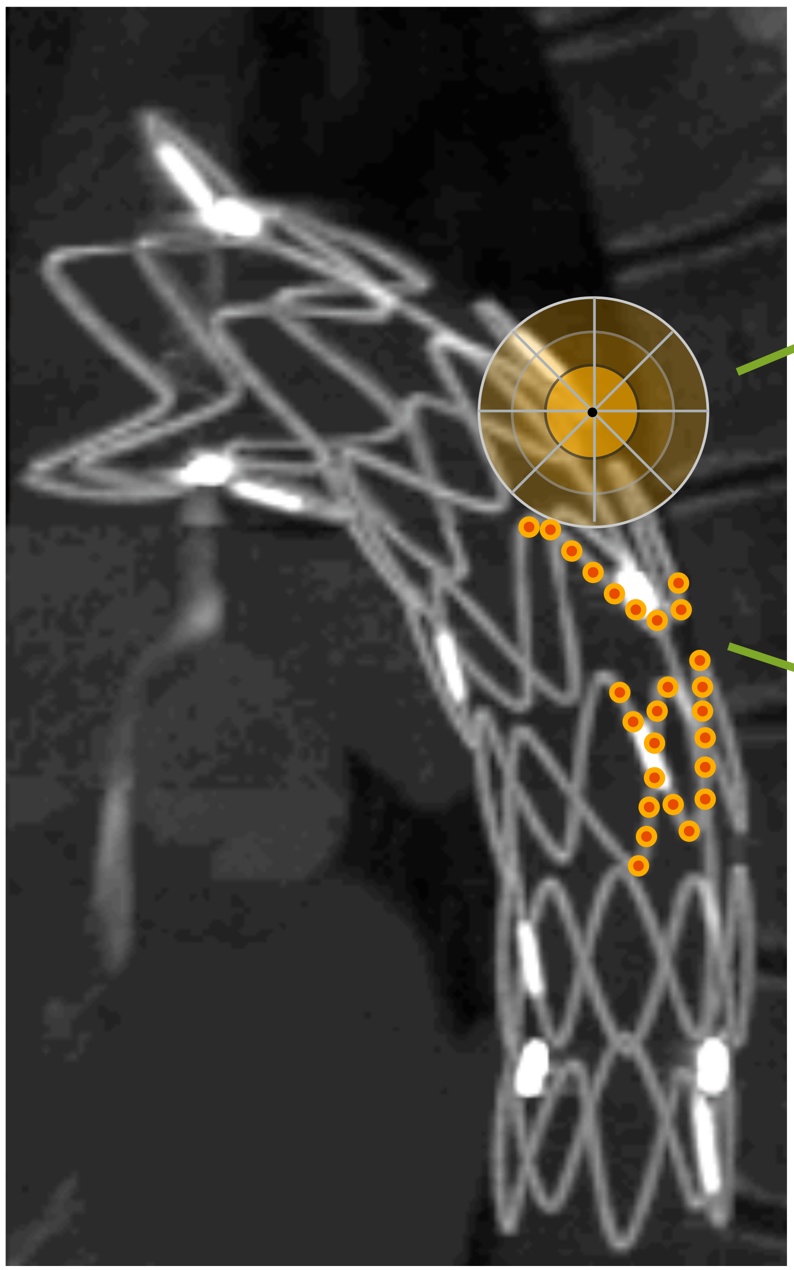
Group-wise registration



Reduce the problem to a set of interest points



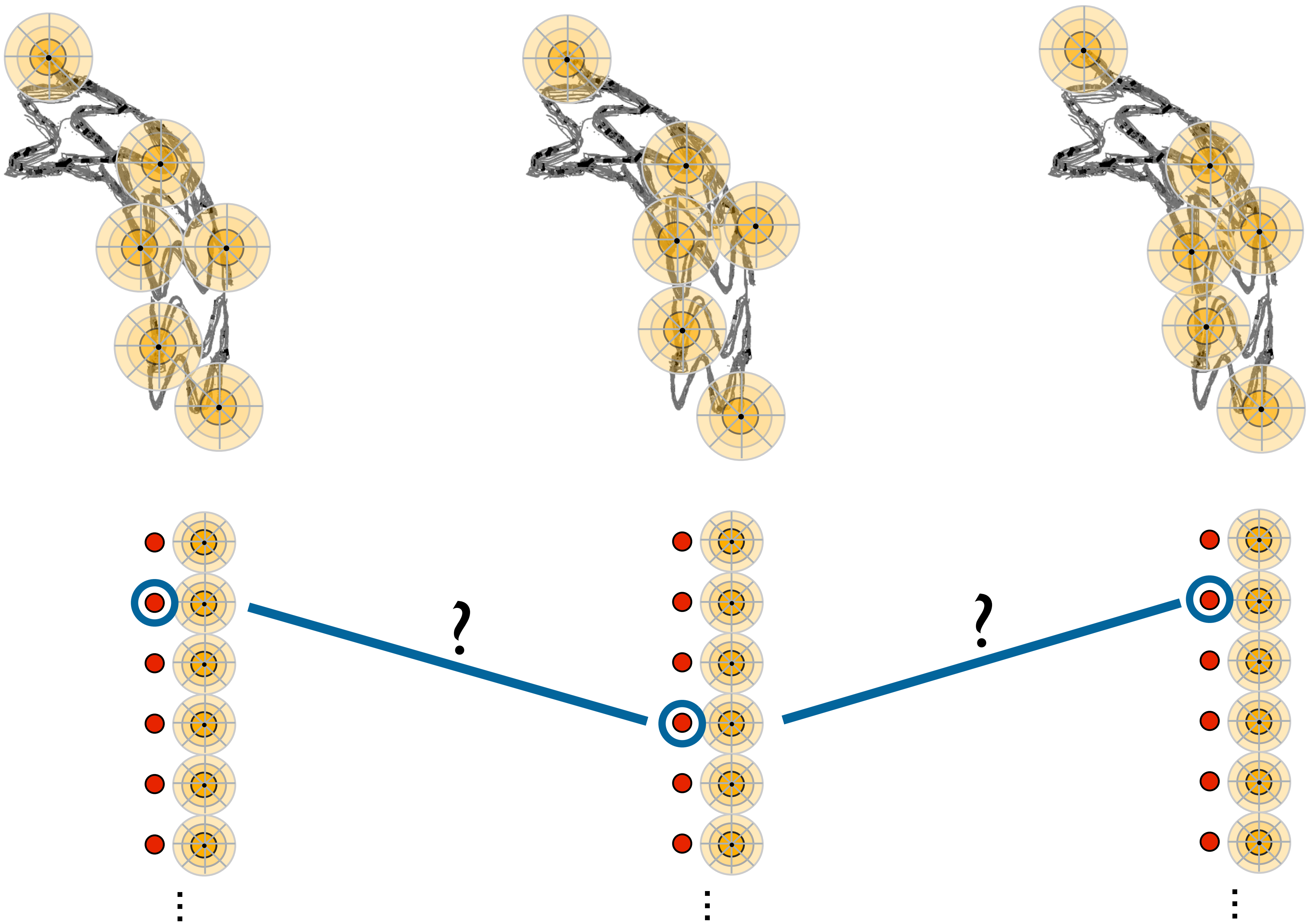
Representing data differently



Local appearance descriptors

A set of interest points

Correspondences on point sets



Minimum Description Length

$$\mathcal{L} = \mathcal{L}(M) + \mathcal{L}(D|M) + \mathcal{R}$$

transmit model

transmit data encoded by model

residual error

DL expresses the compactness of a model given certain training data. Minimize description length to improve generalization behavior

Shape **Local Texture**

$$\mathcal{C} = \mathcal{C}_S + \mathcal{C}_T + a(t)\mathcal{C}_E$$

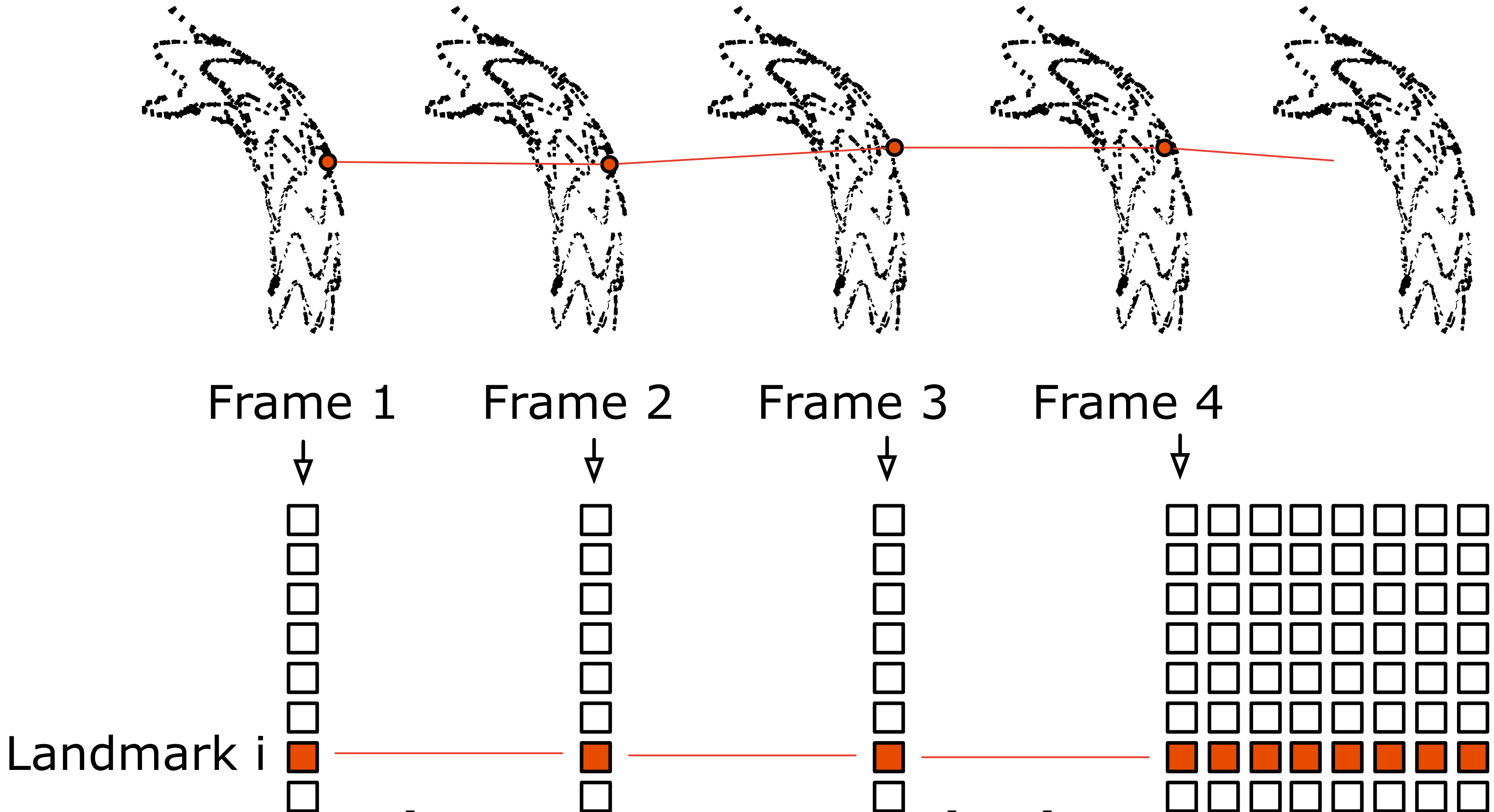
Criterion function: costs for encoding of shapes (C_S), local texture (C_T), and elasticity regularization (C_E)

Evolving shape model of hands

Before optimization

After optimization

Correspondence encoding

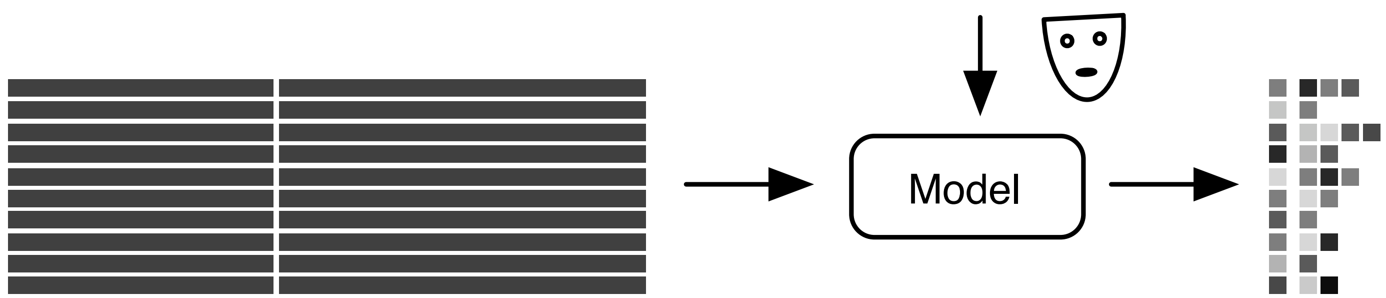


Correspondences are coded as indices into the lists of interest points for each image, resulting in matrix *G*

Optimization

Pick an example

Project single example into the model

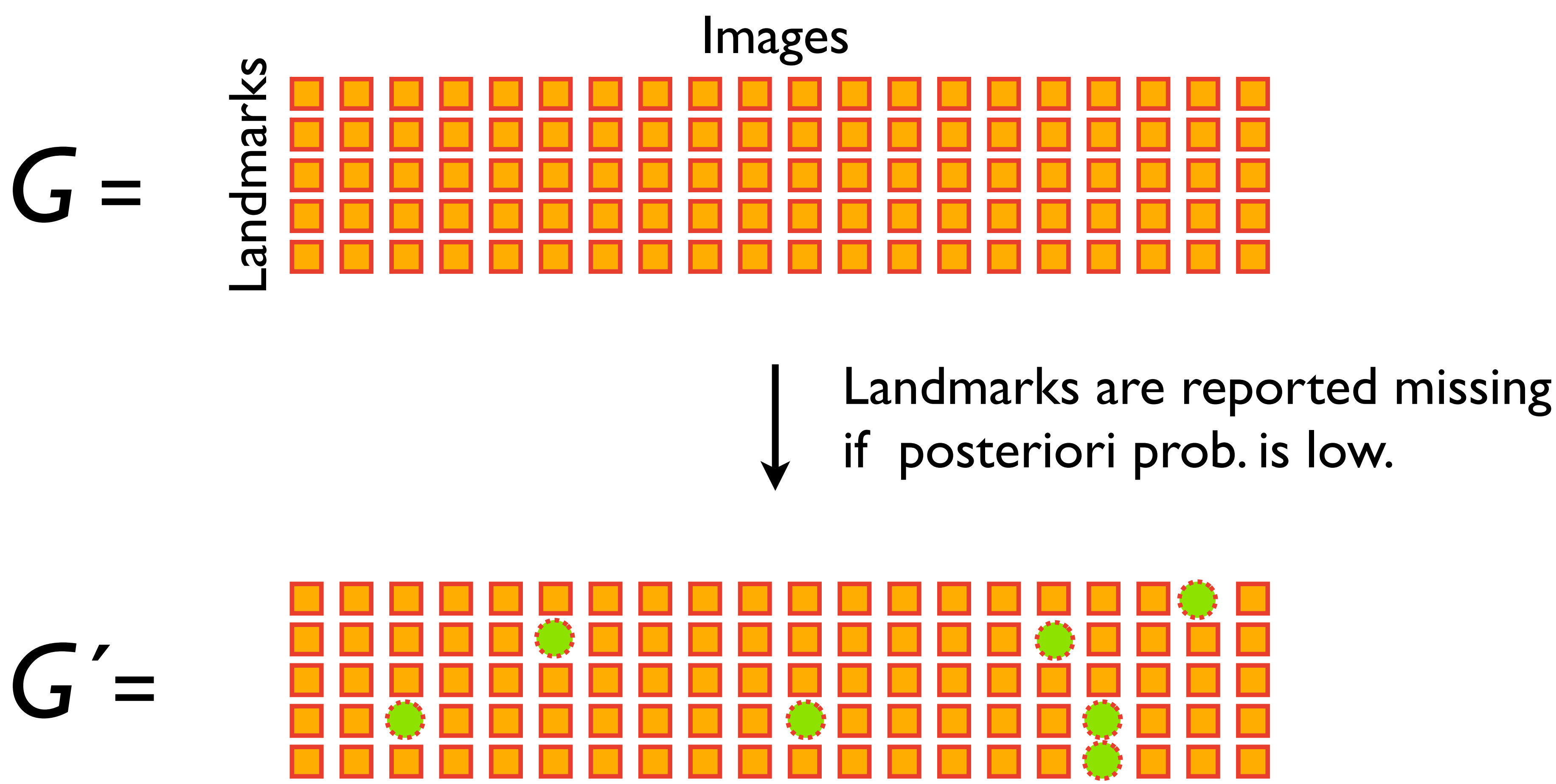


Build model from remaining examples

Search for best points in neighbourhood of projection

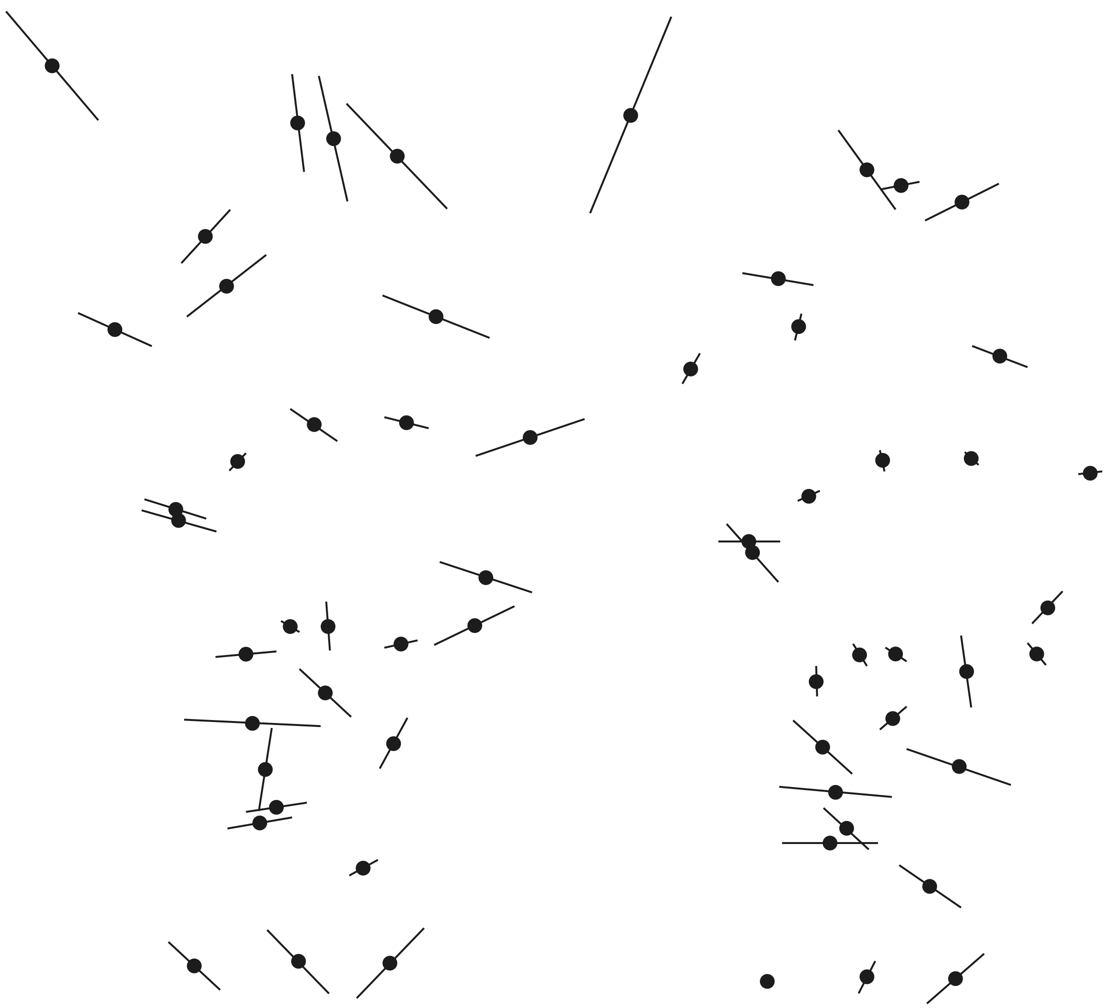
Correspondences are optimized by changing *G*, after convergence additional landmarks are added, by TPS interpolation.

Dealing with incomplete data

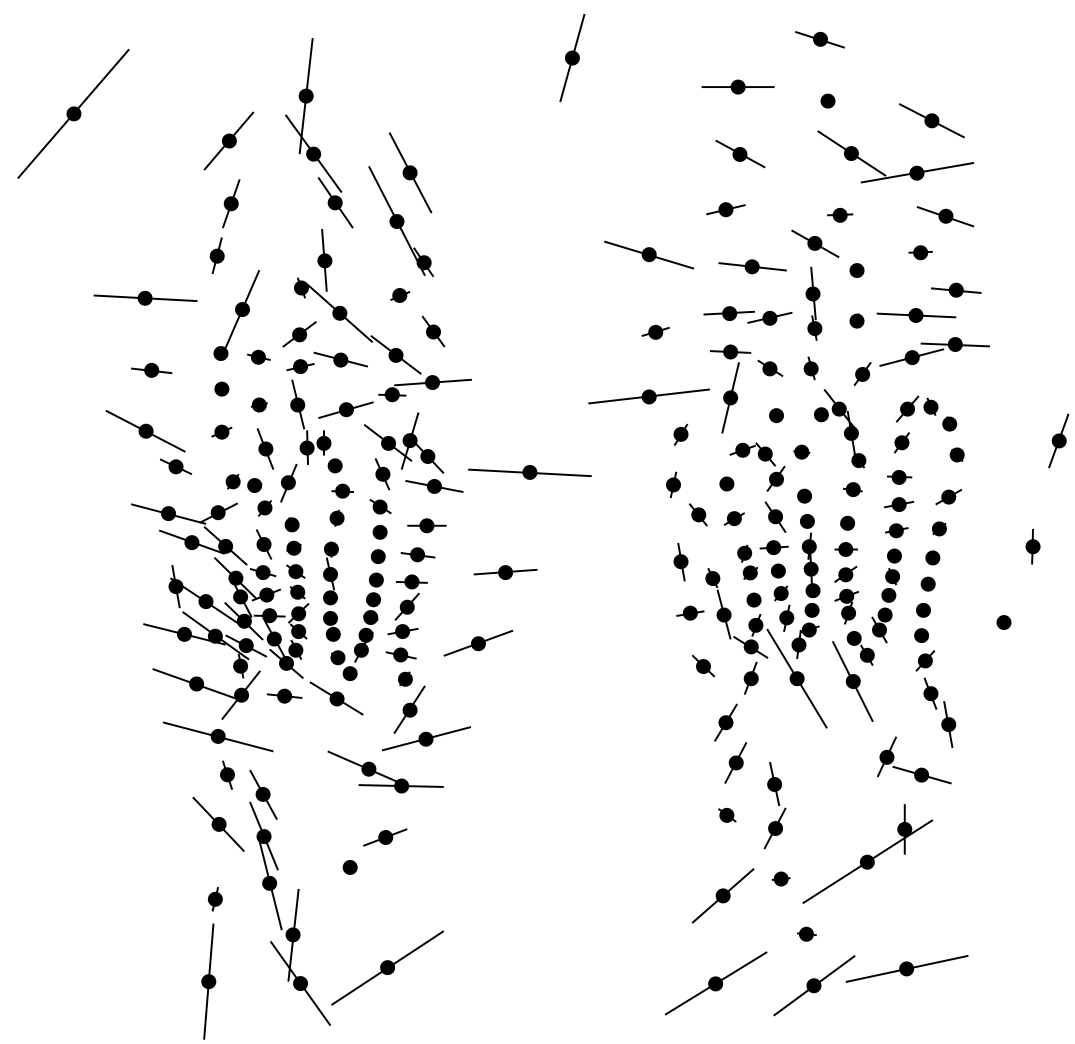


The positions of missing landmarks are imputed by the preceding shape model. (i.e. similar to EM imputation)

Examples



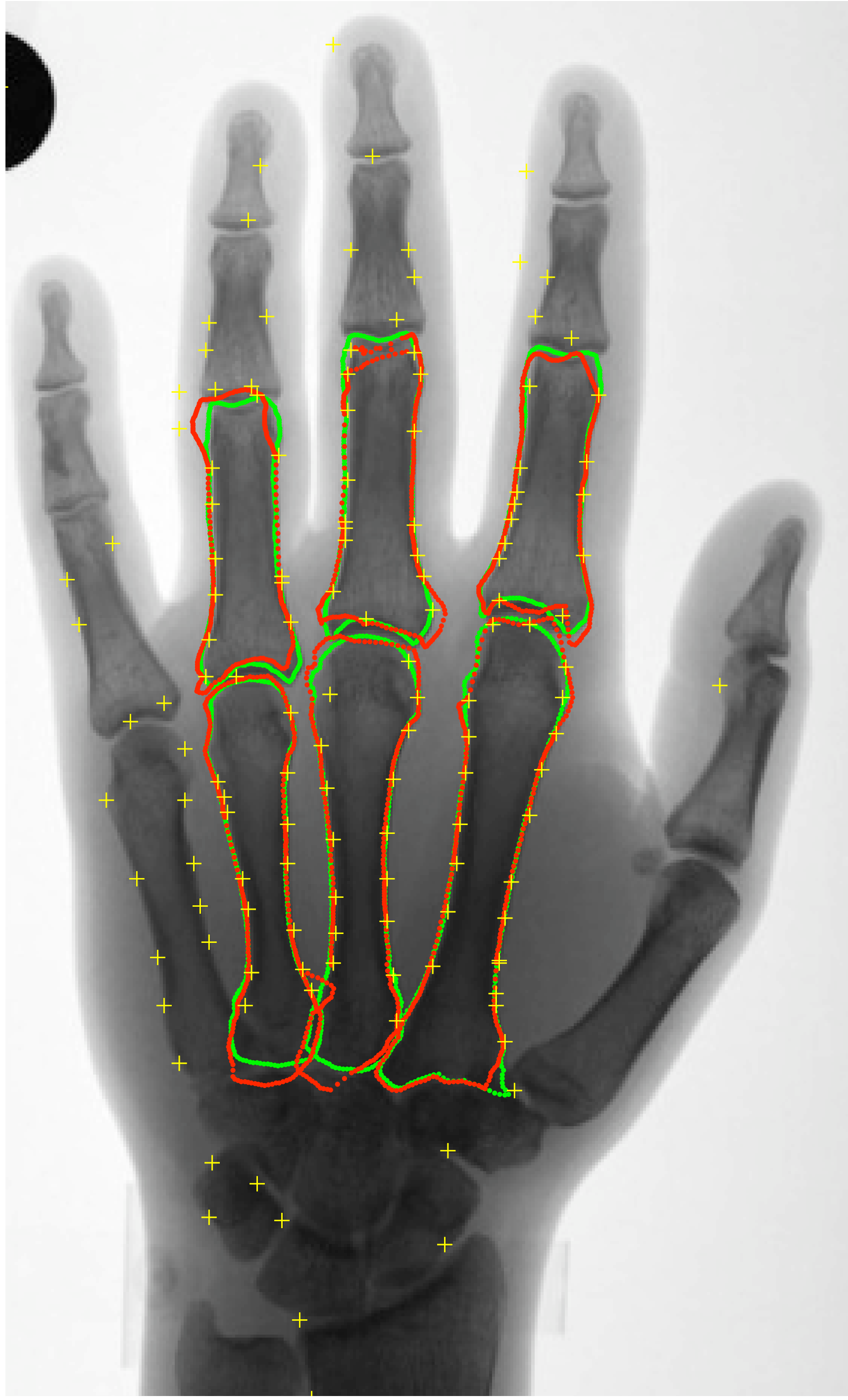
Early model

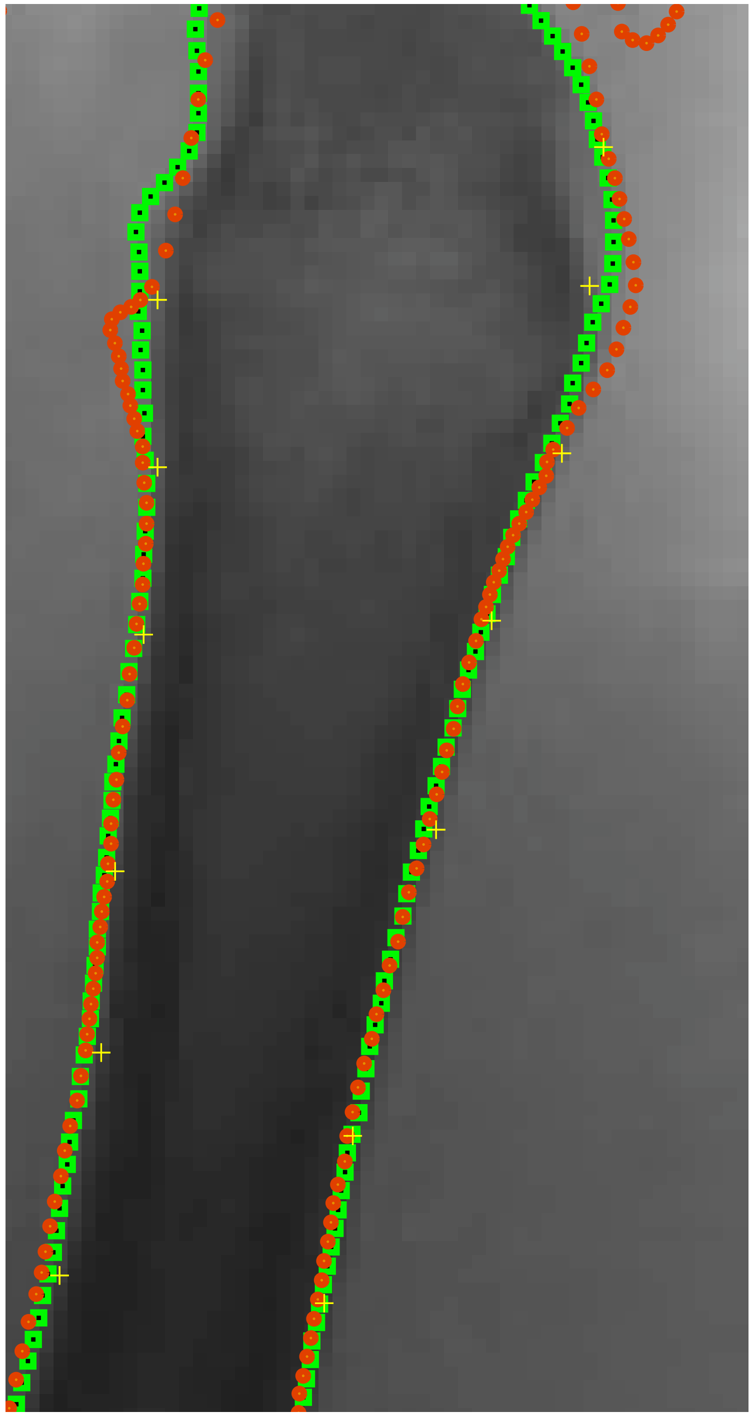


Optimization result

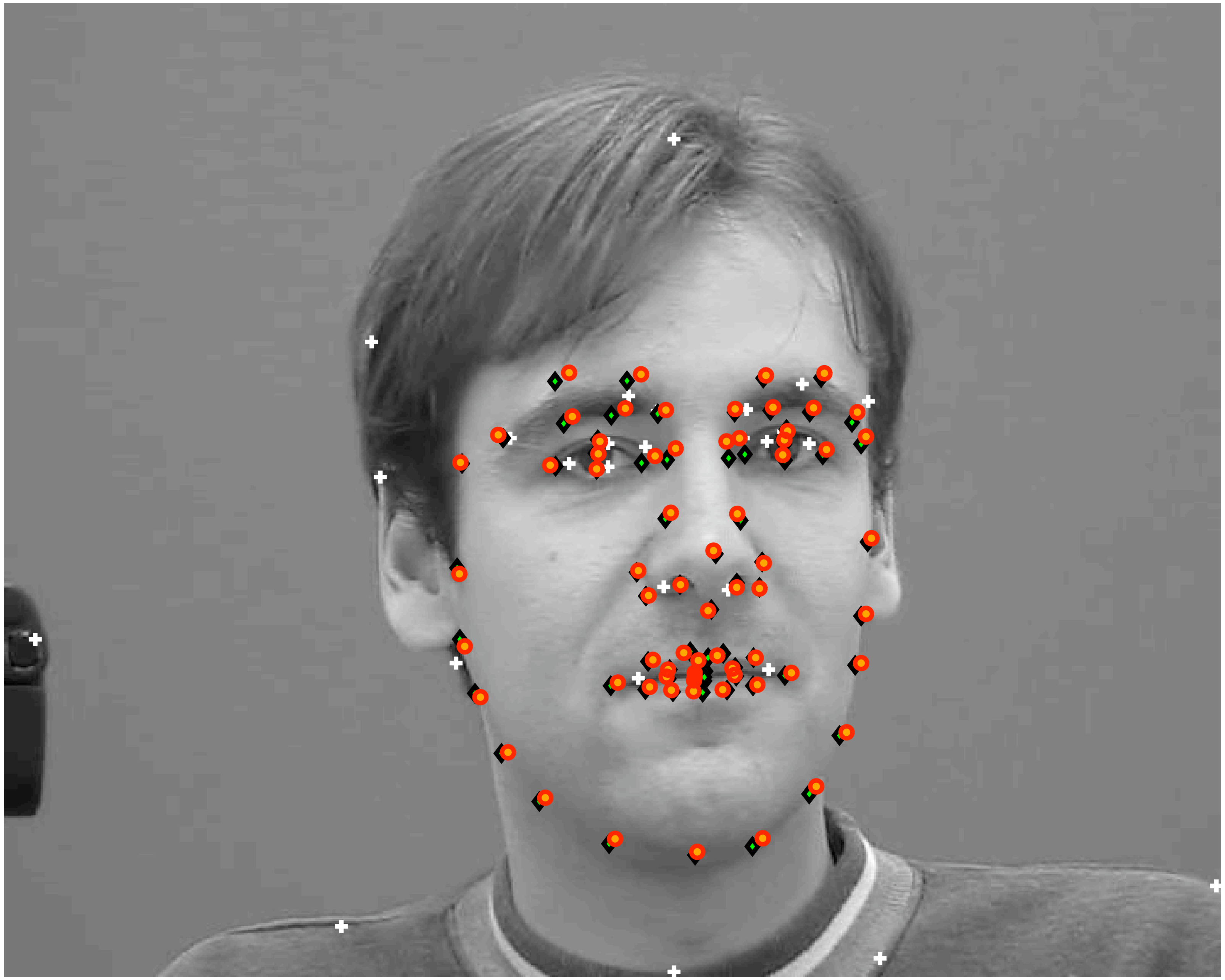
Landmark error to groundtruth: 5.84 px, contour error: 2.27 px

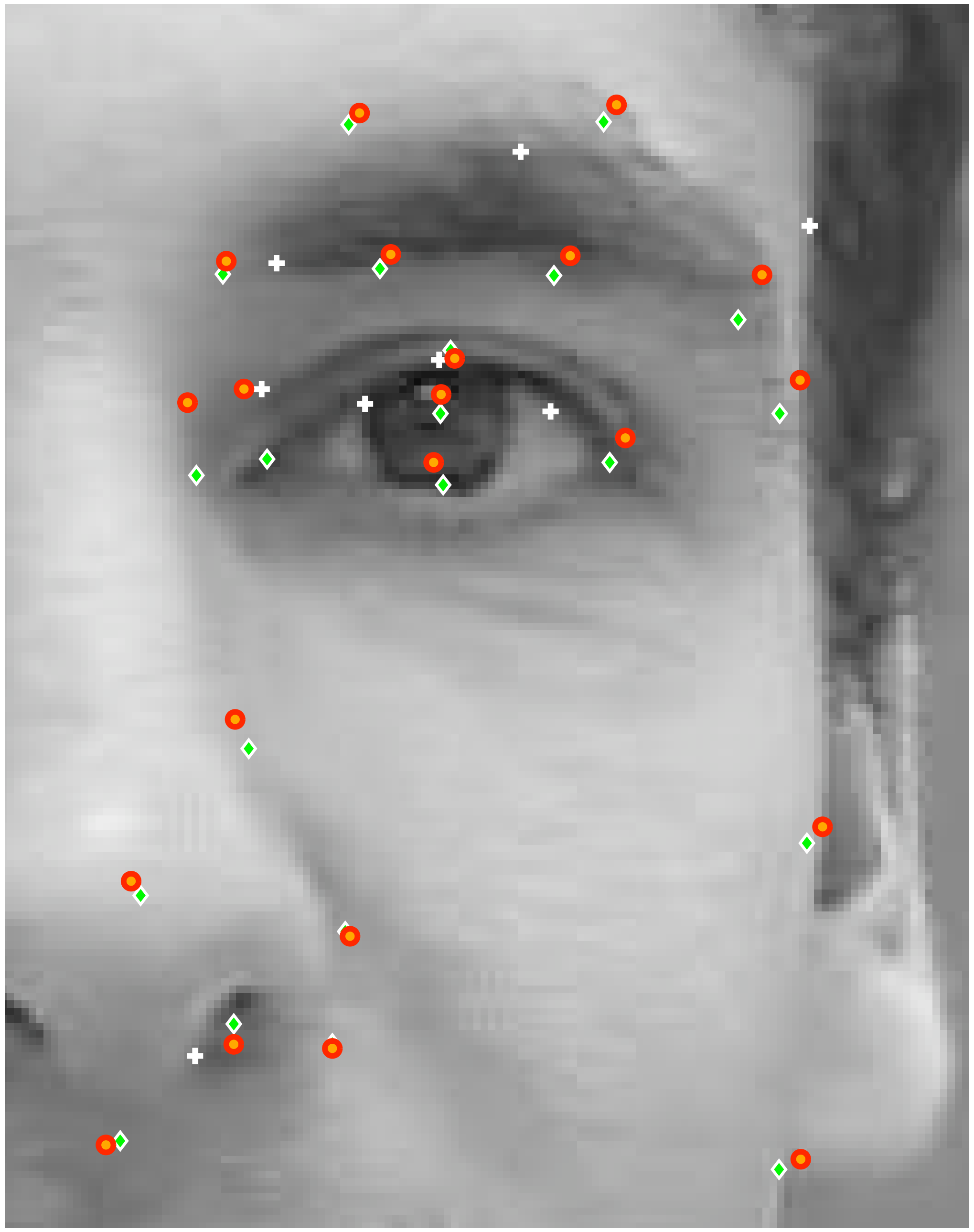
Examples





Examples





Examples

modes for 85% variation: before optim: 5   after optim: 2

Examples

After tracking

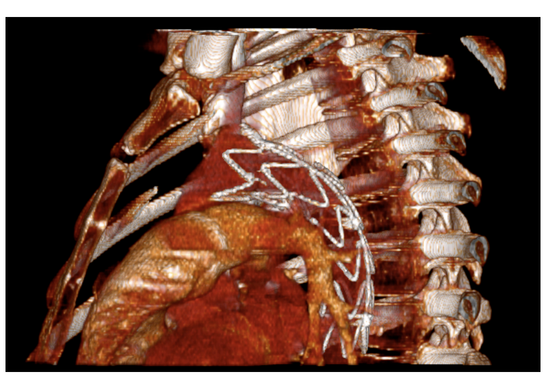
After MDL based registration

1st mode

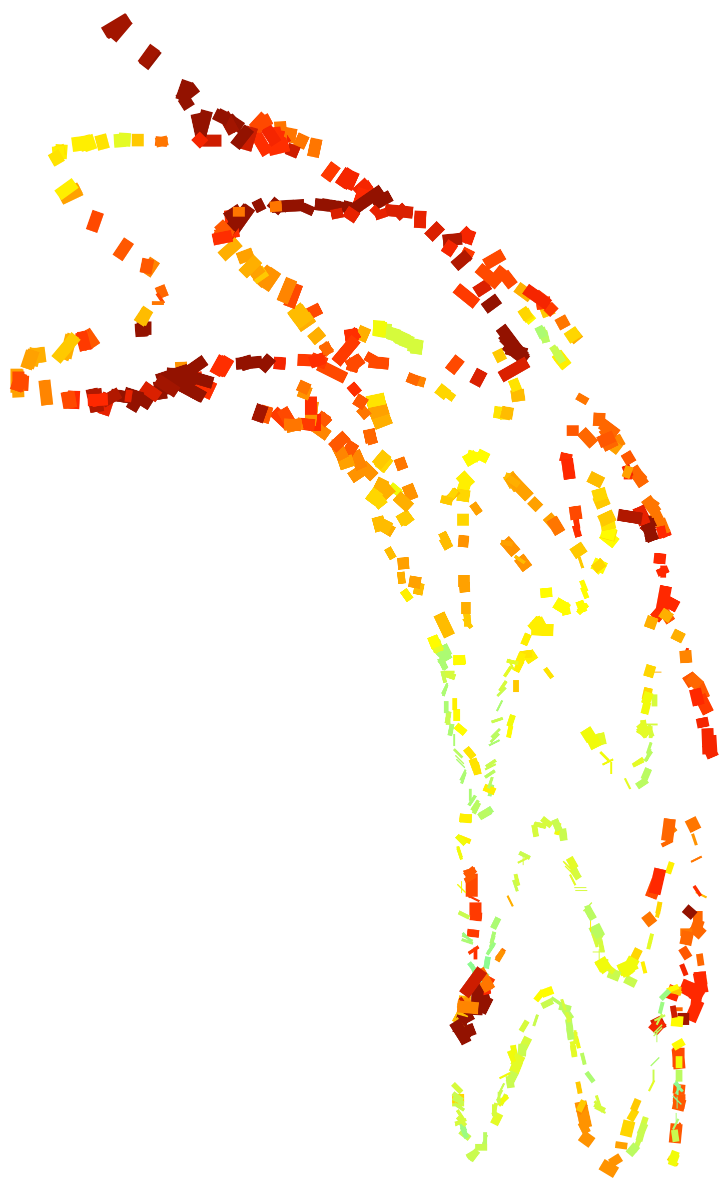
2nd mode

1st mode

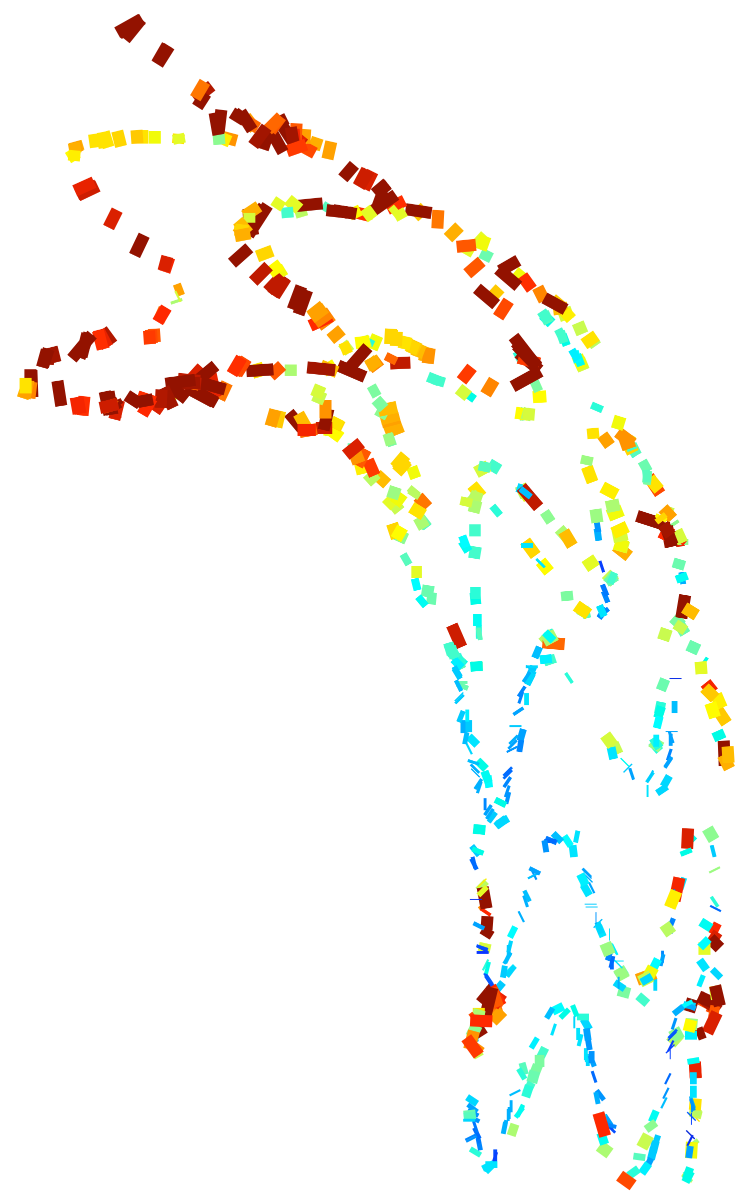
2nd mode



Stent deformation

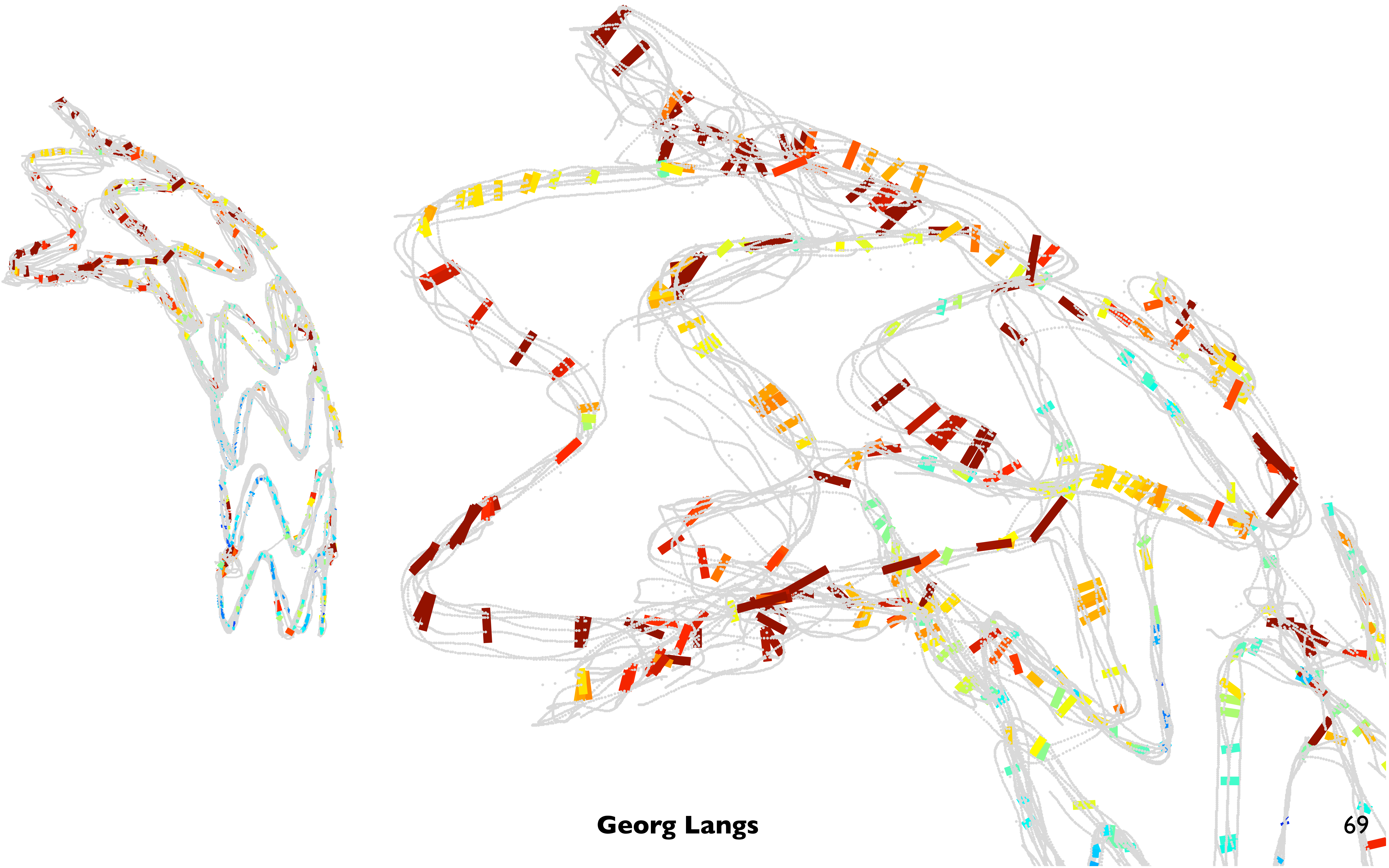


Local compactness

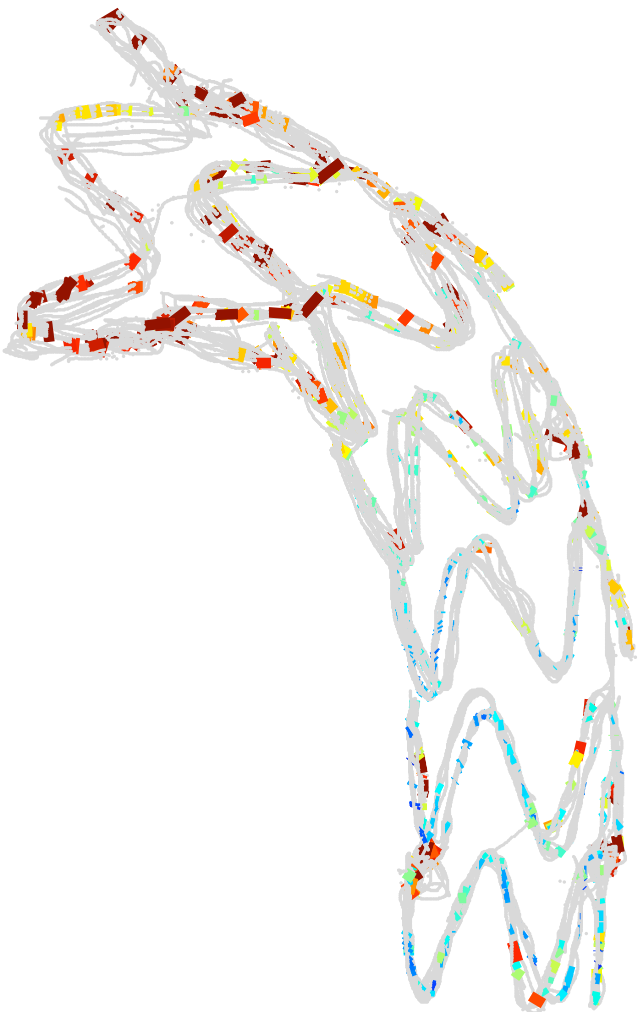


Deformation

Stent deformation



Stent deformation

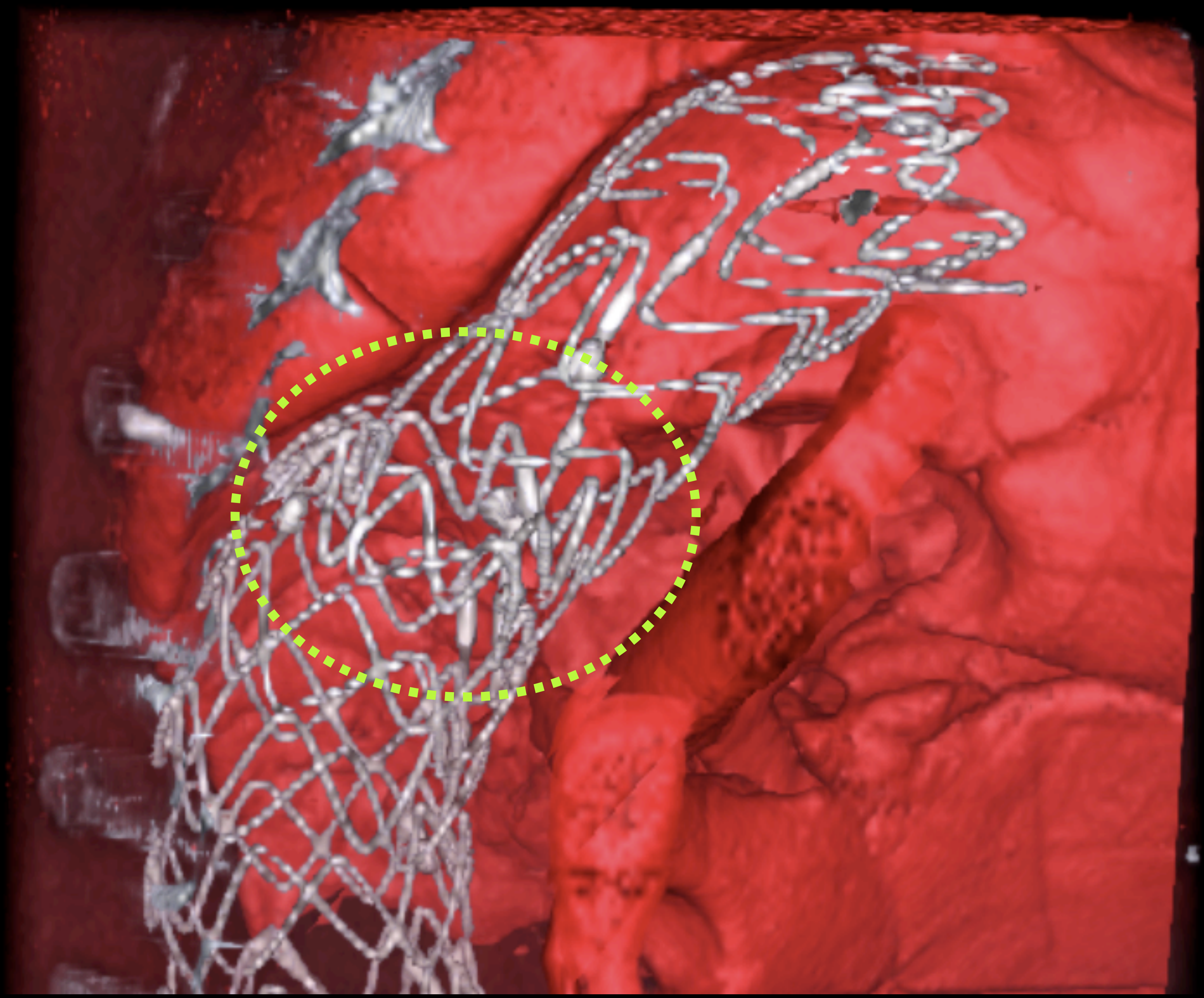


Wrap Up

- Learn model: un-supervised or weakly supervised
- No manual annotation
- Learn correspondences between images by optimizing MDL criterion
- Necessary for models of complex data
- Can be used to capture information not accessible otherwise to human experts

1. The Structure of Models

Stent-Grafts

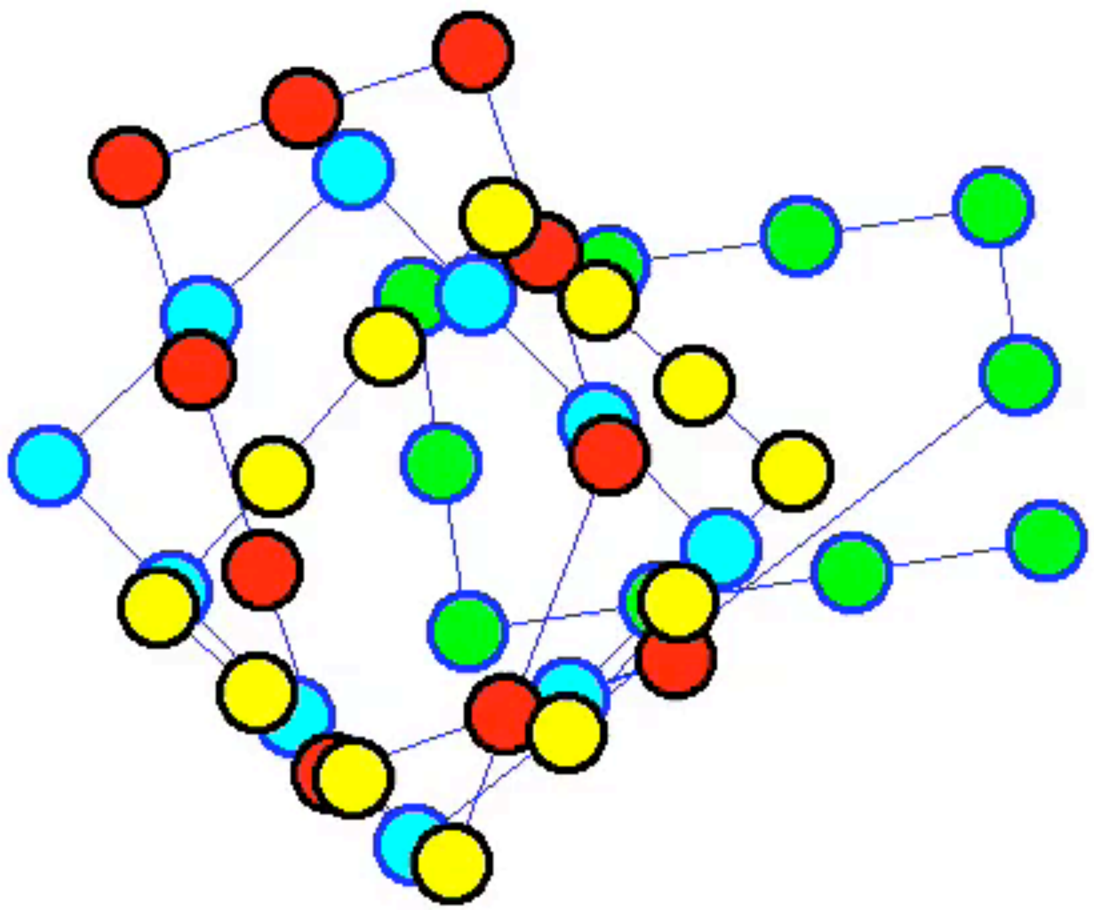


Georg Langs

Topology?

- In modelling we use topology to propagate information, to express and use dependencies
- Lets not define topology a priori
- Let the observed dependencies establish topology

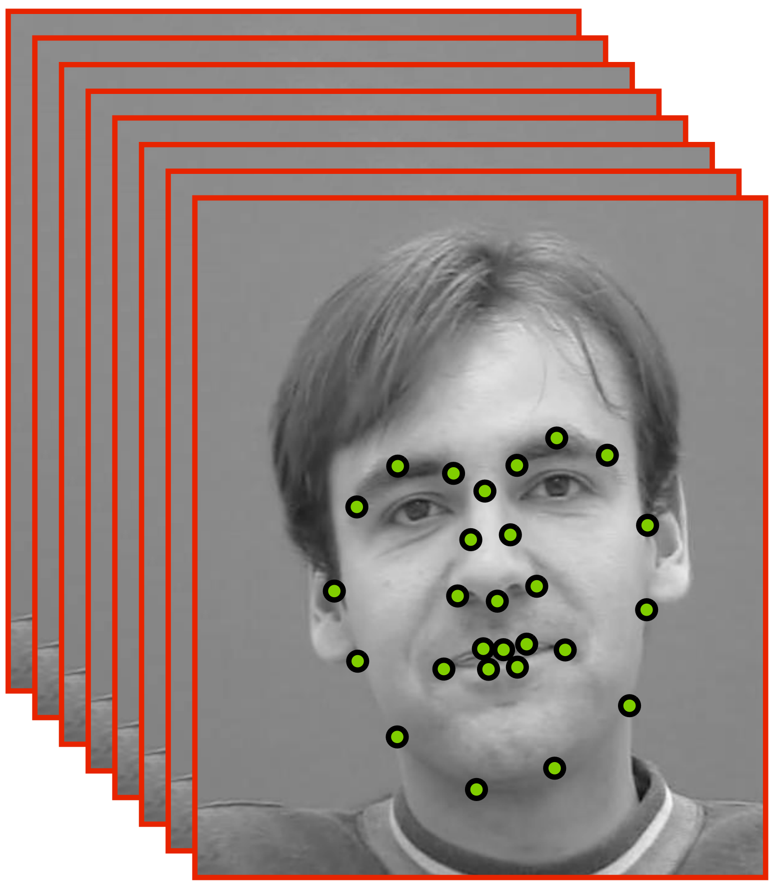
Artificial Example



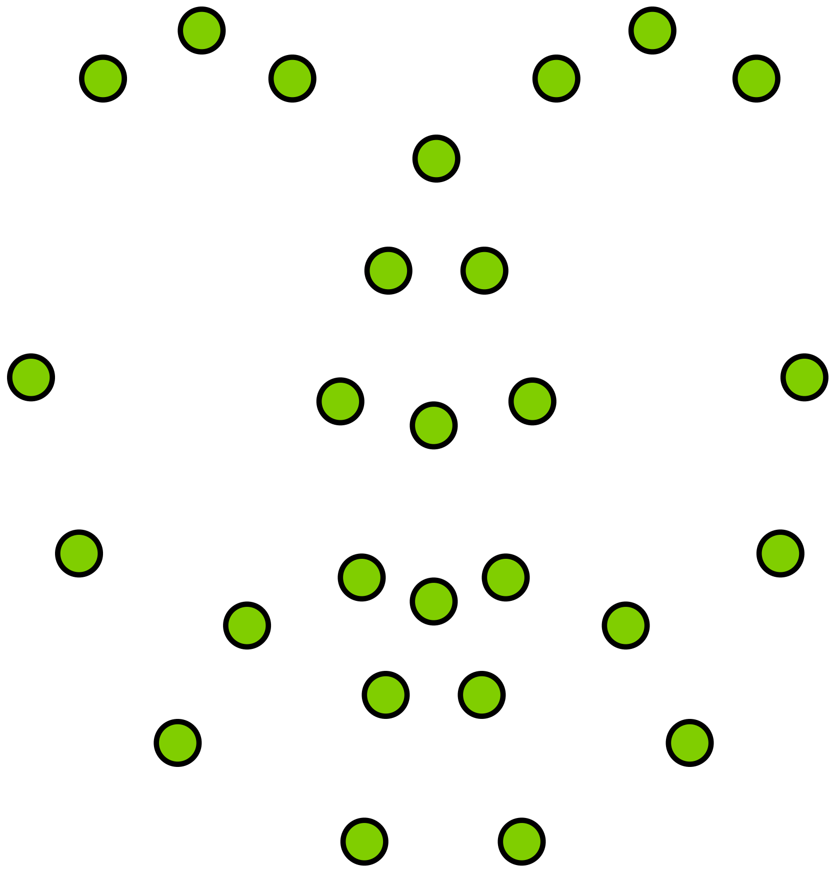
Measure dependencies

- Quantify dependencies between landmarks in the structure
  - tool: model complexity / description length
- Represent the landmarks as vertices in a graph
- Weights of the edges correspond to the complexity of a model encompassing the two landmarks
- This is a Markov chain, and thus has nice properties

Building a Markov chain

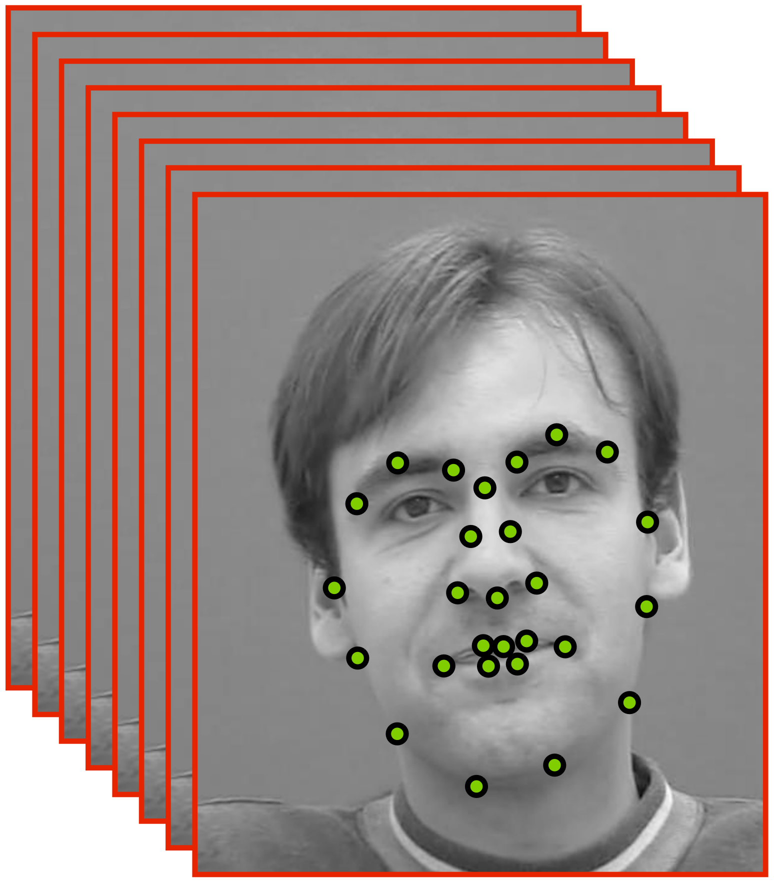


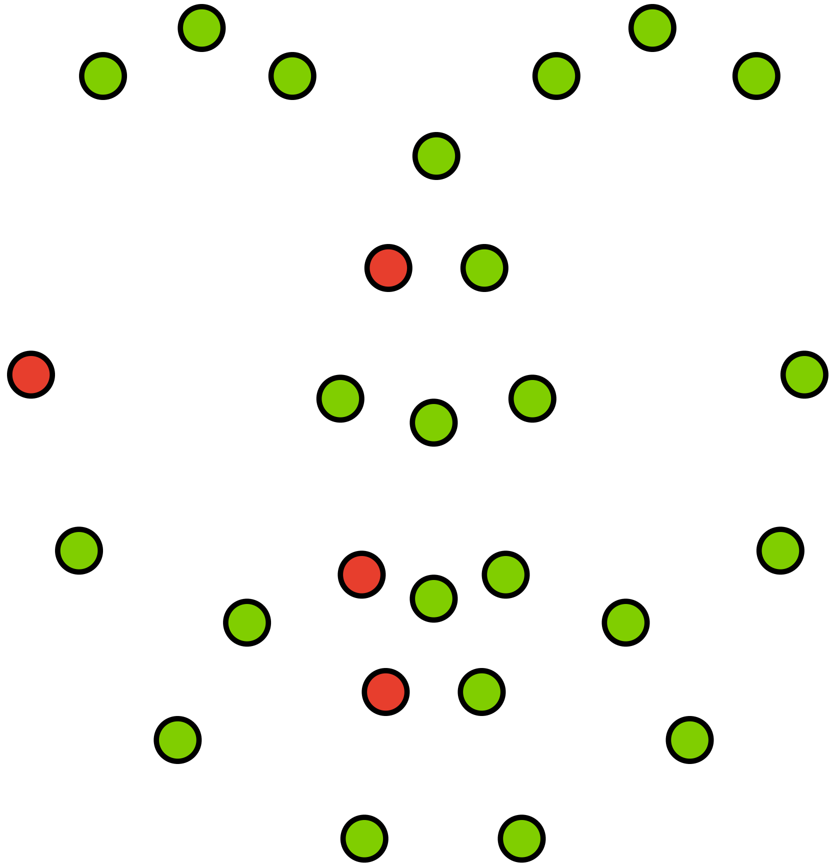
Set of examples each
with landmark positions



Landmarks

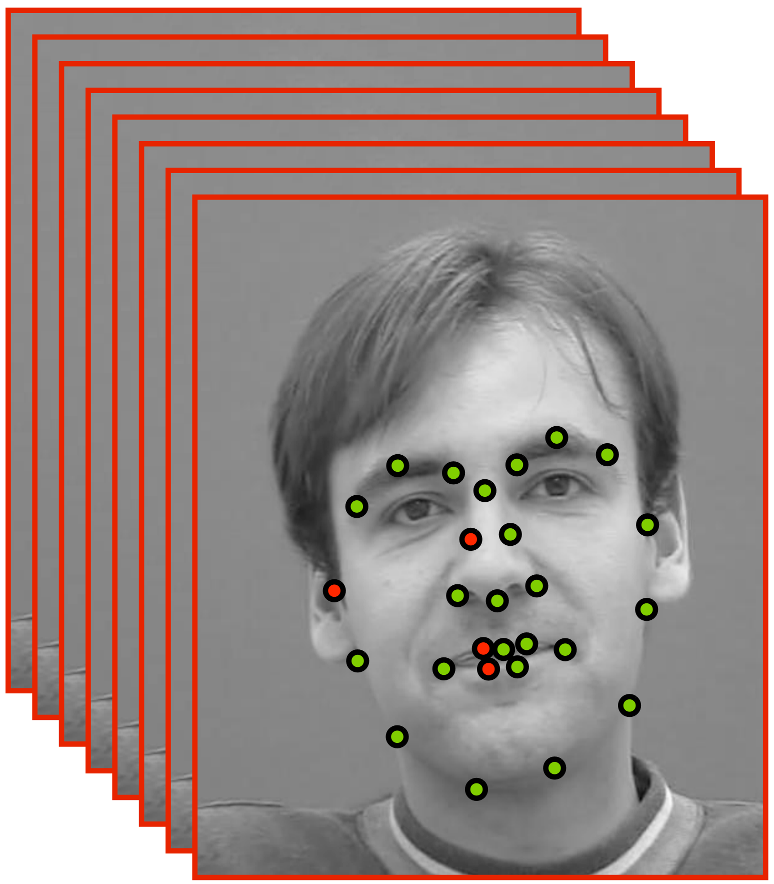
Sampling with sub-models

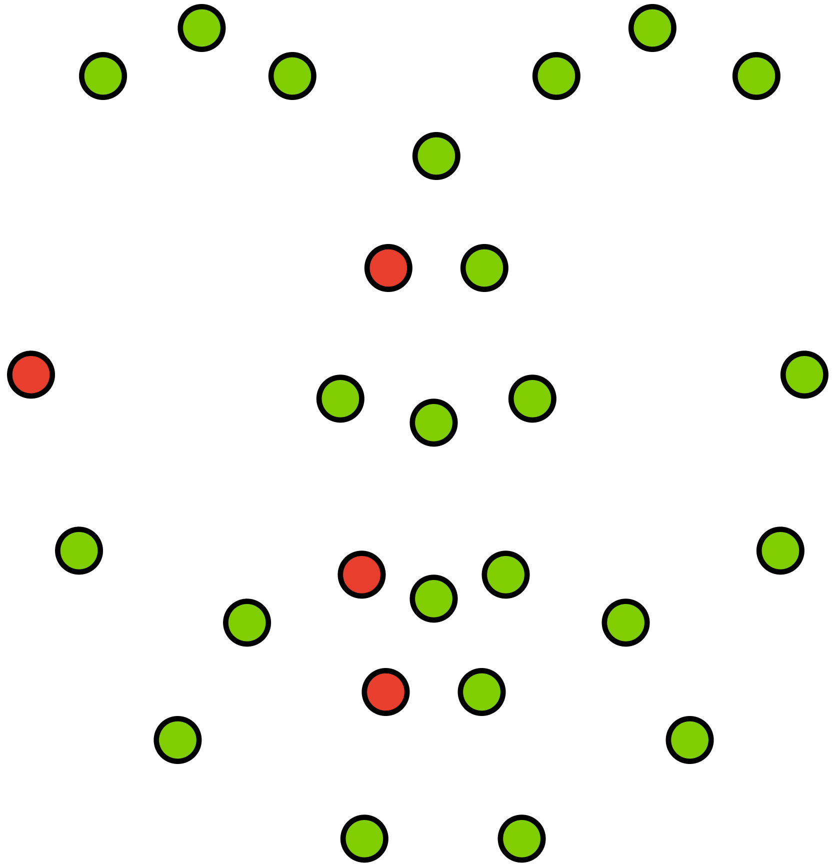




Choose a sub-set of landmarks

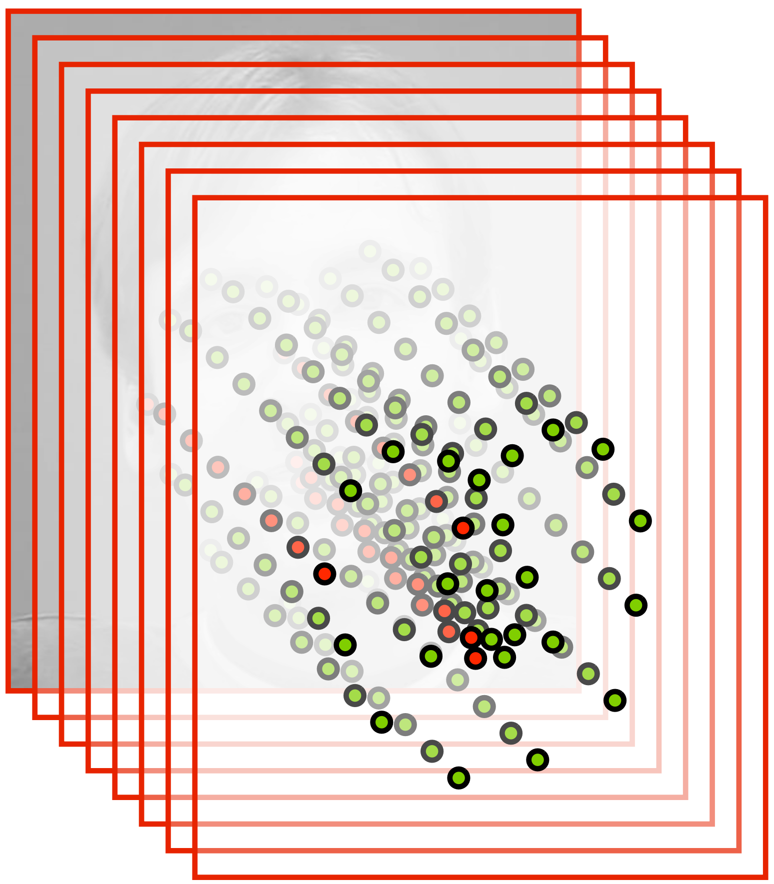
Sampling with sub-models

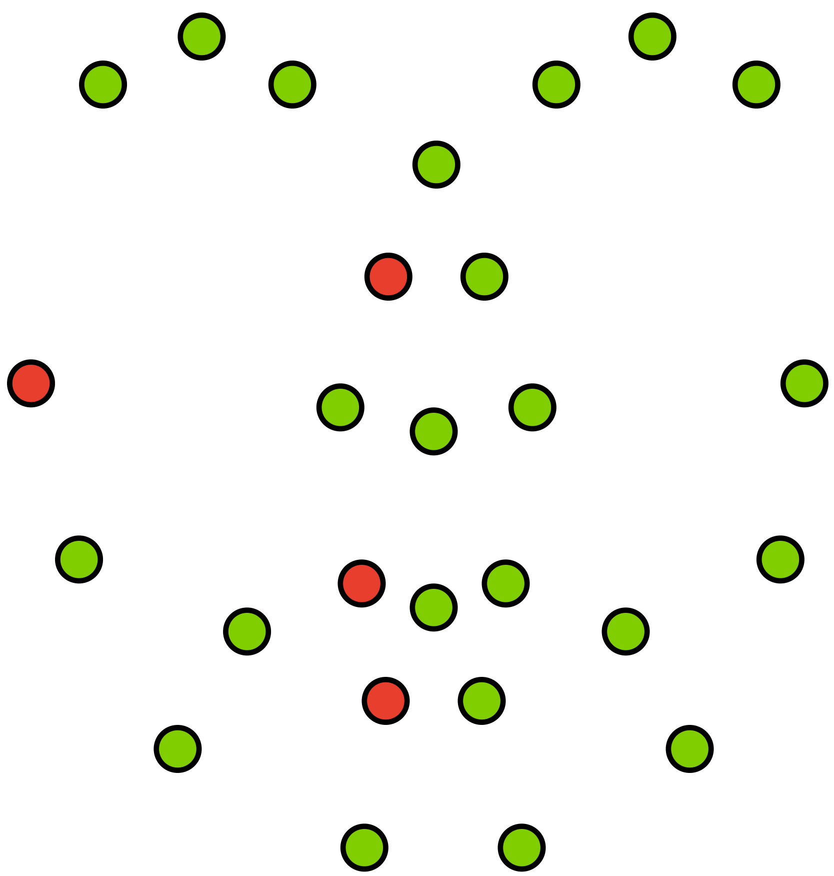




Gather the positions of these landmarks in the training set

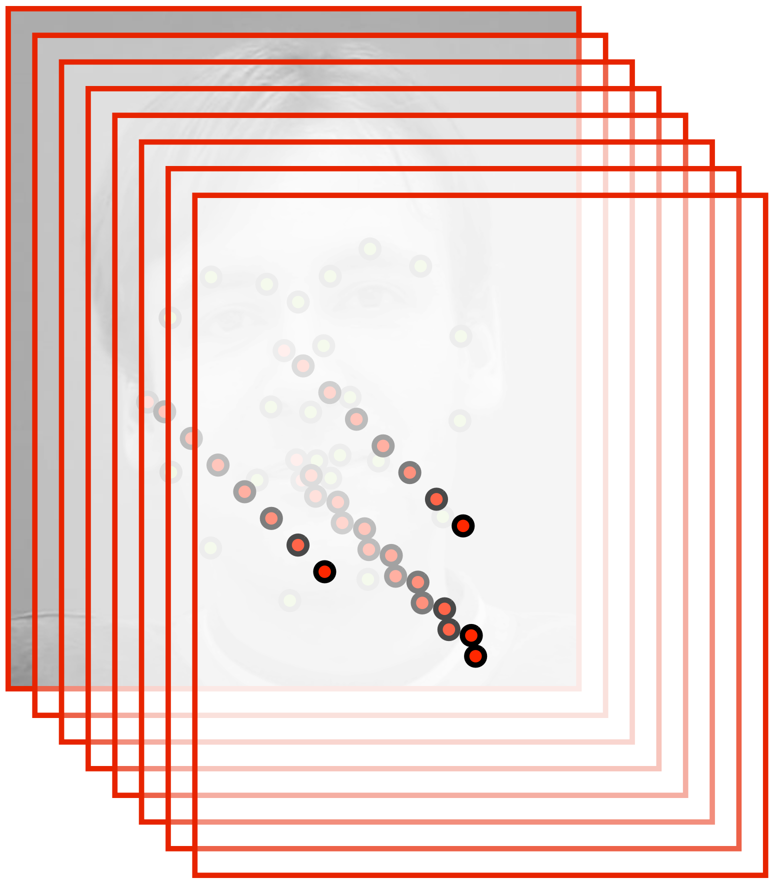
Sampling with sub-models

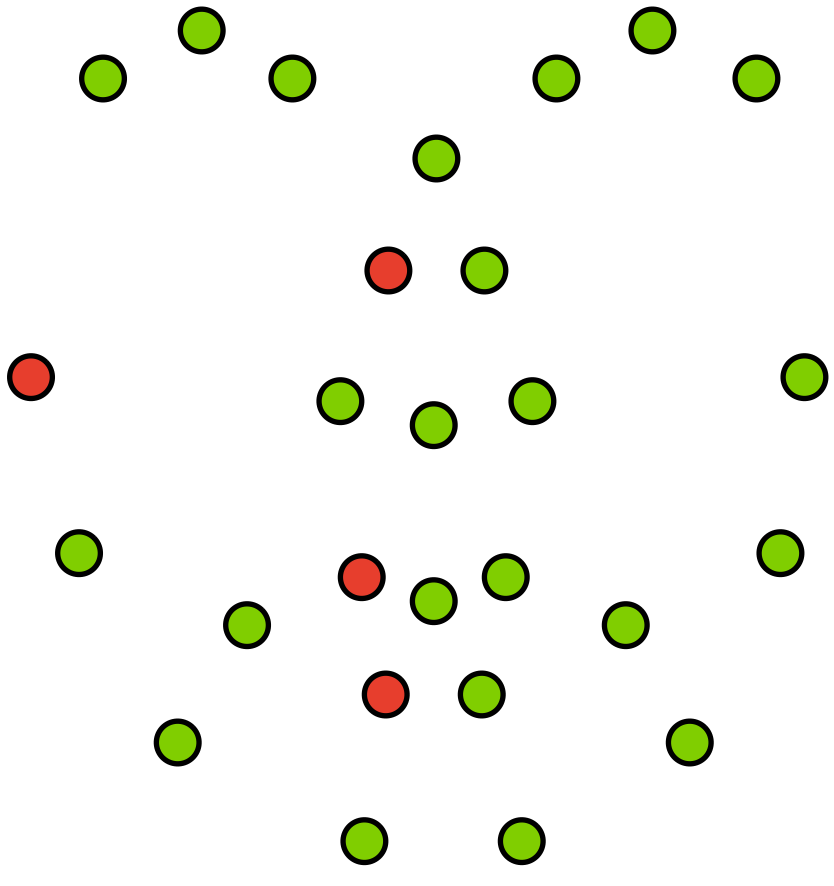




Gather the positions of these landmarks in the training set

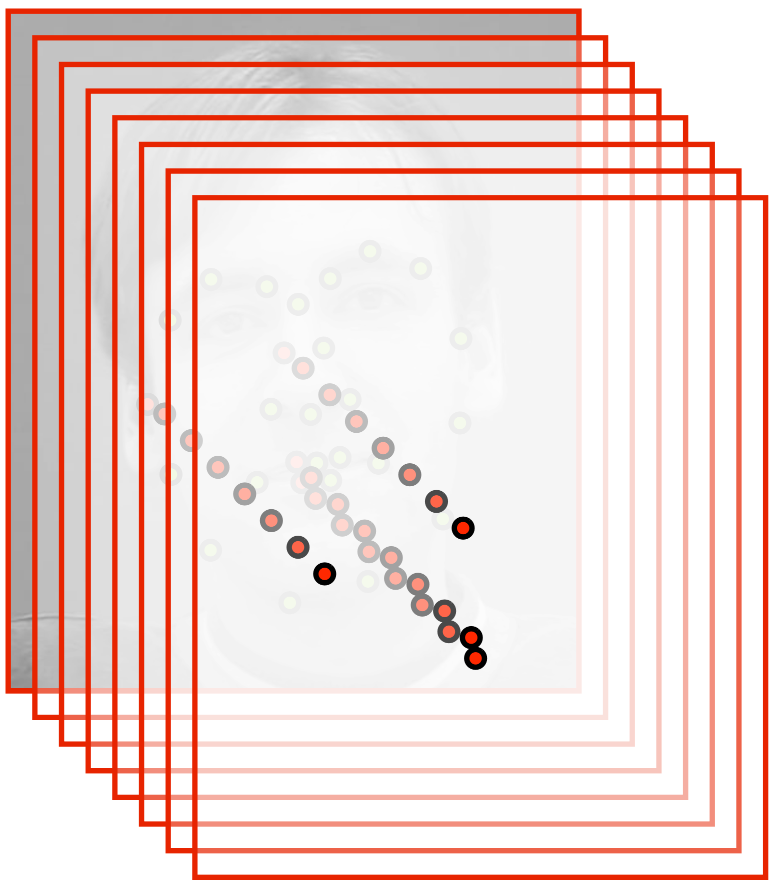
Sampling with sub-models





Gather the positions of these landmarks in the training set

Calculating sub-model complexity

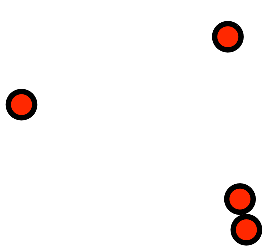


$$\mathcal{L}_{S_{sub}} = L(M) + L(D|M) + \mathcal{R}$$

Model — Data encoded with model

For the sub-set of landmarks: calculate description length of the model and the data encoded with the model.

S_{sub} :



Can be viewed as affinity between these landmarks.

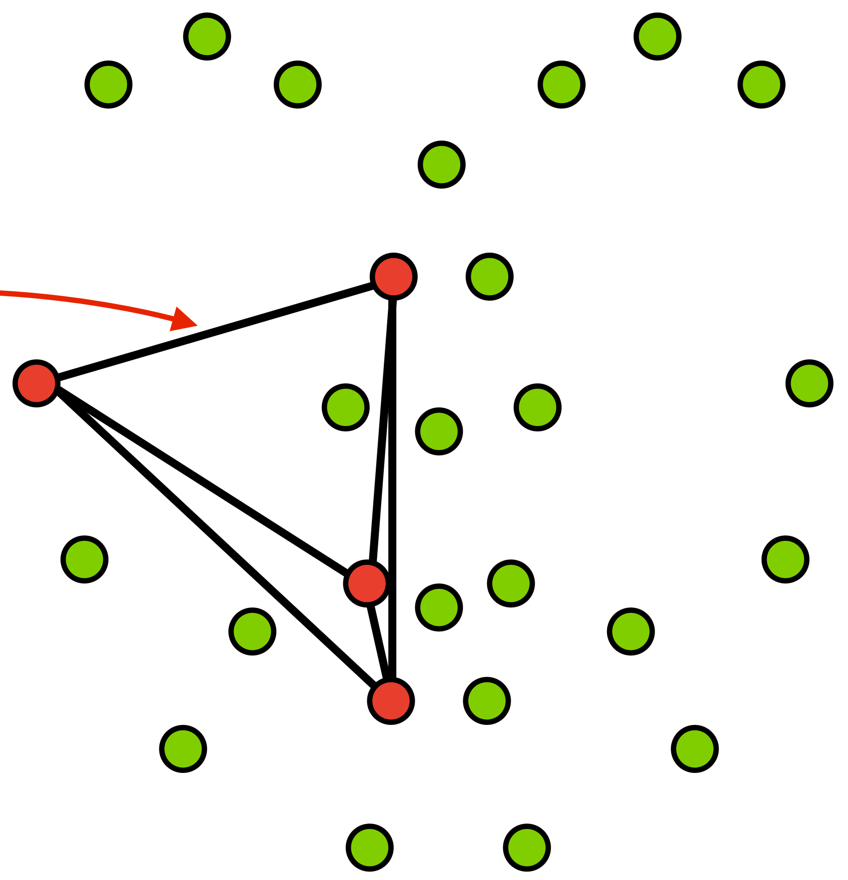
Building a Markov chain

$$d(i,j) = \min_{S_{sub}}(\mathcal{L}_{S_{sub}} | i,j \subseteq S_{sub} \text{ and } \#S_{sub} = k)$$

$$k(i,j) = e^{-\frac{d(i,j)}{\epsilon}}$$

$$d(i) = \sum_j k(i,j)$$

$$p(i,j) = \frac{k(i,j)}{d(i)}$$



Assign the edges connecting the landmarks values

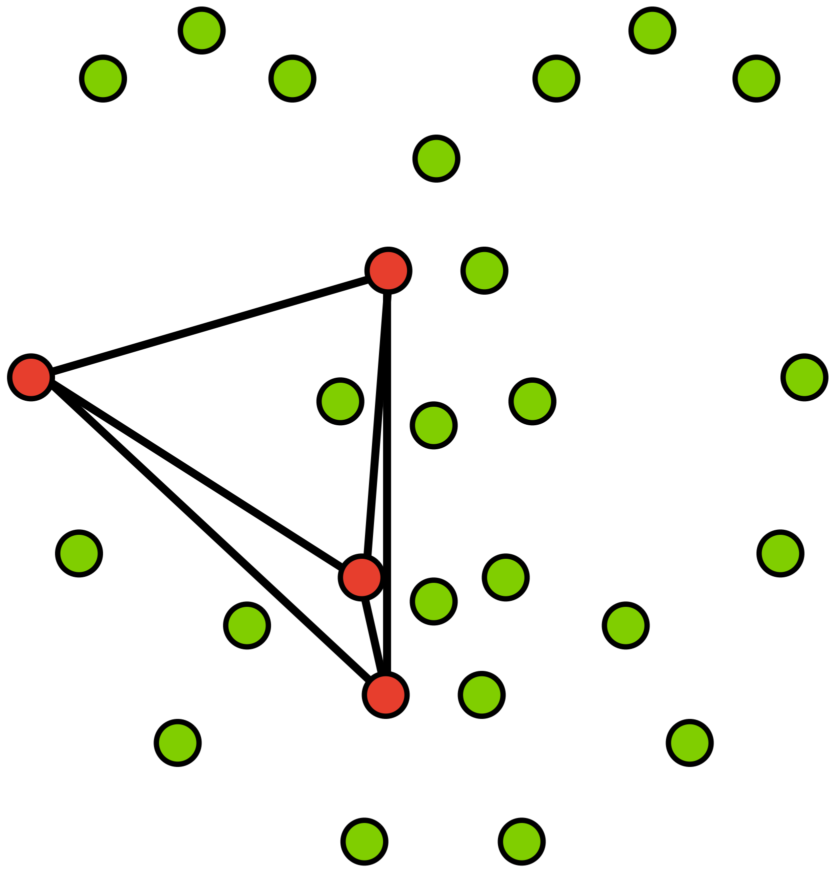
Building a Markov chain

$$p(i,j) = \frac{k(i,j)}{d(i)}$$

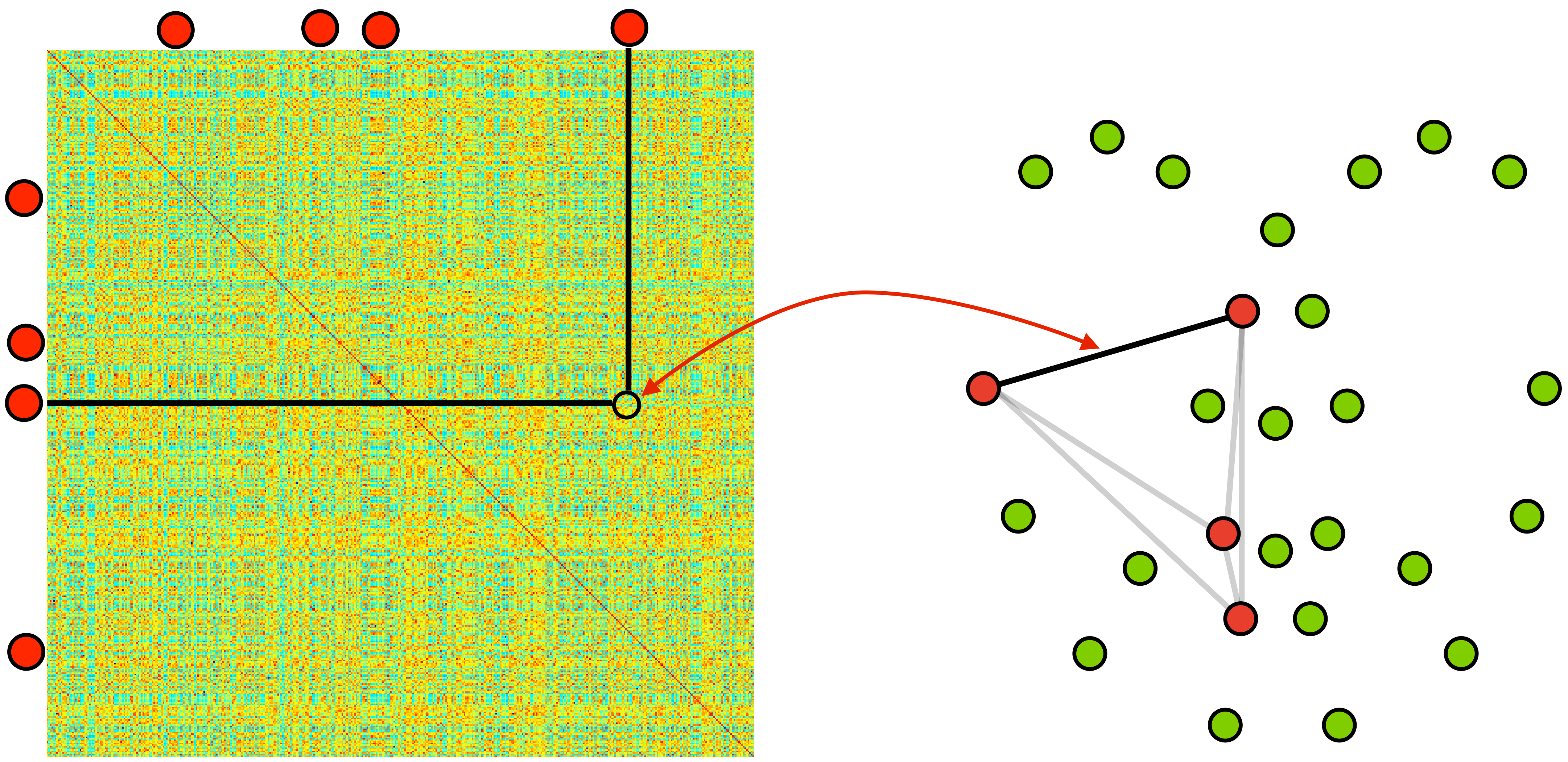
satisfies

$$\sum_j p(i,j) = 1$$

and can be interpreted as the probability of the transition from i to j in one step

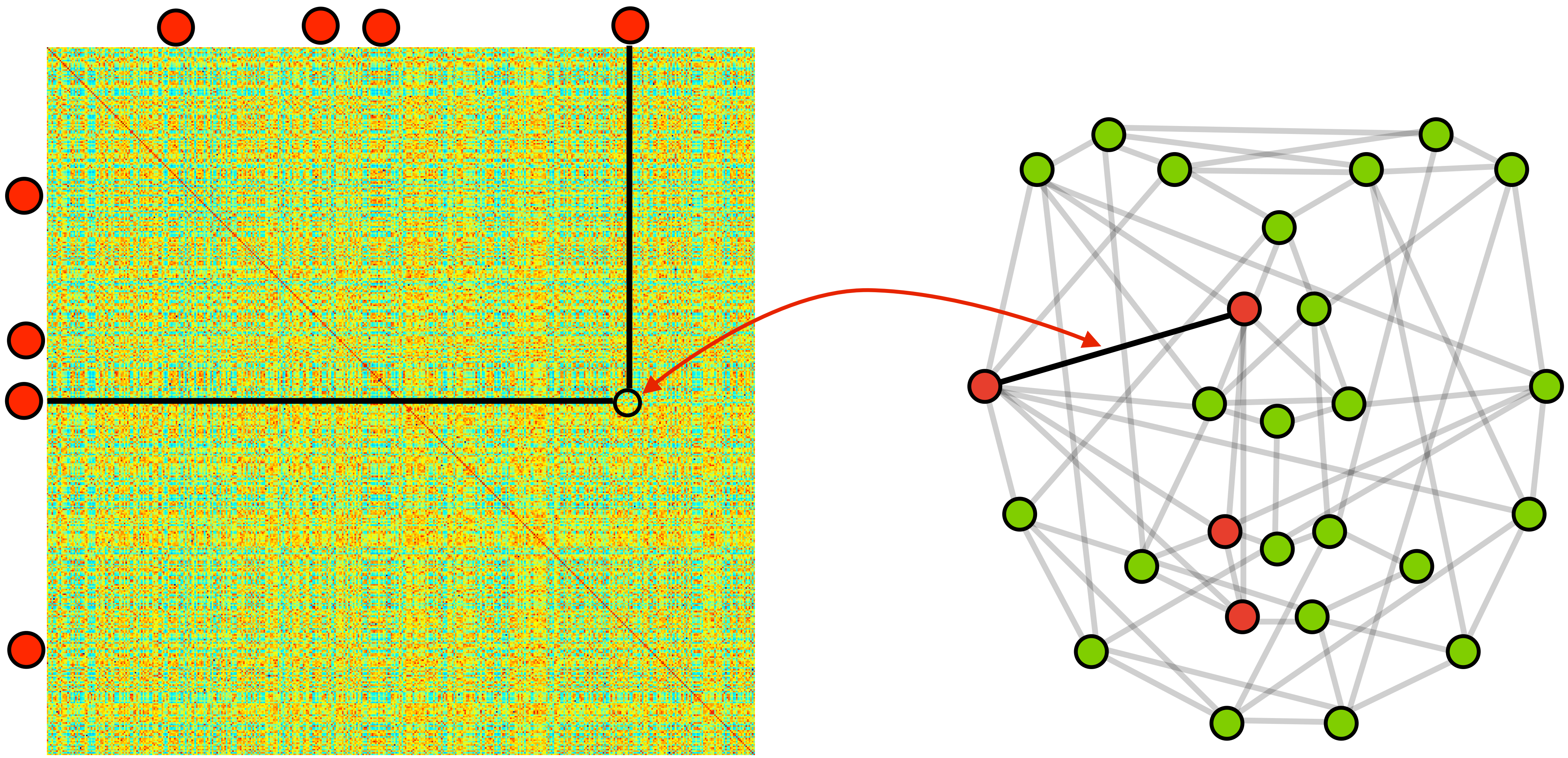


Building a Markov chain



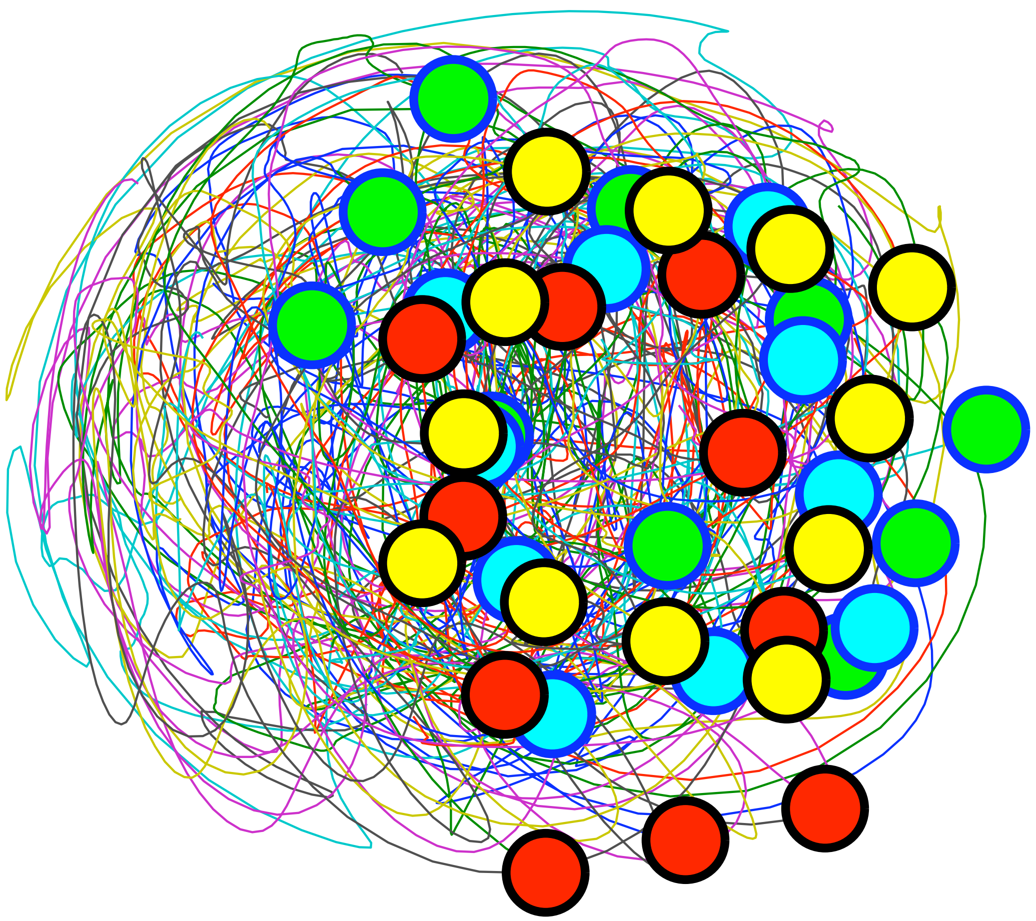
The Markov chain can be represented by a matrix

Building a Markov chain

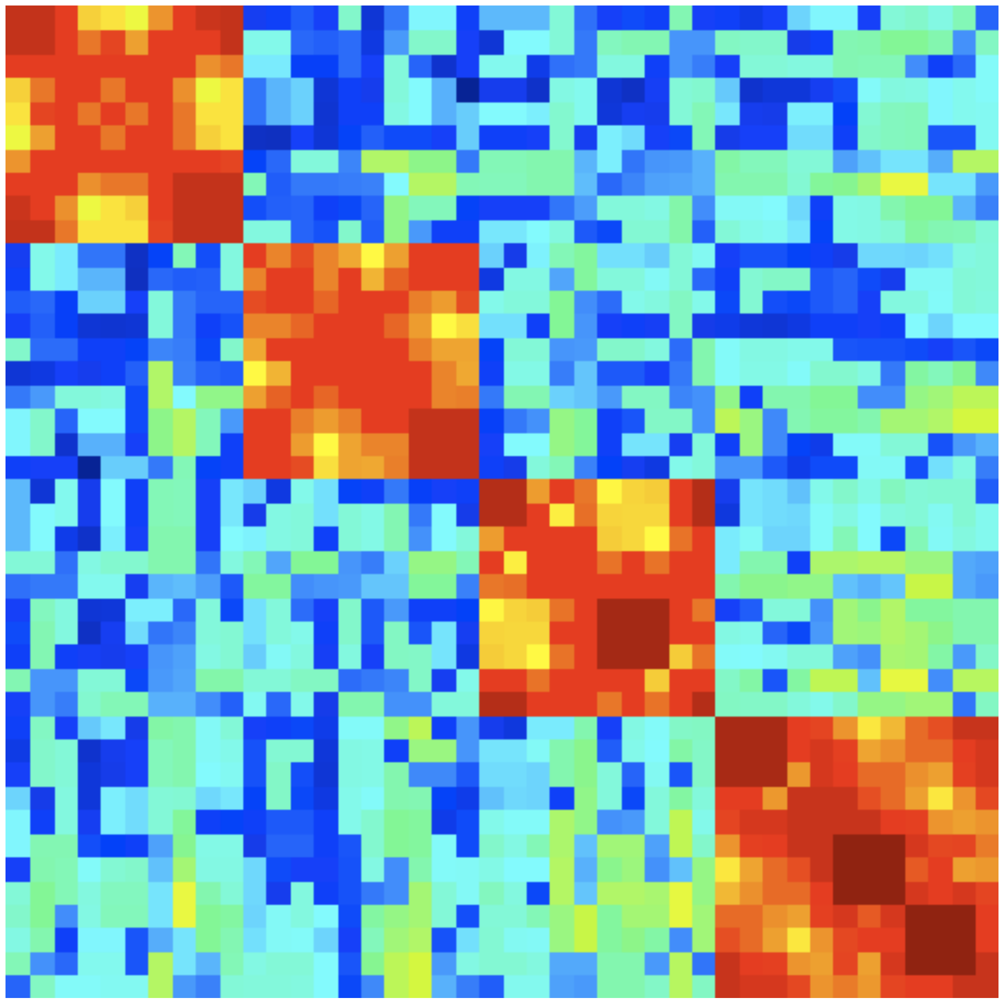


The Markov chain can be represented by a matrix **P**

Example: rotating boxes

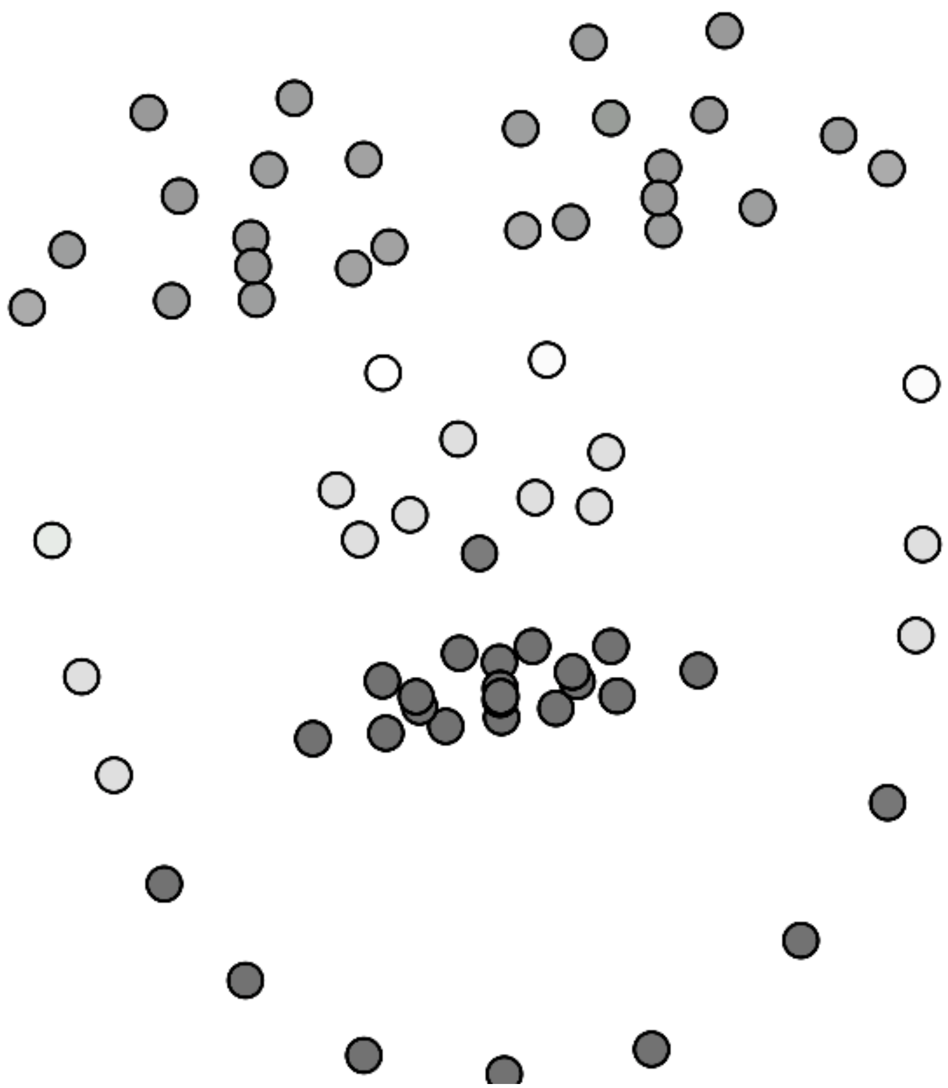


Sequence of 300 examples of 4 rotating boxes

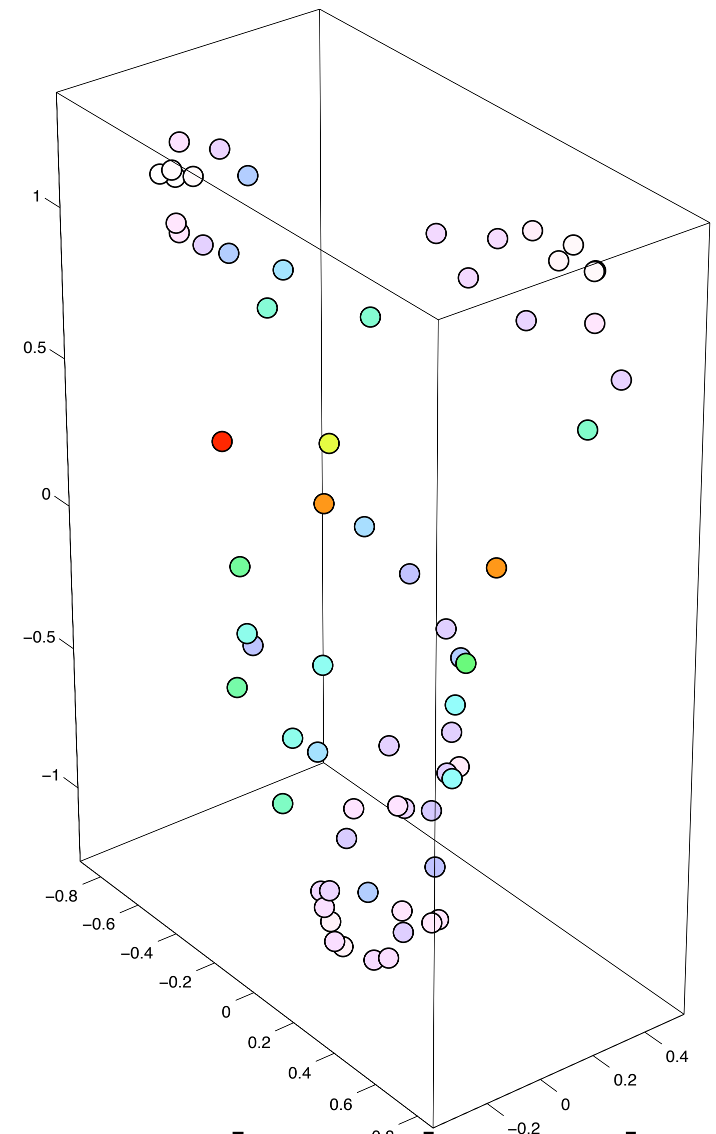


Resulting matrix **P**

Eigenspace: faces

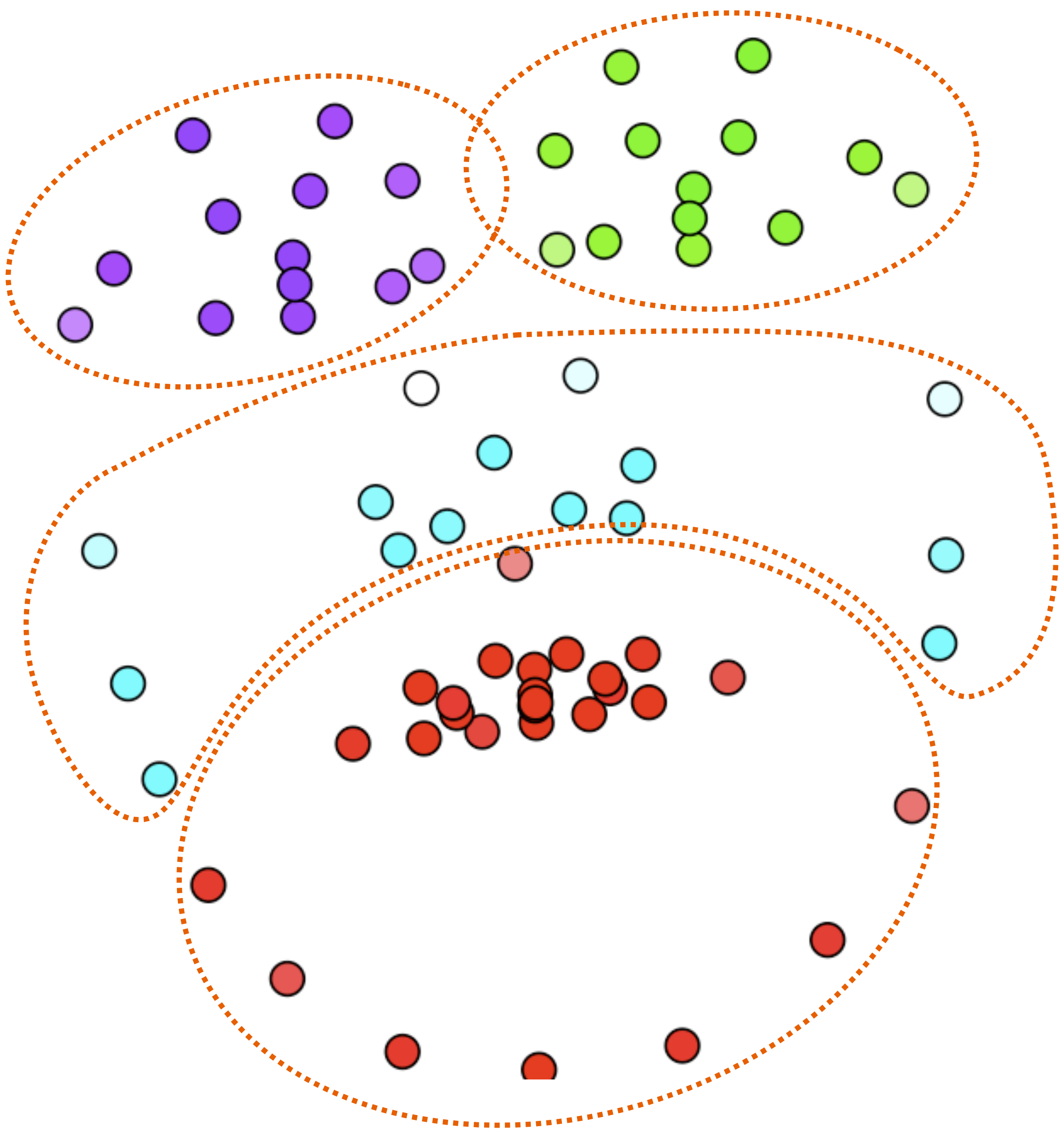


Landmarks

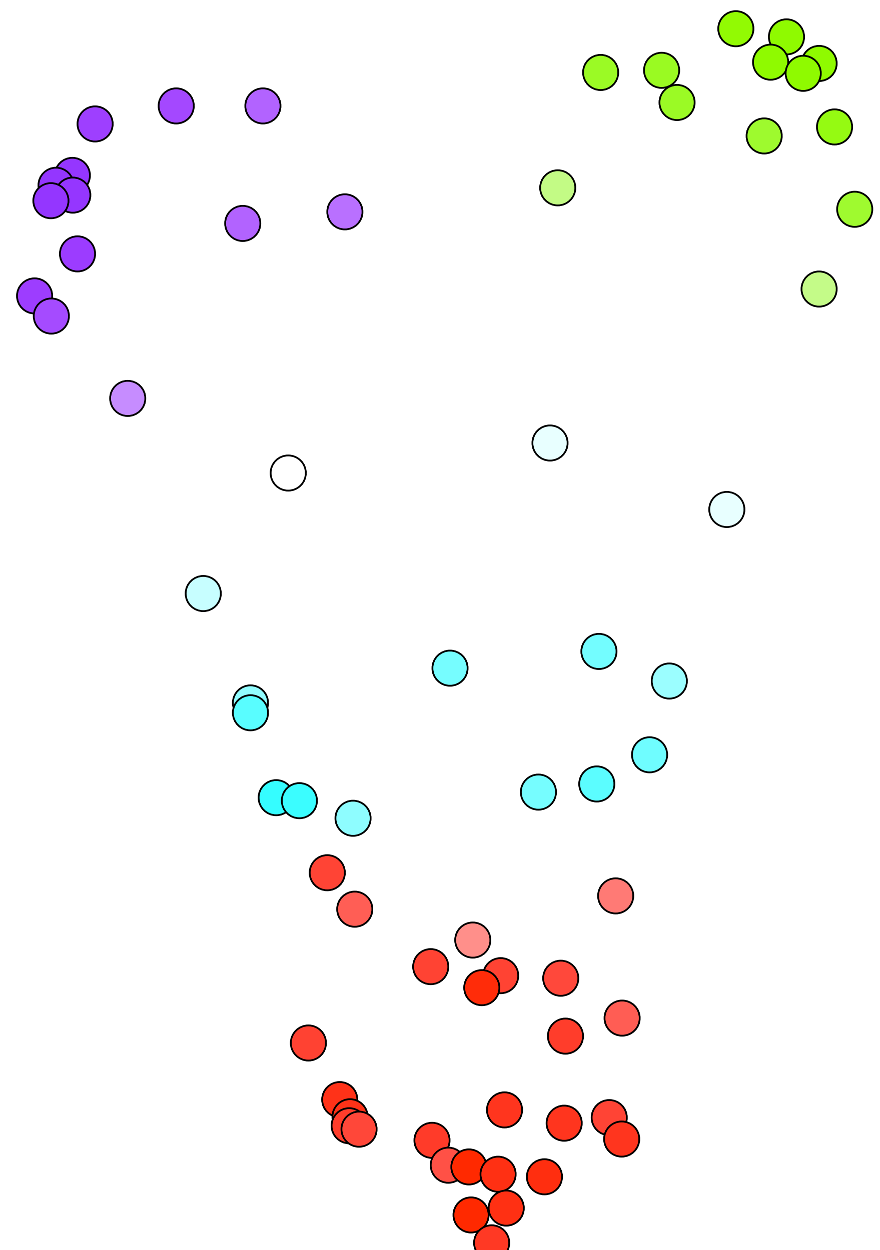


Landmarks in eigenspace: colors encode density

Clusters in the eigenspace

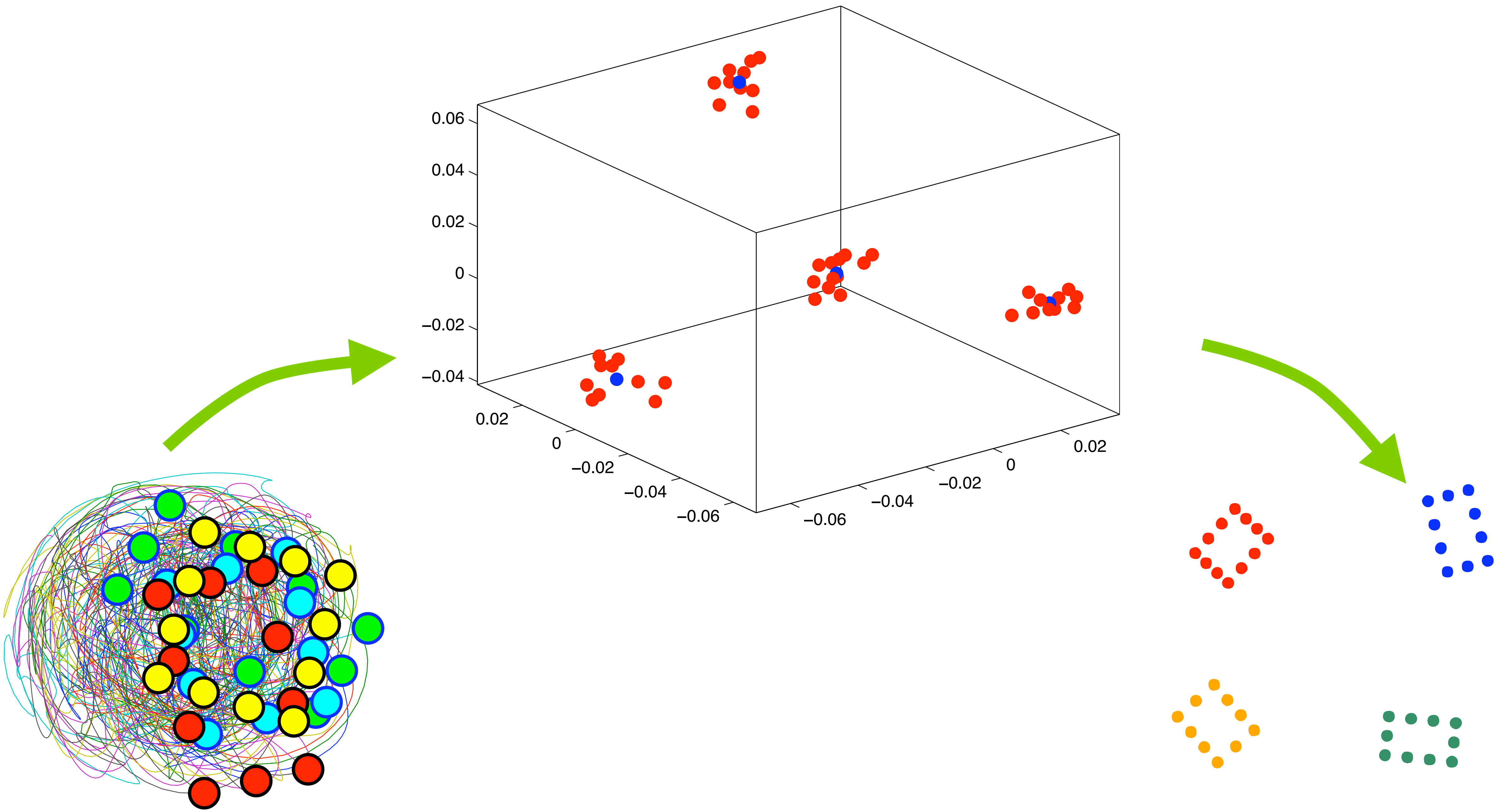


Landmarks

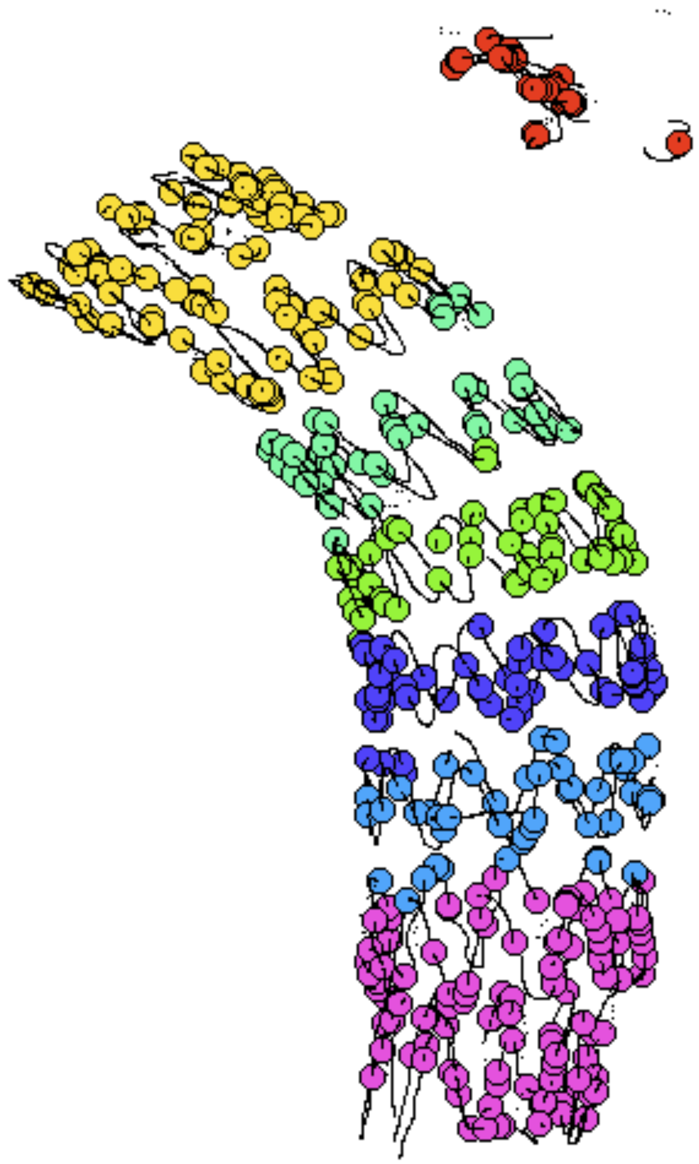


Landmarks in eigenspace: 4 clusters

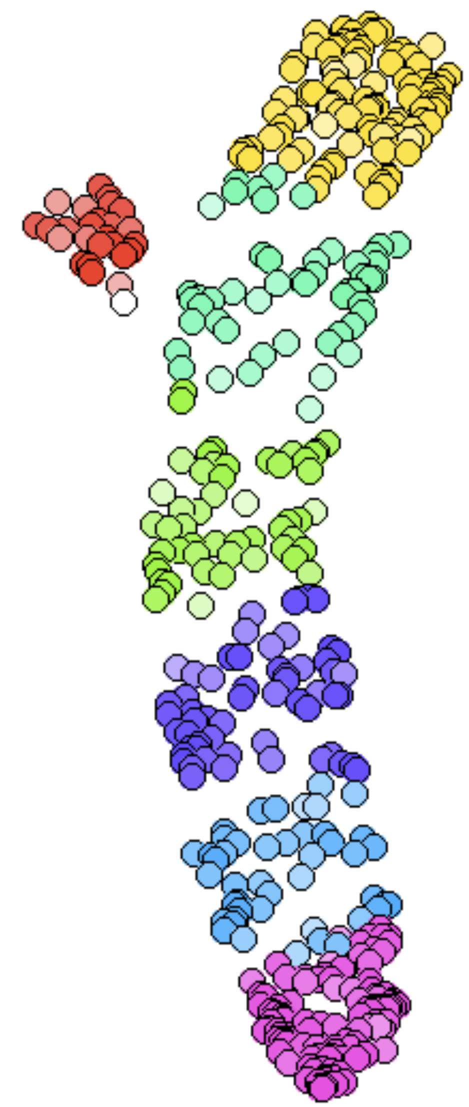
Clusters: boxes



Stents: deformation segmentation

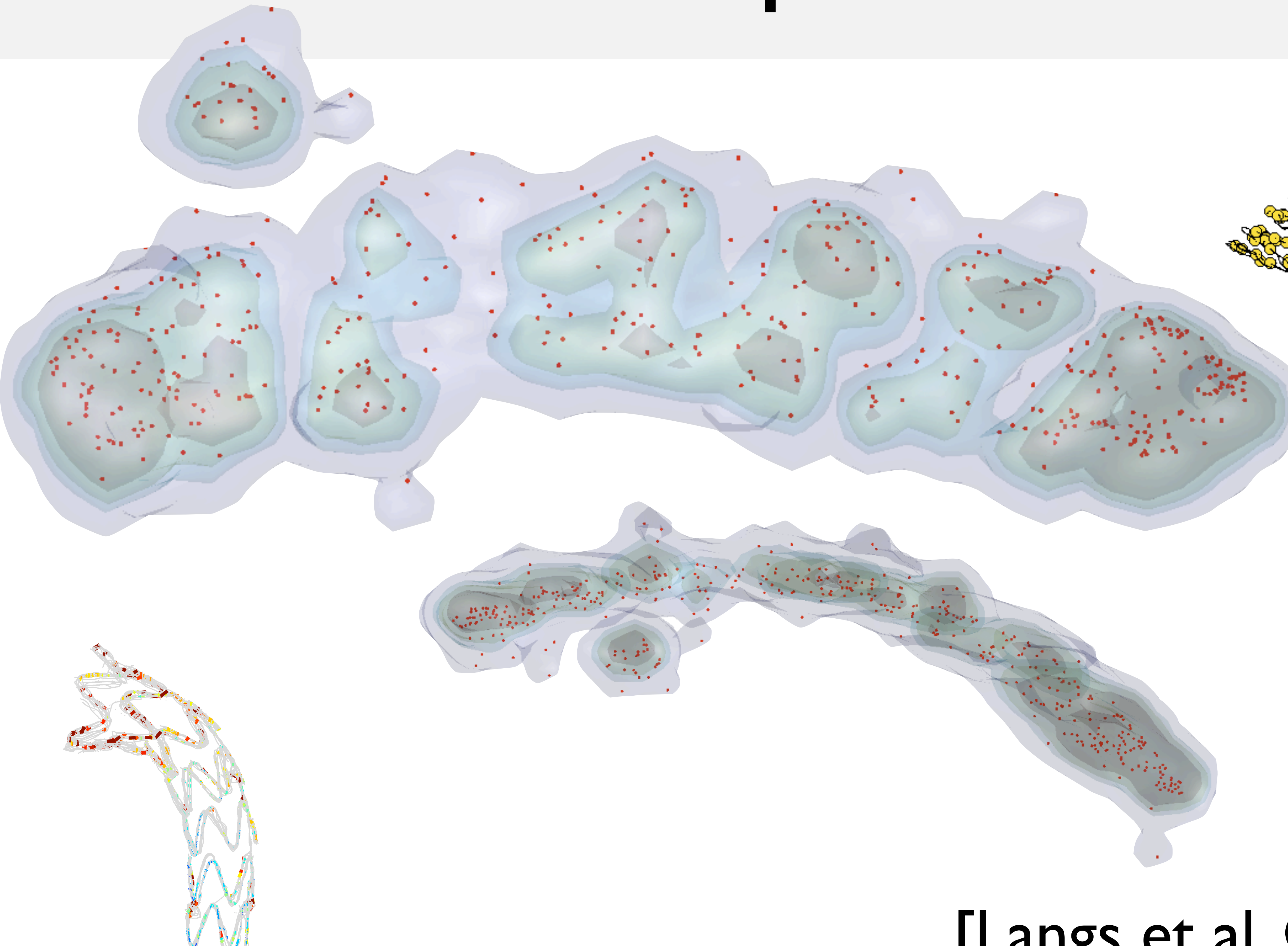


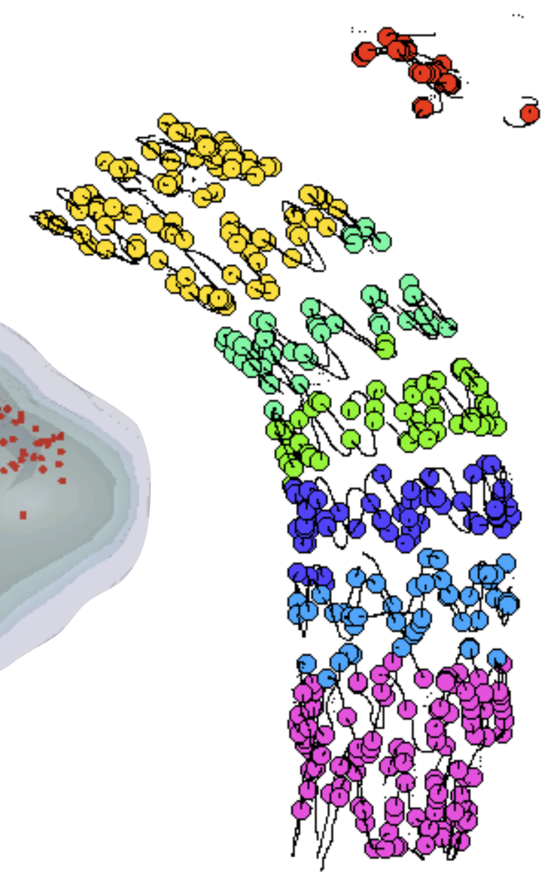
Landmarks



Landmarks in eigenspace

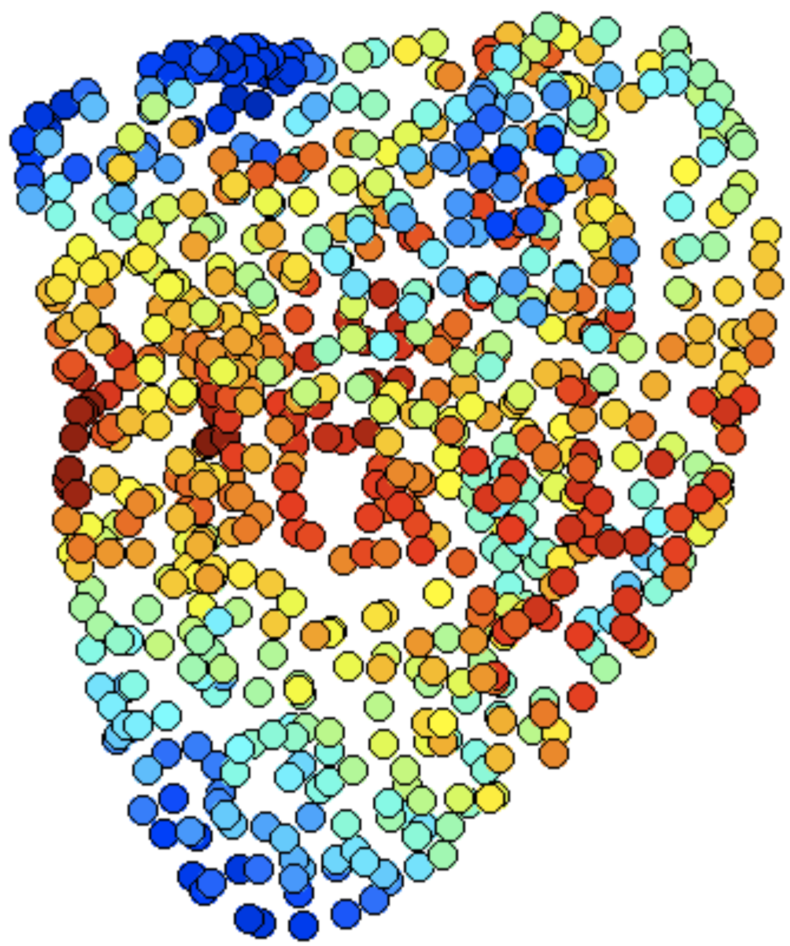
Motion patterns



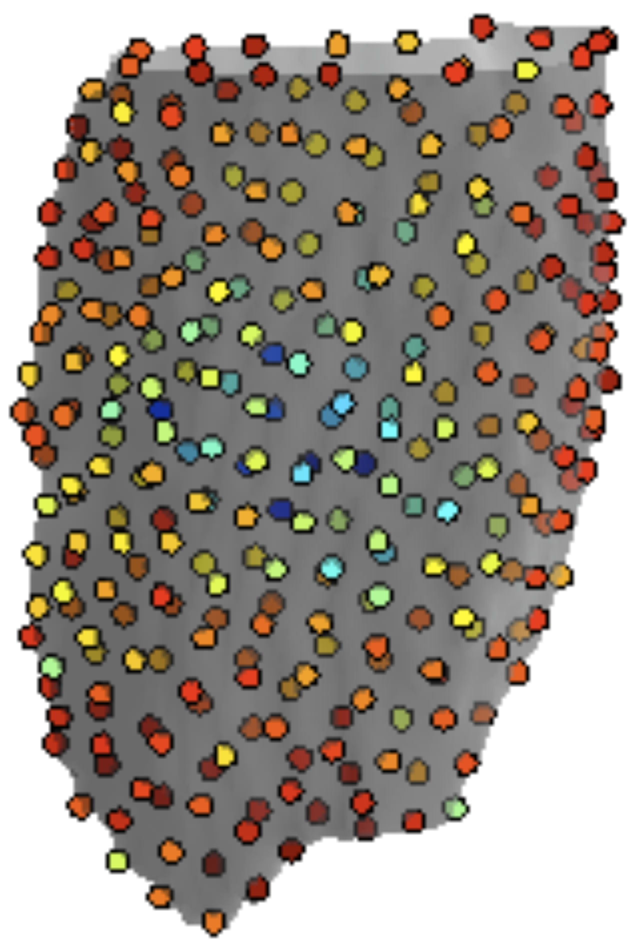


[Langs et al. CVPR'08]

Local deformation complexity



Heart data
(Maxime Taron, Ahmed Besbes)



Muscle data
(Salma Essafi)

Wrap Up

- Models of appearance and shape variation:
    - Active Appearance Models
- Learn models autonomously from un-annotated data
- Find the structure of behavior in the data
- Necessary prerequisites to use the vast amount of information acquired with medical imaging modalities

Thank you

Contact:
georg.langs@ecp.fr

Try some of the code (coming soon)
and have a look at literature:
www.mas.ecp.fr / vision / Personnel / langs