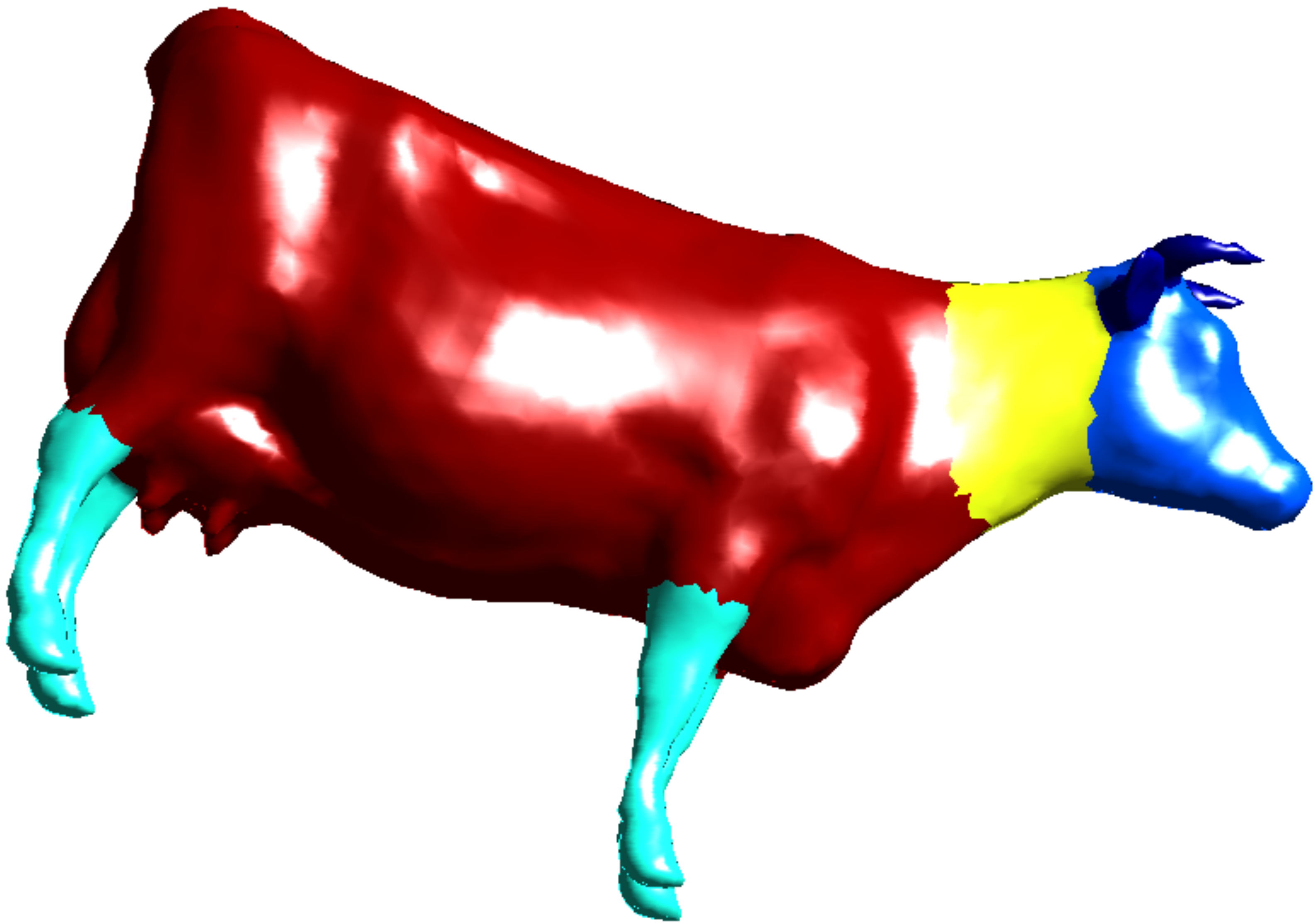
3D Shape Analysis Using Machine Learning

Student: Michael Lindsey

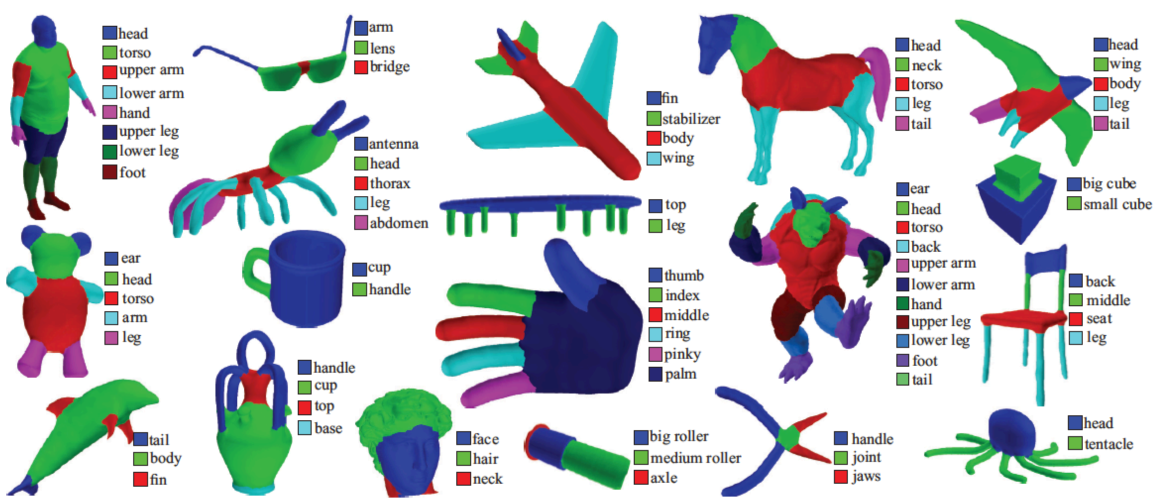
CURIS Project

Guibas Lab



Introduction

- Immediate goal of project: mesh segmentation with labeling
- Only previous work: Kalogerakis et al. 2010
  - Good results
  - Very slow: 8 hours to train on 6 meshes (Xeon E5355 2.66 GHz)



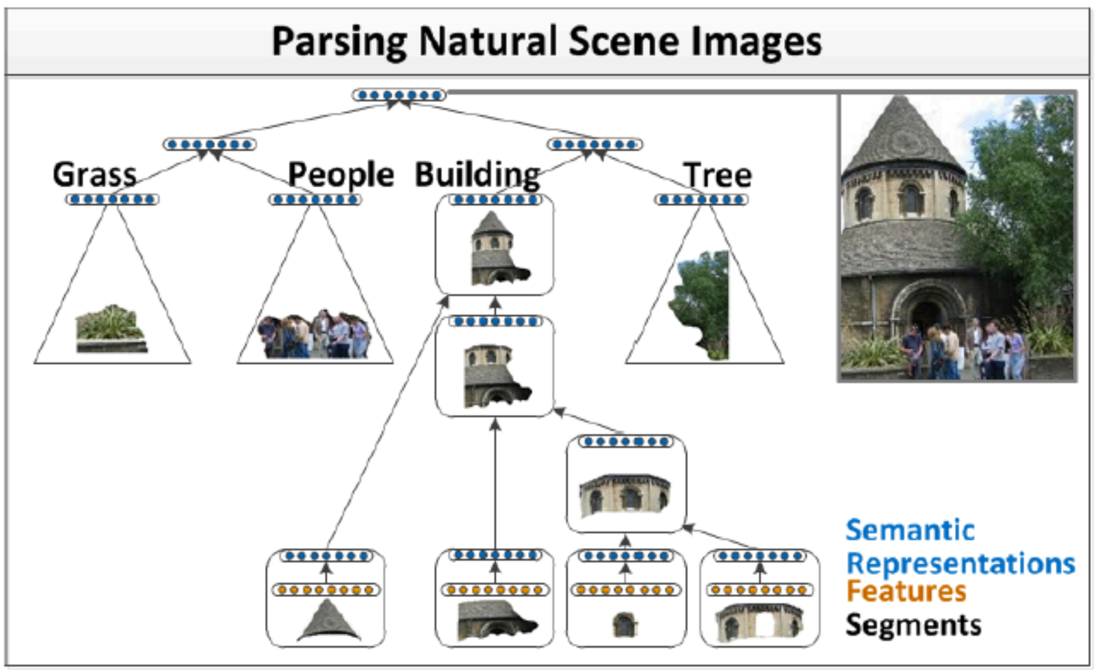
Introduction

- Idea: use features at level of superpixel-like patches
- Adapt method of Socher et al. 2011
  - Used for image segmentation, sentence parsing
  - RNN which learns semantic embedding
- More nebulous/more interesting goal: learn a meaningful embedding of per-superpixel features into "semantic space"
  - Investigate the structure of this space
  - Recover descriptors at all levels

Outline

- Machine learning method
- Oversegmentation
- Features (old and new)
- Segmentation results
- Applications of semantic embedding to shape understanding
- Future directions

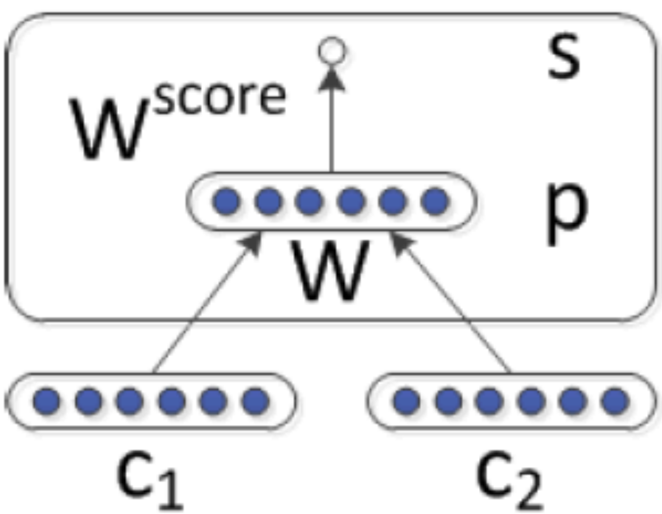
Socher's recursive neural network for image segmentation



Semantic embedding of superpixels:

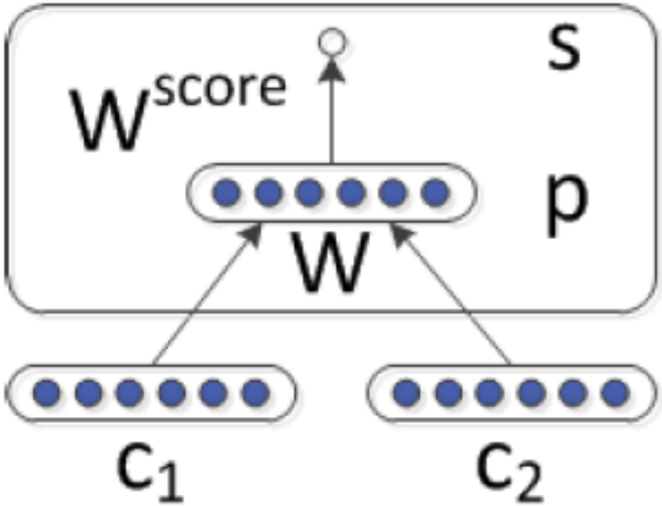
$$a_i = f(W^{sem} F_i + b^{sem})$$

Semantic embedding of patches:



$$p = f(W[c_1; c_2] + b)$$

Scoring the joined patches:



$$s = W^{score} p$$

Labeling joined patches:

$$label_p = softmax(W^{label} p)$$

Oversegmentation

- Need to respect true segment boundaries
- Concave creases
- Edge function from Lai et al. 2009:

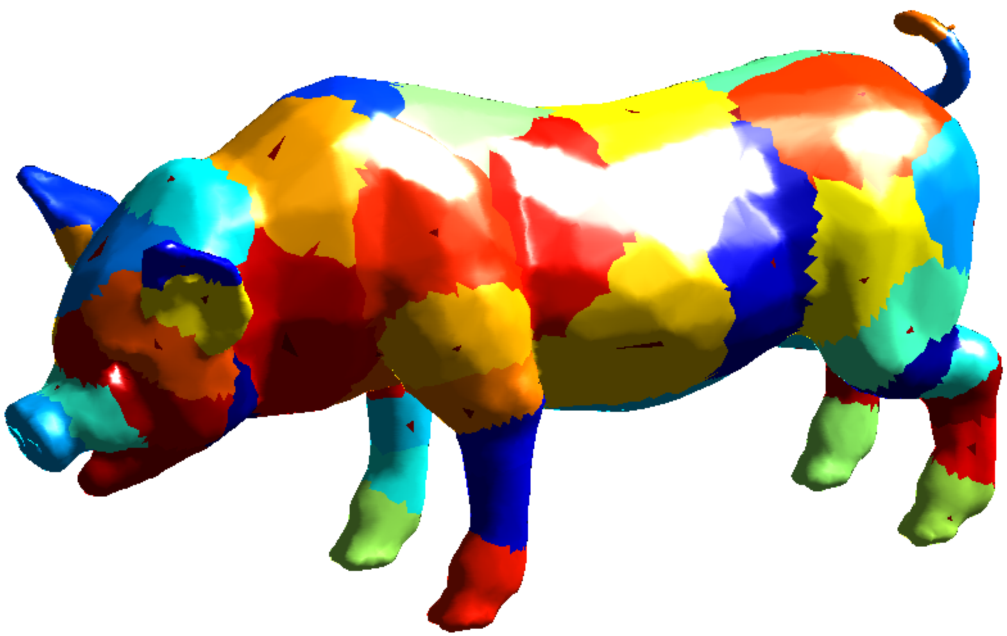
$$d_1(f_i, f_{i,k}) = \eta\left[1 - \cos\left(\mathrm{dihedral}(f_i, f_{i,k})\right)\right] = \frac{\eta}{2}\,||\mathbf{N}_i - \mathbf{N}_{i,k}||^2$$

$$d(f_i, f_{i,k}) = \frac{d_1(f_i, f_{i,k})}{\bar{d}_1}$$

$$p_{i,k} = |e_{i,k}|\exp\left\{-\frac{d(f_i, f_{i,k})}{\sigma}\right\}$$

Oversegmentation

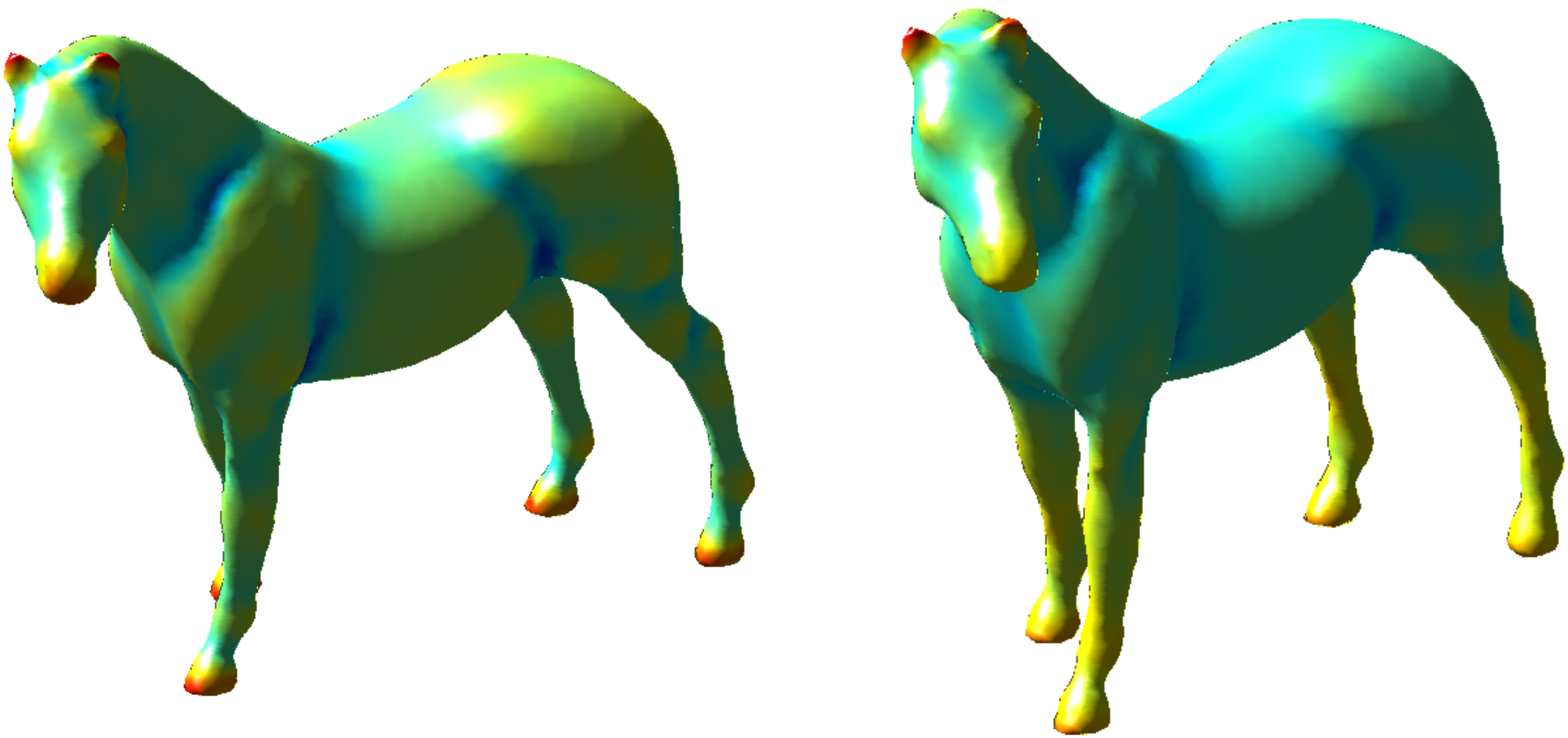
- Use this function for weighted adjacency matrix
- Construct Laplacian from weighted adjacency matrix
- *k*-means on spectral embedding



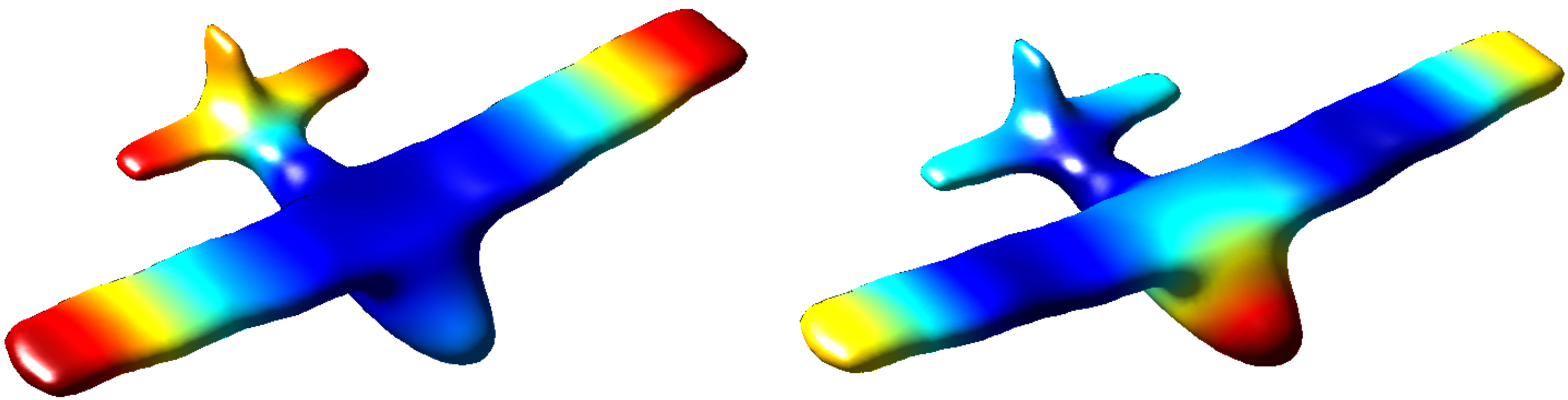
Features

$$a_i \quad = \quad f(W^{sem} F_i + b^{sem})$$

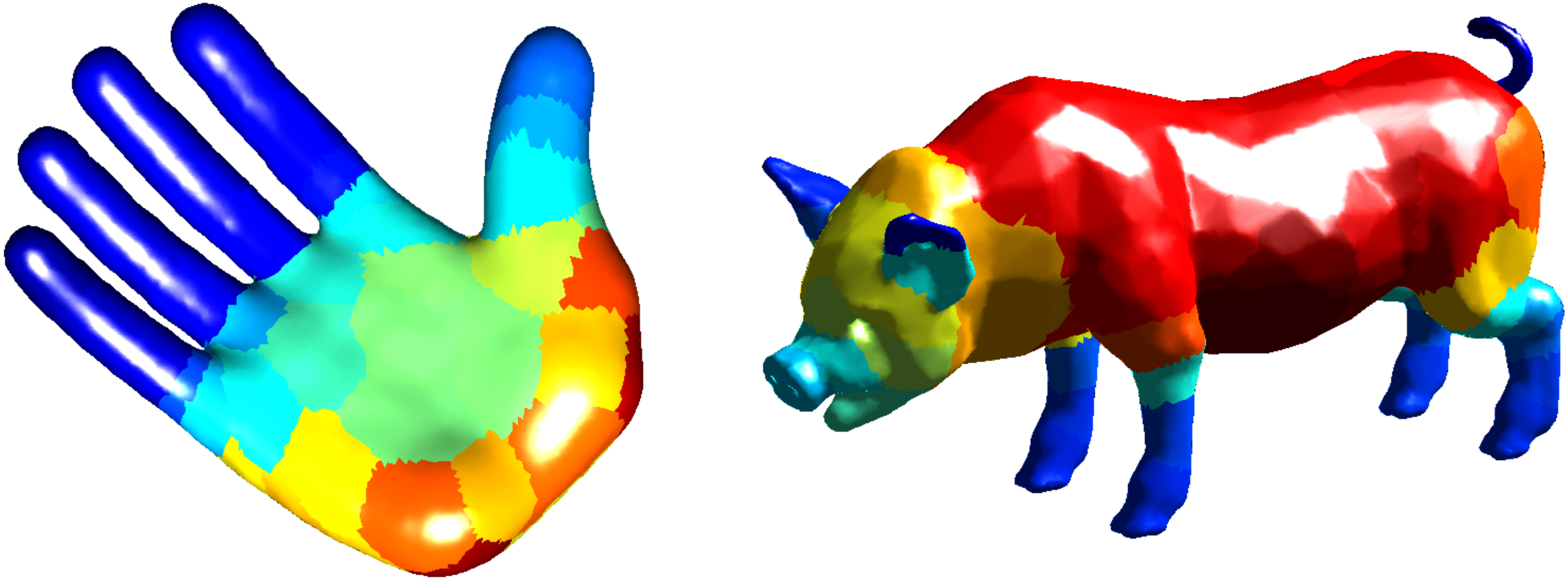
Features: curvatures



Features: HKS, WKS

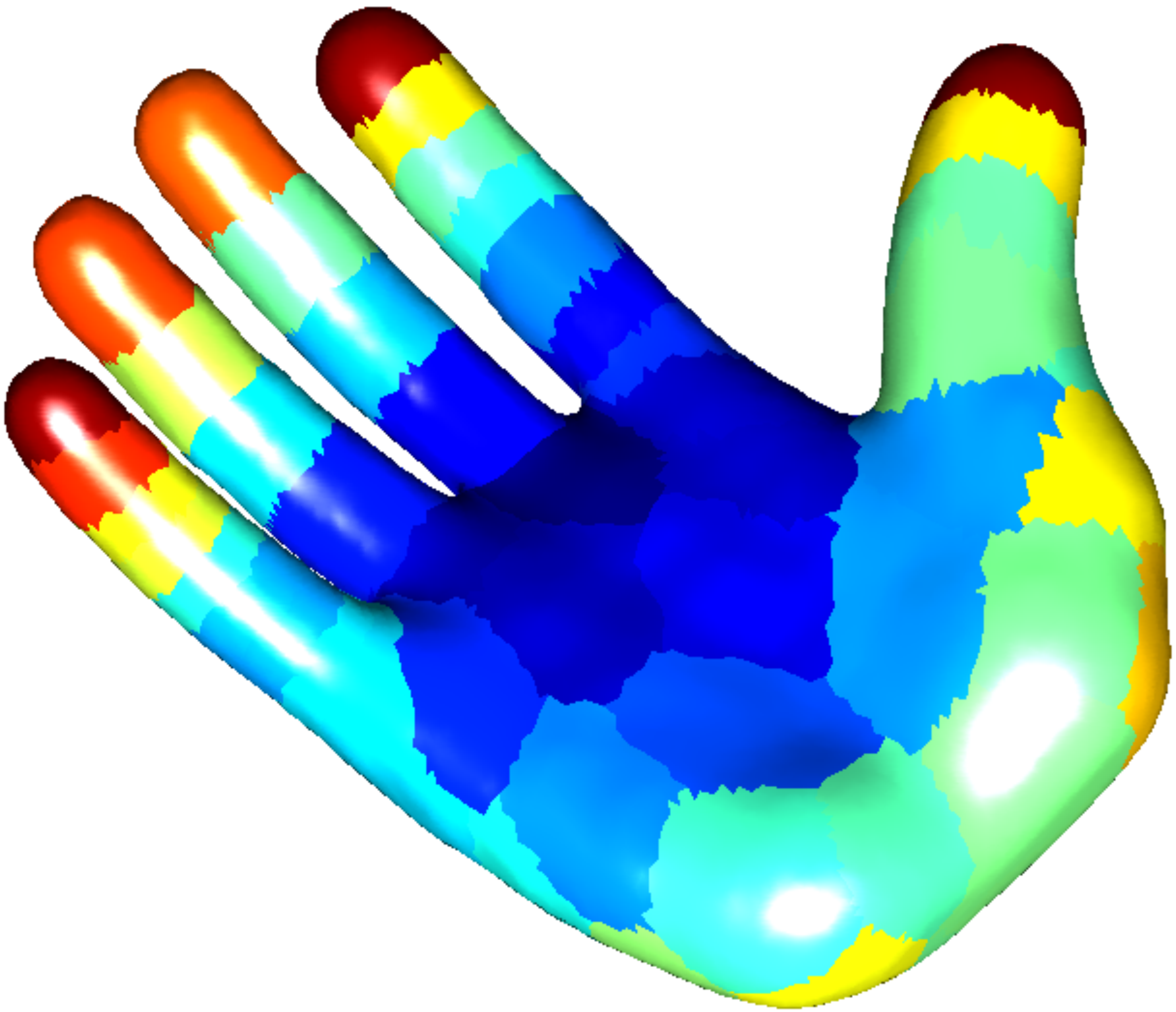


Features: shape diameters



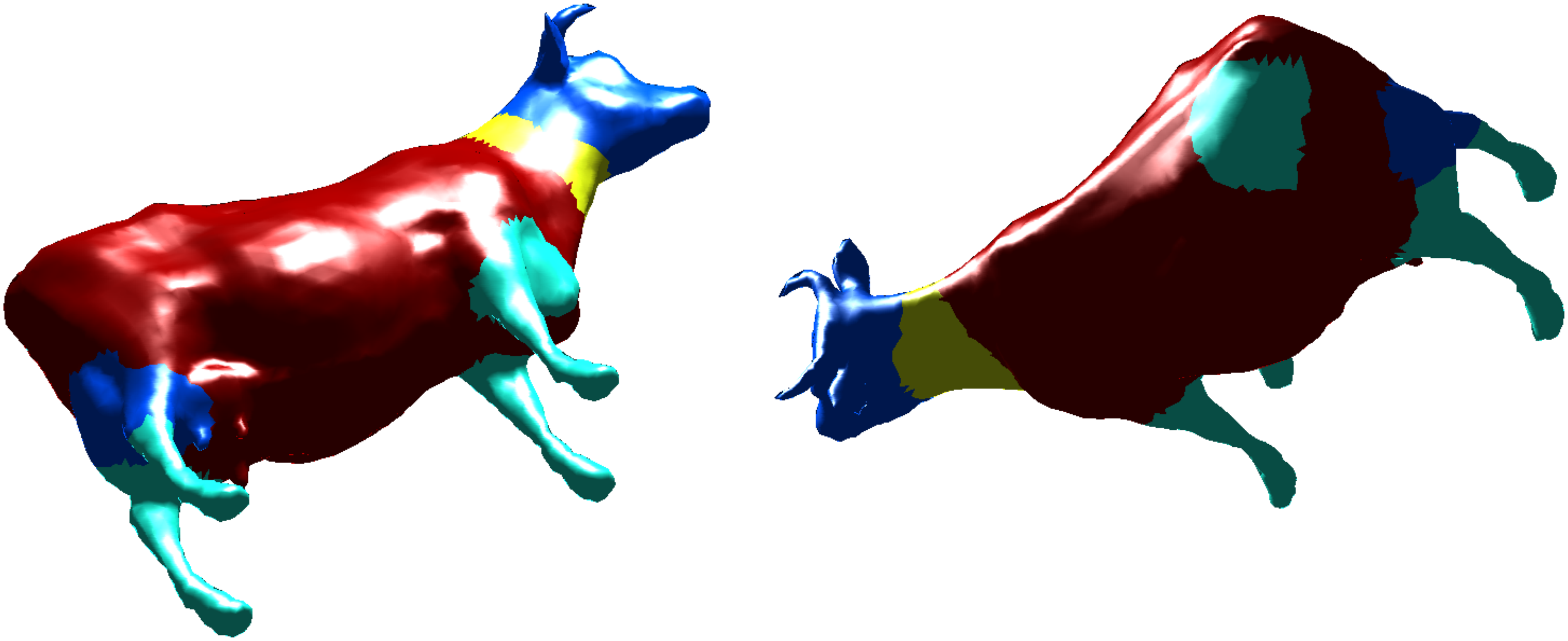
Features: shape contexts

Features: average geodesic distance



(Old) segmentation results

Difficult to encode intrinsic location information

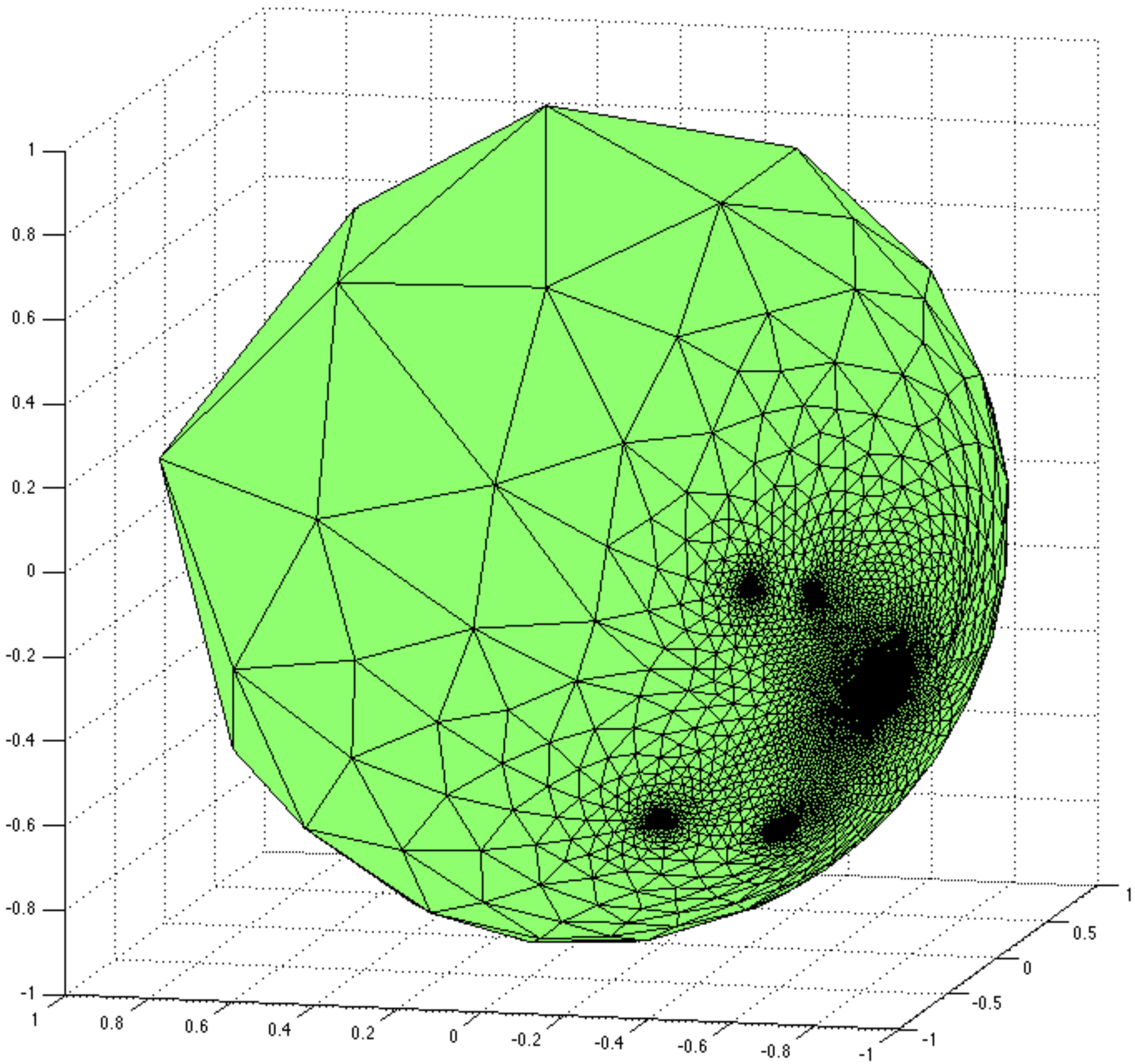


Conformal features

Steps:

1. Initial conformal mapping to sphere (Haker et al. 2000)

Bad area distortion



Conformal features

Steps:

1. Initial conformal mapping to sphere (Haker et al. 2000)
2. Minimize area distortion, as measured by

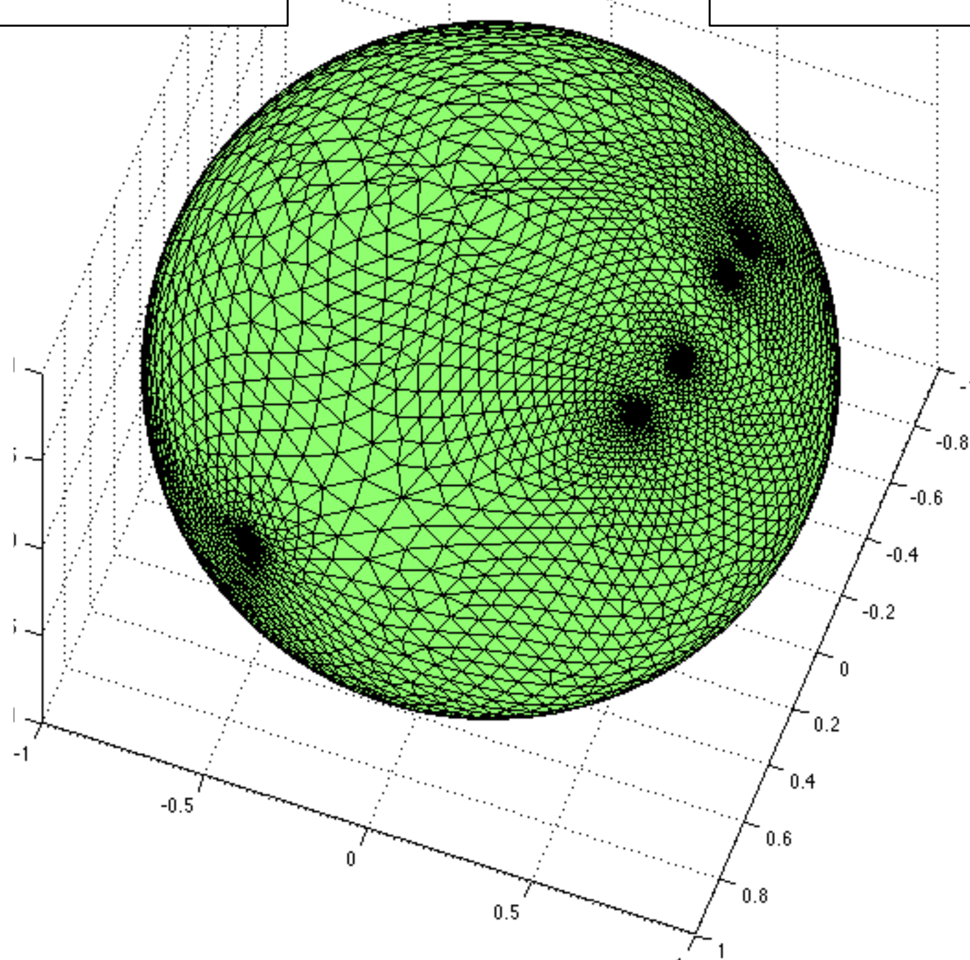
$$\boxed{E_1 = \max_i |A_i - A'_i|}$$

$$\boxed{E_2 = \max_i \left| 1 - \frac{A'_i}{A_i} \right|}$$

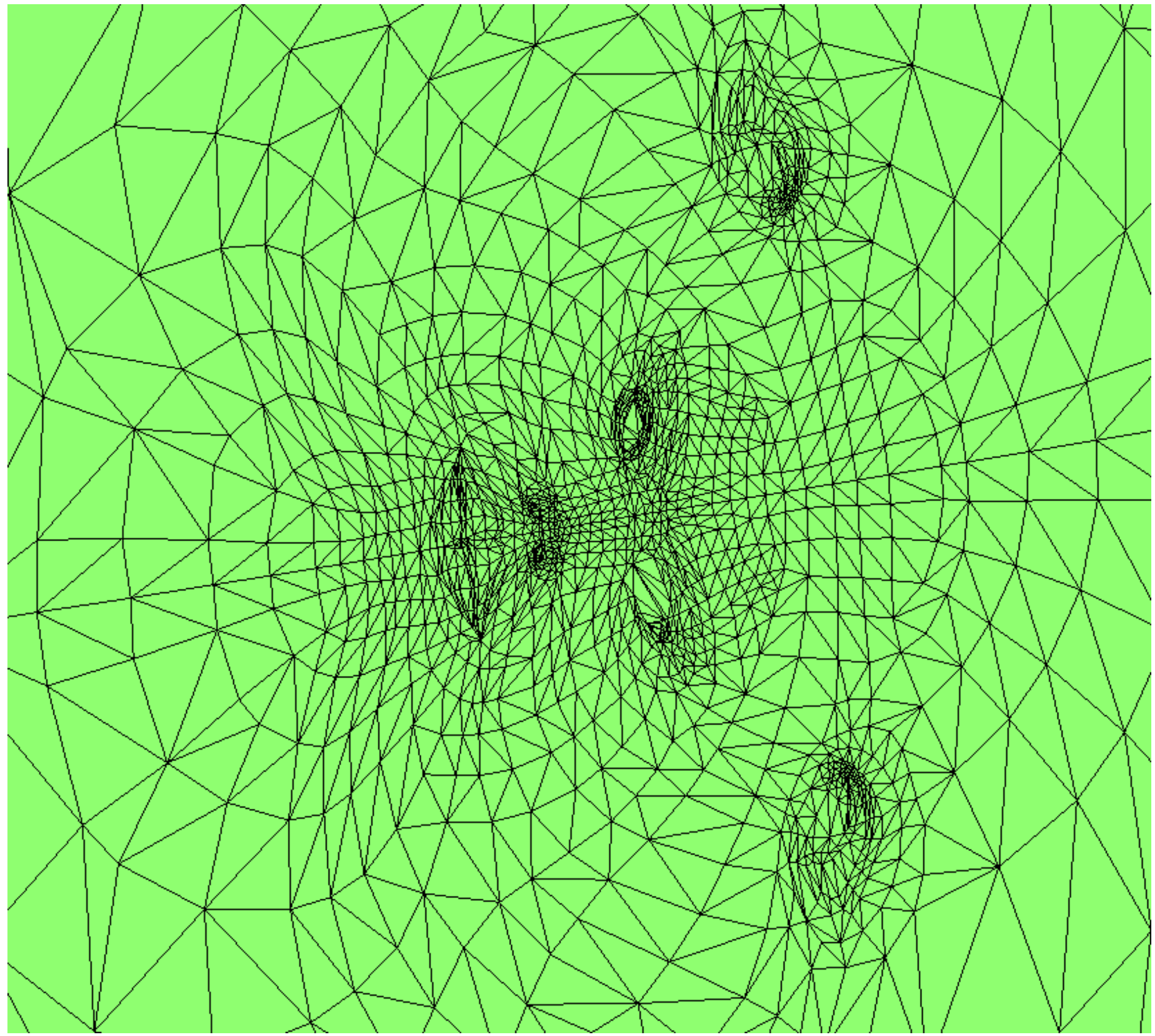
Desirable area distortion

$$E_1 = \max_i |A_i - A_i'|$$

$$E_2 = \max_i \left|1 - \frac{A_i'}{A_i}\right|$$



Desirable area distortion



Conformal features

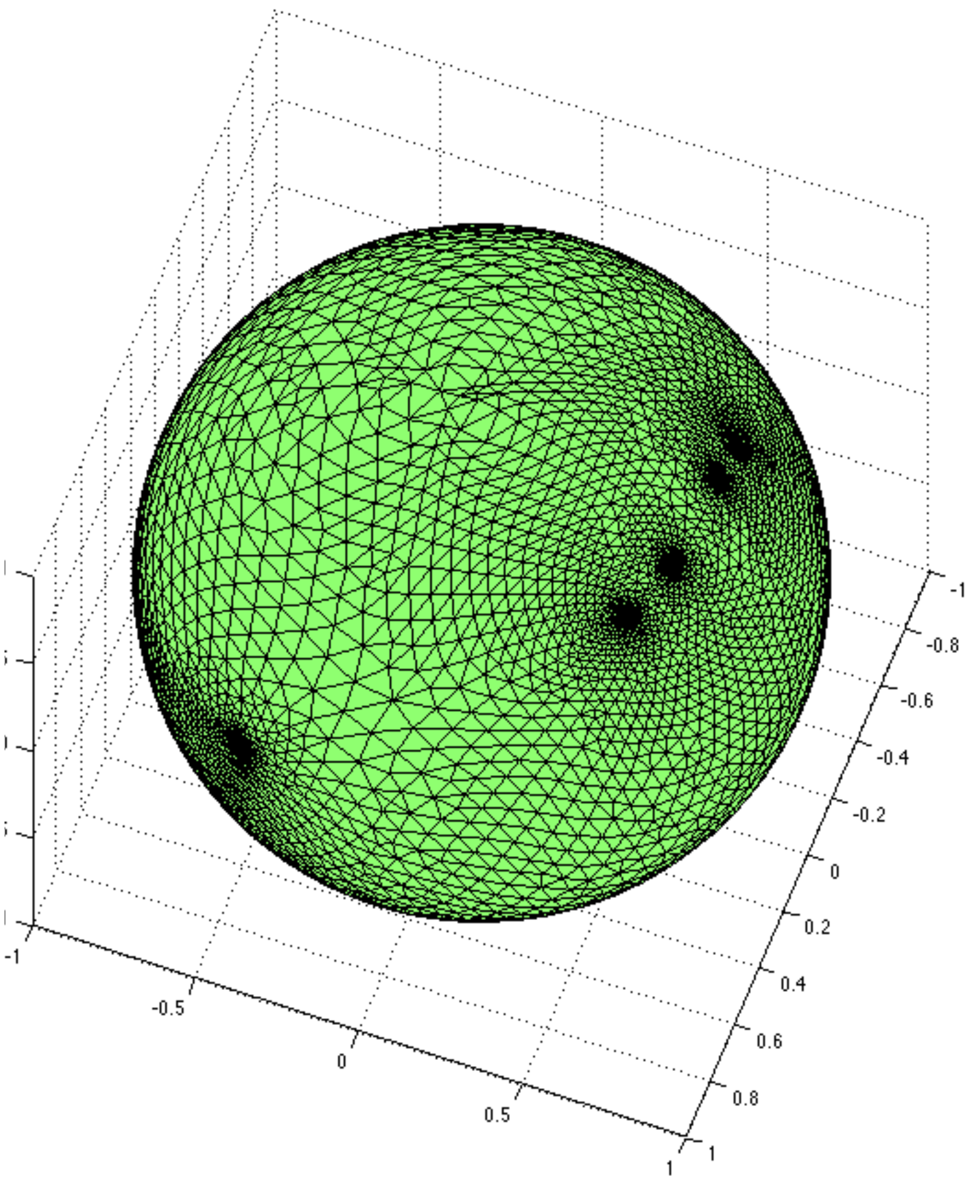
Steps:

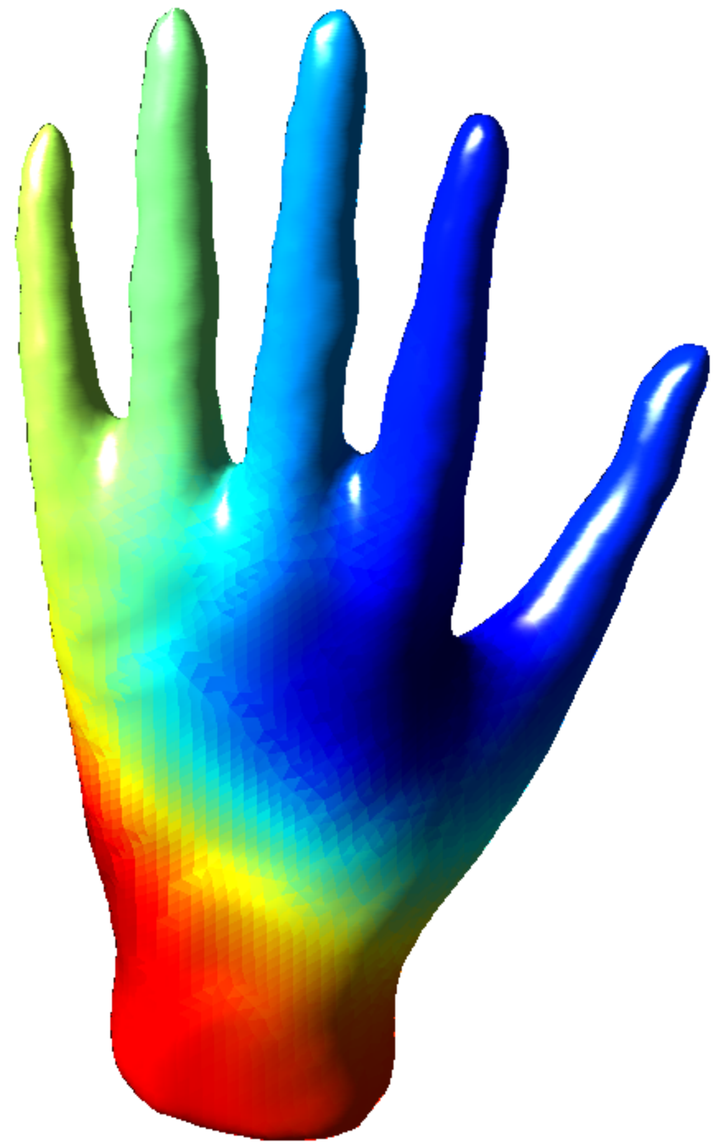
1. Initial conformal mapping to sphere (Haker et al. 2000)
2. Minimize area distortion, as measured by

$$\boxed{E_1 = \max_i |A_i - A_i'|} \quad \boxed{E_2 = \max_i \left|1 - \frac{A_i'}{A_i}\right|}$$

3. Extract and average features
   - Distance to *k*-th nearest neighbor
   - Mean geodesic distance to *p*% closest vertices
   - Mean square distance to *p*% closest vertices
   - Mean log(distance^{-1}) to *p*% closest vertices

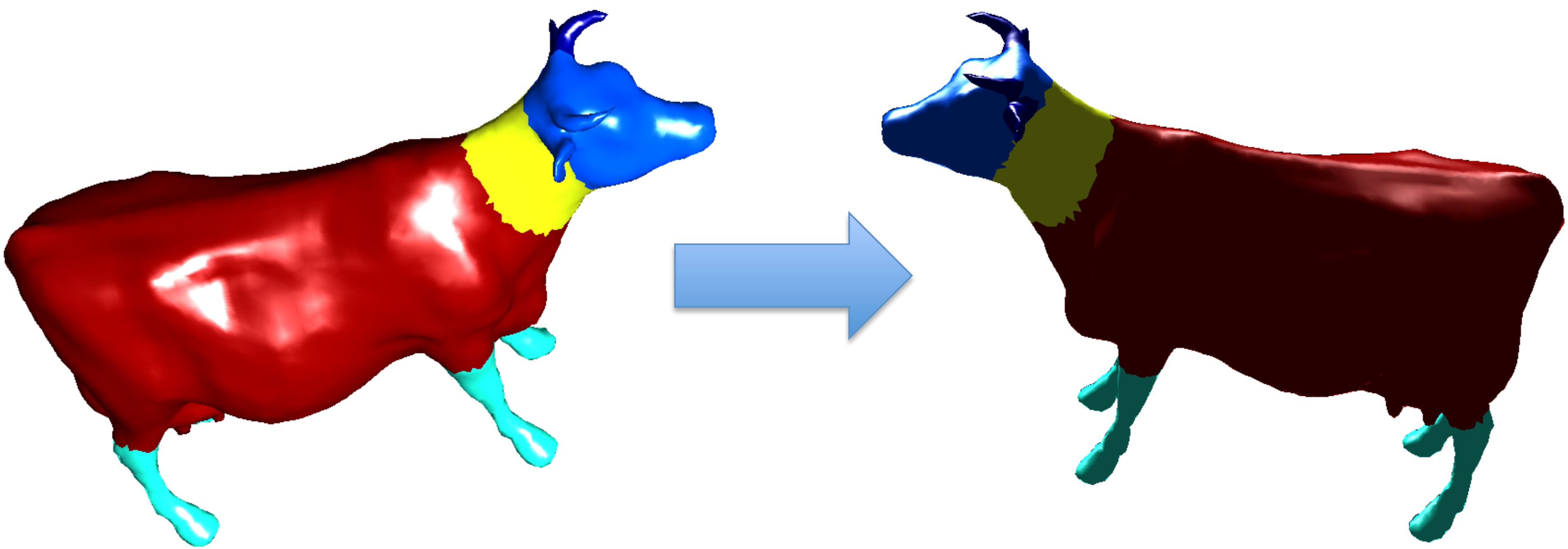
Conformal features



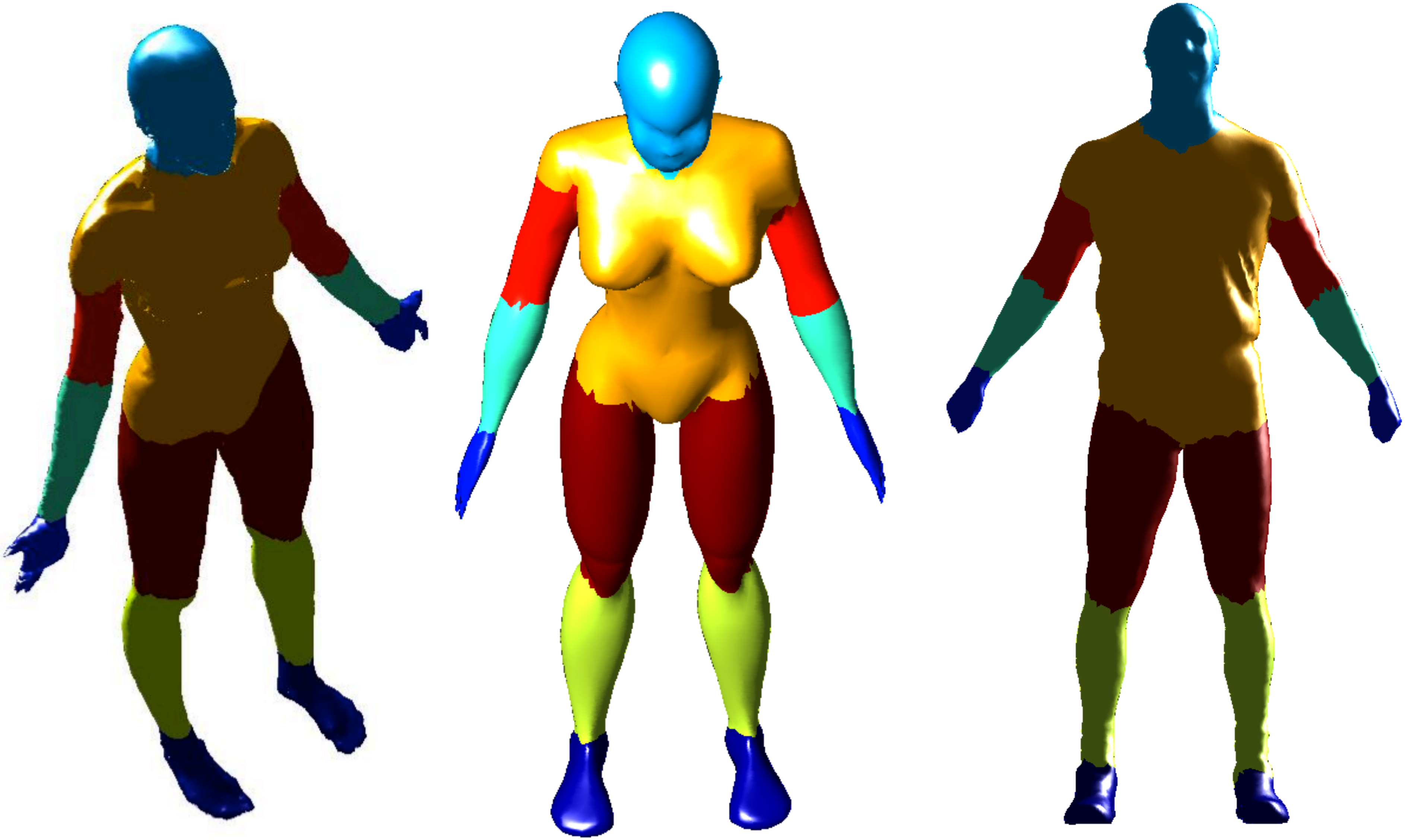


Features: contextual label features

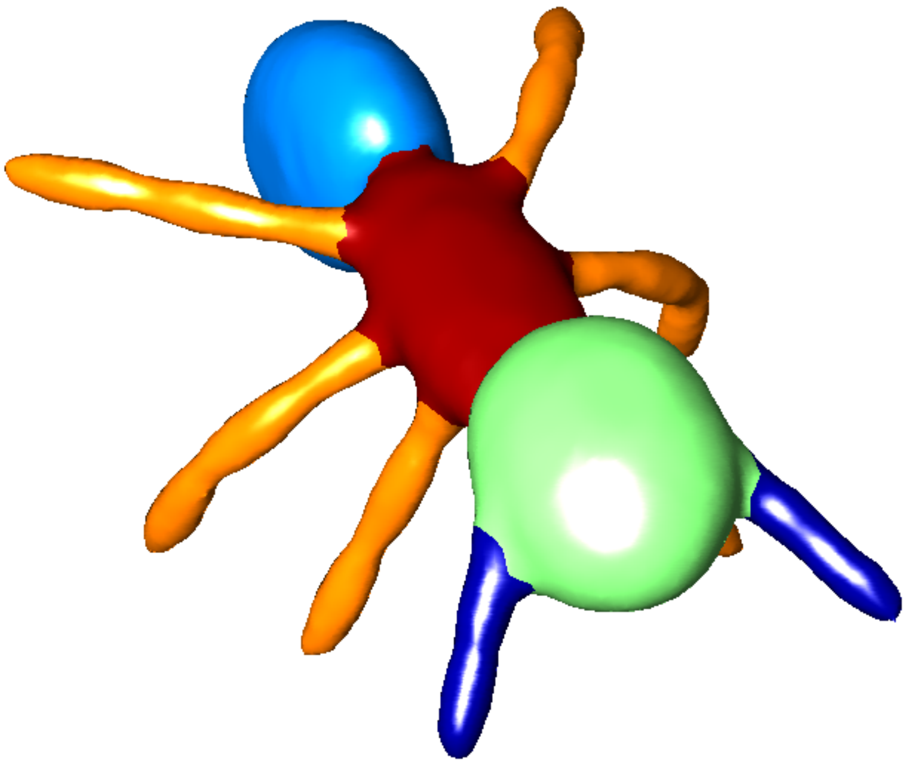
$$p_i^l = \sum_{j:\, d_b \leq \mathrm{dist}(i,j) < d_{b+1}} a_j \cdot P(c_j = l)$$



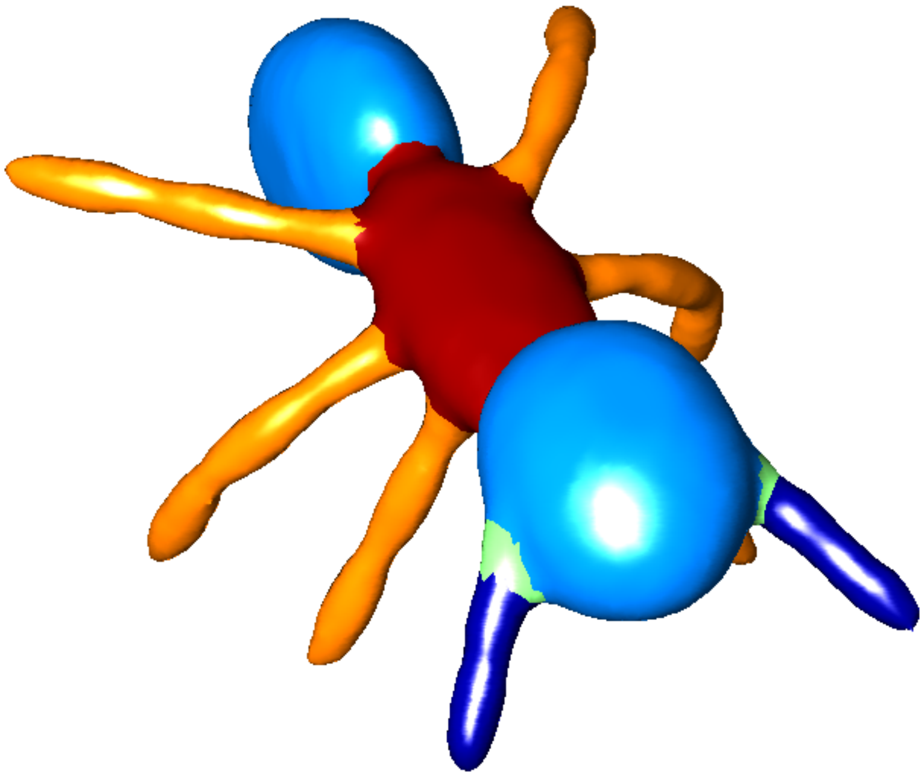
Segmentation results



Segmentation results

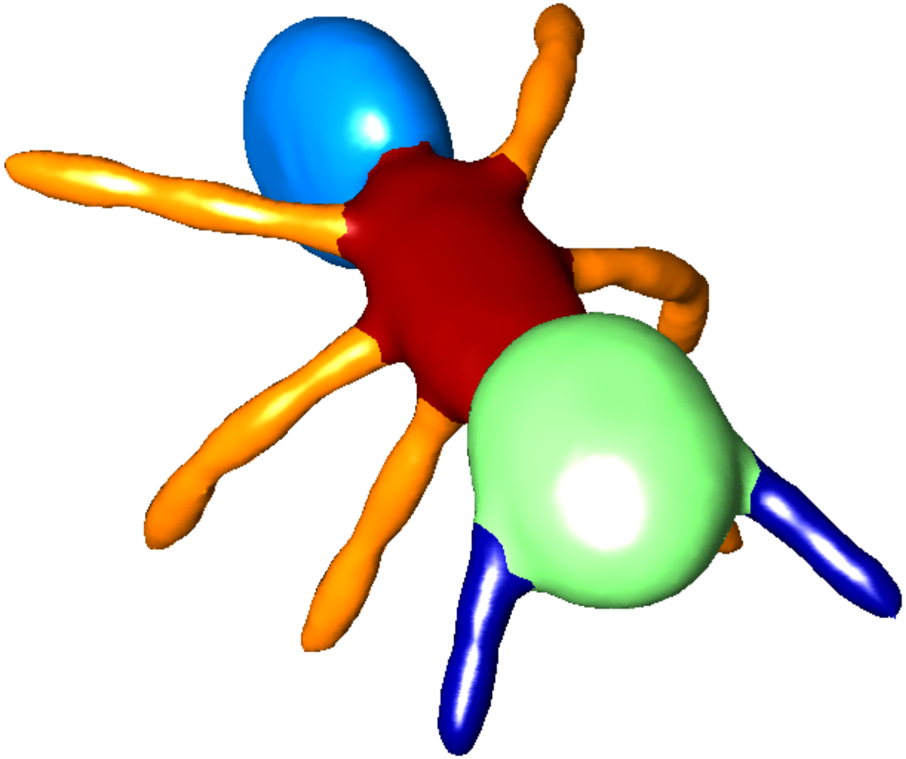


Ground truth

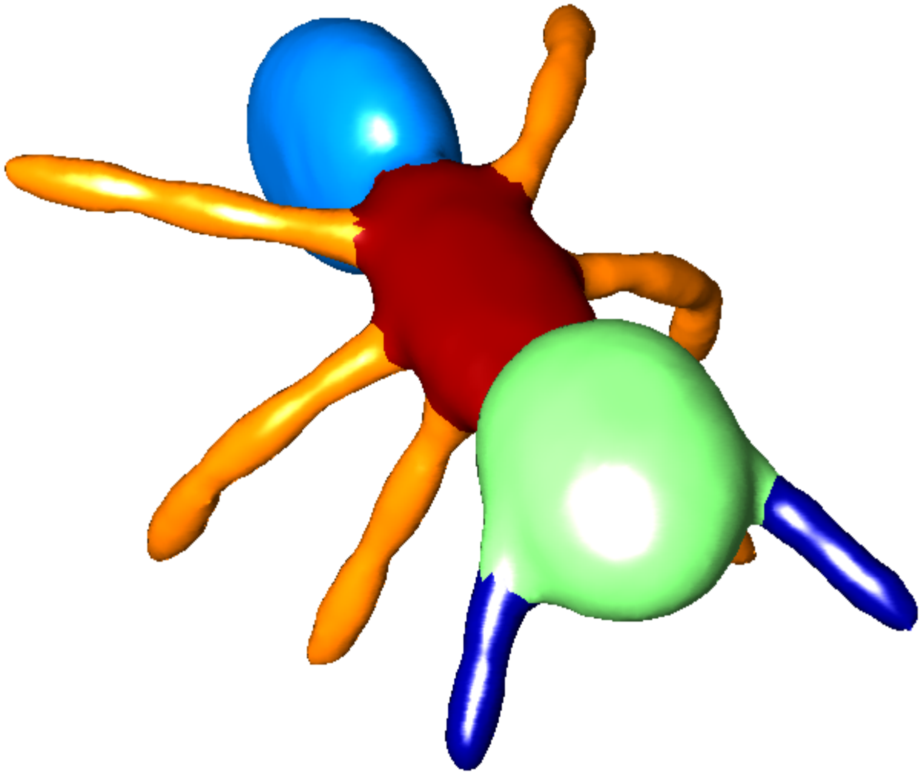


Segmentation without conformal features

Segmentation results

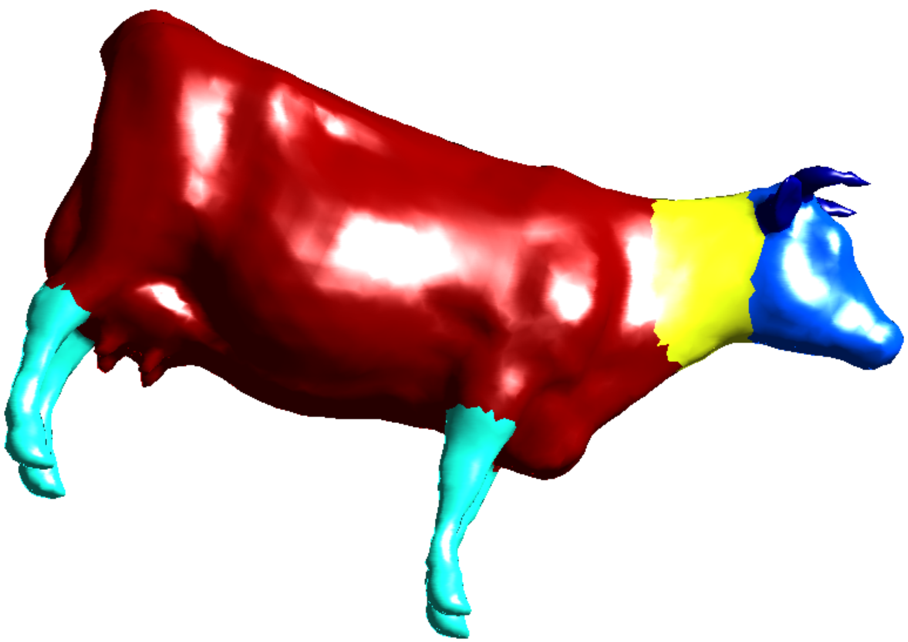


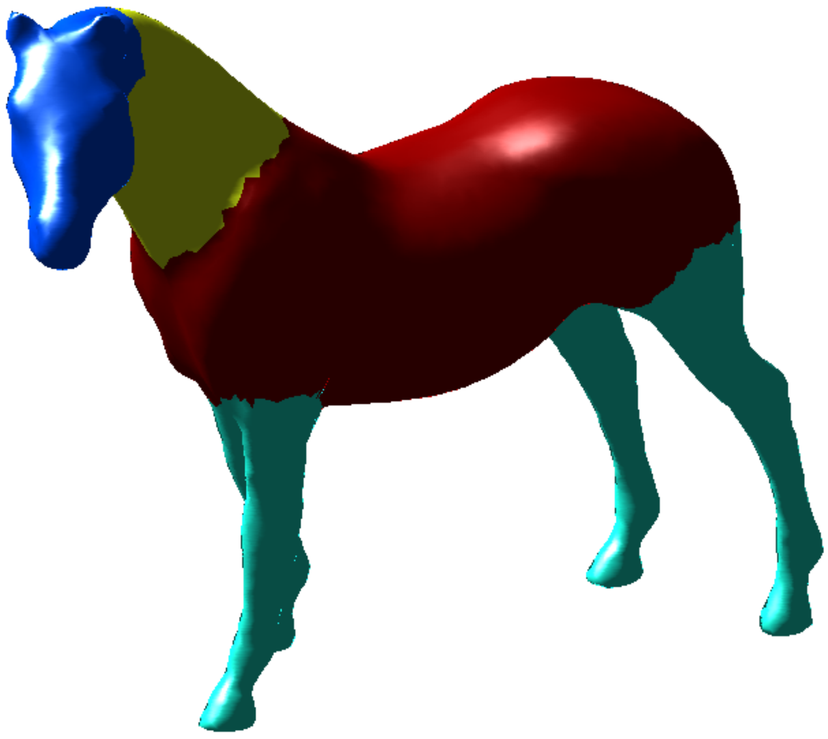
Ground truth



Segmentation with conformal features

Segmentation results





Investigating the semantic embedding

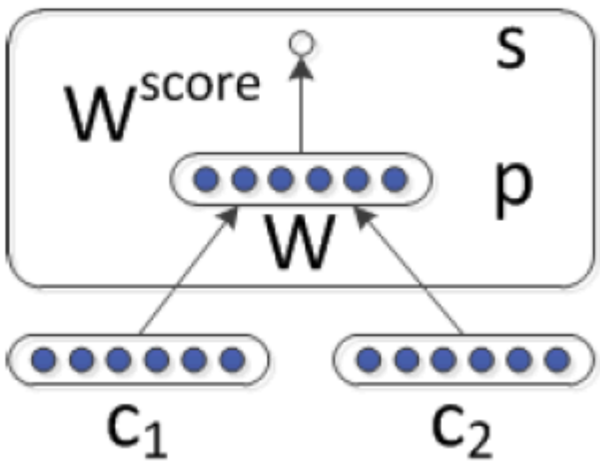
- Trained on SCAPE dataset (72 different poses of same human)
  - Used point-to-point correspondences for defining training segmentation
- Then recovered semantic embedding of feature vectors for superpixels

$$a_i \quad = \quad f(W^{sem} F_i + b^{sem})$$

Ground truth

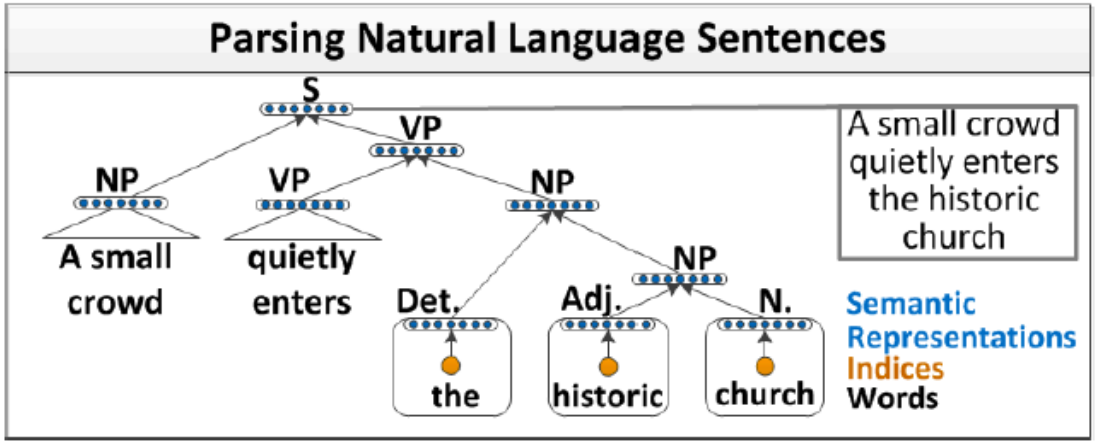
Segmentation result (does not use point correspondences)

How to join superpixels?



$$p = f(W[c_1; c_2] + b)$$

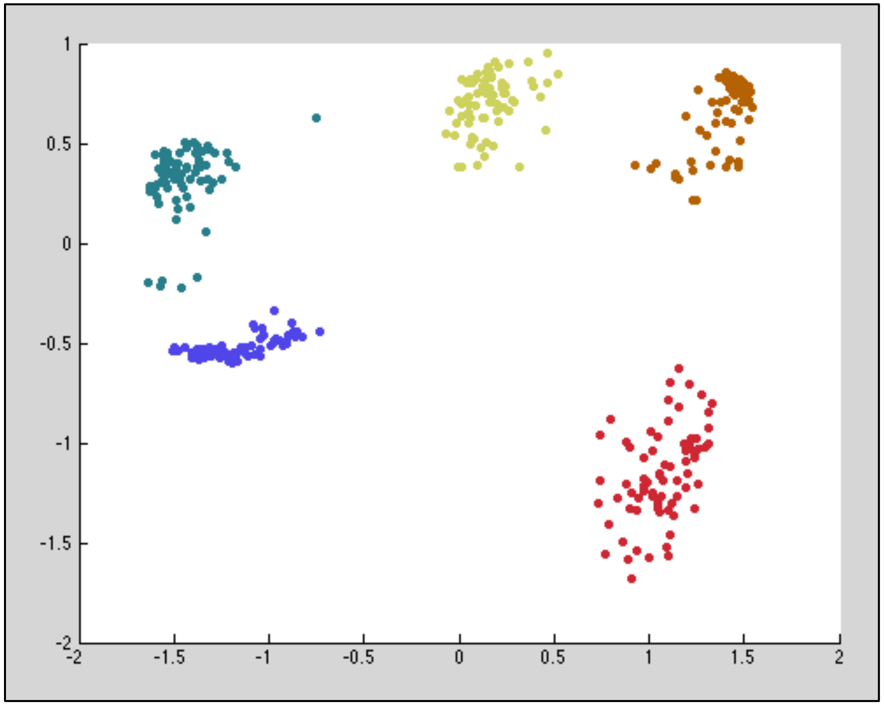
- Socher uses greedy approach to build trees
- Get degenerate tree structures which are undesirable for meshes

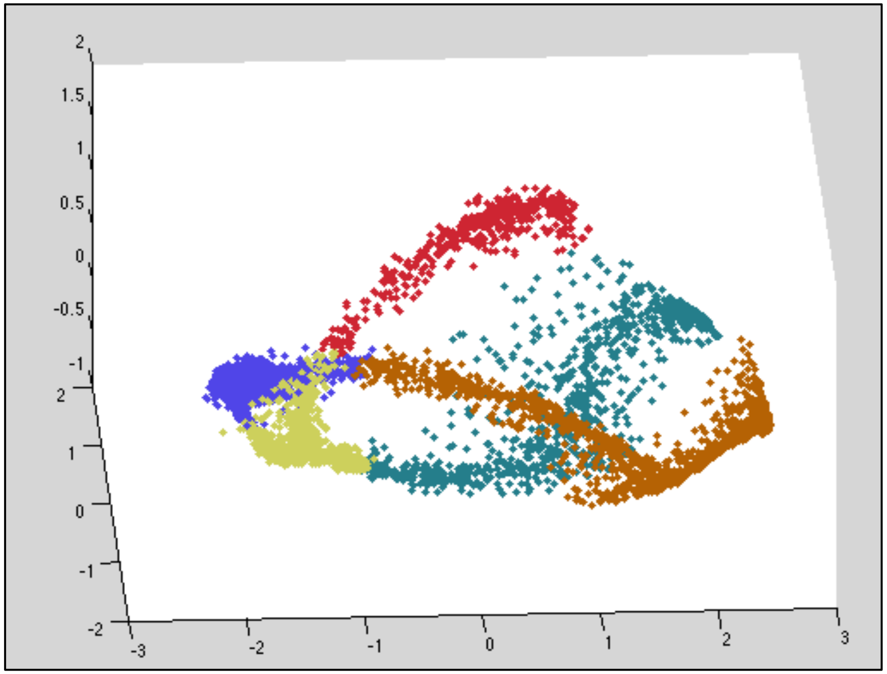


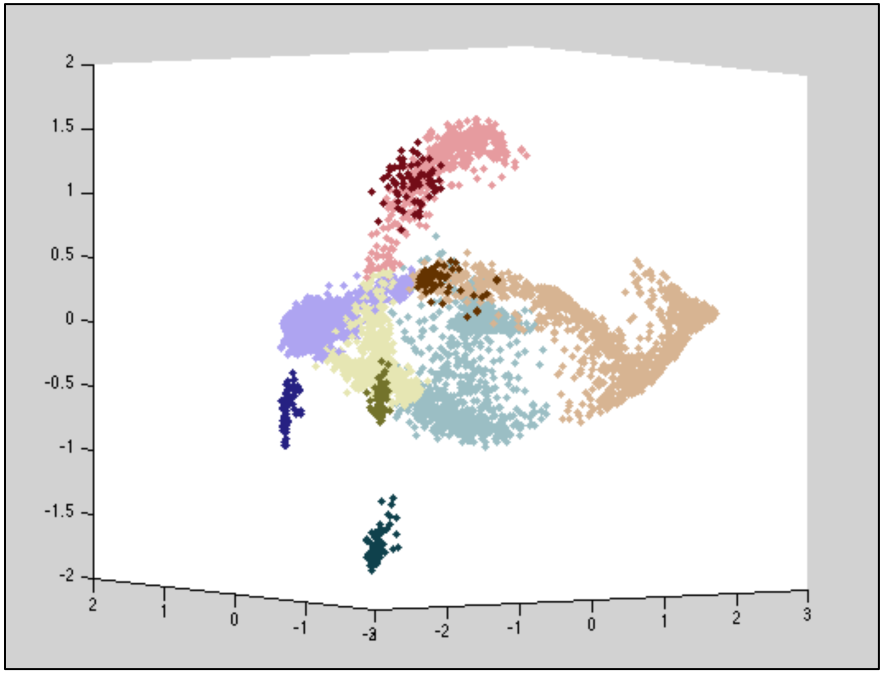
Semantic space PCA plots

Bottom left: semantic features at superpixel level
Top right: semantic features at segment level
Bottom right: both in same plot

(Red = head, burnt orange = legs, purple = waist, teal-green = arms, yellow-green = torso)





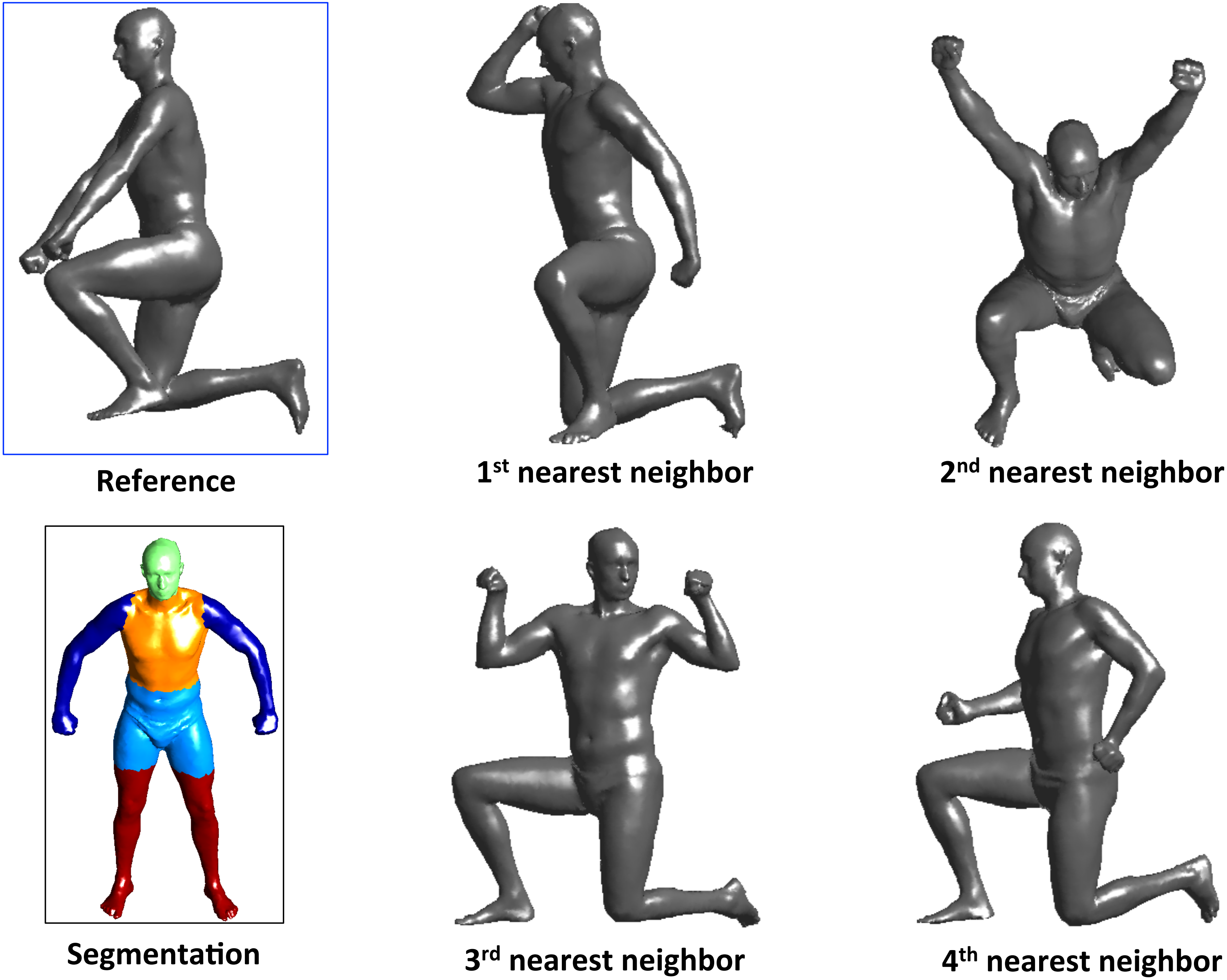


Reference

Segmentation

Shape nearest neighbors: LEGS

- Reference pose shown at left
- We take the nearest/farthest neighbors of the semantic vector for reference legs



Reference

1^{st} nearest neighbor

2^{nd} nearest neighbor

Segmentation

3^{rd} nearest neighbor

4^{th} nearest neighbor

Reference

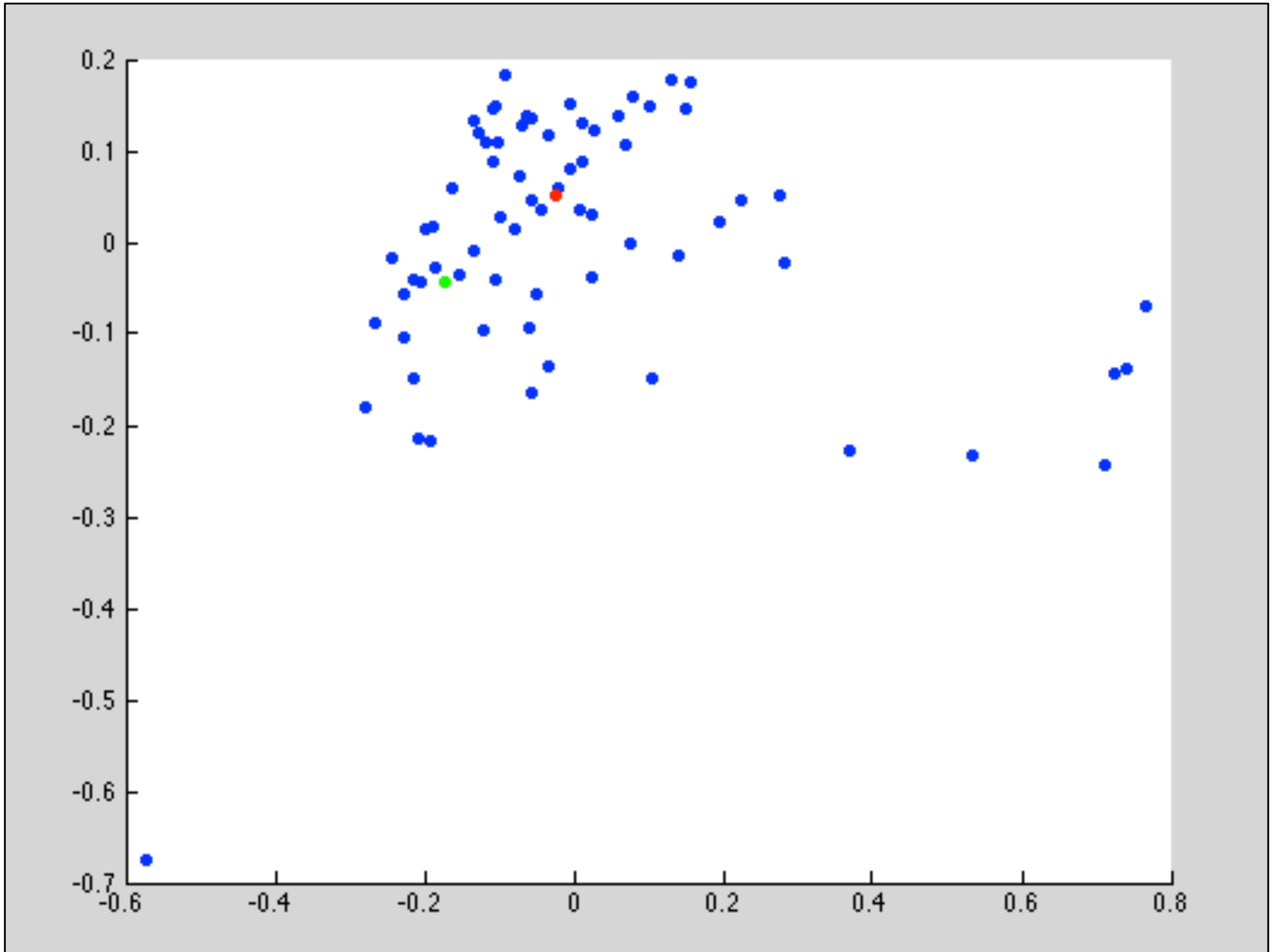
1^{st} farthest neighbor

Segmentation

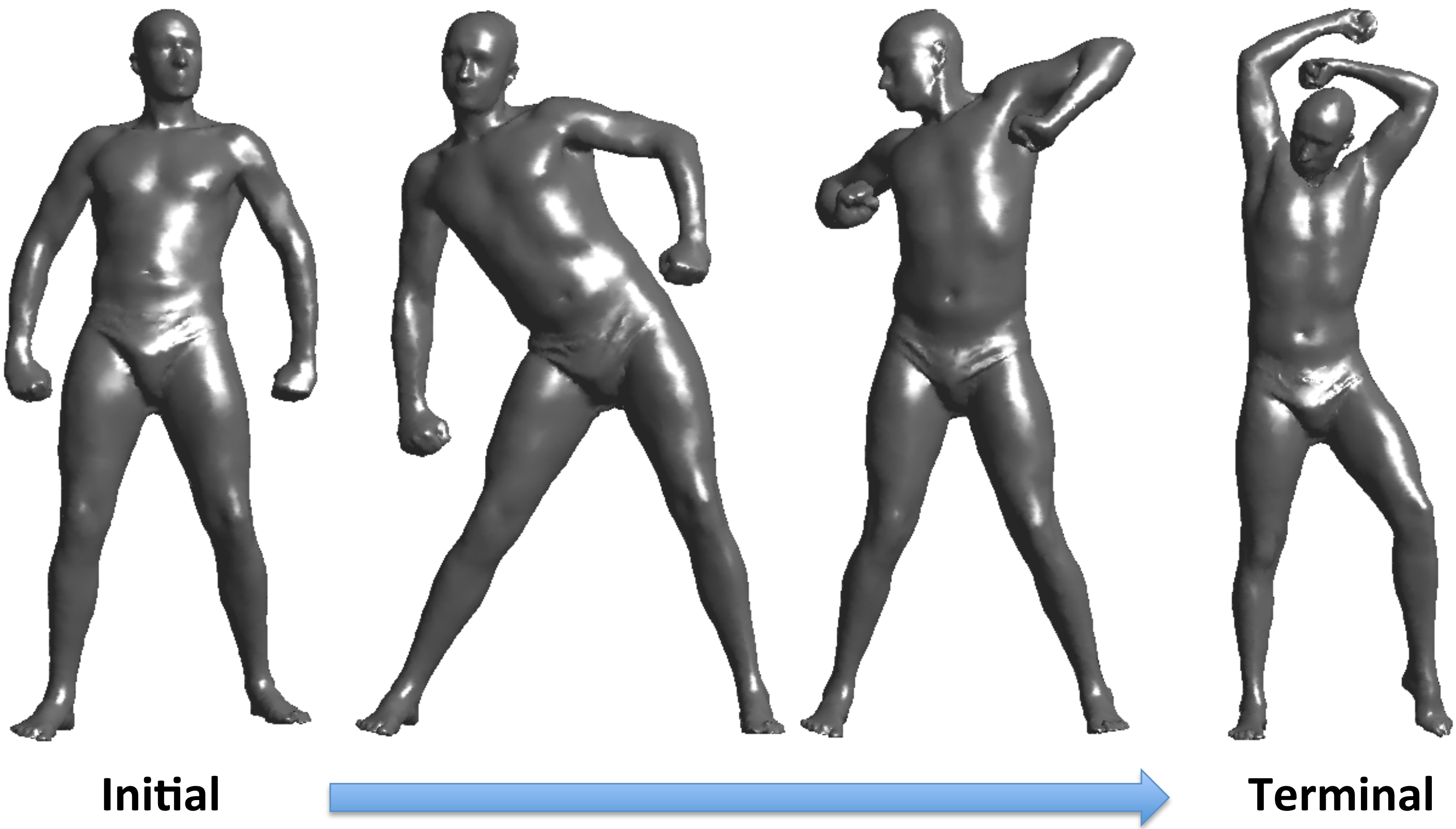
2^{nd} farthest neighbor

3^{rd} farthest neighbor

Shape interpolation: ARMS



Shape interpolation



Further work/future directions

- Tuning the model, removing redundant or useless features for increased speed
- Investigating alterations to learning model/objective function
  - Dealing with combinatorial explosion to learn correct tree structure
- Adding boundary features

Conclusions

- Segmentation
  - Shape correspondences between non-(nearly)-isometric meshes (e.g., horse and cow)
- Interesting shape descriptor from semantic embedding
  - Shape understanding

Objective function:

$$\begin{aligned} J(\theta) &= \frac{1}{N}\sum_{i=1}^{N} r_i(\theta) + \frac{\lambda}{2}||\theta||^2, \quad \text{where} \\ r_i(\theta) &= \max_{\hat{y}\in\mathcal{T}(x_i)} \left(s(\mathrm{RNN}(\theta, x_i, \hat{y})) + \Delta(x_i, l_i, \hat{y}) \right) \\ &- \max_{y_i\in Y(x_i,l_i)} \left(s(\mathrm{RNN}(\theta, x_i, y_i)) \right) \end{aligned}$$

Margin loss:

$$\Delta(x, l, \hat{y}) = \kappa \sum_{d\in N(\hat{y})} \mathbf{1}\{subTree(d) \notin Y(x, l)\}$$