CS 523: Computer Graphics, Spring 2009

Interactive Shape Modeling

Space deformations

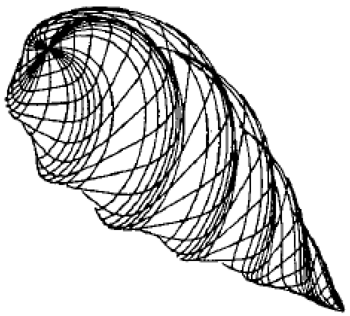
Space Deformation

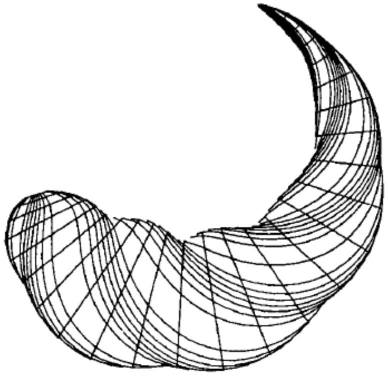
- Displacement function defined on the ambient space

$$\mathbf{d} : \mathrm{R}^3 \rightarrow \mathrm{R}^3$$

- Evaluate the function on the points of the shape embedded in the space

$$\mathbf{x}' = \mathbf{x} + \mathbf{d}(\mathbf{x})$$





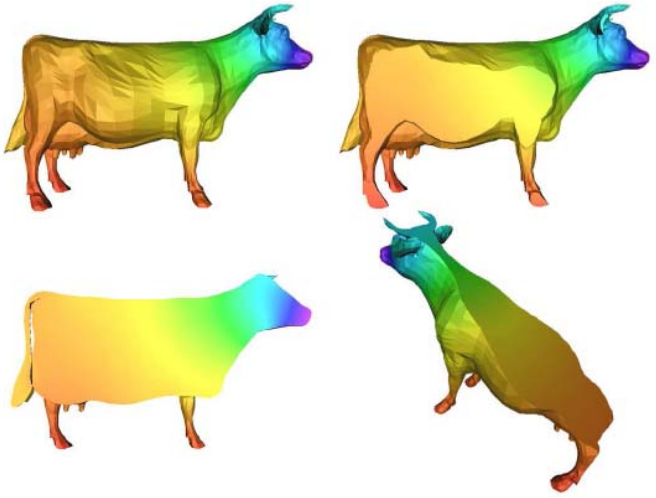
Twist warp
Global and local deformation of solids
[A. Barr, SIGGRAPH 84]

Freeform Deformations

- Control object
- User defines displacements $\mathbf{d}_i$ for each element of the control object
- Displacements are interpolated to the entire space using basis functions $B_i(\mathbf{x}) : \mathrm{R}^3 \to \mathrm{R}$

$$\mathbf{d}(\mathbf{x}) = \sum_{i=1}^{k} \mathbf{d}_i B_i(\mathbf{x})$$

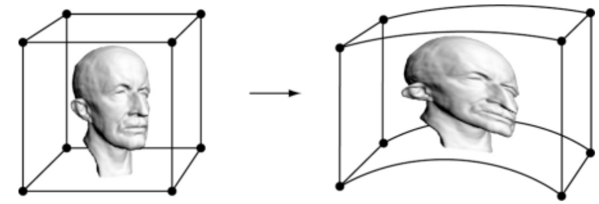
- Basis functions should be smooth for aesthetic results



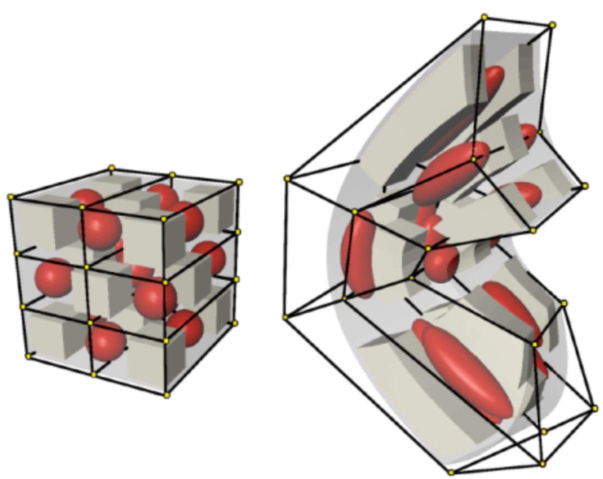
Trivariate Tensor Product Bases

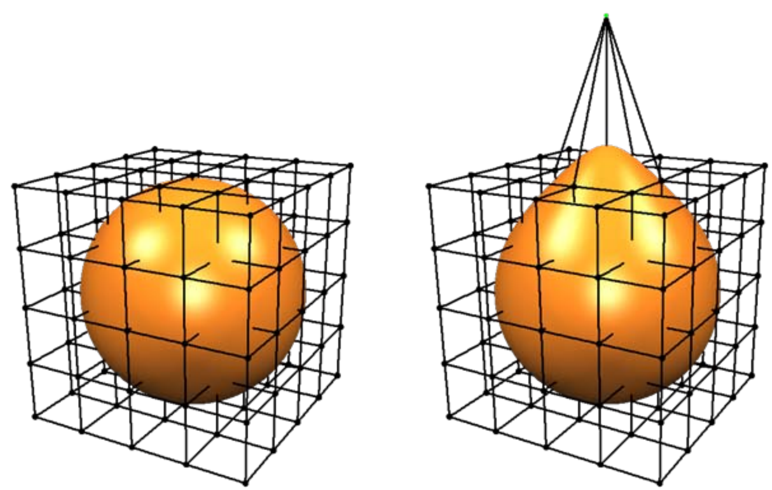
[Sederberg and Parry 86]

- Control object = lattice
- Basis functions $B_i(\mathbf{x})$ are trivariate tensor-product splines:



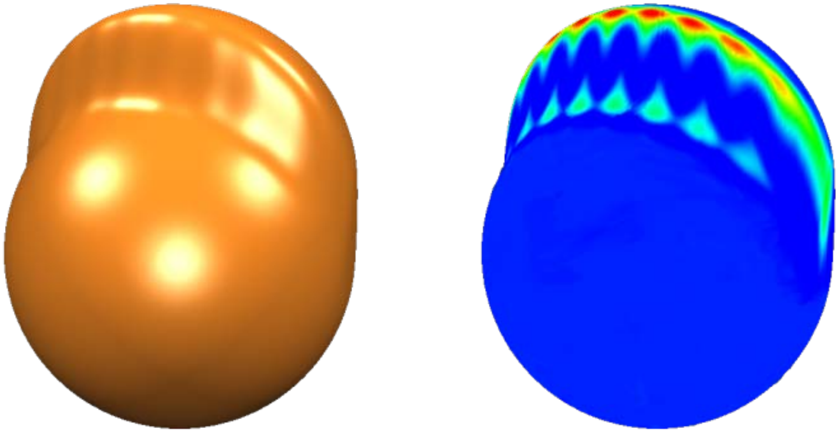
$$\mathbf{d}(x,y,z) = \sum_{i=0}^{l} \sum_{j=0}^{m} \sum_{k=0}^{n} \mathbf{d}_{ijk} \, N_i(x) N_j(y) N_i(z)$$





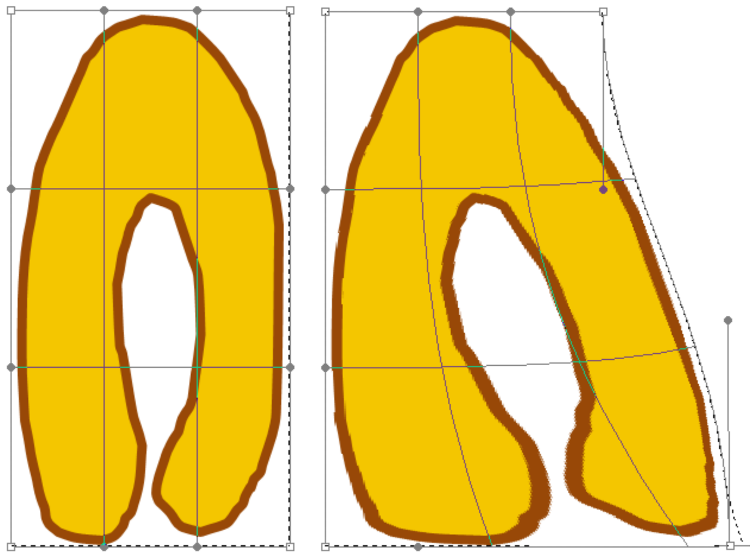
Trivariate Tensor Product Bases

- Similar to the surface case
  - Aliasing artifacts
- Interpolate deformation constraints?
  - Only in least squares sense



Lattice as Control Object

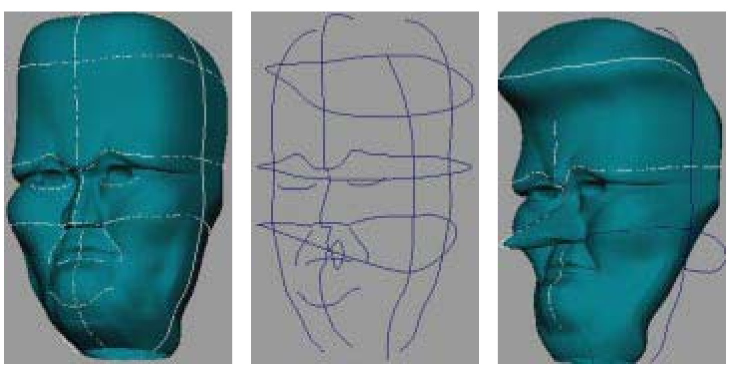
- Difficult to manipulate
- The control object is not related to the shape of the edited object
- Part of the shape in close Euclidean distance always deform similarly, even if geodesically far

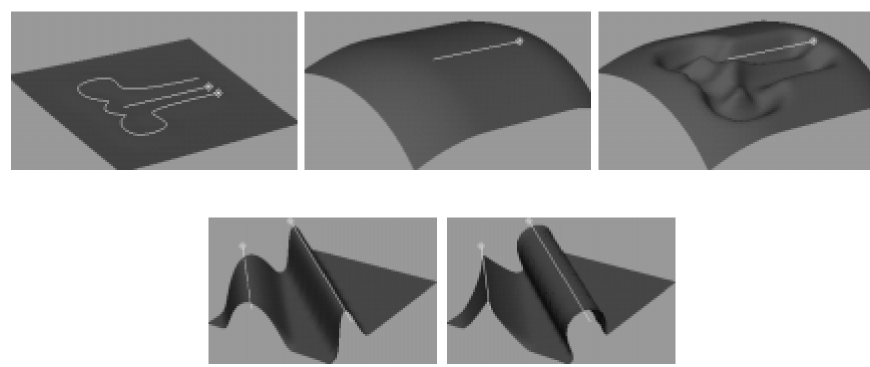


Wires

[Singh and Fiume 98]

- Control objects are arbitrary space curves
- Can place curves along meaningful features of the edited object
- Smooth deformations around the curve with decreasing influence

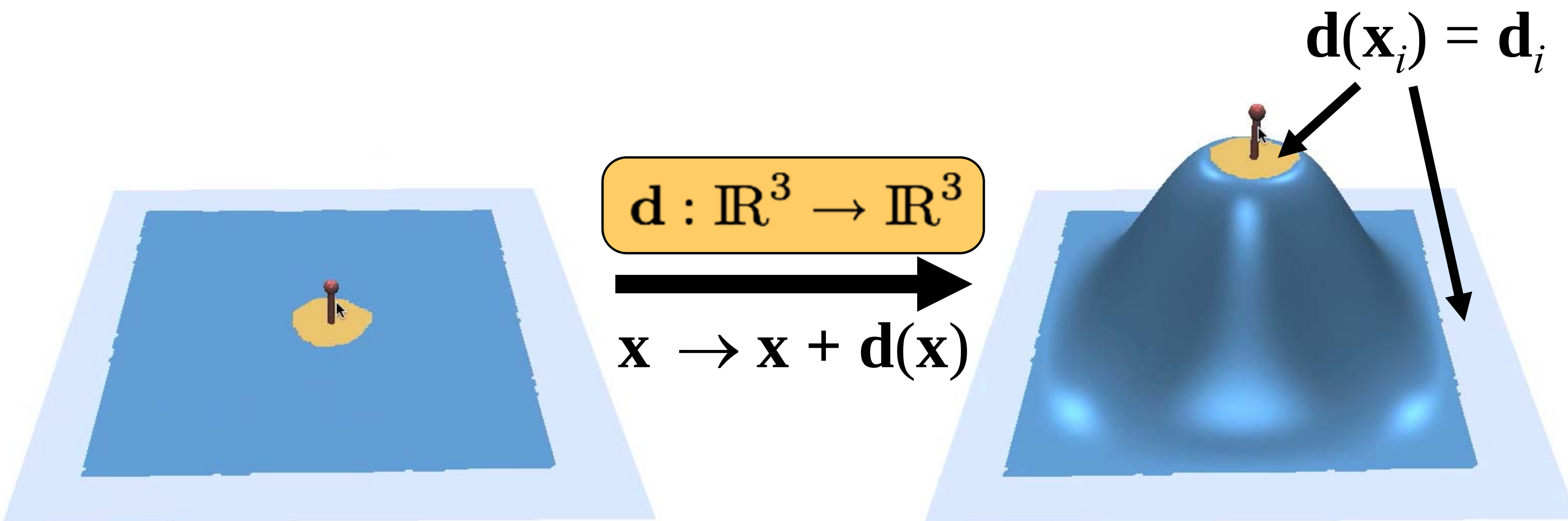




Handle Metaphor

[RBF, Botsch and Kobbelt 05]

- Wish list for the displacement function $\mathbf{d}(\mathbf{x})$:
  - Interpolate prescribed constraints
  - Smooth, intuitive deformation



Volumetric Energy Minimization

[RBF, Botsch and Kobbelt 05]

- Minimize similar energies to surface case

$$\int_{\Re^3} \left\|\mathbf{d}_{xx}\right\|^2 + \left\|\mathbf{d}_{xy}\right\|^2 + \ldots + \left\|\mathbf{d}_{zz}\right\|^2 \mathrm{d}x\mathrm{d}y\mathrm{d}z \;\rightarrow\; \min$$

- But displacements function lives in 3D...
  - Need a volumetric space tessellation?
  - No, same functionality provided by RBFs!

Radial Basis Functions

[RBF, Botsch and Kobbelt 05]

- Represent deformation by RBFs

$$\mathbf{d}(\mathbf{x}) = \sum_j \mathbf{w}_j \cdot \varphi(\|\mathbf{c}_j - \mathbf{x}\|) + \mathbf{p}(\mathbf{x})$$

- Triharmonic basis function $\varphi(r) = r^3$
  - C^2 boundary constraints
  - Highly smooth / fair interpolation

$$\int_{\Re^3} \left\|\mathbf{d}_{xxx}\right\|^2 + \left\|\mathbf{d}_{xyy}\right\|^2 + \ldots + \left\|\mathbf{d}_{zzz}\right\|^2 \mathrm{d}x\mathrm{d}y\mathrm{d}z \;\rightarrow\; \min$$

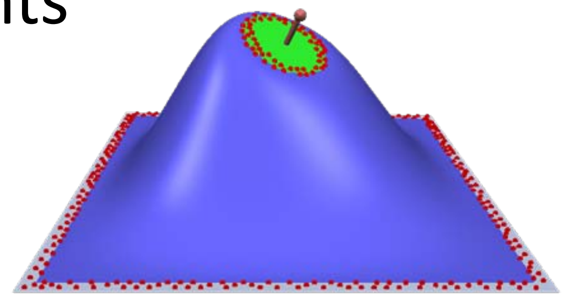
RBF Fitting

[RBF, Botsch and Kobbelt 05]

- Represent deformation by RBFs

$$\mathbf{d}(\mathbf{x}) = \sum_j \mathbf{w}_j \cdot \varphi(\| \mathbf{c}_j - \mathbf{x} \|) + \mathbf{p}(\mathbf{x})$$

- RBF fitting
  - Interpolate displacement constraints
  - Solve linear system for $\mathbf{w}_j$ and $\mathbf{p}$



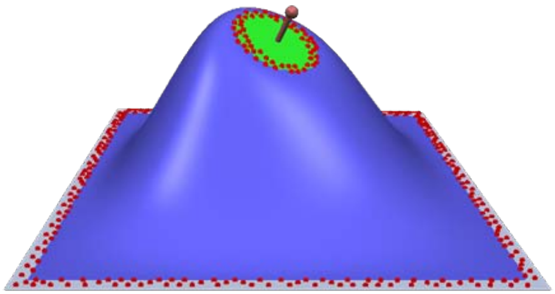
RBF Fitting

[RBF, Botsch and Kobbelt 05]

- Represent deformation by RBFs

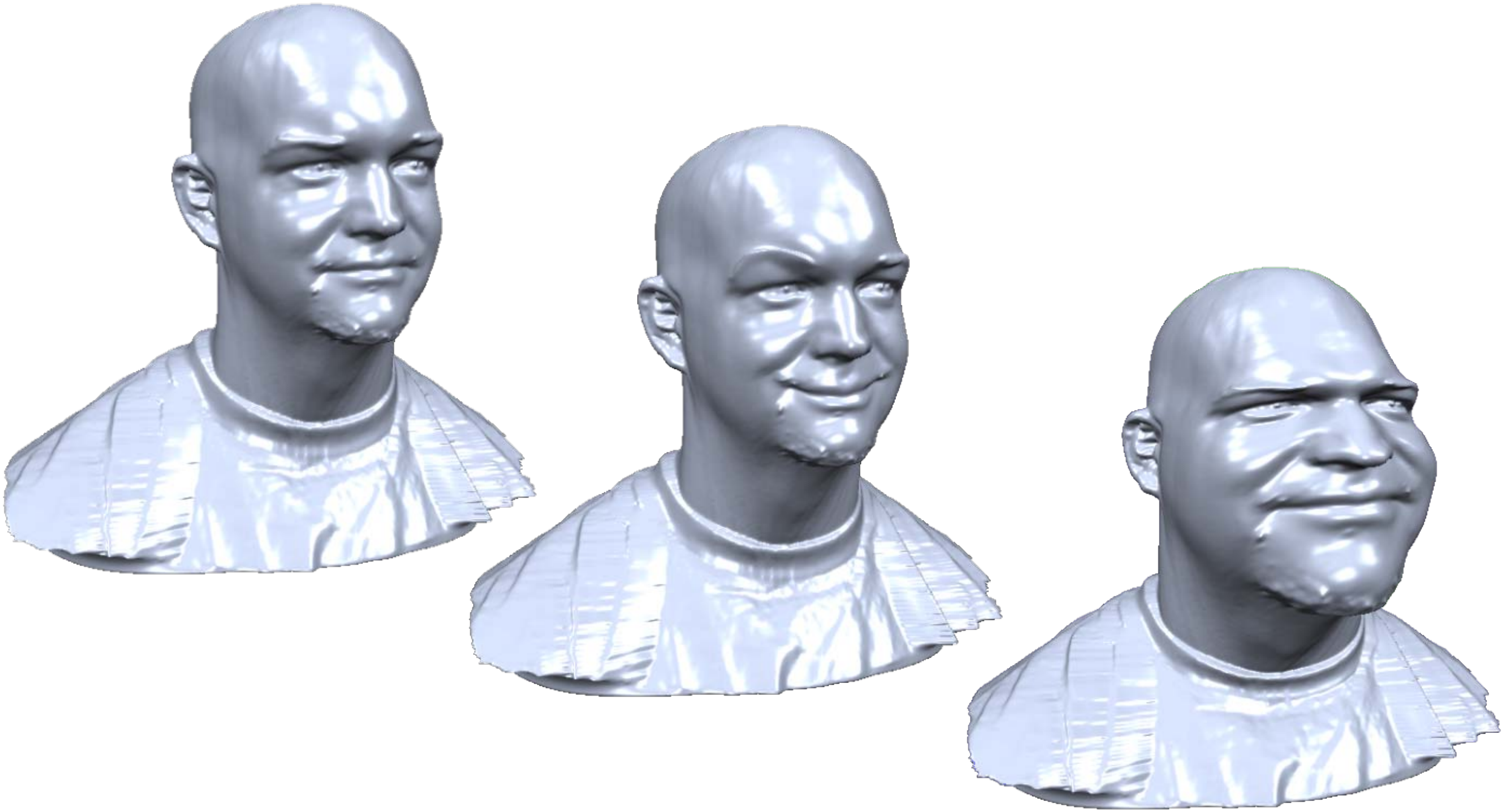
$$\mathbf{d}(\mathbf{x}) = \sum_j \mathbf{w}_j \cdot \varphi(\|\mathbf{c}_j - \mathbf{x}\|) + \mathbf{p}(\mathbf{x})$$

- RBF evaluation
  - Function $\mathbf{d}$ transforms points
  - Jacobian $\nabla\mathbf{d}$ transforms normals
  - Precompute basis functions
  - Evaluate on the GPU!



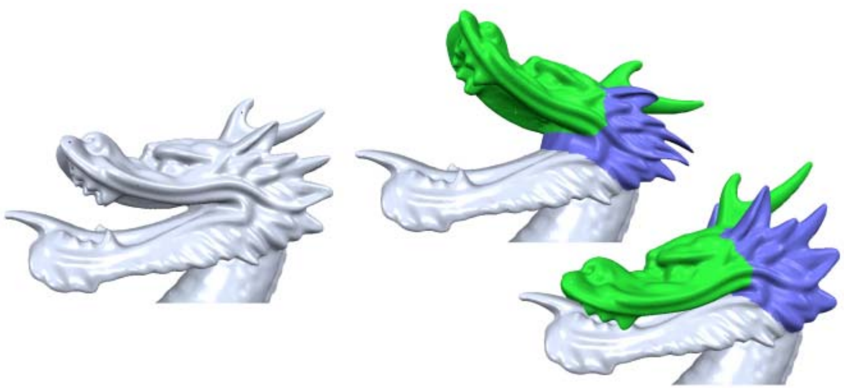
Local & Global Deformations

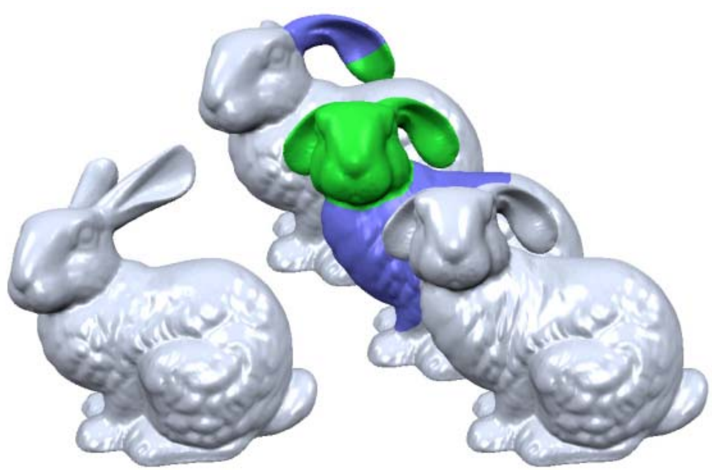
[RBF, Botsch and Kobbelt 05]



Local & Global Deformations

[RBF, Botsch and Kobbelt 05]



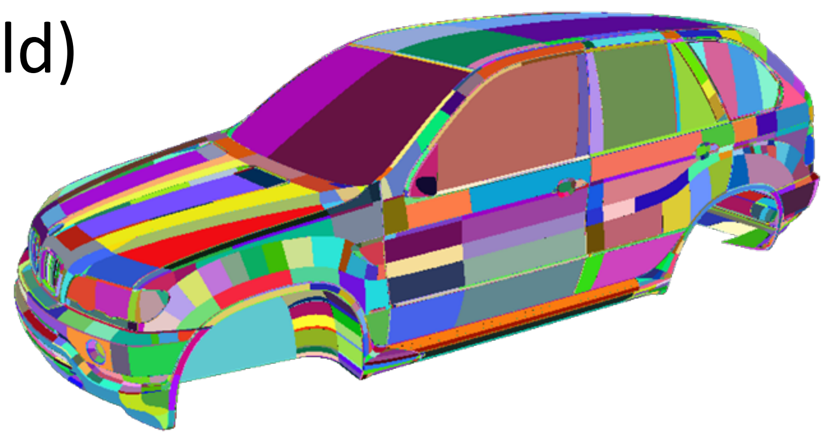


1M vertices
movie

Space Deformations

Summary so far

- Handle arbitrary input
  - Meshes (also non-manifold)
  - Point sets
  - Polygonal soups
  - …



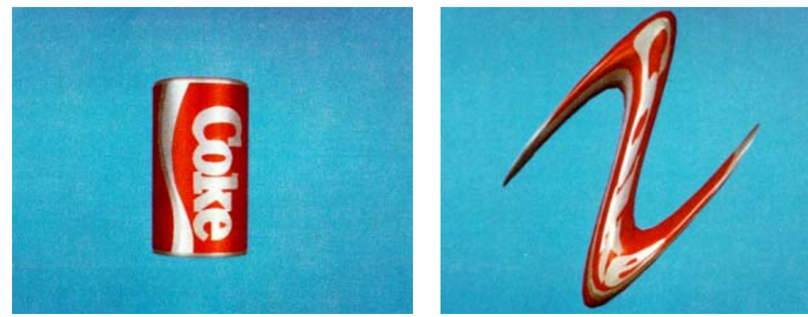
- 3M triangles
- 10k components
- Not oriented
- Not manifold

- Complexity mainly depends on the control object, not the surface

Space Deformations

Summary so far

- Handle arbitrary input
  - Meshes (also non-manifold)
  - Point sets
  - Polygonal soups
  - …



$$\mathbf{F}(x,y,z) = (F(x,y,z), G(x,y,z), H(x,y,z))$$

then the Jacobian is the determinant

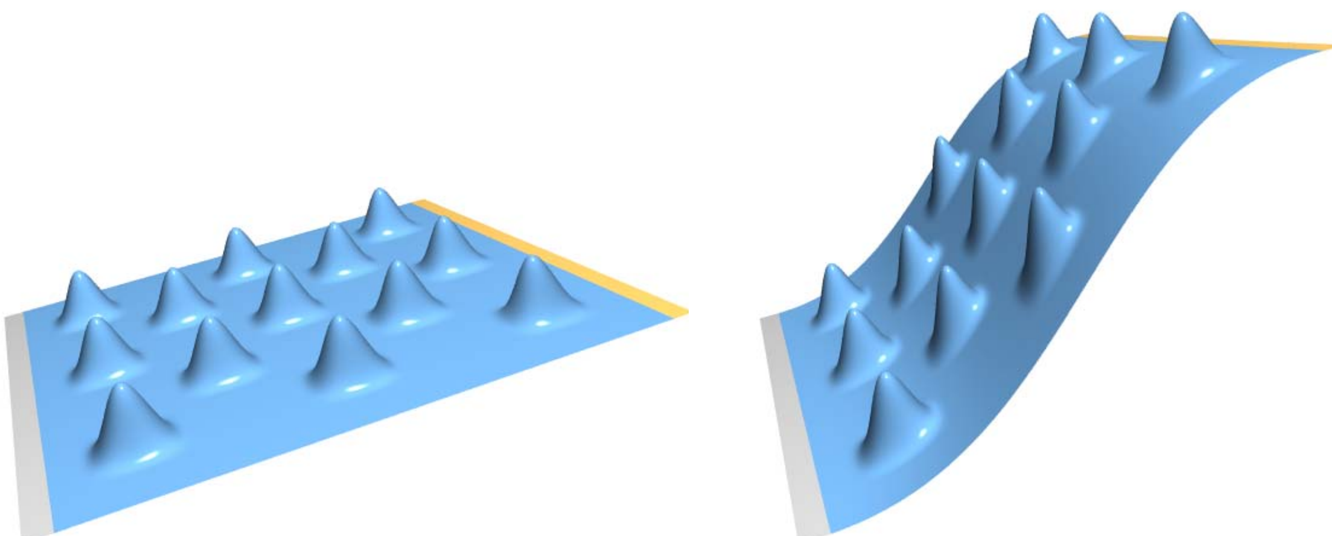
$$Jac(\mathbf{F}) = \begin{vmatrix} \frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} & \frac{\partial F}{\partial z} \\ \frac{\partial G}{\partial x} & \frac{\partial G}{\partial y} & \frac{\partial G}{\partial z} \\ \frac{\partial H}{\partial x} & \frac{\partial H}{\partial y} & \frac{\partial H}{\partial z} \end{vmatrix}$$

- Easier to analyze: functions on Euclidean domain
  - Volume preservation: |Jacobian| = 1

Space Deformations

Summary so far

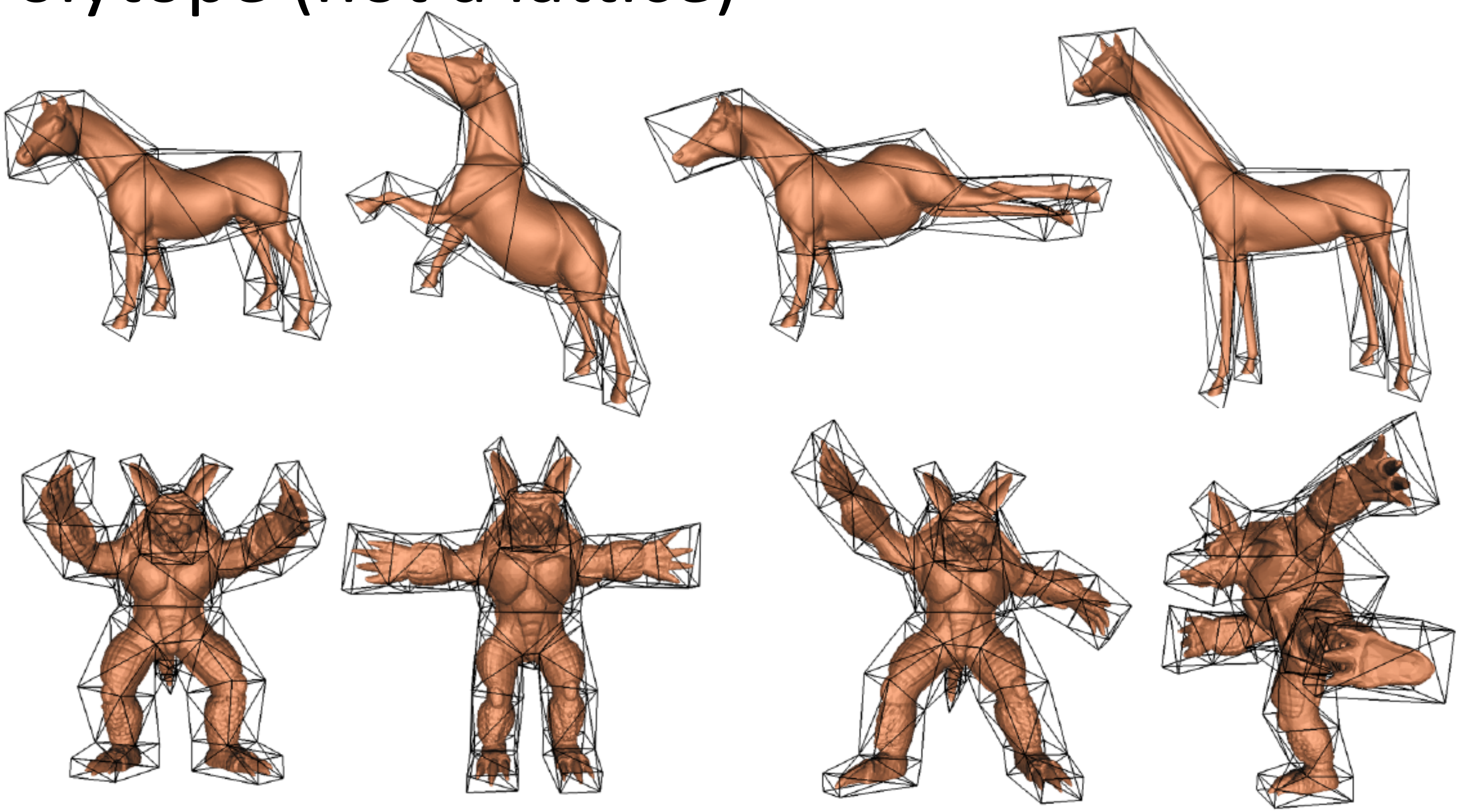
- The deformation is only loosely aware of the shape that is being edited
- Small Euclidean distance $\rightarrow$ similar deformation
- Local surface detail may be distorted



Cage-based Deformations

[Ju et al. 2005]

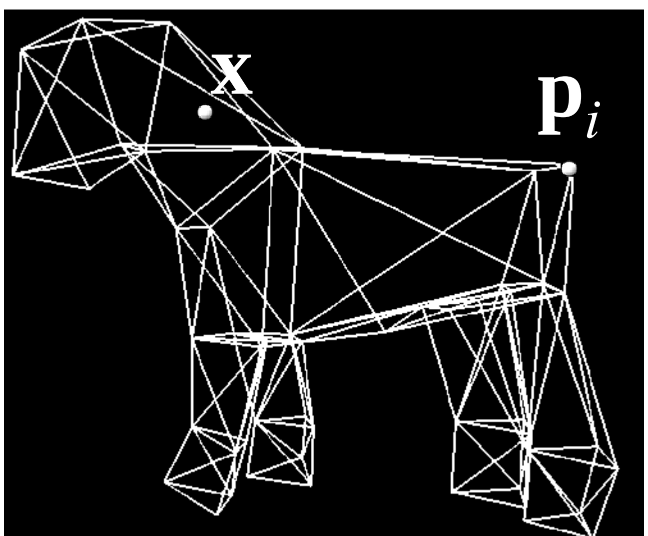
- Cage = crude version of the input shape
- Polytope (not a lattice)



Cage-based Deformations

[Ju et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)
- Each point $\mathbf{x}$ in space is represented w.r.t. to the cage elements using coordinate functions

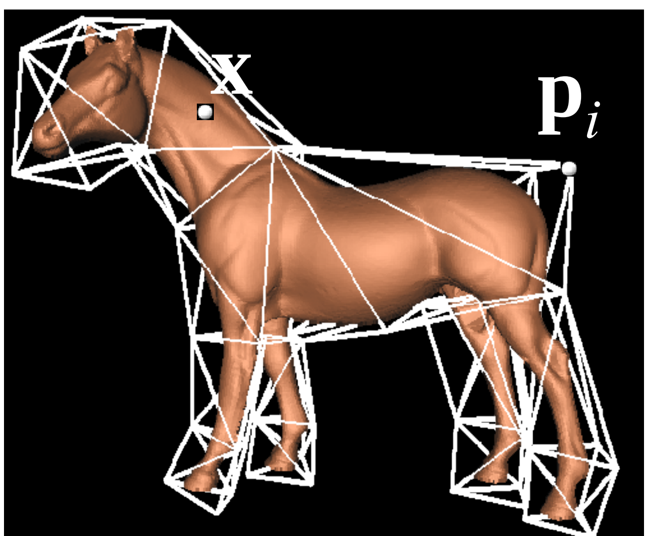


$$\mathbf{x} = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}_i$$

Cage-based Deformations

[Ju et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)
- Each point **x** in space is represented w.r.t. to the cage elements using coordinate functions

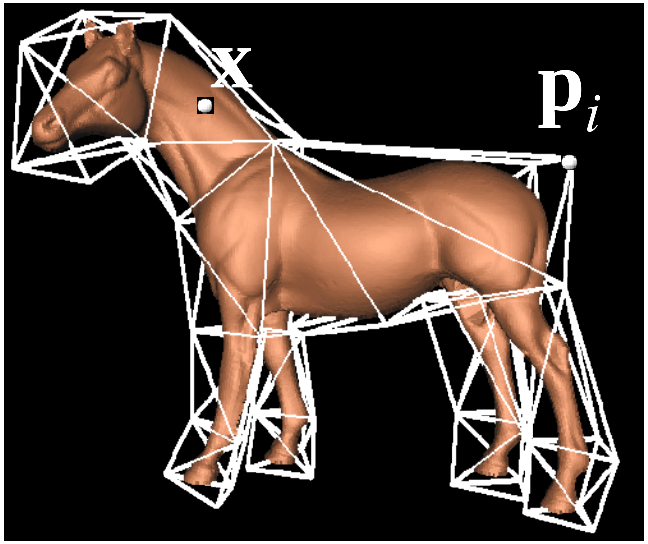


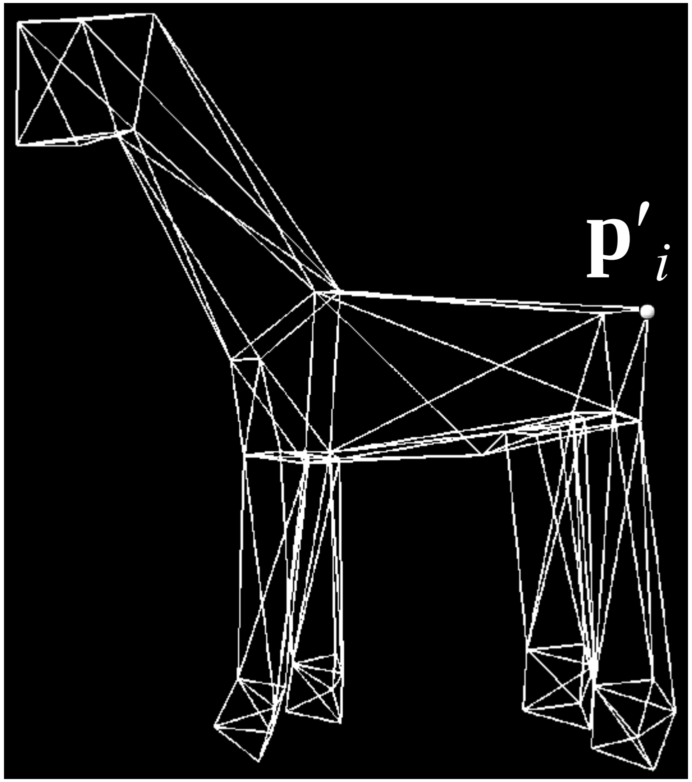
$$\mathbf{x} = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}_i$$

Cage-based Deformations

[Ju et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)



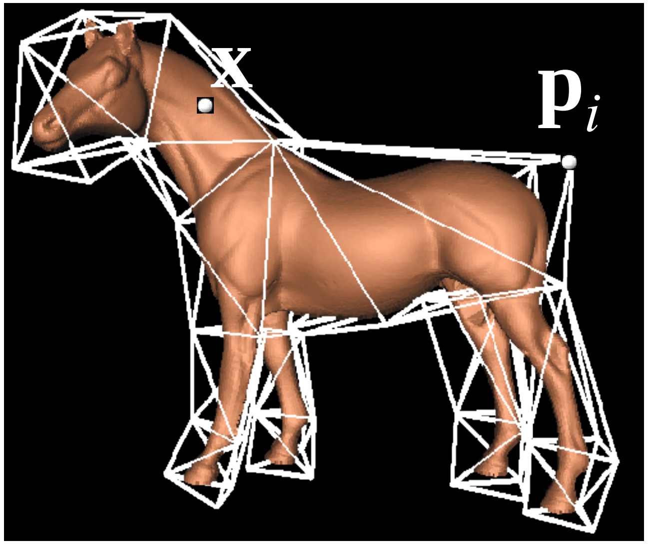


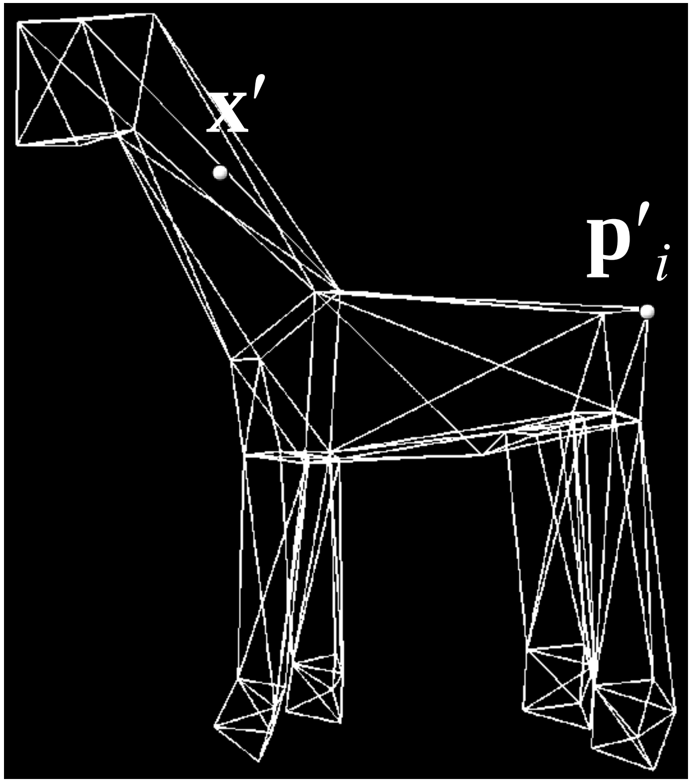
Cage-based Deformations

[Ju et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)

$$\mathbf{x}' = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}_i'$$



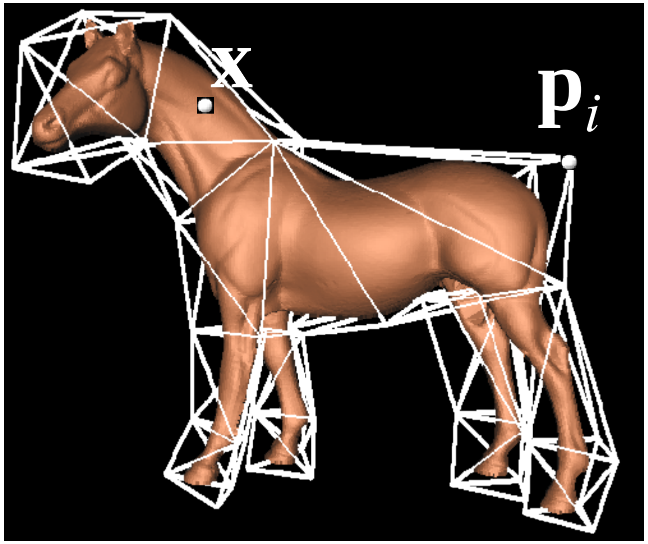


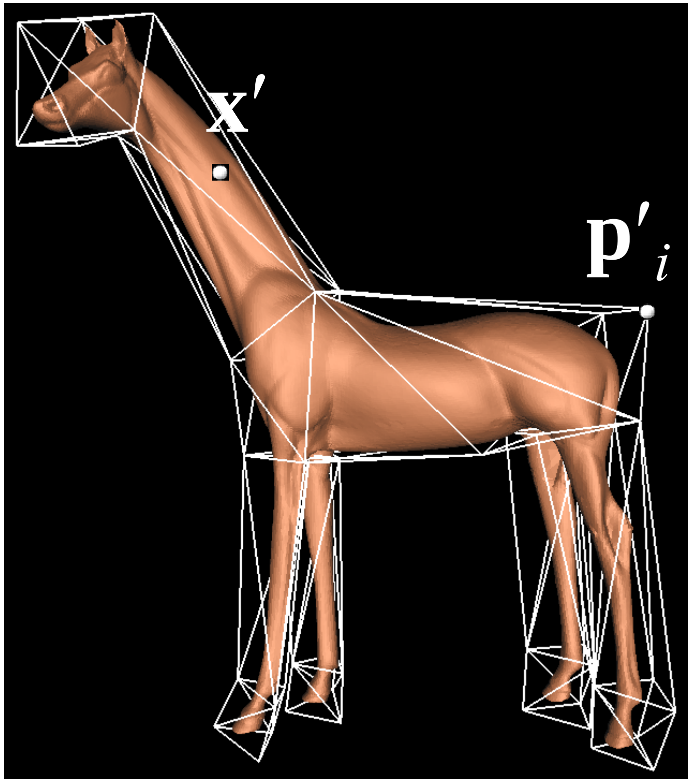
Cage-based Deformations

[Ju et al. 2005]

- Cage = crude version of the input shape
- Polytope (not a lattice)

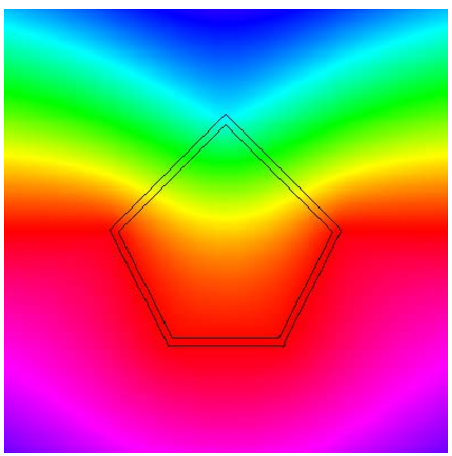
$$\mathbf{x}' = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}'_i$$

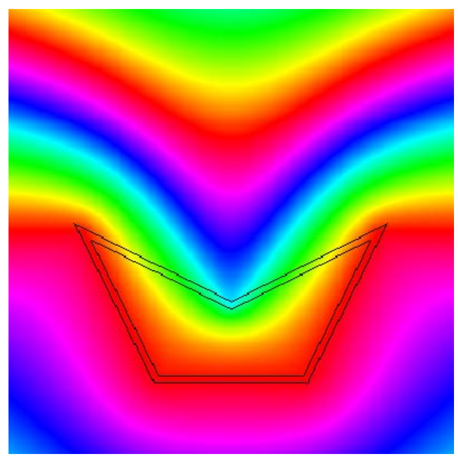


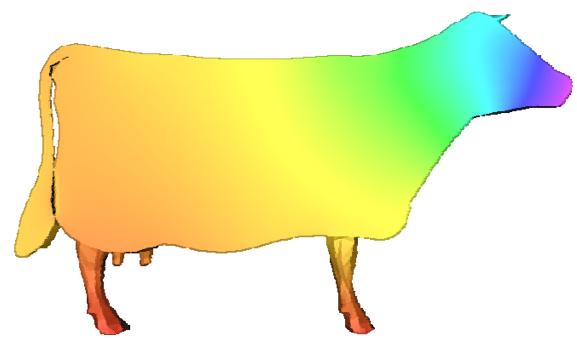


Coordinate Functions

- Mean-value coordinates (Floater, Ju et al. 2005)
  - Generalization of barycentric coordinates
  - Closed-form solution for $w_i(\mathbf{x})$

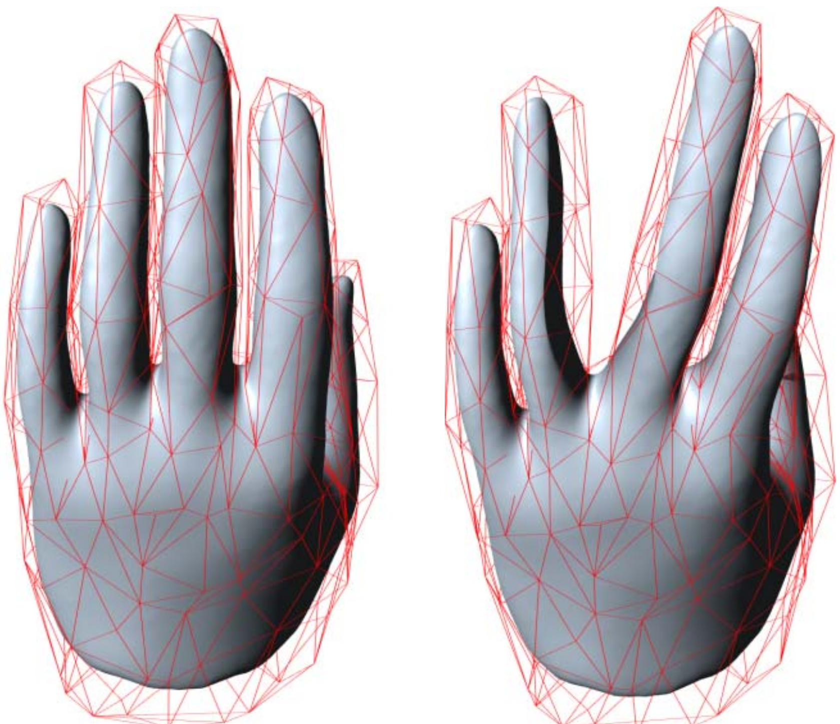






Coordinate Functions

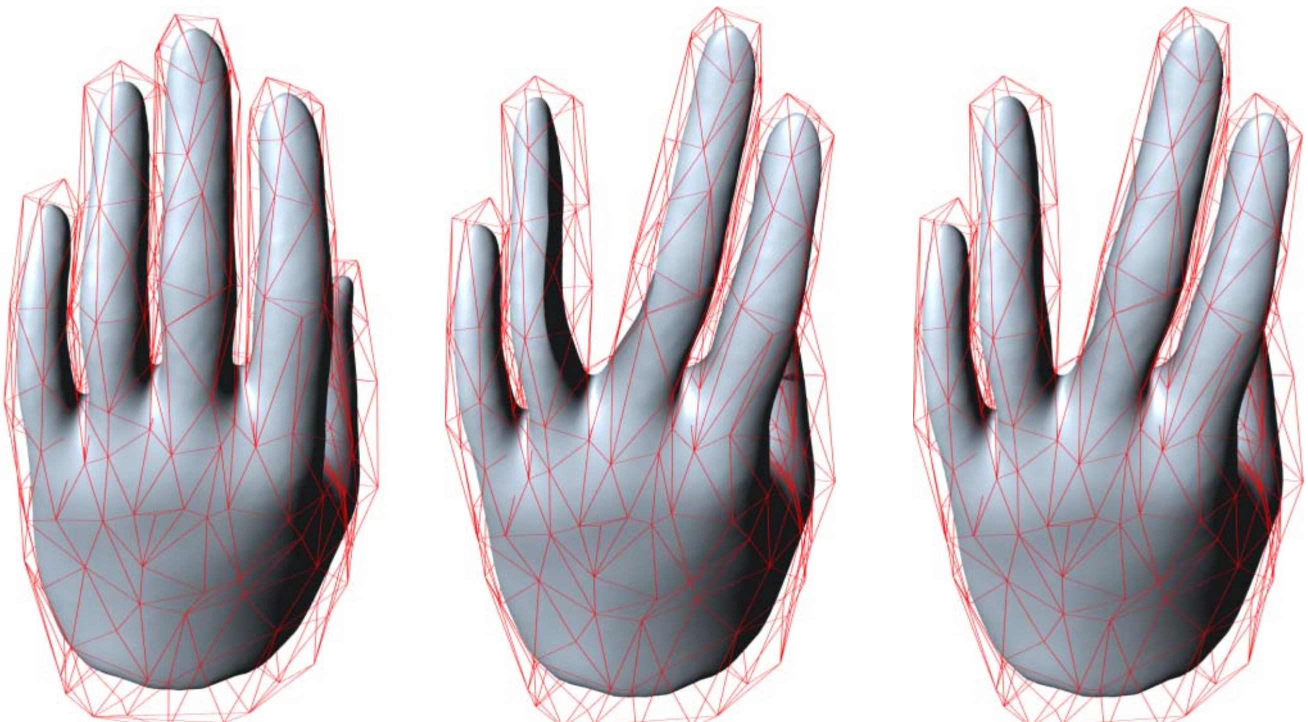
- Mean-value coordinates (Floater, Ju et al. 2005)
  - Not necessarily positive on non-convex domains



MVC

Coordinate Functions

- PMVC (Lipman et al. 2007) – ensures positivity, but no longer closed-form and only C^0



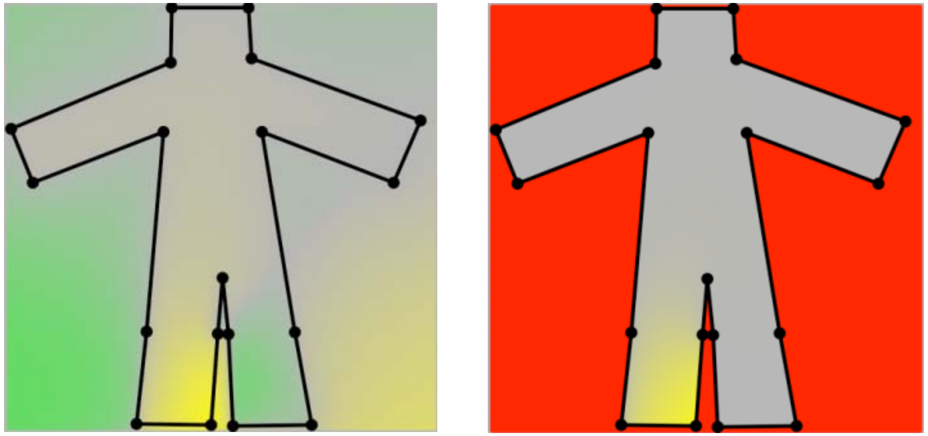
MVC PMVC

Coordinate Functions

- Harmonic coordinates (Joshi et al. 2007)
  - Harmonic functions $h_i(\mathbf{x})$ for each cage vertex $\mathbf{p}_i$
  - Solve

$$\Delta h = 0$$

subject to: h_i linear on the boundary s.t. $h_i(\mathbf{p}_j) = \delta_{ij}$



MVC HC

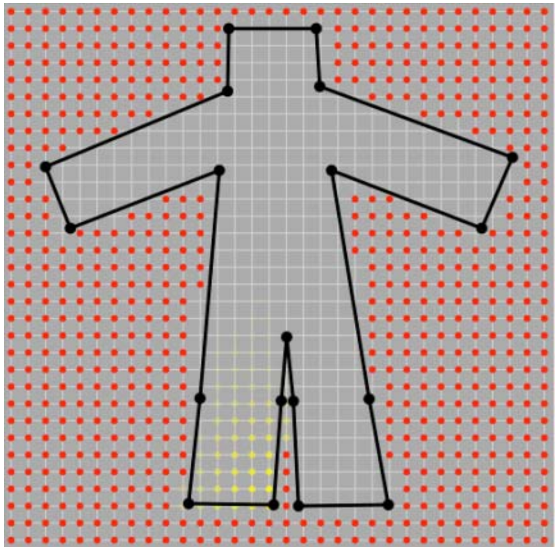
Coordinate Functions

- Harmonic coordinates (Joshi et al. 2007)
  - Harmonic functions $h_i(\mathbf{x})$ for each cage vertex $\mathbf{p}_i$
  - Solve

$$\Delta h = 0$$

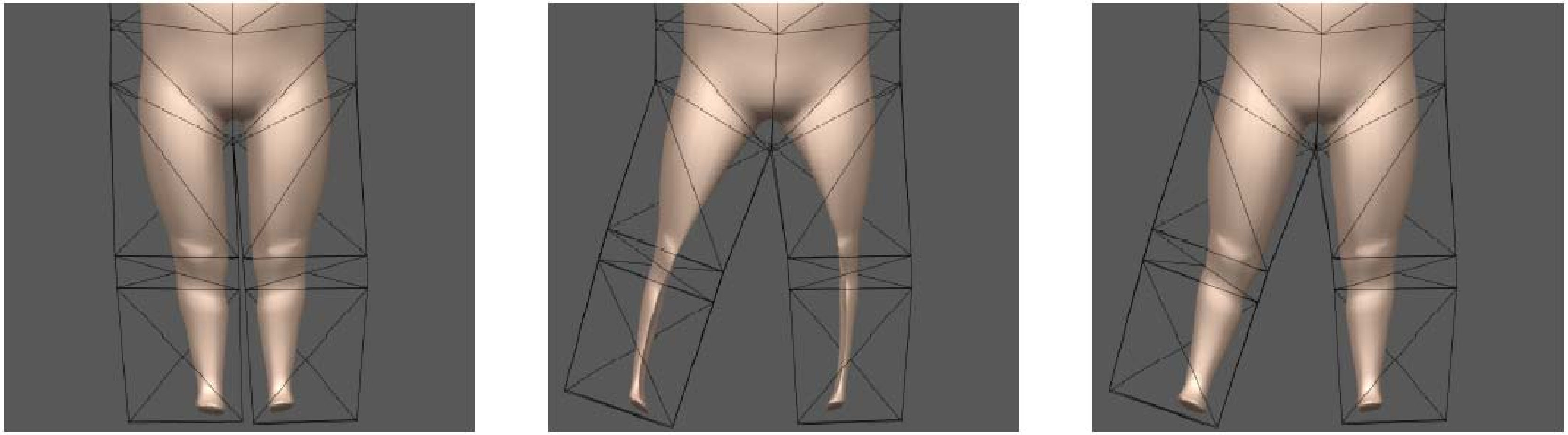
subject to: h_i linear on the boundary s.t. $h_i(\mathbf{p}_j) = \delta_{ij}$

  - Volumetric Laplace equation
  - Discretization, no closed-form



Coordinate Functions

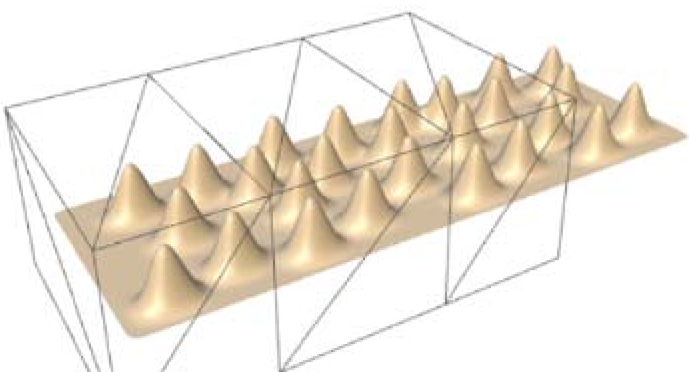
- Harmonic coordinates (Joshi et al. 2007)

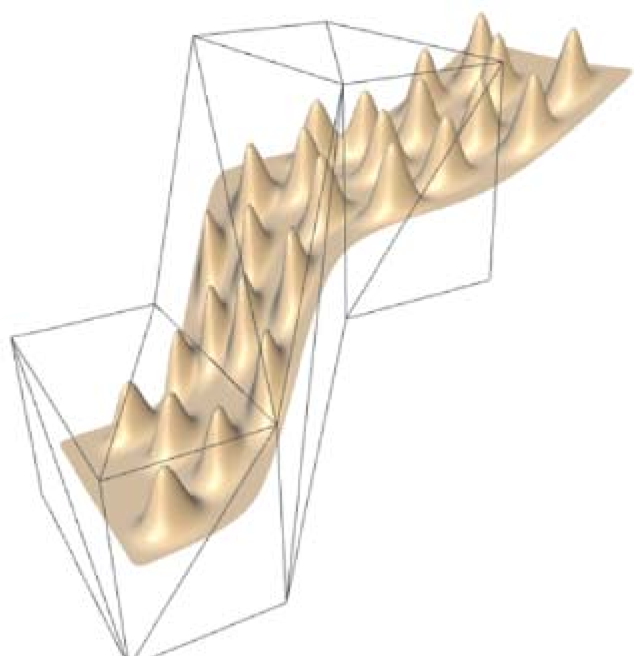


MVC HC

Coordinate Functions

- Green coordinates (Lipman et al. 2008)
- Observation: previous vertex-based basis functions always lead to affine-invariance!

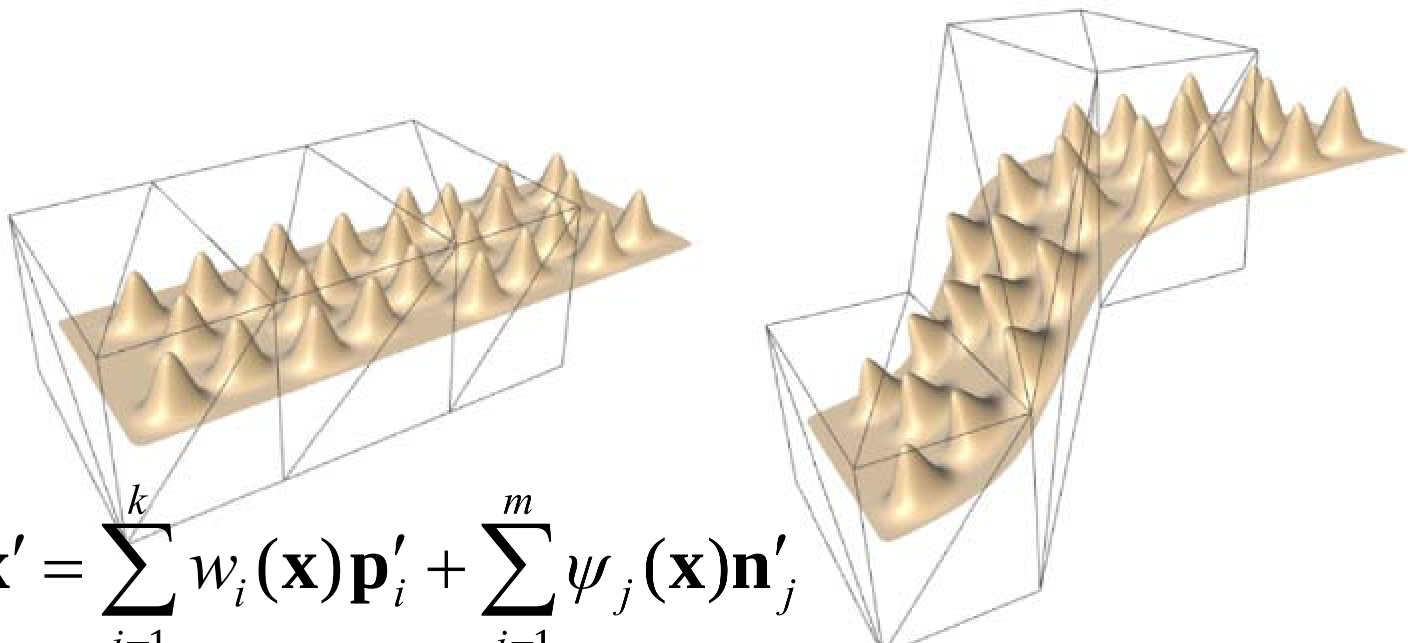




$$\mathbf{x}' = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}'_i$$

Coordinate Functions

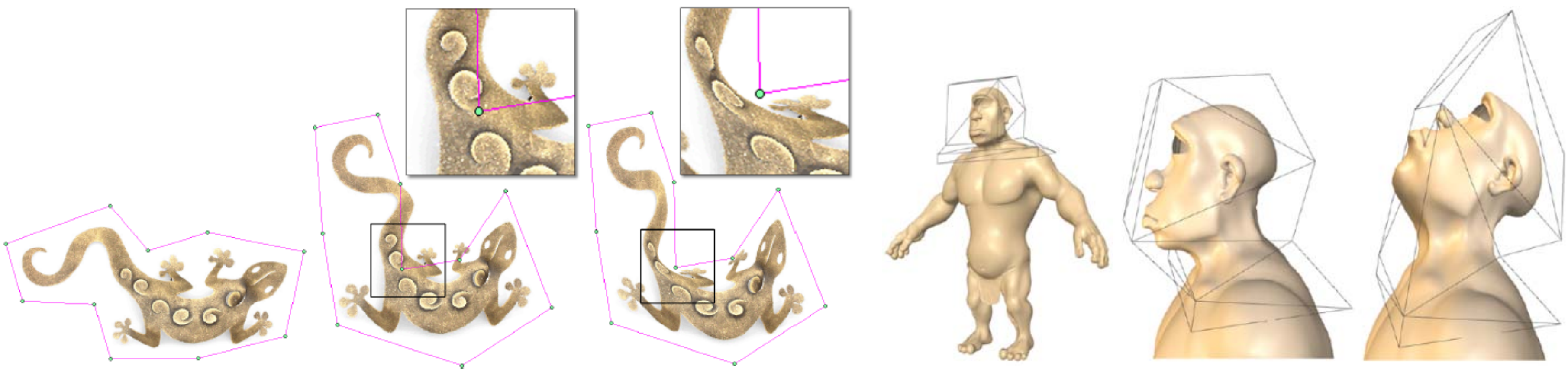
- Green coordinates (Lipman et al. 2008)
- Correction: Make the coordinates depend on the cage faces as well



$$\mathbf{x}' = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}'_i + \sum_{j=1}^{m} \psi_j(\mathbf{x})\mathbf{n}'_j$$

Coordinate Functions

- Green coordinates (Lipman et al. 2008)
- Closed-form solution
- Conformal in 2D, quasi-conformal in 3D



GC MVC GC

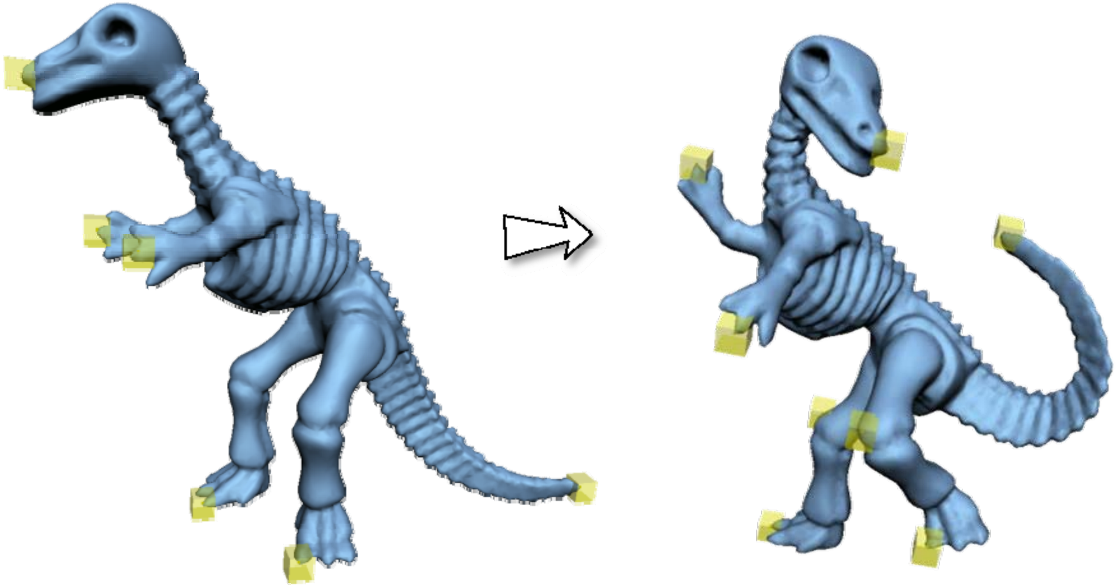
Coordinate Functions

- Green coordinates (Lipman et al. 2008)
- Closed-form solution
- Conformal in 2D, quasi-conformal in 3D

Alternative interpretation in 2D via holomorphic functions and extension to point handles **: Weber et al. Eurographics 2009**

Nonlinear Space Deformations

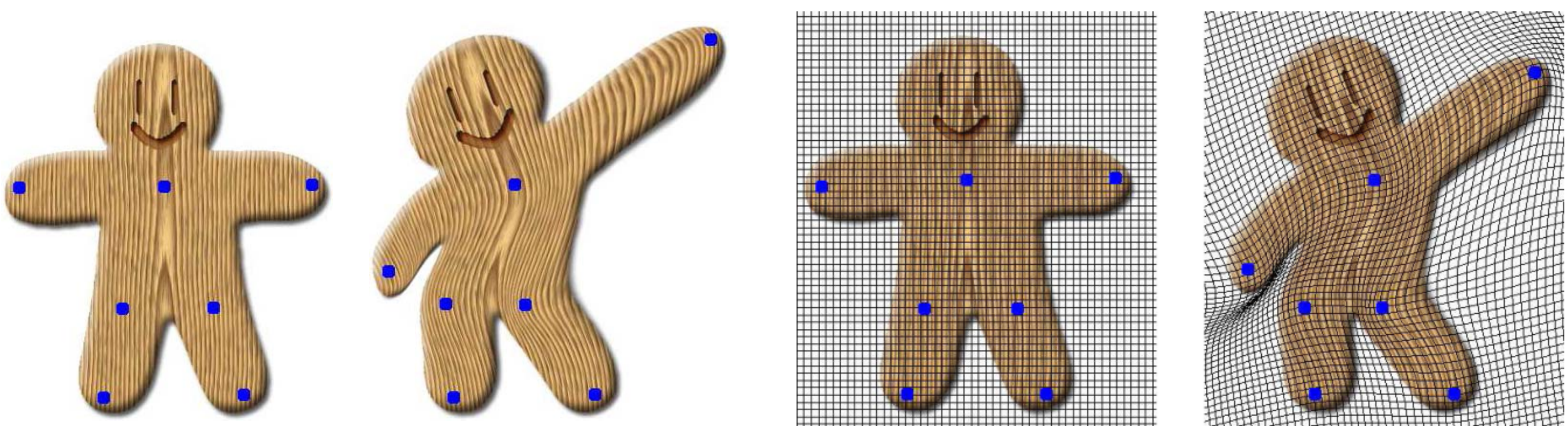
- Involve nonlinear optimization
- Enjoy the advantages of space warps
- Additionally, have shape-preserving properties



As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

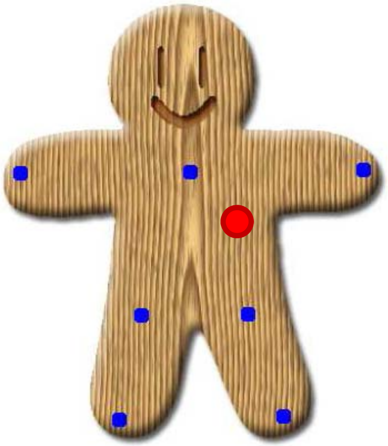
- Points or segments as control objects
- First developed in 2D and later extended to 3D by Zhu and Gortler (2007)



As-Rigid-As-Possible Deformation

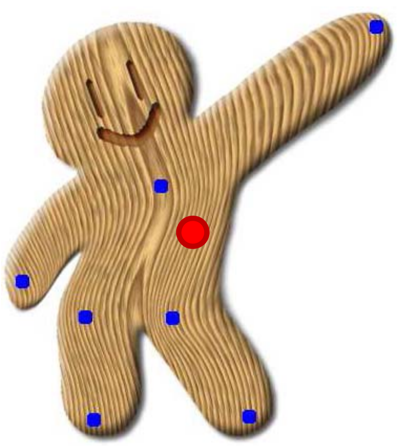
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Attach an affine transformation to each point $\mathbf{x} \in \mathrm{R}^3$:



$$A_{\mathbf{x}}(\mathbf{p}) = M_{\mathbf{x}}\mathbf{p} + \mathbf{t}_{\mathbf{x}}$$

- The space warp:



$$\mathbf{x} \rightarrow A_{\mathbf{x}}(\mathbf{x})$$

As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

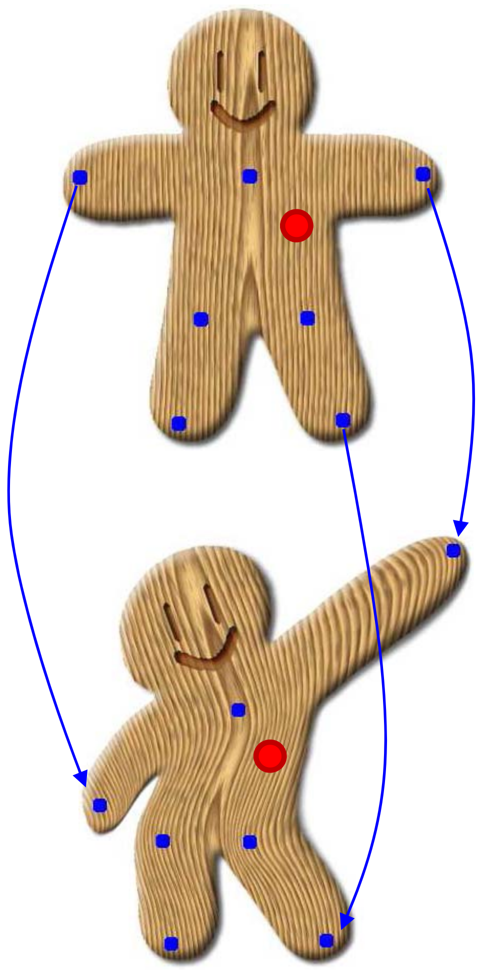
- Handles $\mathbf{p}_i$ are displaced to $\mathbf{q}_i$
- The local transformation at $\mathbf{x}$:

$$A_{\mathbf{x}}(\mathbf{p}) = M_{\mathbf{x}}\mathbf{p} + \mathbf{t}_{\mathbf{x}} \quad \text{s.t.}$$

$$\sum_{i=1}^{k} w_i(\mathbf{x}) \left\| A_{\mathbf{x}}(\mathbf{p}_i) - \mathbf{q}_i \right\|^2 \rightarrow \min$$

- The weights depend on $\mathbf{x}$:

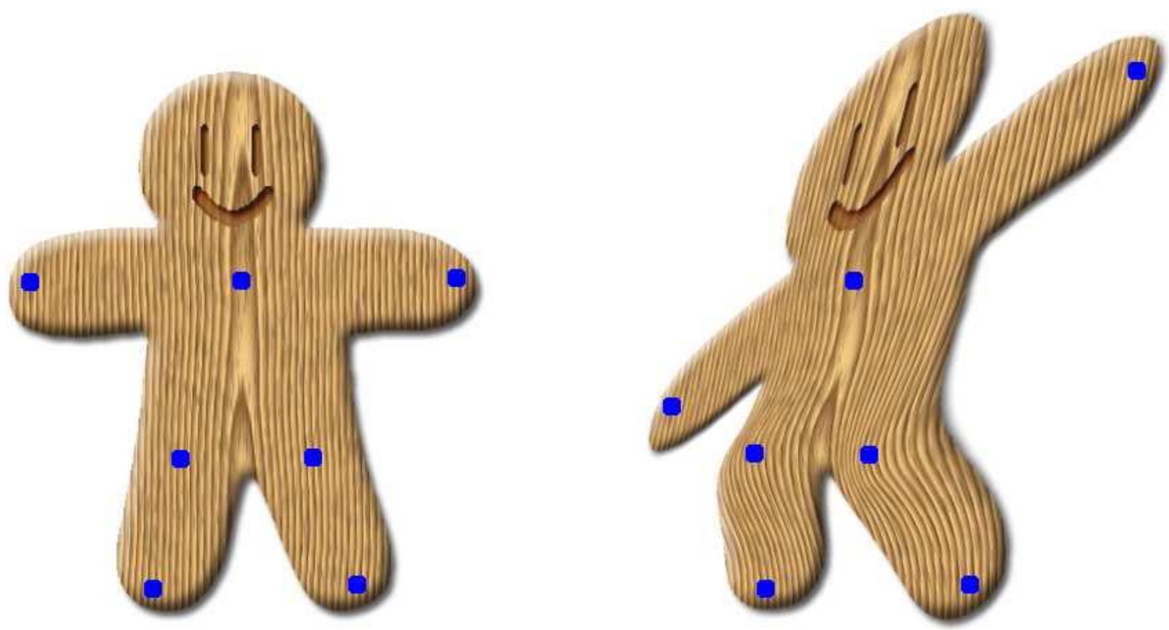
$$w_i(\mathbf{x}) = \|\mathbf{p}_i - \mathbf{x}\|^{-2\alpha}$$



As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

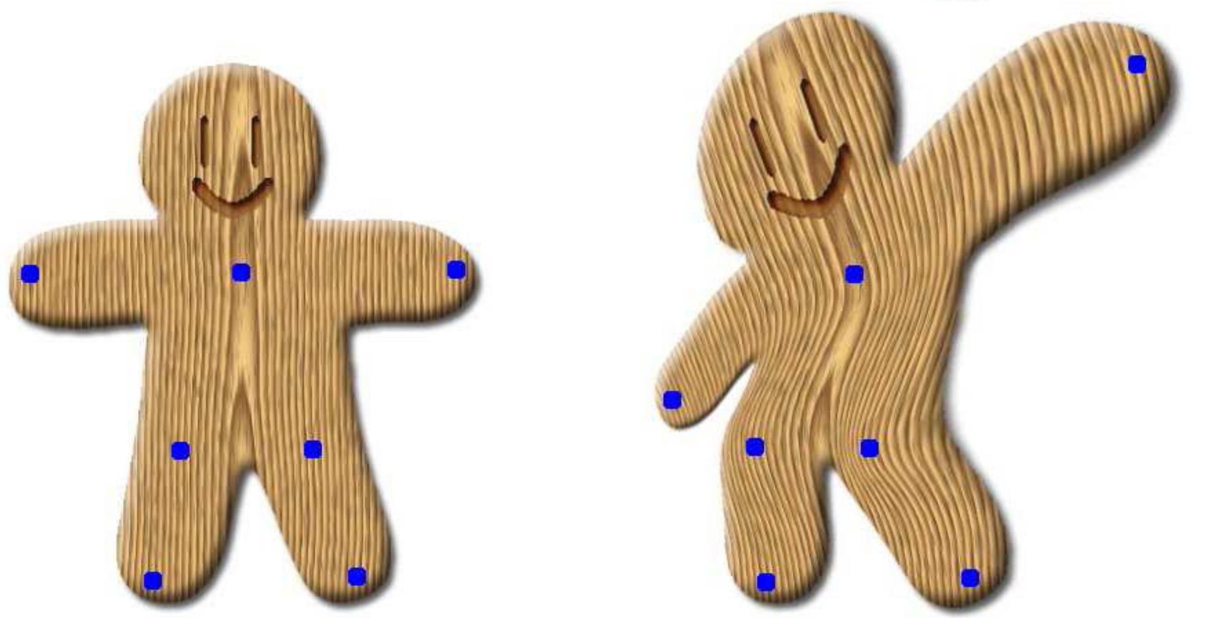
- No additional restriction on $\mathrm{A}_{\mathbf{x}}(\cdot)$ – affine local transformations



As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $\mathrm{A}_{\mathbf{x}}(\cdot)$ to similarity

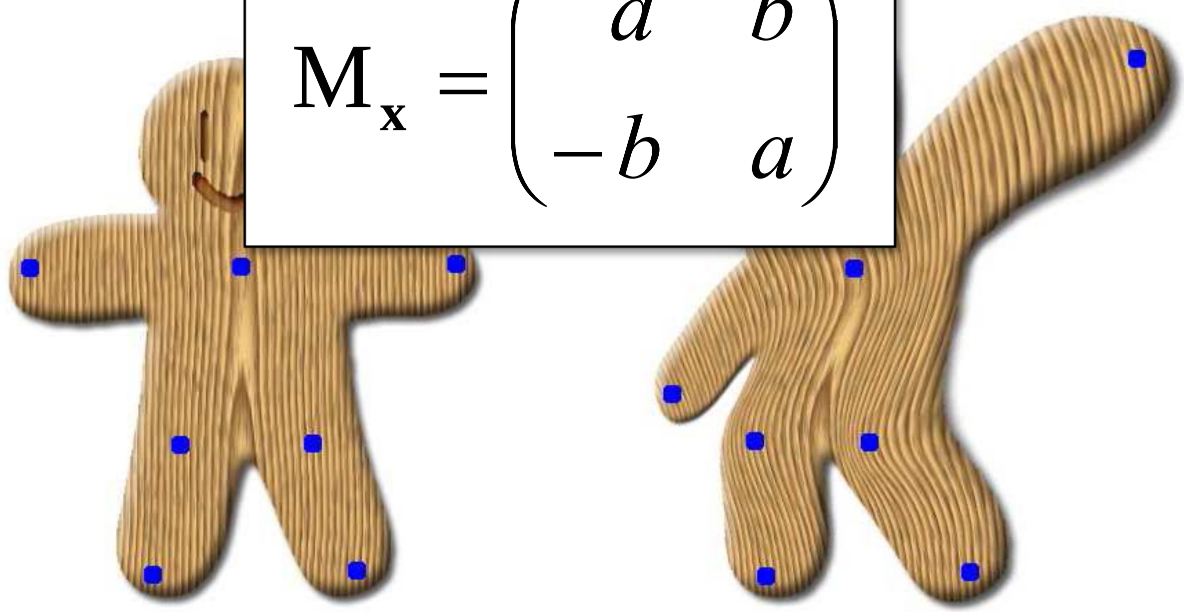


As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $\mathrm{A}_{\mathbf{x}}(\cdot)$ to similarity

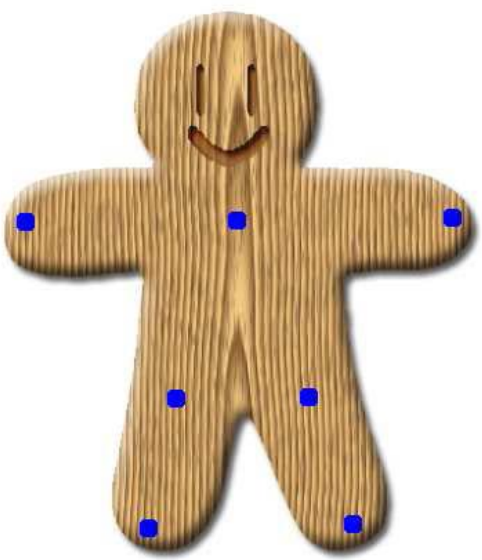
$$\mathrm{M}_{\mathbf{x}} = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$$

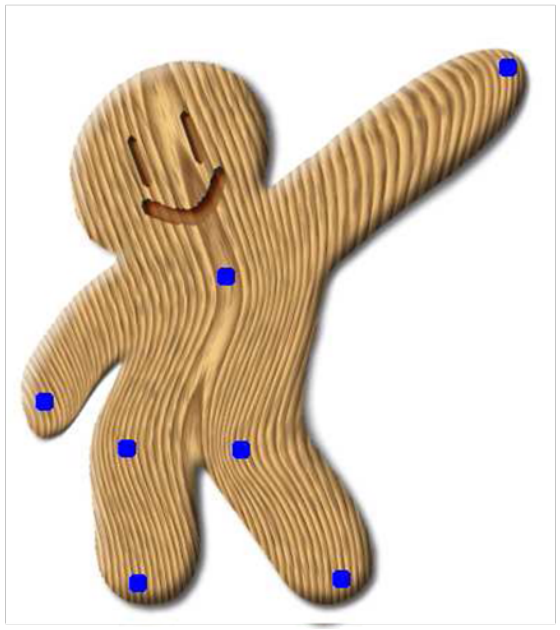


As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $\mathrm{A}_{\mathbf{x}}(\cdot)$ to rigid

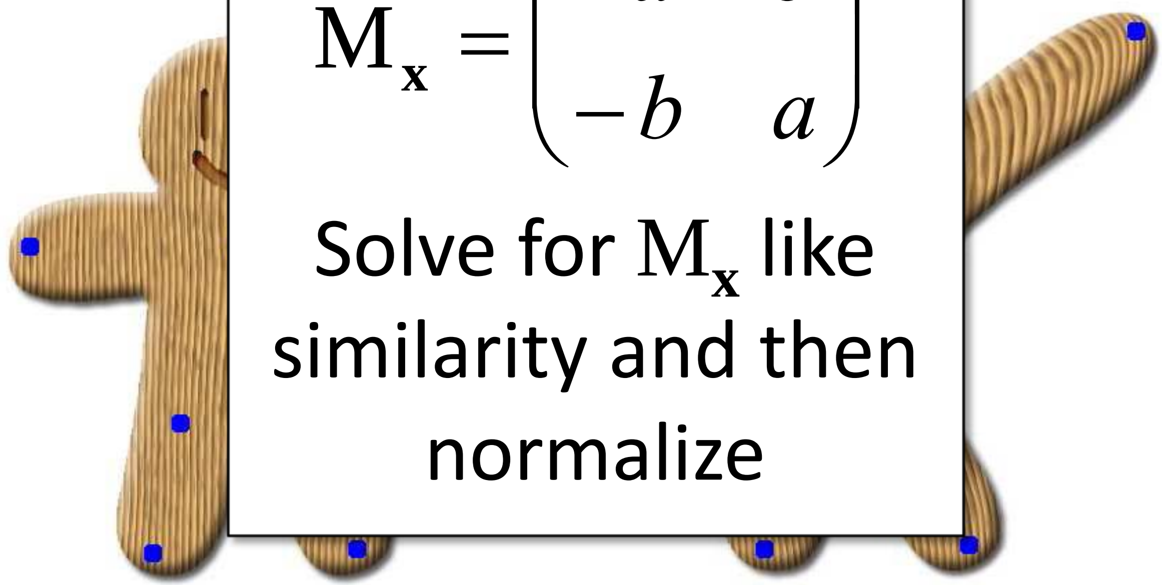




As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $\mathrm{A}_{\mathbf{x}}(\cdot)$ to rigid



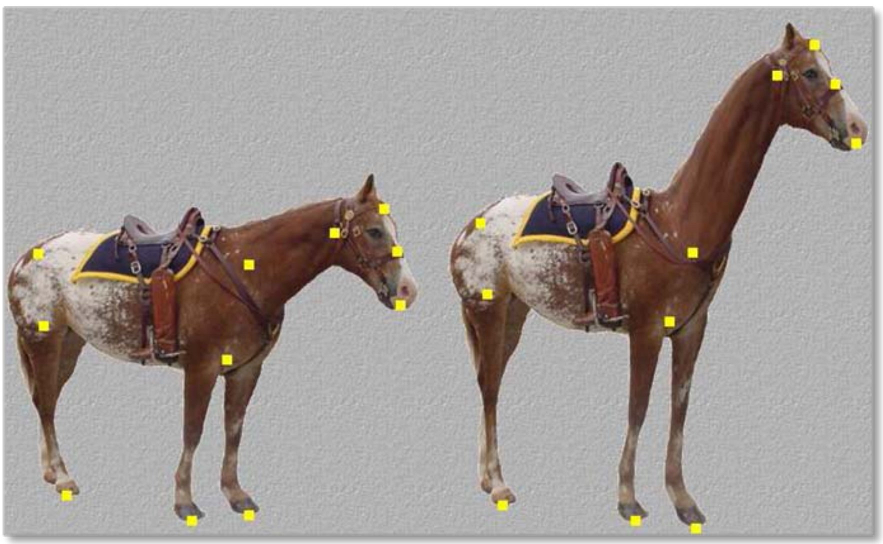
$$\mathrm{M}_{\mathbf{x}} = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$$

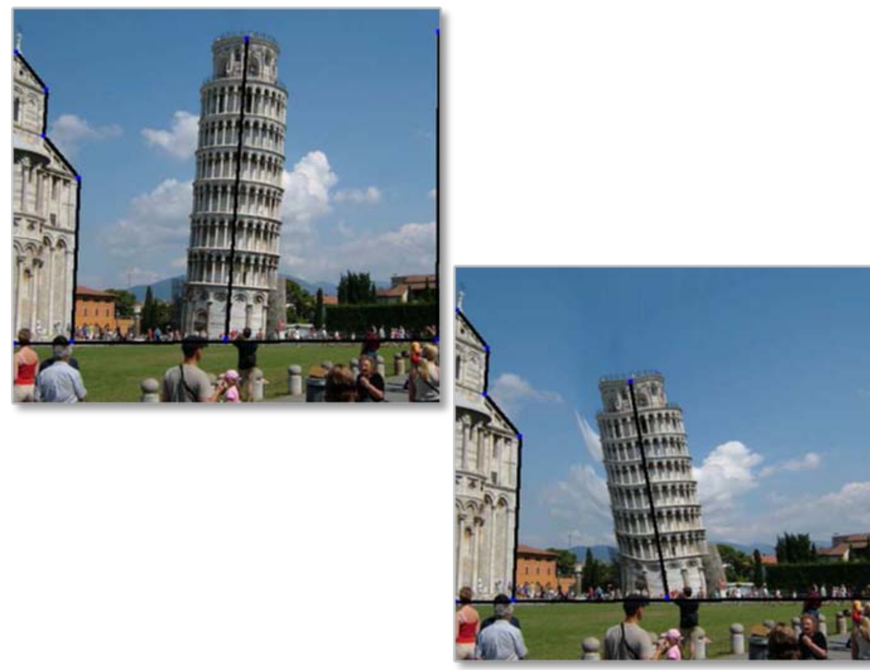
Solve for $\mathrm{M}_{\mathbf{x}}$ like similarity and then normalize

As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Examples





As-Rigid-As-Possible Deformation

MLS approach – extension to 3D [Zhu & Gortler 2007]

- No linear expression for similarity in 3D
- Instead, can solve for the minimizing rotation

$$\underset{\mathrm{R}\in SO(3)}{\arg\min} \sum_{i=1}^{k} w_i(\mathbf{x}) \left\| \mathrm{R}\mathbf{p}_i - \mathbf{q}_i \right\|^2$$

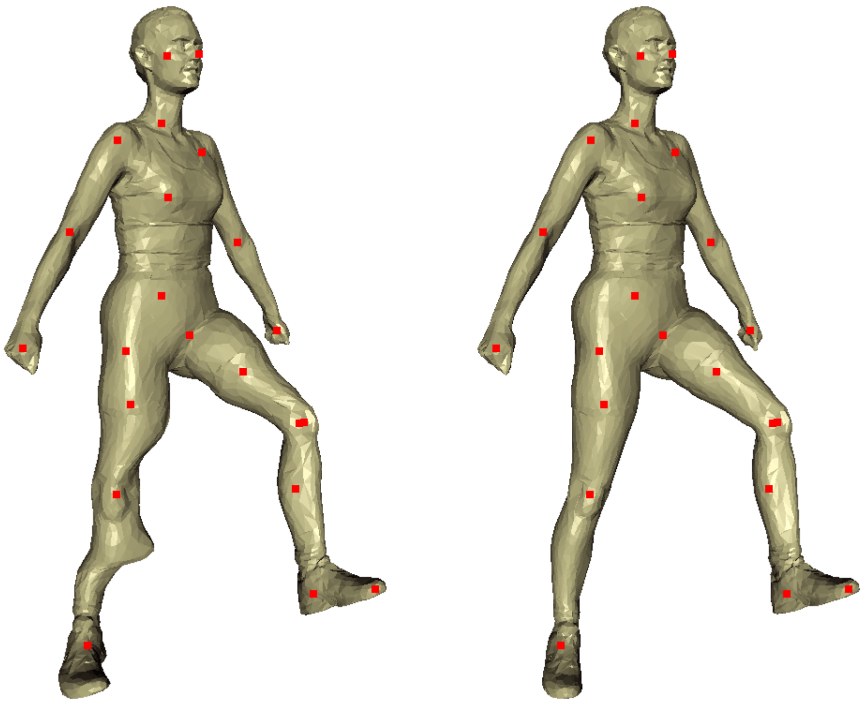
by polar decomposition of the 3×3 covariance matrix

As-Rigid-As-Possible Deformation

MLS approach – extension to 3D [Zhu & Gortler 2007]

- Zhu and Gortler also replace the Euclidean distance in the weights by "distance within the shape"

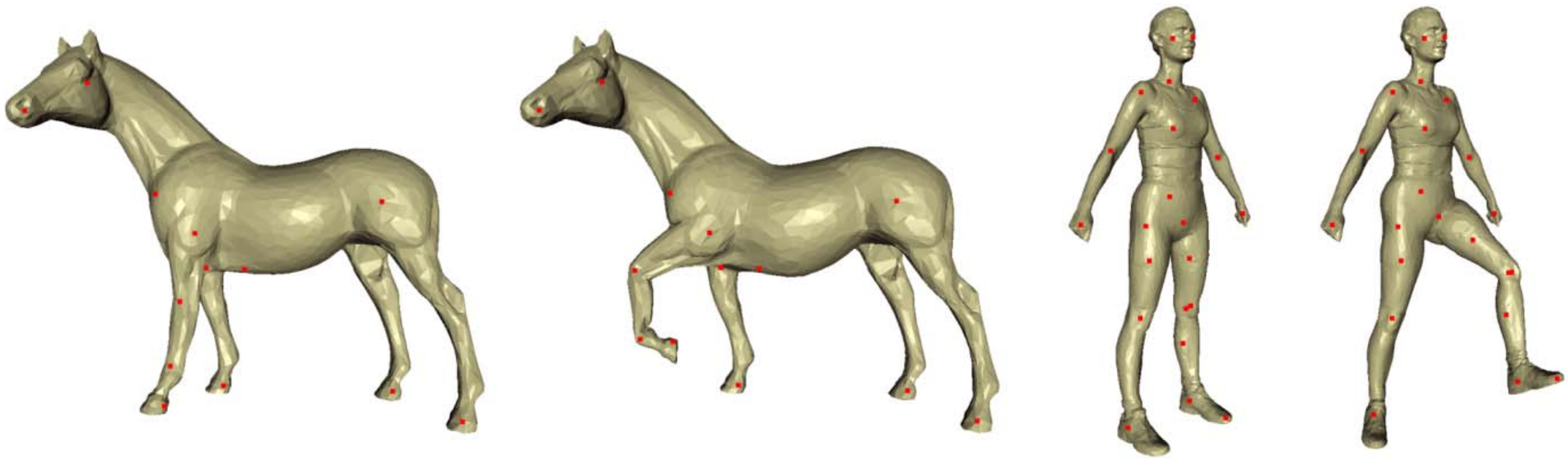
$$w_i(\mathbf{x}) = d(\mathbf{p}_i, \mathbf{x})^{-2\alpha}$$

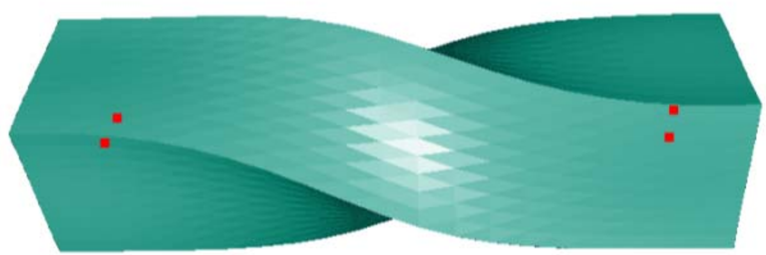


As-Rigid-As-Possible Deformation

MLS approach – extension to 3D [Zhu & Gortler 2007]

- More results

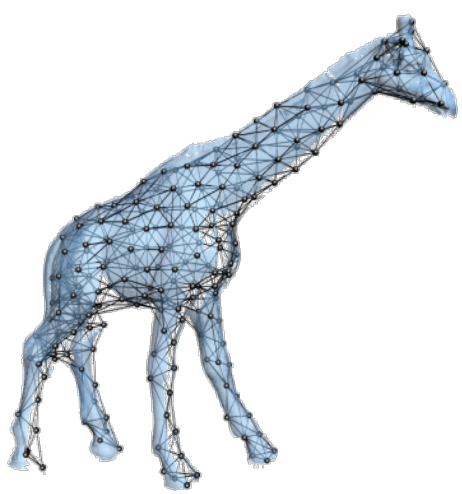




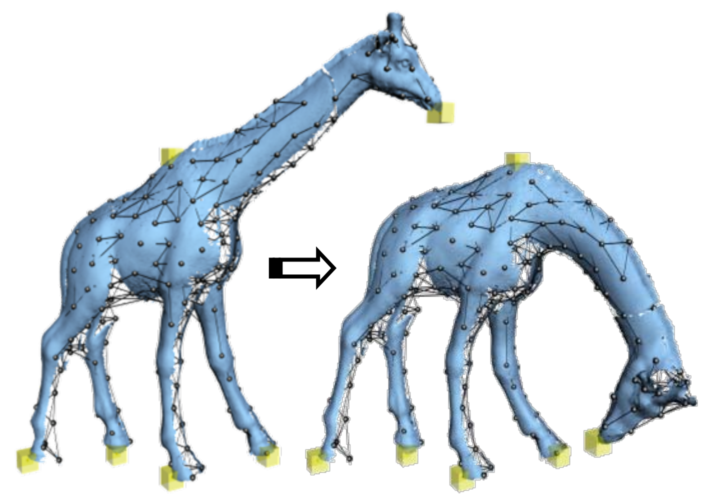
As-Rigid-As-Possible Deformation

Deformation Graph approach [Sumner et al. 2007]

- Surface handles as interface
- Underlying graph to represent the deformation; nodes store rigid transformations
- Decoupling of handles from def. representation



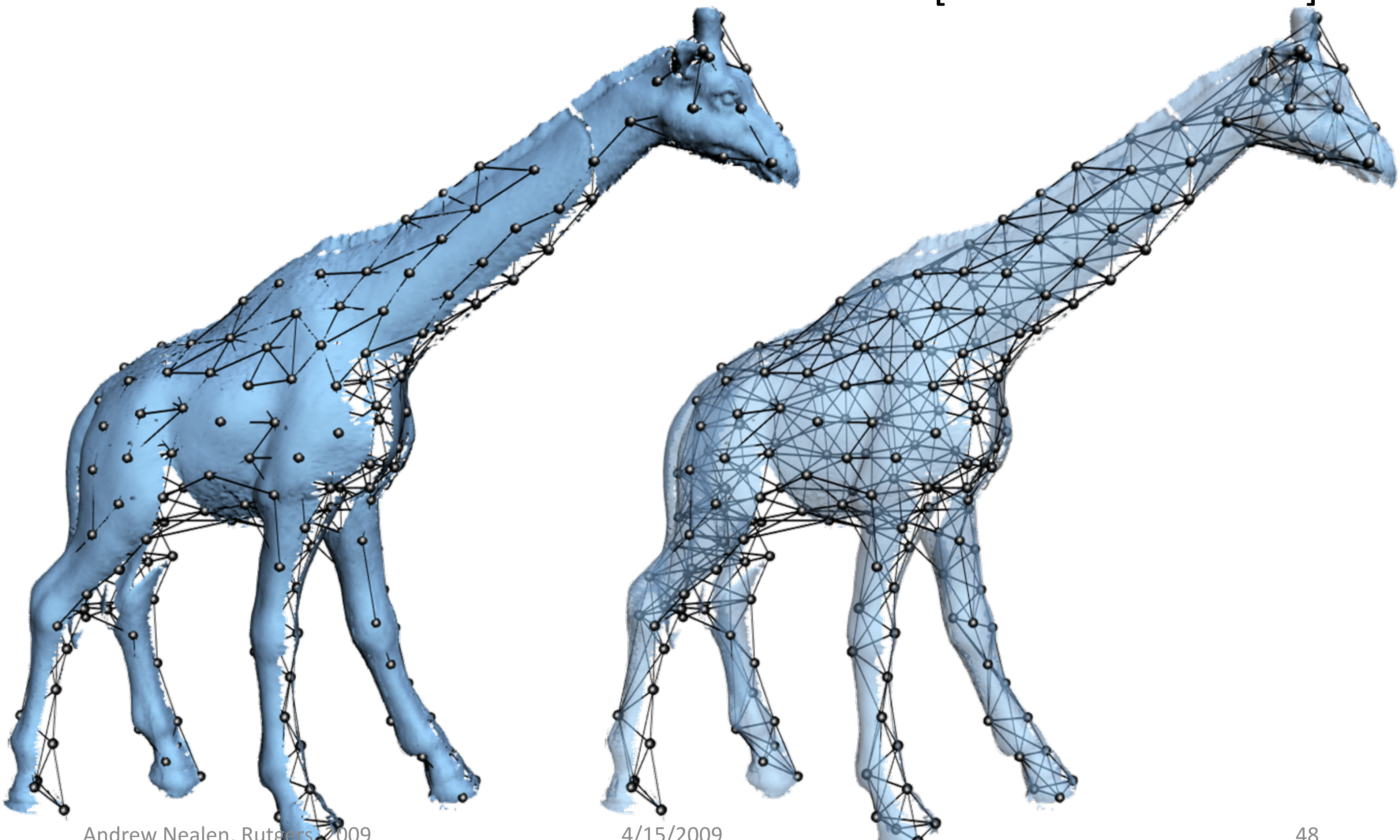
Deformation Graph



Optimization Procedure

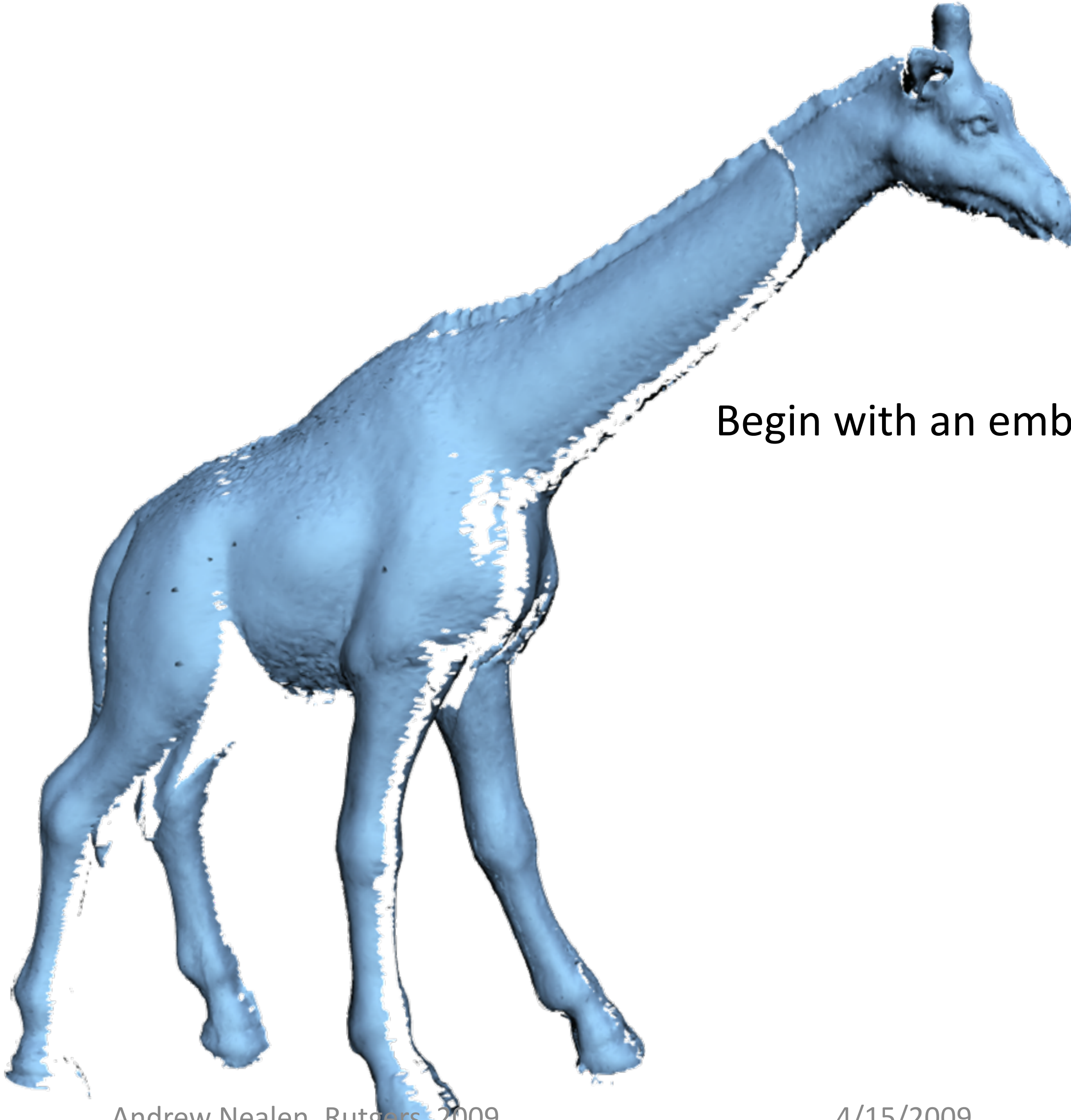
Deformation Graph

[Sumner et al. 2007]



Deformation Graph

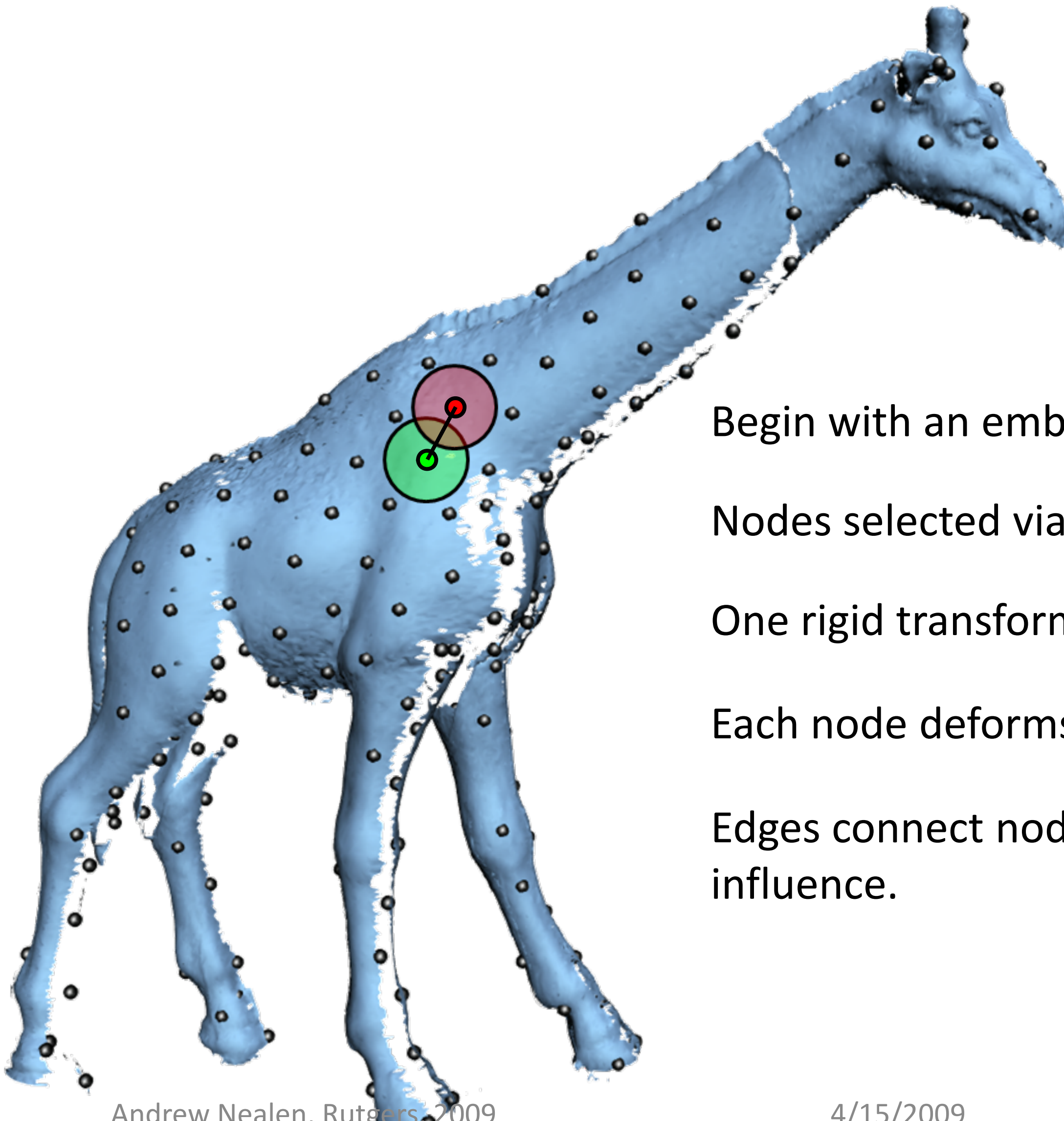
[Sumner et al. 2007]



Begin with an embedded object.

Deformation Graph

[Sumner et al. 2007]



Begin with an embedded object.

Nodes selected via uniform sampling; located at $\mathbf{g}_j$

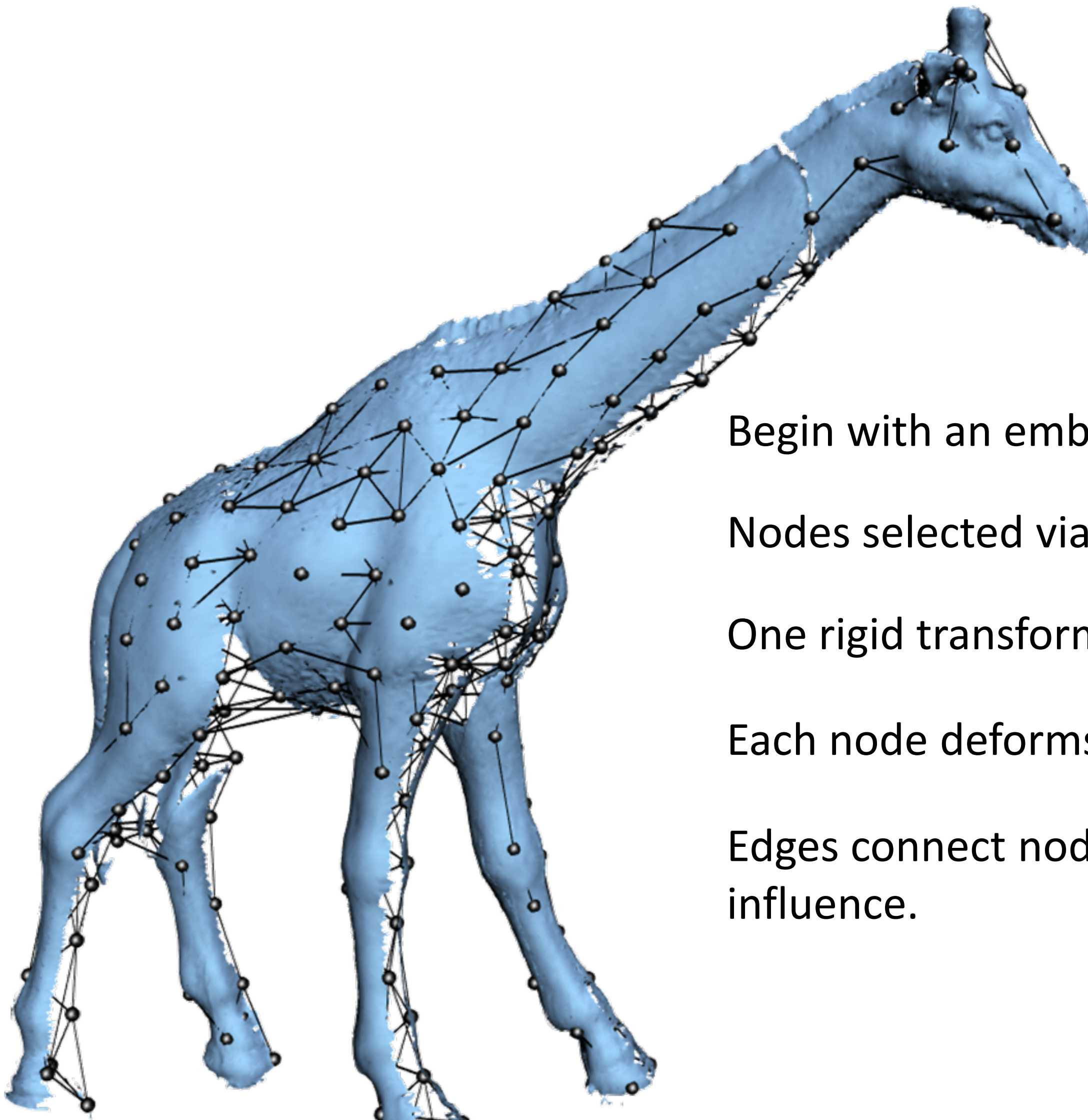
One rigid transformation for each node: R_j , $\mathbf{t}_j$

Each node deforms nearby space.

Edges connect nodes of overlapping influence.

Deformation Graph

[Sumner et al. 2007]



Begin with an embedded object.

Nodes selected via uniform sampling; located at $\mathbf{g}_j$

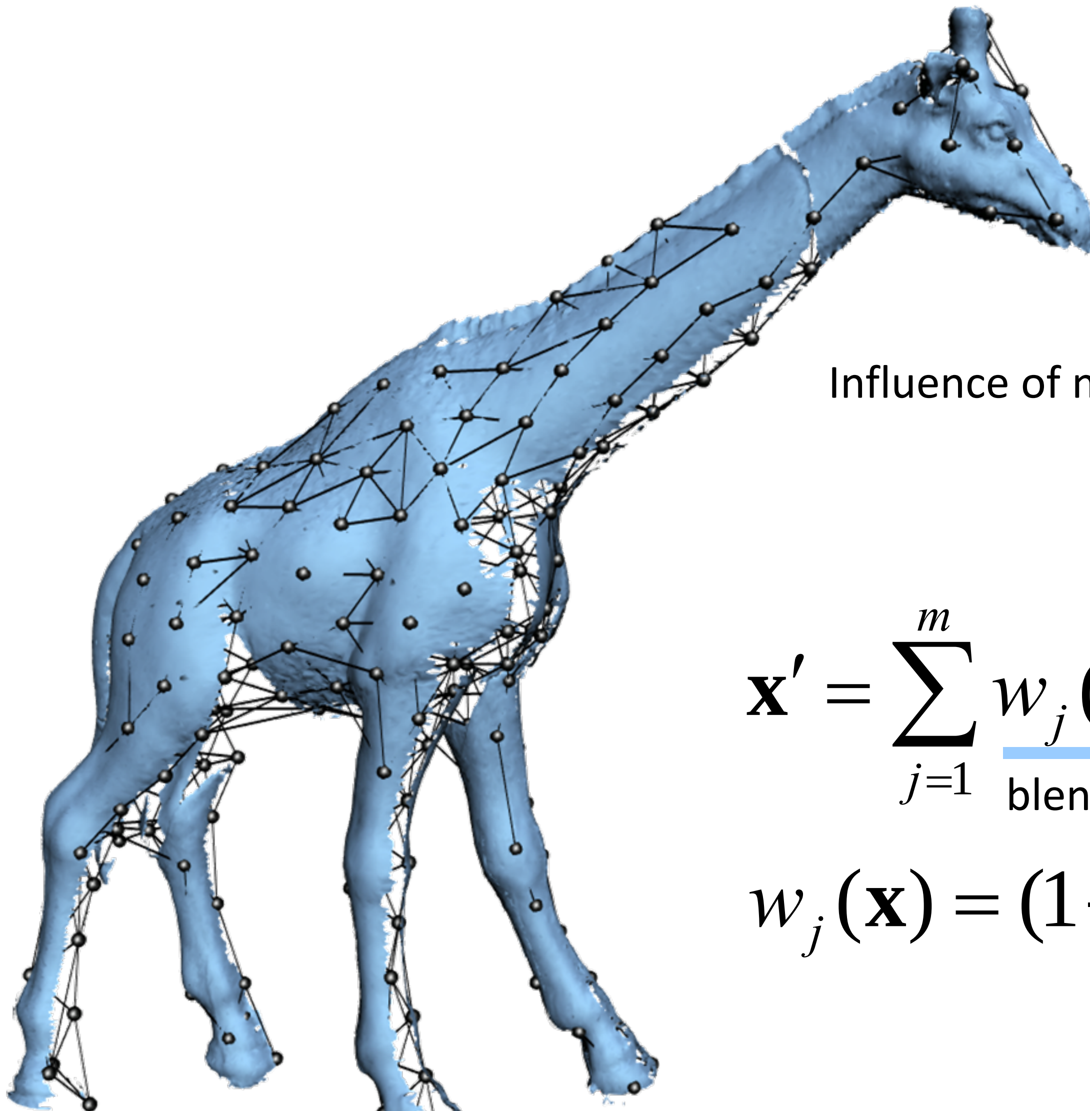
One rigid transformation for each node: $\mathrm{R}_j\ ,\ \mathbf{t}_j$

Each node deforms nearby space.

Edges connect nodes of overlapping influence.

Deformation Graph

[Sumner et al. 2007]



Influence of nearby transformations is blended.

point $\mathbf{x}$ transformed by node j

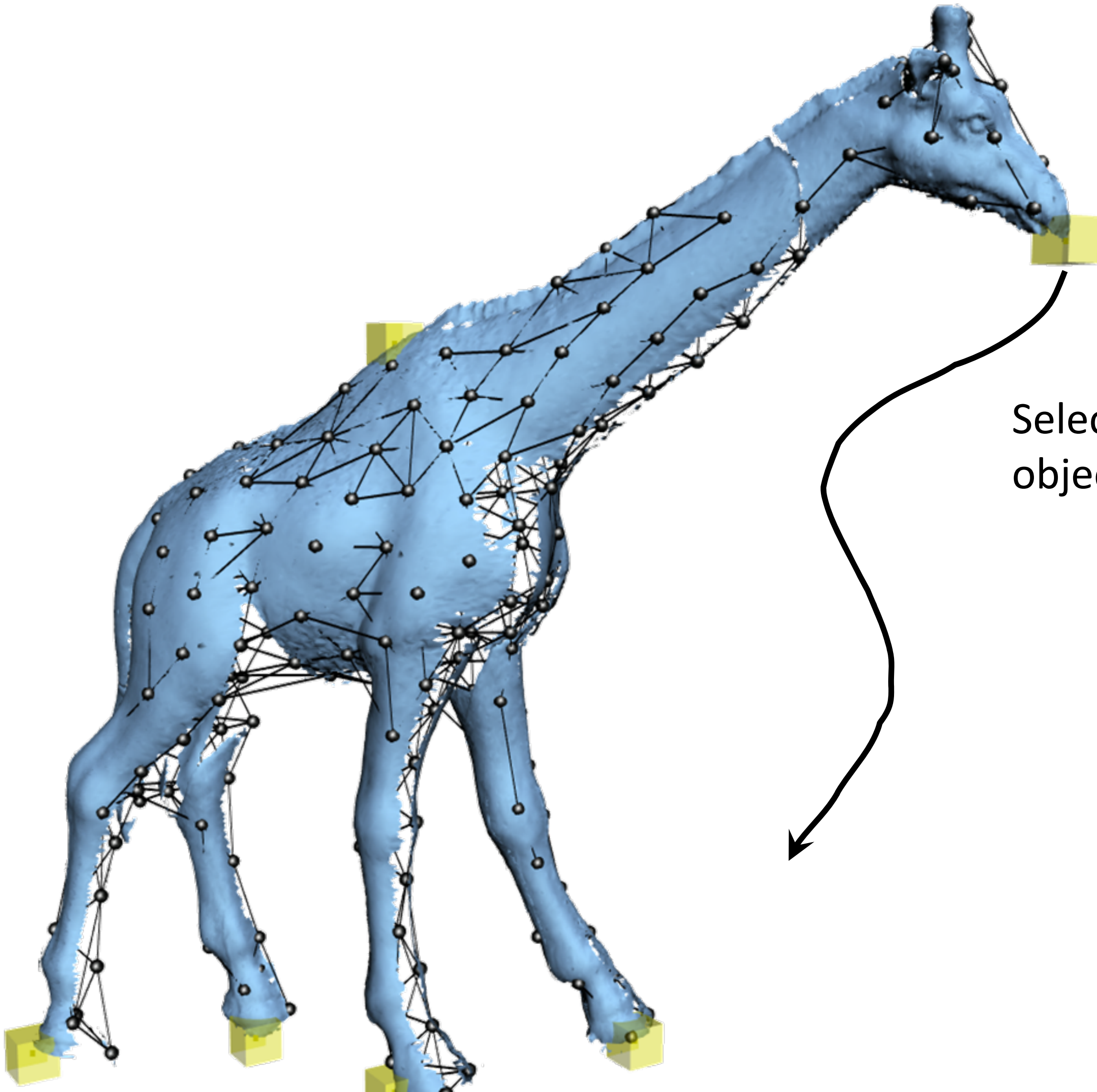
$$\mathbf{x}' = \sum_{j=1}^{m} w_j(\mathbf{x}) \left[\mathrm{R}_j (\mathbf{x} - \mathbf{g}_j) + \mathbf{g}_j + \mathbf{t}_j \right]$$

blending weights

$$w_j(\mathbf{x}) = (1 - \left\| \mathbf{x} - \mathbf{g}_j \right\| / d_{\max})^2$$

Optimization

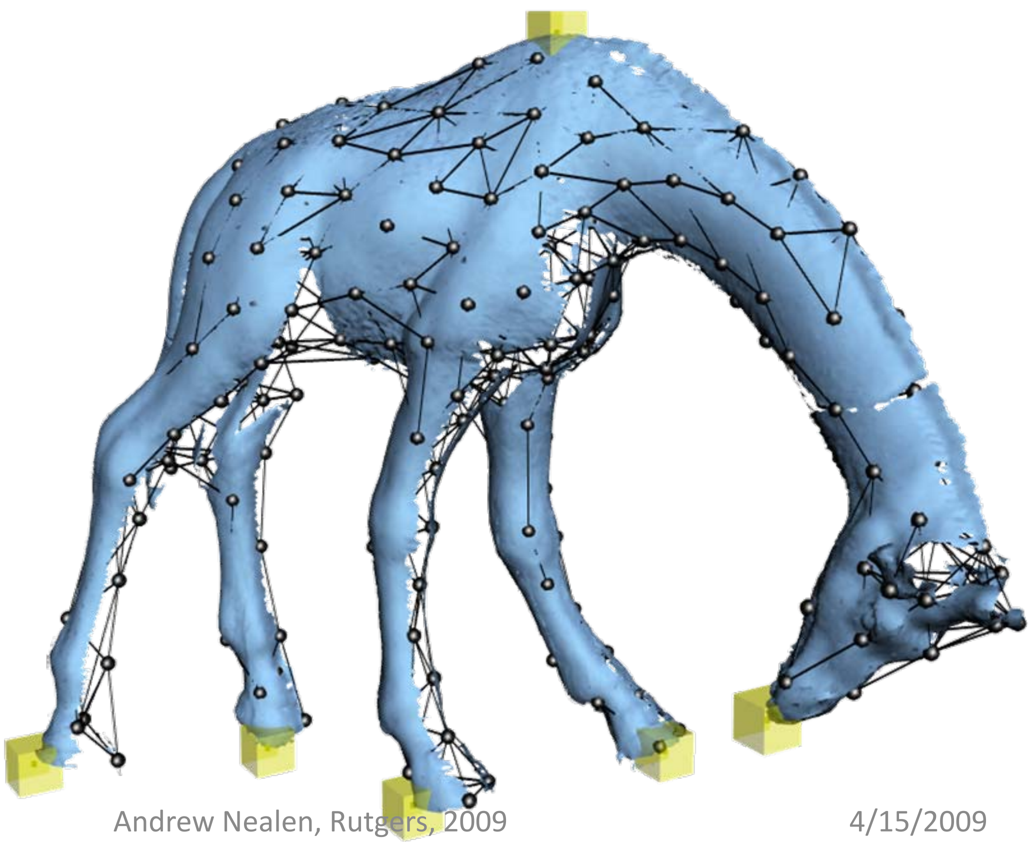
[Sumner et al. 2007]



Select & drag vertices of embedded object.

Optimization

[Sumner et al. 2007]

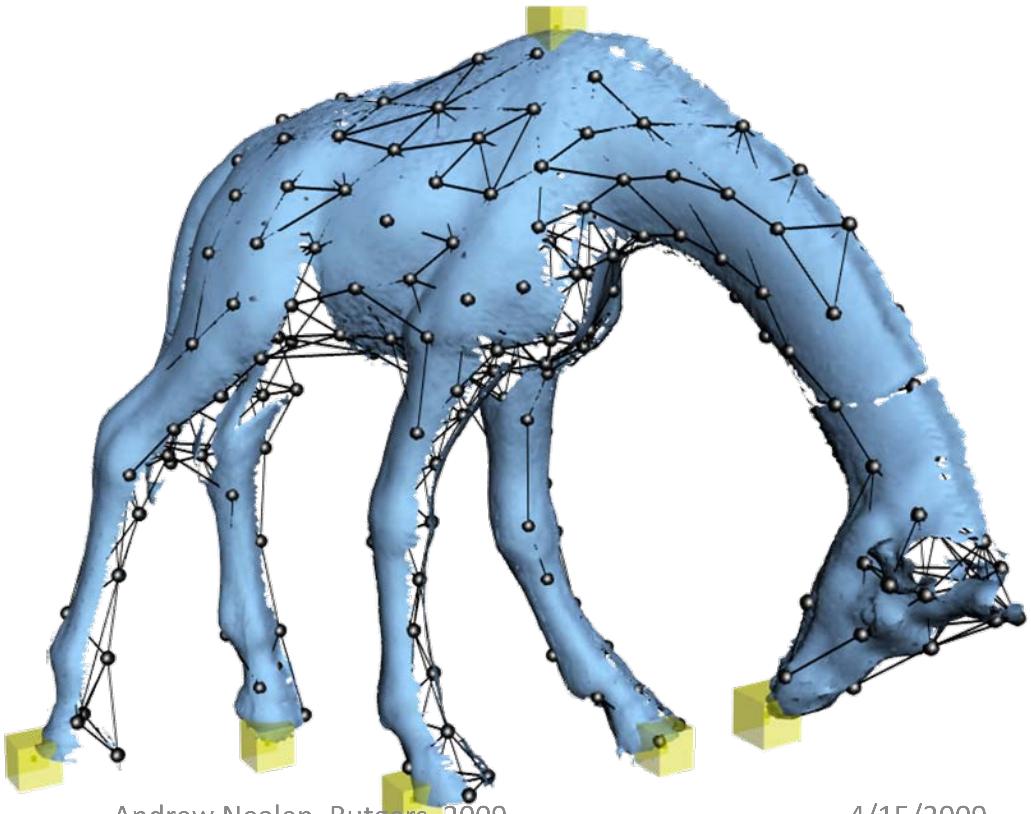


Select & drag vertices of embedded object.

Optimization finds deformation parameters R_j , $\mathbf{t}_j$.

$$\min_{\mathbf{R}_1, \mathbf{t}_1, \ldots, \mathbf{R}_m, \mathbf{t}_m} \quad w_{\text{rot}} \mathrm{E}_{\text{rot}} + w_{\text{reg}} \mathrm{E}_{\text{reg}} + w_{\text{con}} \mathrm{E}_{\text{con}}$$

Graph parameters | Rotation term | Regularization term | Constraint term



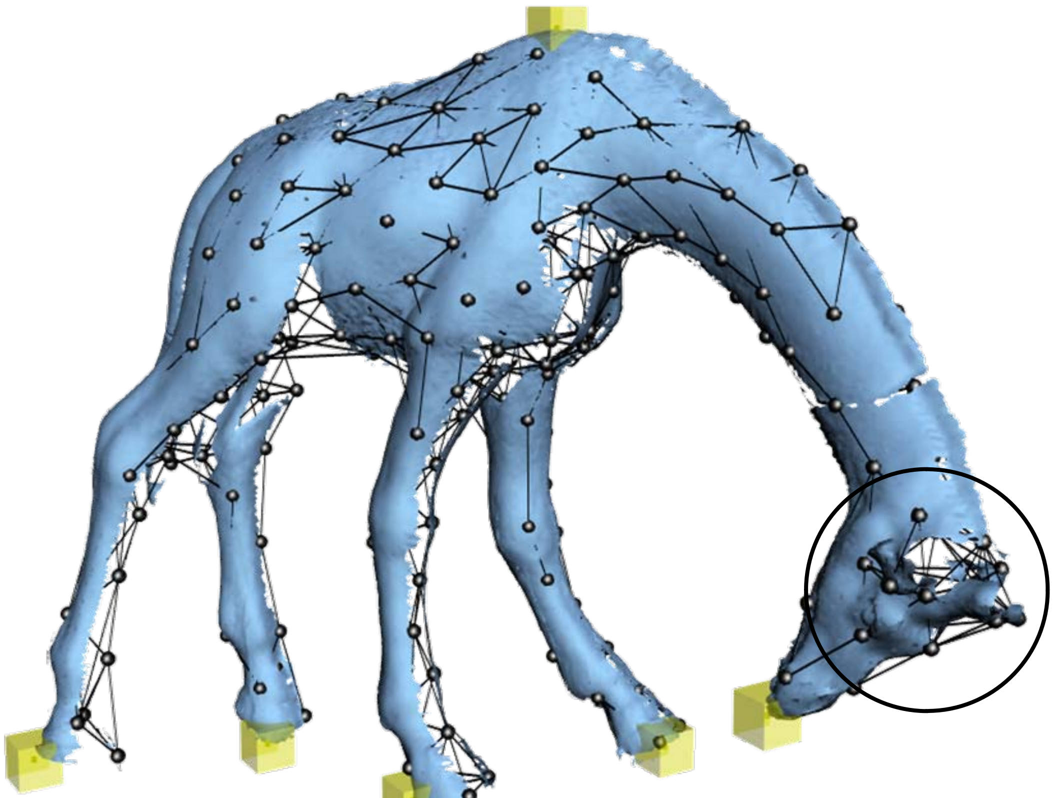
Select & drag vertices of embedded object.

Optimization finds deformation parameters R_j , $\mathbf{t}_j$.

$$\min_{\mathbf{R}_1,\mathbf{t}_1,\ldots,\mathbf{R}_m,\mathbf{t}_m} \quad w_{\text{rot}}\mathrm{E}_{\text{rot}} + w_{\text{reg}}\mathrm{E}_{\text{reg}} + w_{\text{con}}\mathrm{E}_{\text{con}}$$

$$\mathrm{Rot}(\mathbf{R}) = (\mathbf{c}_1\cdot\mathbf{c}_2)^2 + (\mathbf{c}_1\cdot\mathbf{c}_3)^2 + (\mathbf{c}_2\cdot\mathbf{c}_3)^2 + (\mathbf{c}_1\cdot\mathbf{c}_1 - 1)^2 + (\mathbf{c}_2\cdot\mathbf{c}_2 - 1)^2 + (\mathbf{c}_3\cdot\mathbf{c}_3 - 1)^2$$

$$\mathrm{E}_{\text{rot}} = \sum_{j=1}^{m} \mathrm{Rot}(\mathbf{R}_j)$$



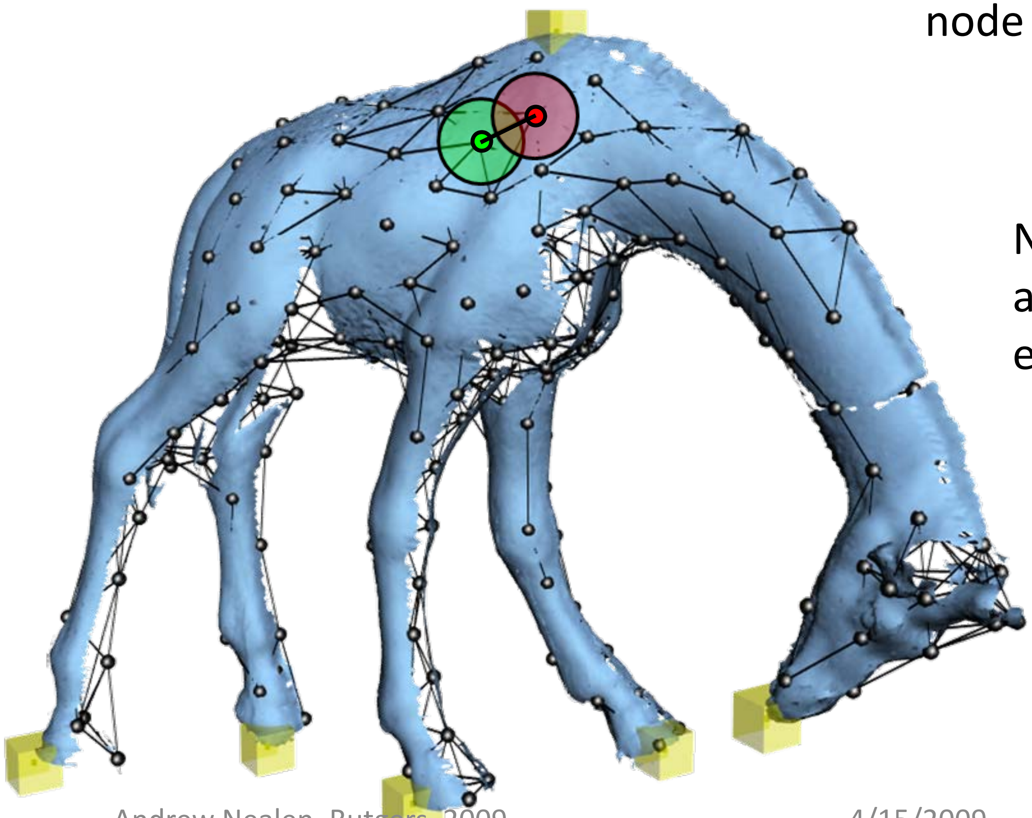
For detail preservation, features should rotate and not scale or skew.

$$\min_{\mathbf{R}_1,\mathbf{t}_1,\ldots,\mathbf{R}_m,\mathbf{t}_m} \quad w_{\mathrm{rot}}\mathrm{E}_{\mathrm{rot}} + w_{\mathrm{reg}}\mathrm{E}_{\mathrm{reg}} + w_{\mathrm{con}}\mathrm{E}_{\mathrm{con}}$$

$$\mathrm{E}_{\mathrm{reg}} = \sum_{j=1}^{m} \sum_{k\in \mathrm{N}(j)} \alpha_{jk} \left\| \mathbf{R}_j(\mathbf{g}_k - \mathbf{g}_j) + \mathbf{g}_j + \mathbf{t}_j - (\mathbf{g}_k + \mathbf{t}_k) \right\|_2^2$$

where node j thinks node k should go

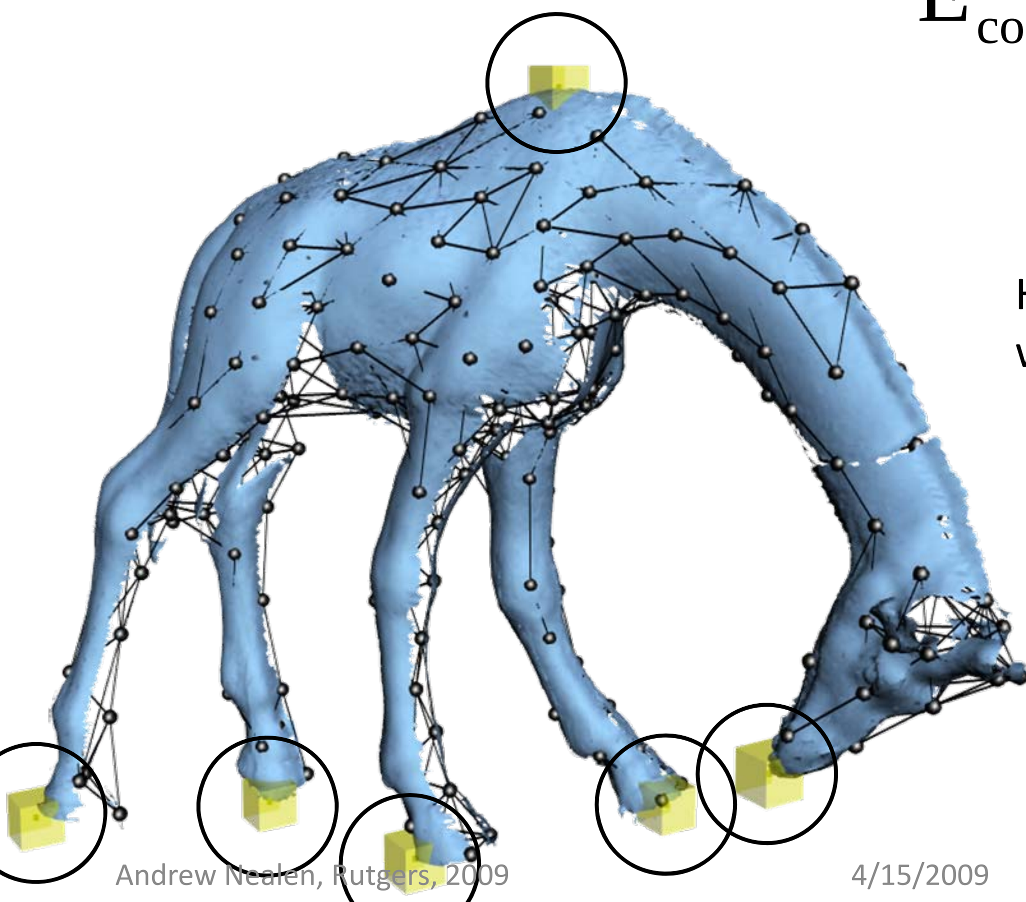
where node k actually goes



Neighboring nodes should agree on where they transform each other.

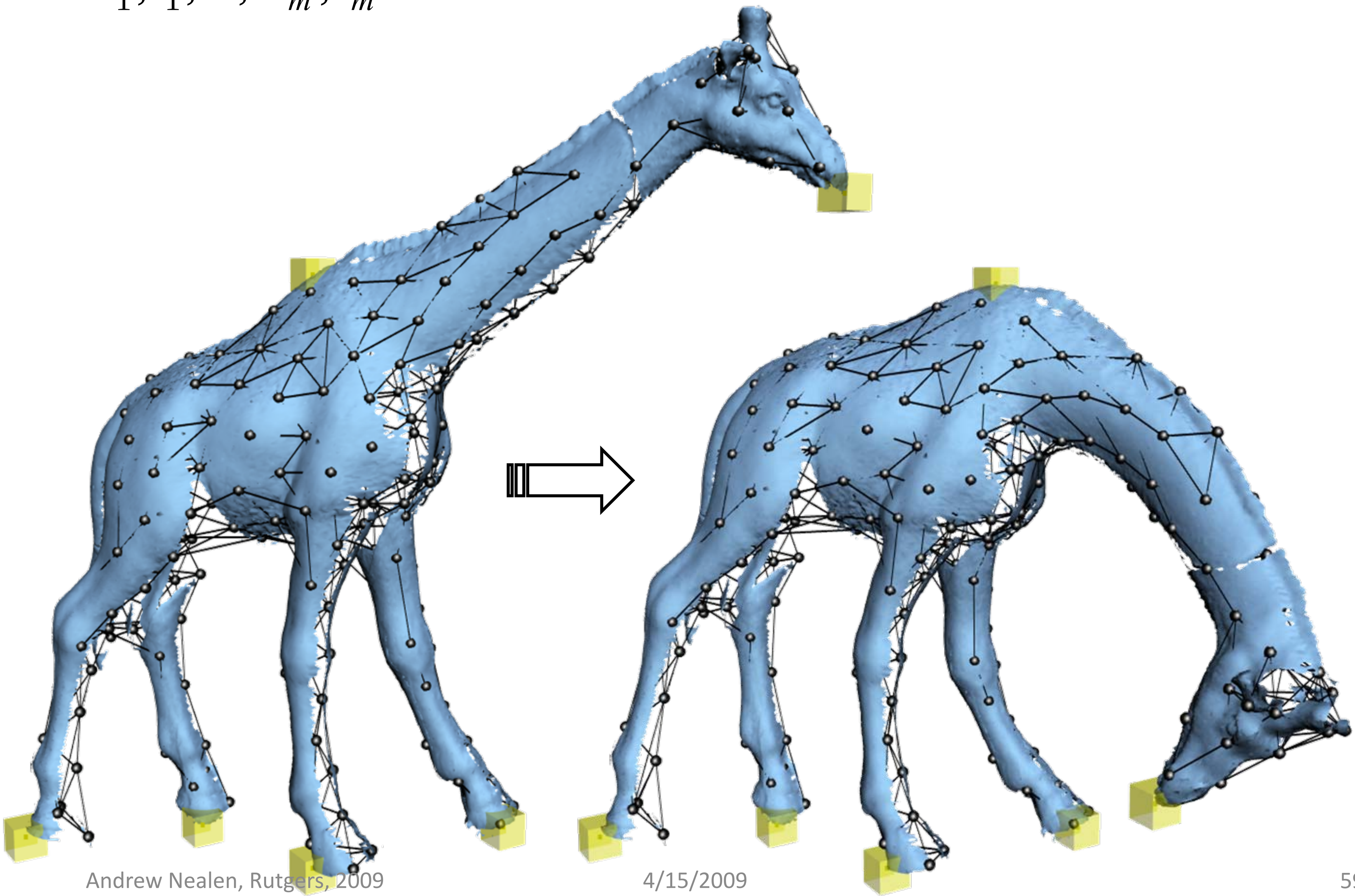
$$\min_{\mathbf{R}_1,\mathbf{t}_1,\ldots,\mathbf{R}_m,\mathbf{t}_m} \quad w_{\text{rot}}\mathrm{E}_{\text{rot}} + w_{\text{reg}}\mathrm{E}_{\text{reg}} + w_{\text{con}}\mathrm{E}_{\text{con}}$$

$$\mathrm{E}_{\text{con}} = \sum_{l=1}^{p} \left\| \tilde{\mathbf{v}}_{\text{index}(l)} - \mathbf{q}_l \right\|_2^2$$



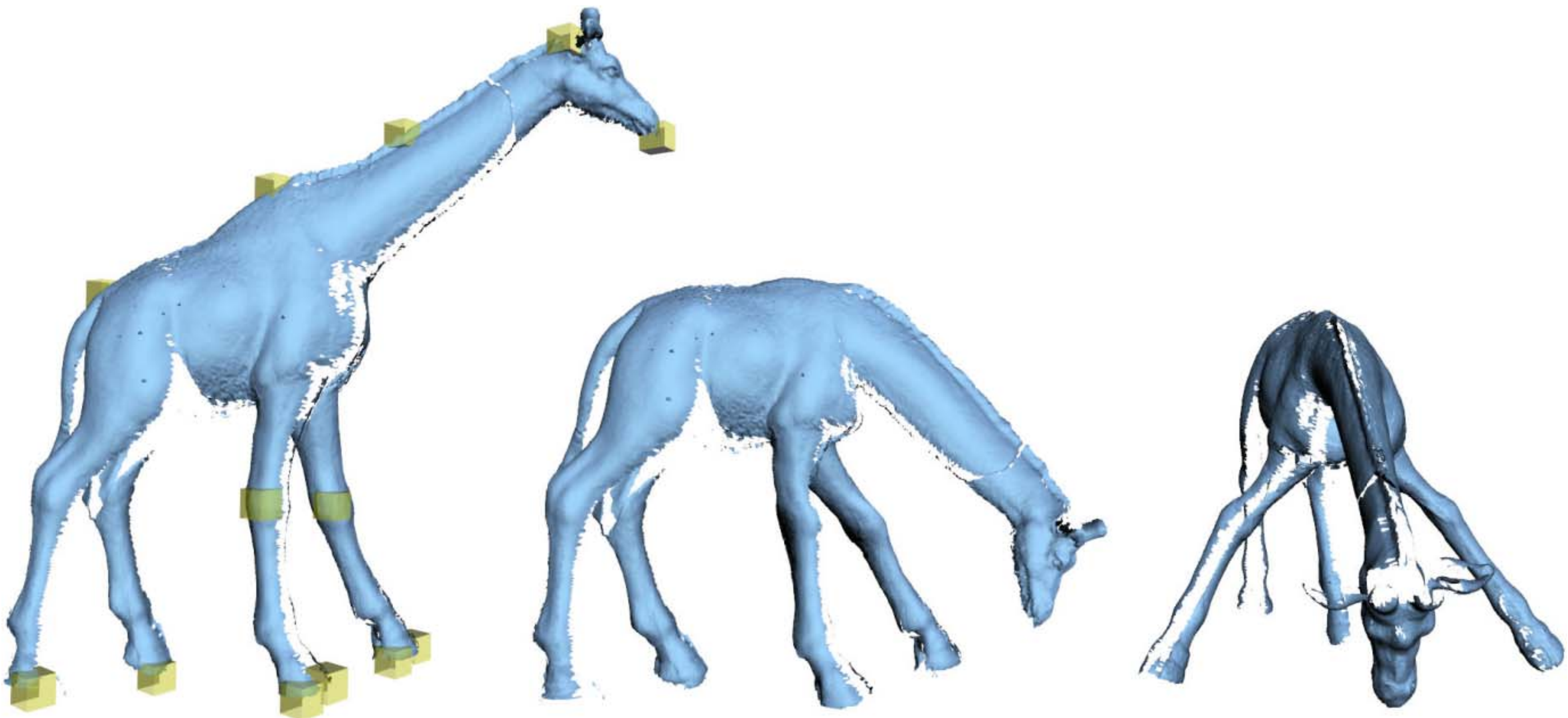
Handle vertices should go where the user puts them.

$$\min_{\mathbf{R}_1,\mathbf{t}_1,\ldots,\mathbf{R}_m,\mathbf{t}_m} \; w_{\text{rot}}\mathrm{E}_{\text{rot}} + w_{\text{reg}}\mathrm{E}_{\text{reg}} + w_{\text{con}}\mathrm{E}_{\text{con}}$$



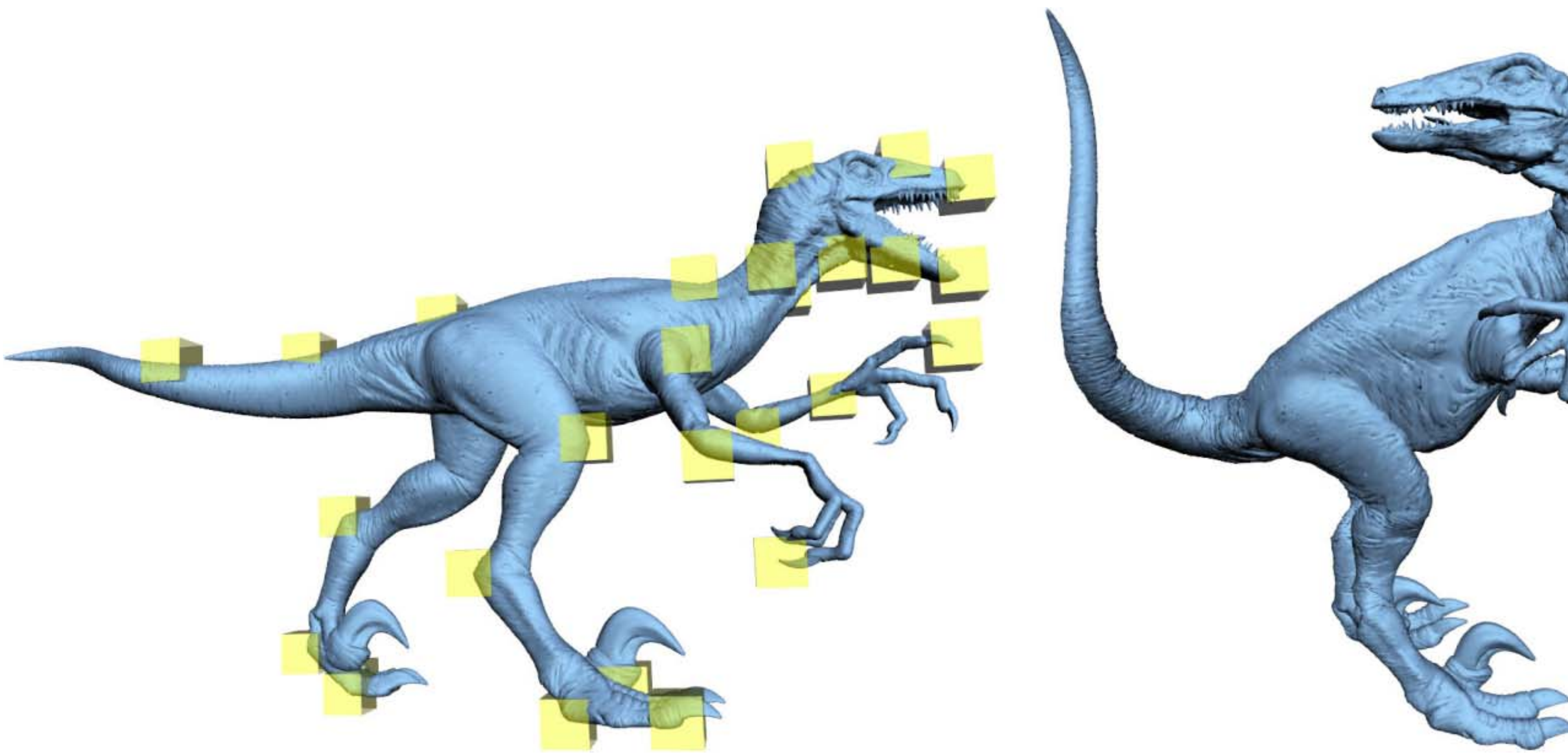
Results: Polygon Soup

[Sumner et al. 2007]



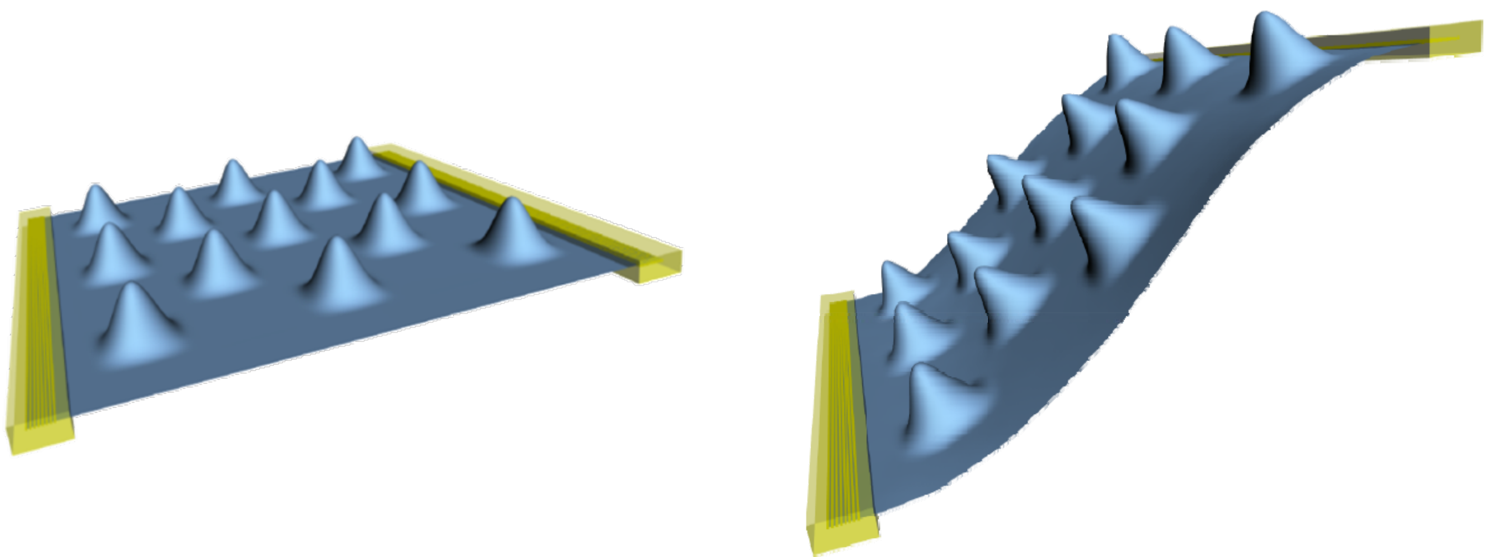
Results: Giant Mesh

[Sumner et al. 2007]



Results: Detail Preservation

[Sumner et al. 2007]



Demo

Discussion

- Decoupling of deformation complexity and model complexity
- Nonlinear energy optimization – results comparable to surface-based approaches