Eurographics 2009

Interactive Shape Modeling and Deformation

T3: Half-Day Tutorial

Introduction, organization

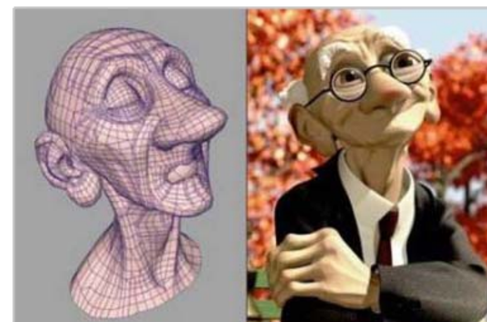
Speakers

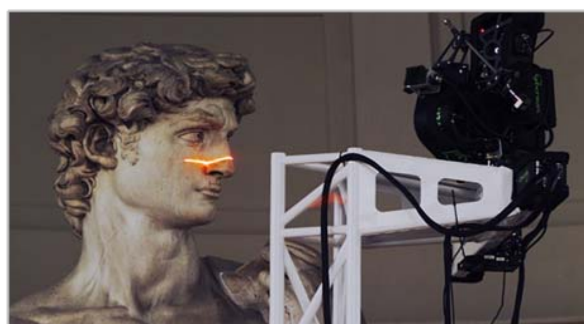
- **Olga Sorkine**
  Media Research Lab, VLG
  Courant Institute, New York University

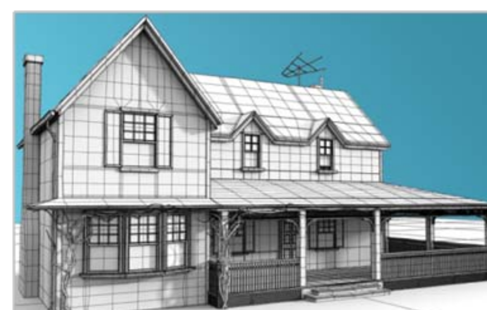
- **Mario Botsch**
  Graphics & Geometry Group
  Bielefeld University

Shapes and Deformations

- Manually modeled and scanned shape data
- Continuous and discrete shape representations

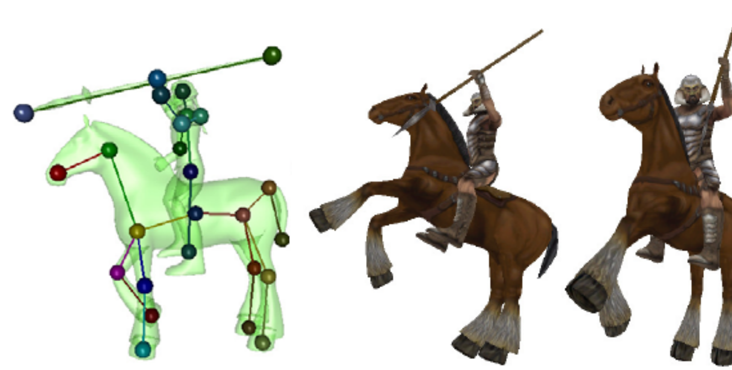


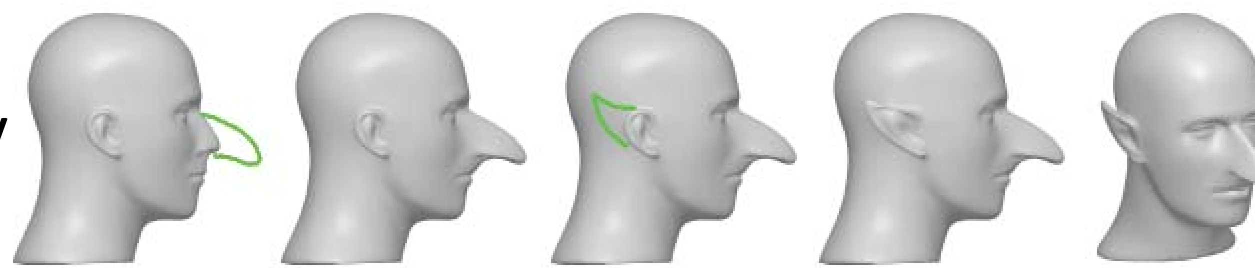




Shapes and Deformations

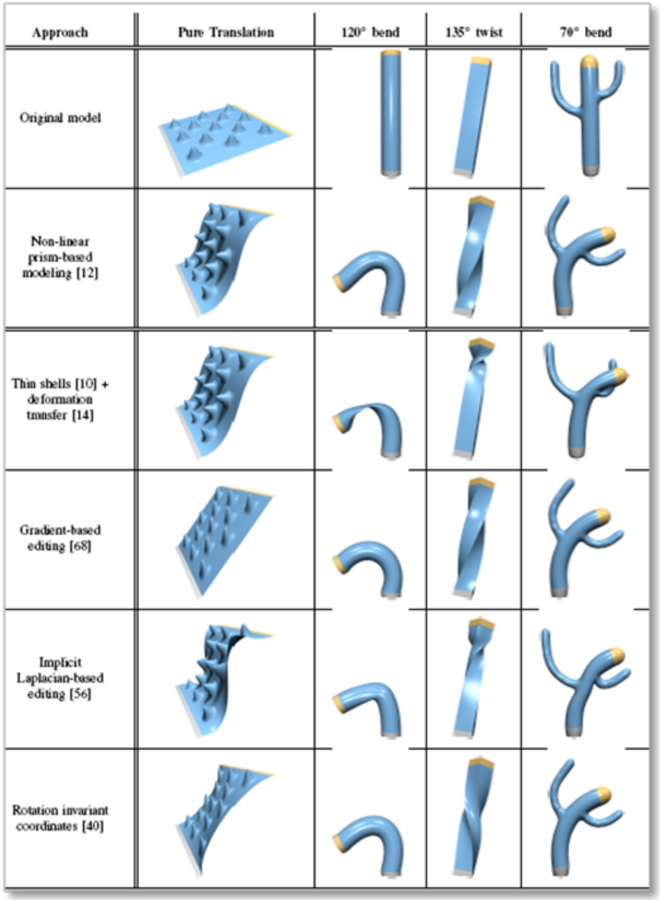
- Why deformations?
  - Sculpting, customization
  - Character posing, animation
- Criteria?
  - Intuitive behavior and interface
  - Interactivity





Tutorial Goals

- Present recent research in shape editing
- Discuss practical considerations
  - Flexibility
  - Numerical issues
  - Admissible interfaces
- Comparison, tradeoffs



Schedule

09:00 – 09:10	Intro (O)
09:10 – 09:25	Shape representations, differential geometry primer (O)
09:25 – 10:05	Linear surface-based deformations (M)
10:05 – 10:30	Linear space deformations (O)
10:30 – 11:00	Break

Schedule (cont'd)

11:00 – 11:10	Summary of linear methods (M)
11:10 – 11:40	Nonlinear surface-based deformations (M)
11:40 – 12:20	Nonlinear space deformations (O)
12:20 – 12:30	Wrap-up (O+M)

Eurographics 2009

Interactive Shape Modeling and Deformation

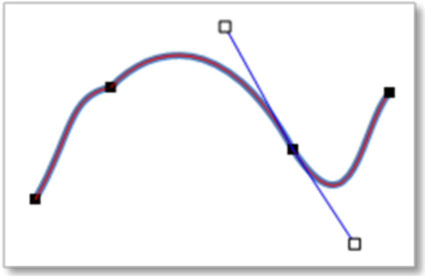
T3: Half-Day Tutorial

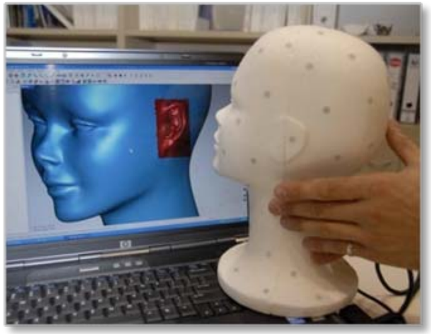
Shape Representations
Differential Geometry Recap

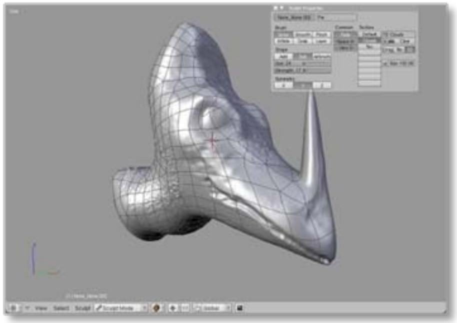
Continuous/analytical surfaces

- Tensor product surfaces (e.g. NURBS)
- Subdivision surfaces

- "Editability" is inherent to the representation



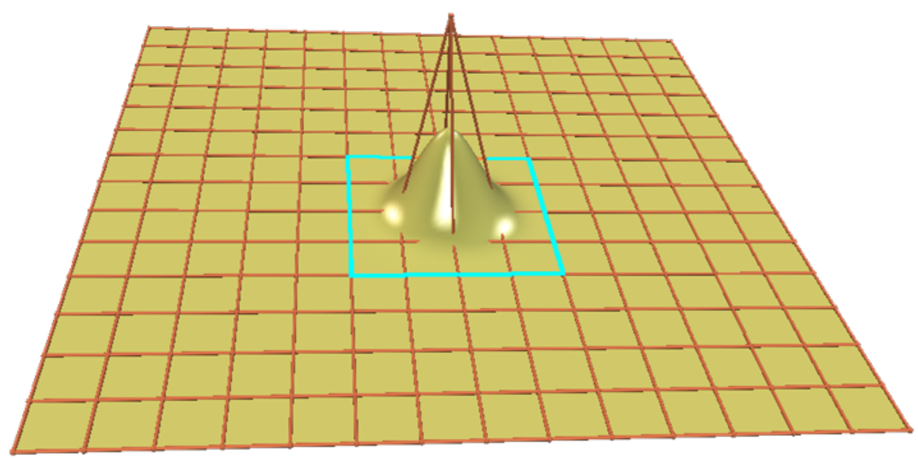




Spline Surfaces

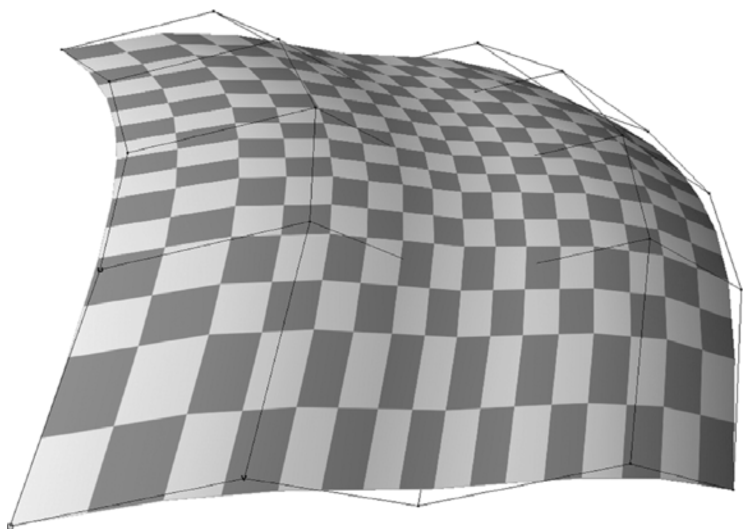
- Tensor product surfaces (“curves of curves”)
  - Rectangular grid of control points

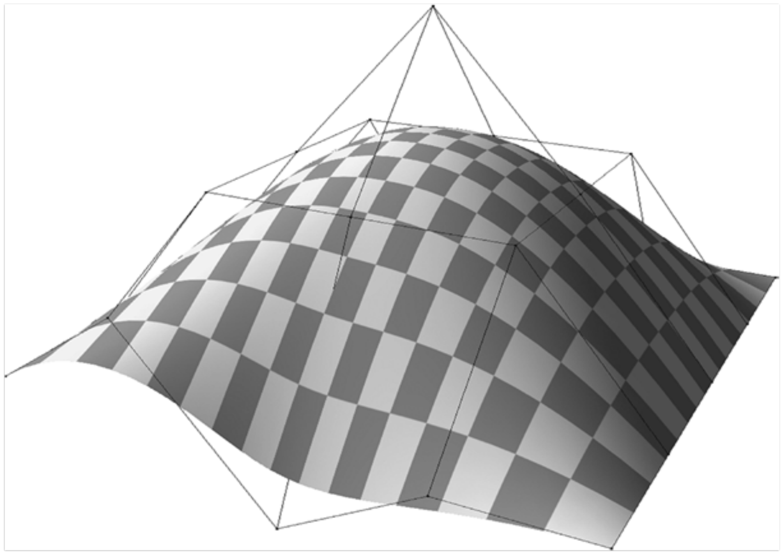
$$\mathbf{p}(u,v) = \sum_{i=0}^{k} \sum_{j=0}^{l} \mathbf{p}_{ij}\, N_i^n(u) N_j^n(v)$$



Spline Surfaces

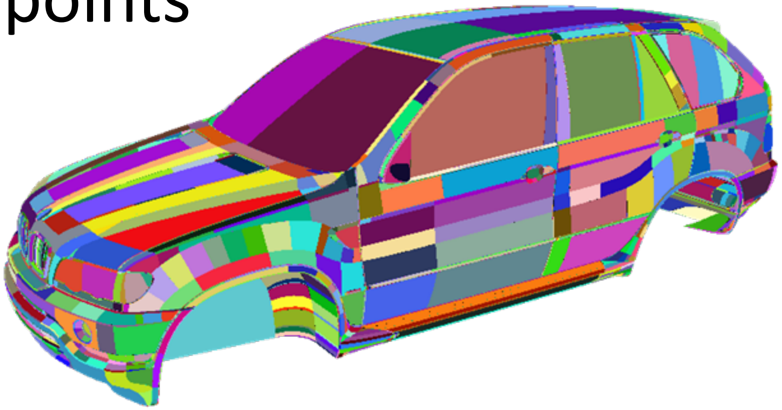
- Tensor product surfaces (“curves of curves”)
  - Rectangular grid of control points
  - Rectangular surface patch





Spline Surfaces

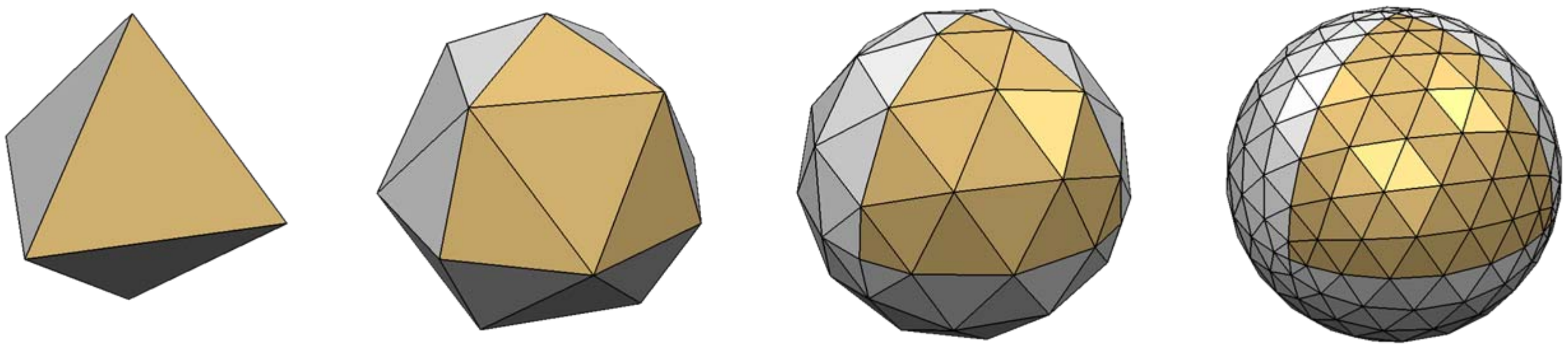
- Tensor product surfaces ("curves of curves")
  - Rectangular grid of control points
  - Rectangular surface patch



- Problems:
  - Many patches for complex models
  - Smoothness across patch boundaries
  - Trimming for non-rectangular patches

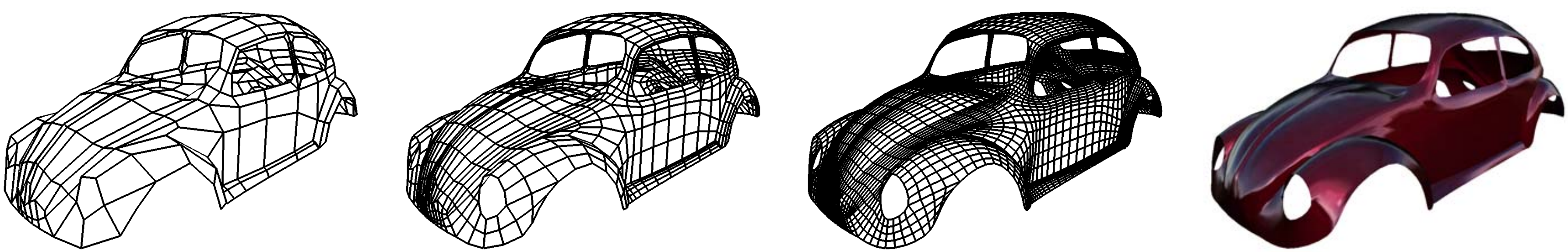
Subdivision Surfaces

- Generalization of spline curves / surfaces
  - Arbitrary control meshes
  - Successive refinement (subdivision)
  - Converges to smooth limit surface
  - Connection between splines and meshes



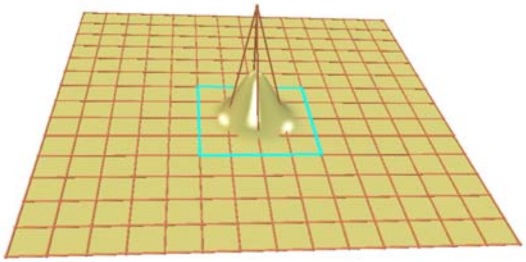
Subdivision Surfaces

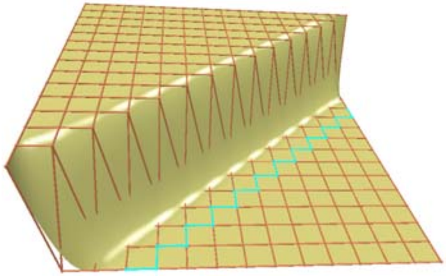
- Generalization of spline curves / surfaces
  - Arbitrary control meshes
  - Successive refinement (subdivision)
  - Converges to smooth limit surface
  - Connection between splines and meshes

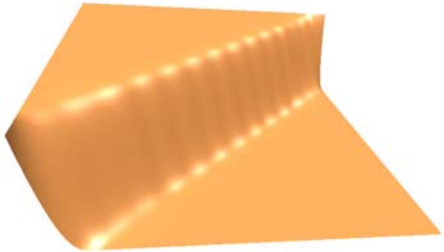


Spline & Subdivision Surfaces

- Basis functions are smooth bumps
  - Fixed support
  - Fixed control grid
- Bound to control points
  - Initial patch layout is crucial
  - Requires experts!
- Decouple deformation from surface representation!

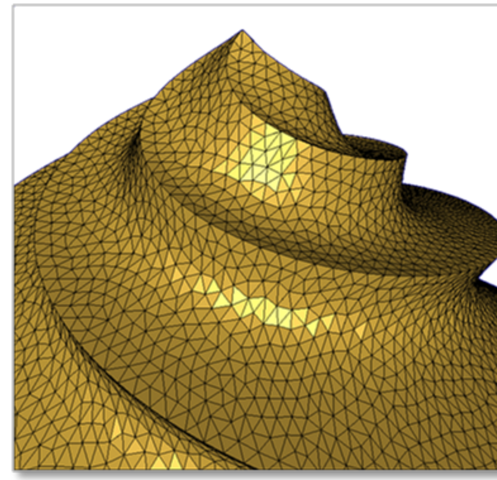






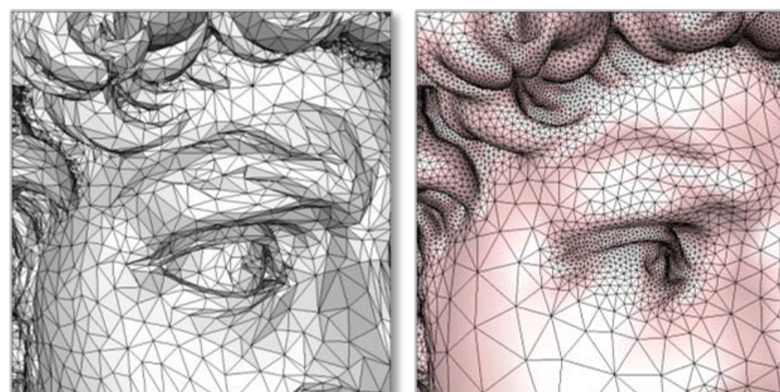
Discrete Surfaces: Point Sets, Meshes

- Flexible
- Suitable for highly detailed scanned data



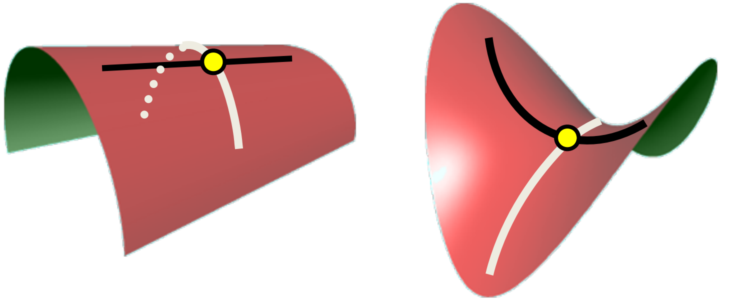
- No analytic surface
- No inherent “editability”

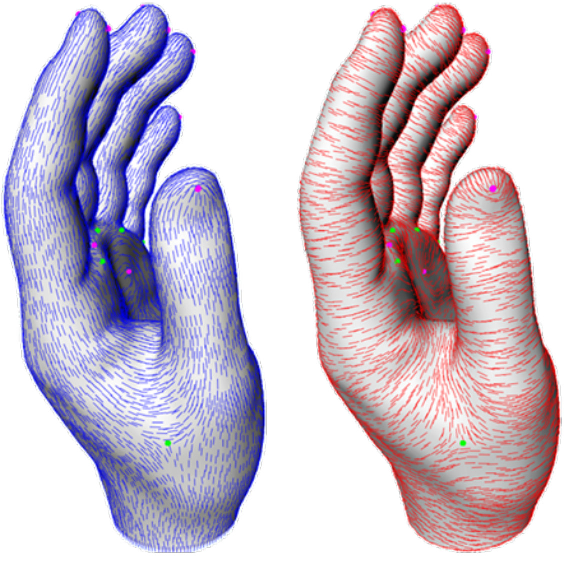
Mesh editing



Differential Geometry

- Tool to analyze shape
- Key notions:
  - Tangents and normals
  - Curvatures
  - Laplace-Beltrami operator





Continuous Case – Parametric

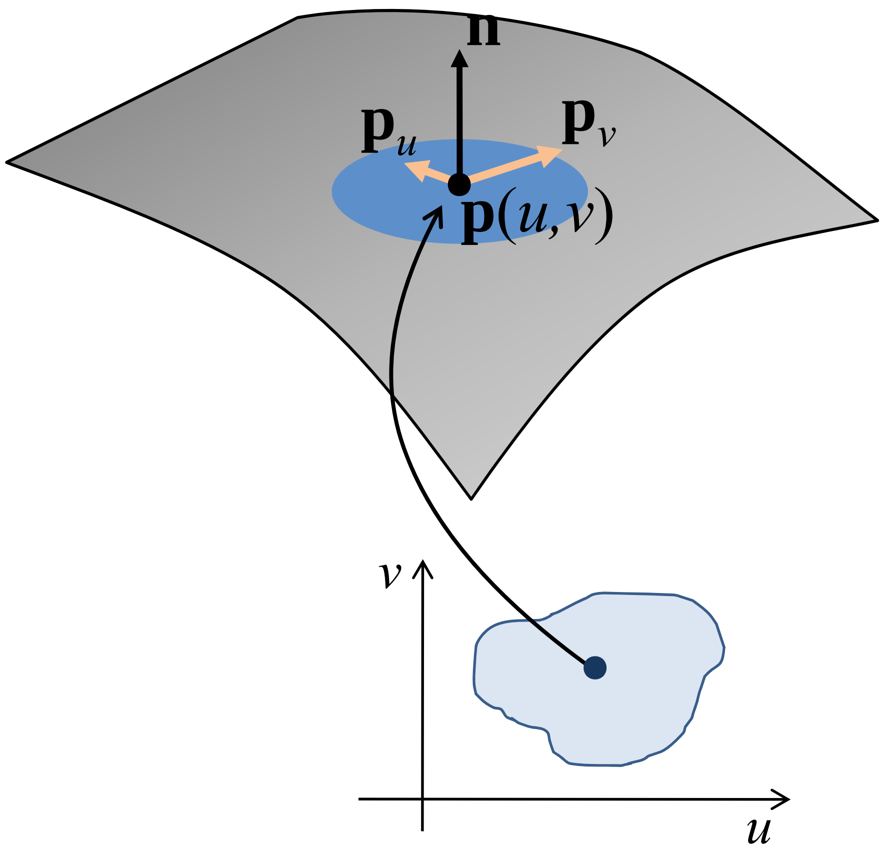
- Local parameterization:

$$\mathbf{p}(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix}, \quad (u,v) \in D \subset \Re^2$$

- Tangent plane at point $\mathbf{p}(u,v)$ is spanned by

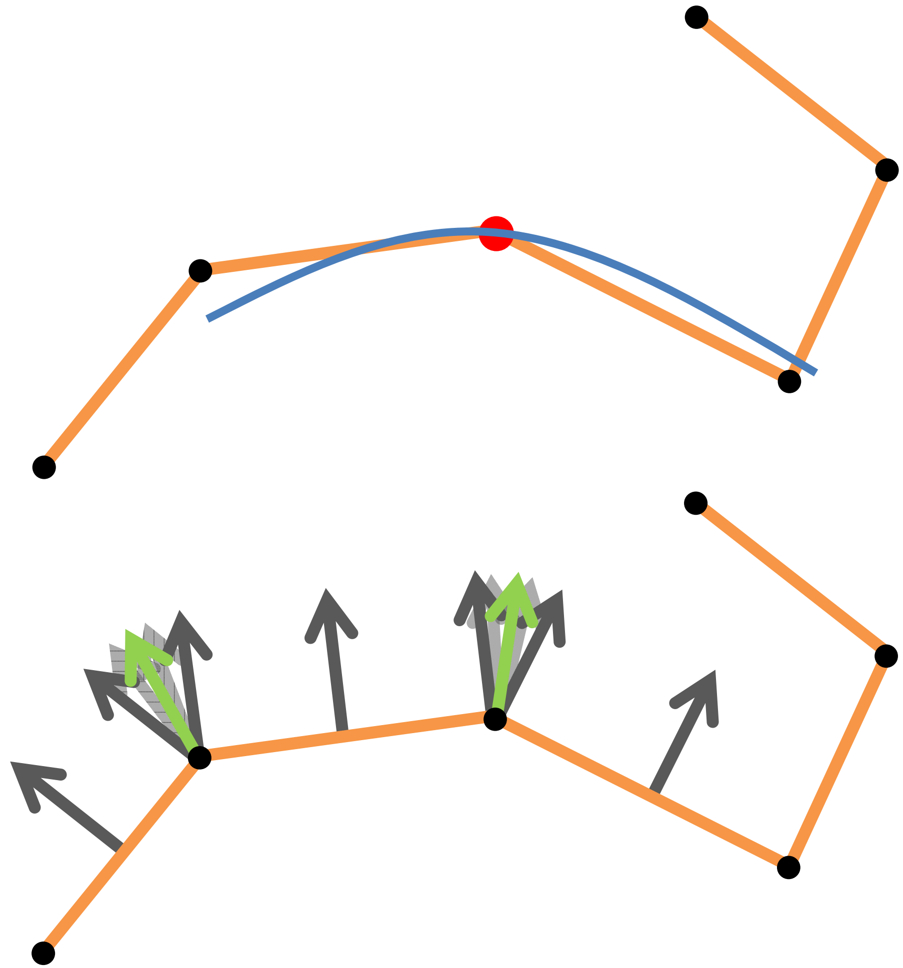
$$\mathbf{p}_u = \frac{\partial \mathbf{p}(u,v)}{\partial u}, \qquad \mathbf{p}_v = \frac{\partial \mathbf{p}(u,v)}{\partial v}$$

- Normal: $\mathbf{n}(u,v) = \dfrac{\mathbf{p}_u \times \mathbf{p}_v}{\left\| \mathbf{p}_u \times \mathbf{p}_v \right\|}$

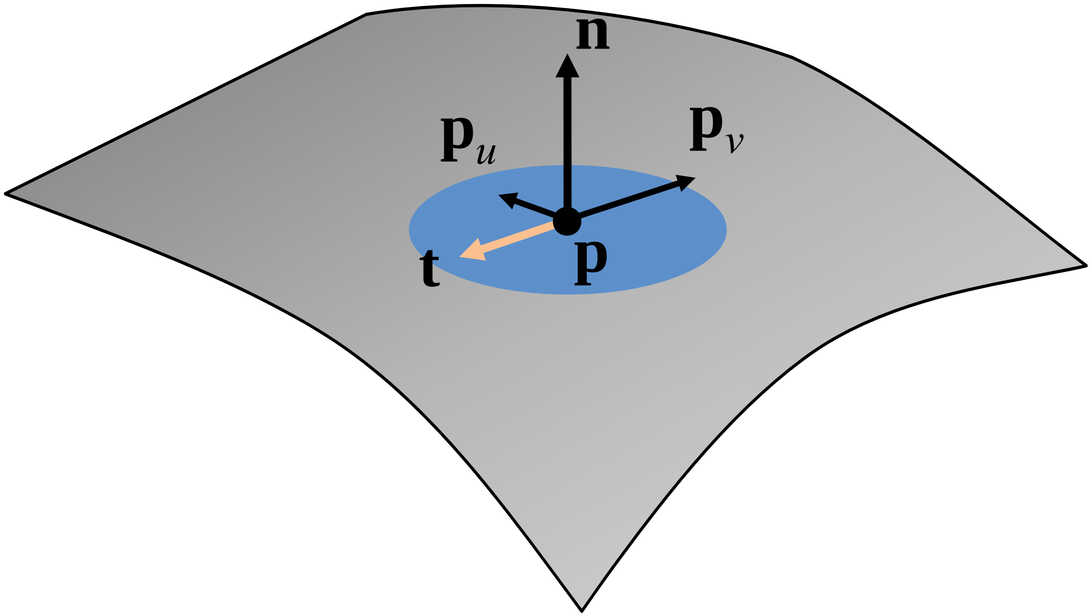


Discrete Case – Piecewise Linear

- No derivatives!
- Strategy 1: locally fit an analytic patch
  - Expensive
- Strategy 2: generalize definitions to discrete case
  - Fast
  - Start from intrinsic notions (non-parametric)



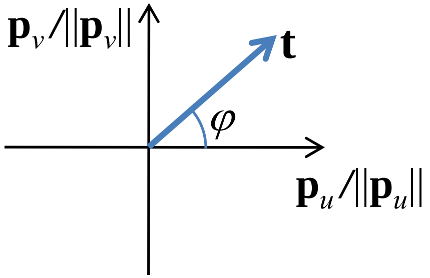
Normal Curvature



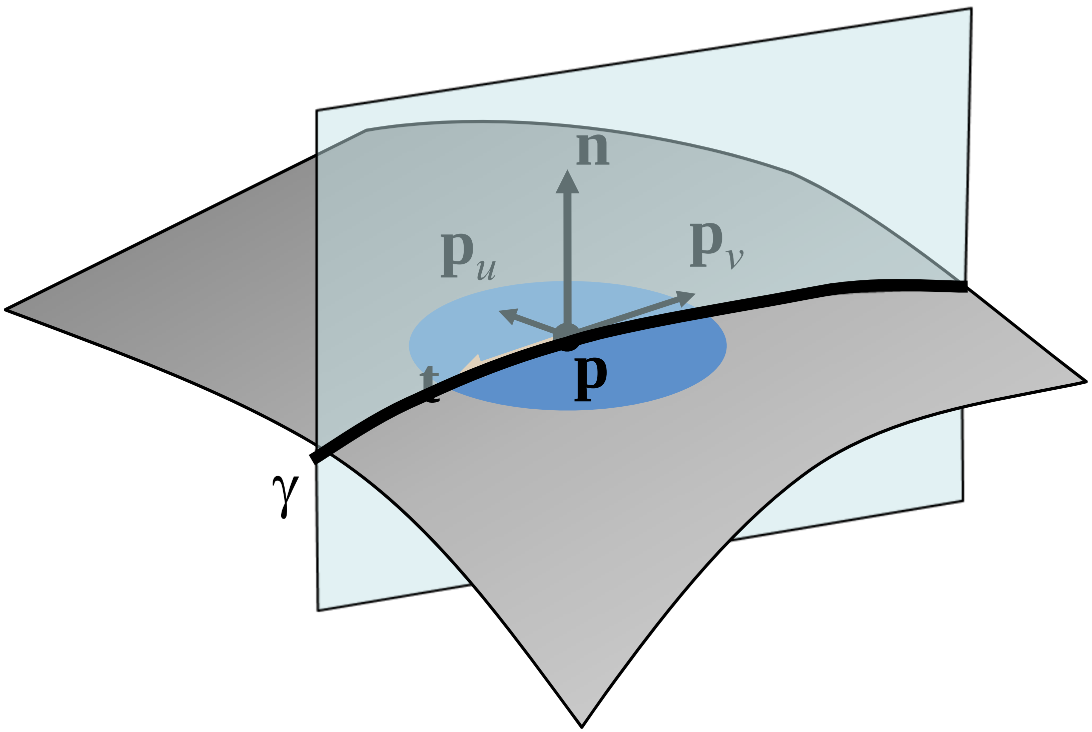
$$\mathbf{n} = \frac{\mathbf{p}_u \times \mathbf{p}_v}{\left\| \mathbf{p}_u \times \mathbf{p}_v \right\|}$$

Direction $\mathbf{t}$ in the tangent plane:

$$\mathbf{t} = \cos\varphi \frac{\mathbf{p}_u}{\left\| \mathbf{p}_u \right\|} + \sin\varphi \frac{\mathbf{p}_v}{\left\| \mathbf{p}_v \right\|}$$



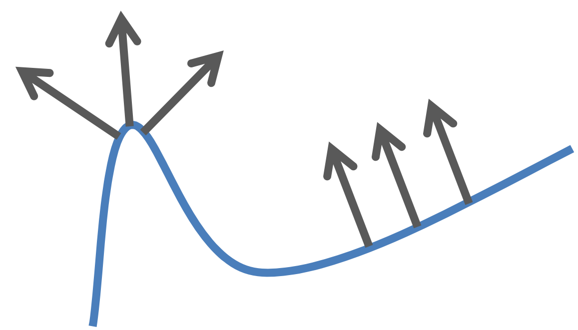
Normal Curvature



The curve γ is the intersection of the surface with the plane through $\mathbf{n}$ and $\mathbf{t}$.

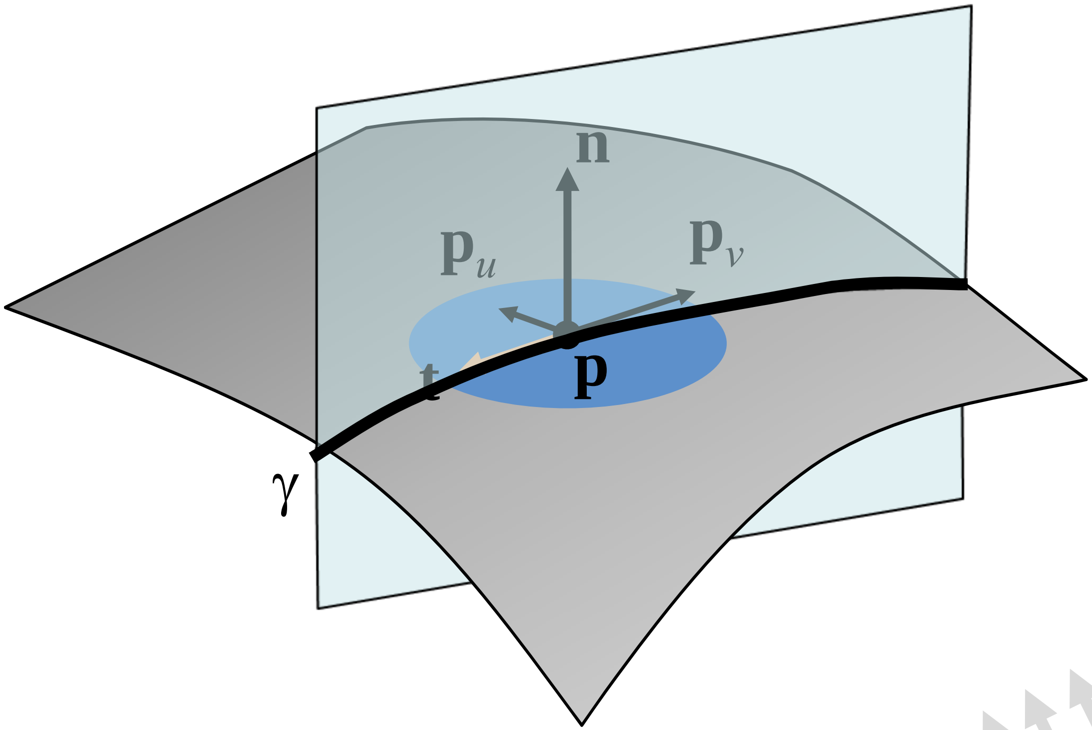
Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(\mathbf{p}))$$



Curvature on a curve: the rate of change in normal

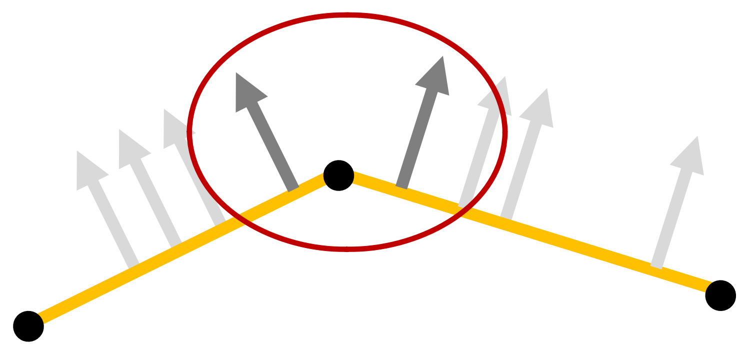
Normal Curvature



The curve γ is the intersection of the surface with the plane through $\mathbf{n}$ and $\mathbf{t}$.

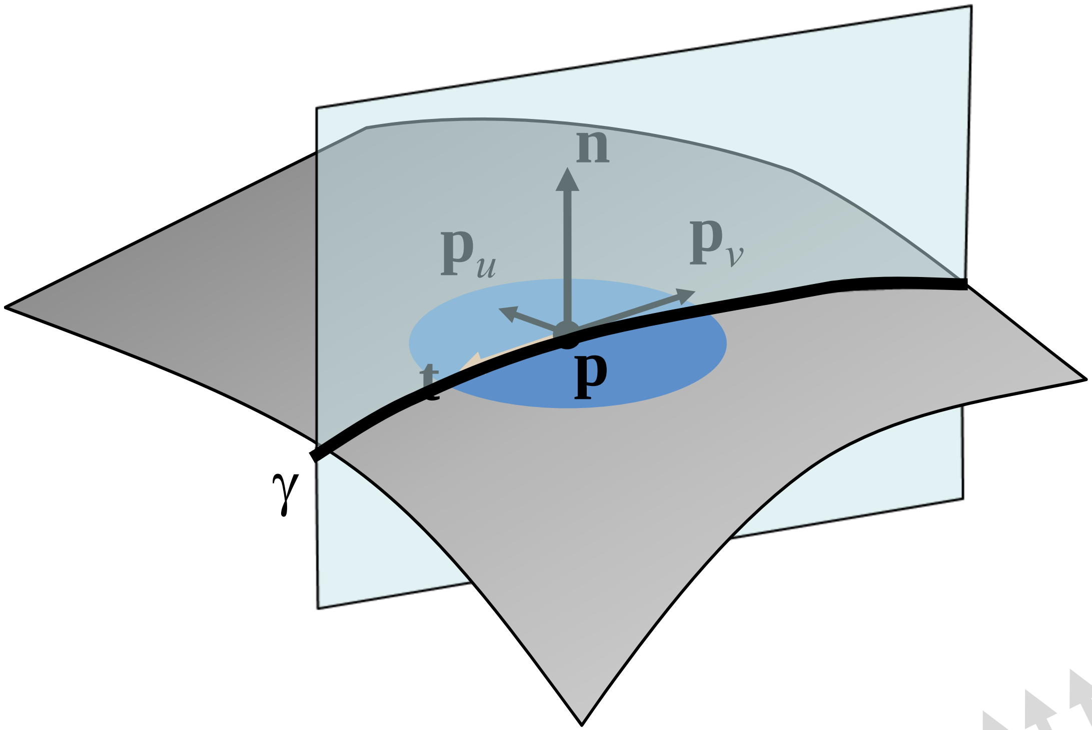
Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(\mathbf{p}))$$



Curvature on a curve: the rate of change in normal

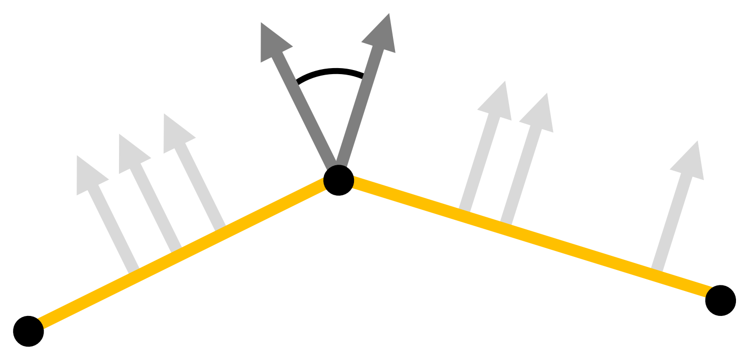
Normal Curvature



The curve γ is the intersection of the surface with the plane through $\mathbf{n}$ and $\mathbf{t}$.

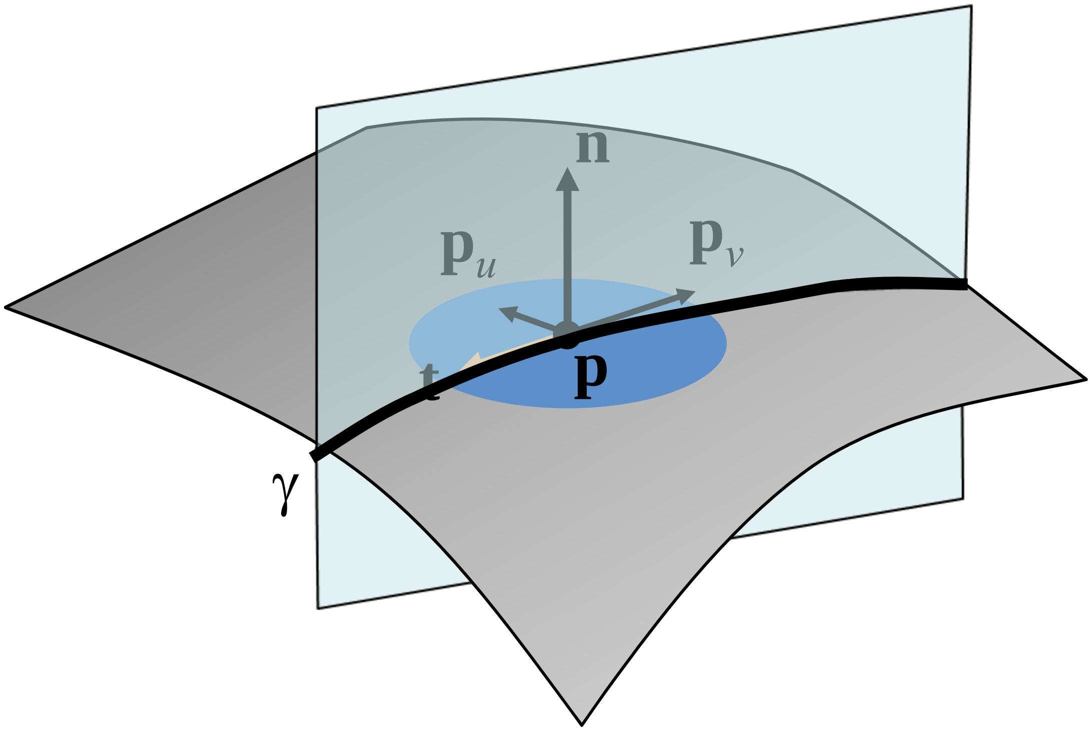
Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(\mathbf{p}))$$



Curvature on a curve: the rate of change in normal

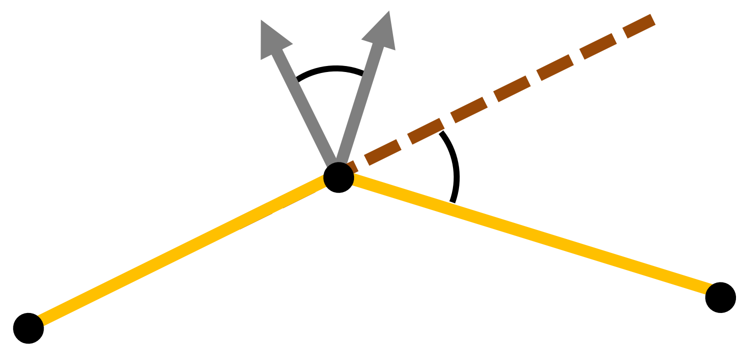
Normal Curvature



The curve γ is the intersection of the surface with the plane through $\mathbf{n}$ and $\mathbf{t}$.

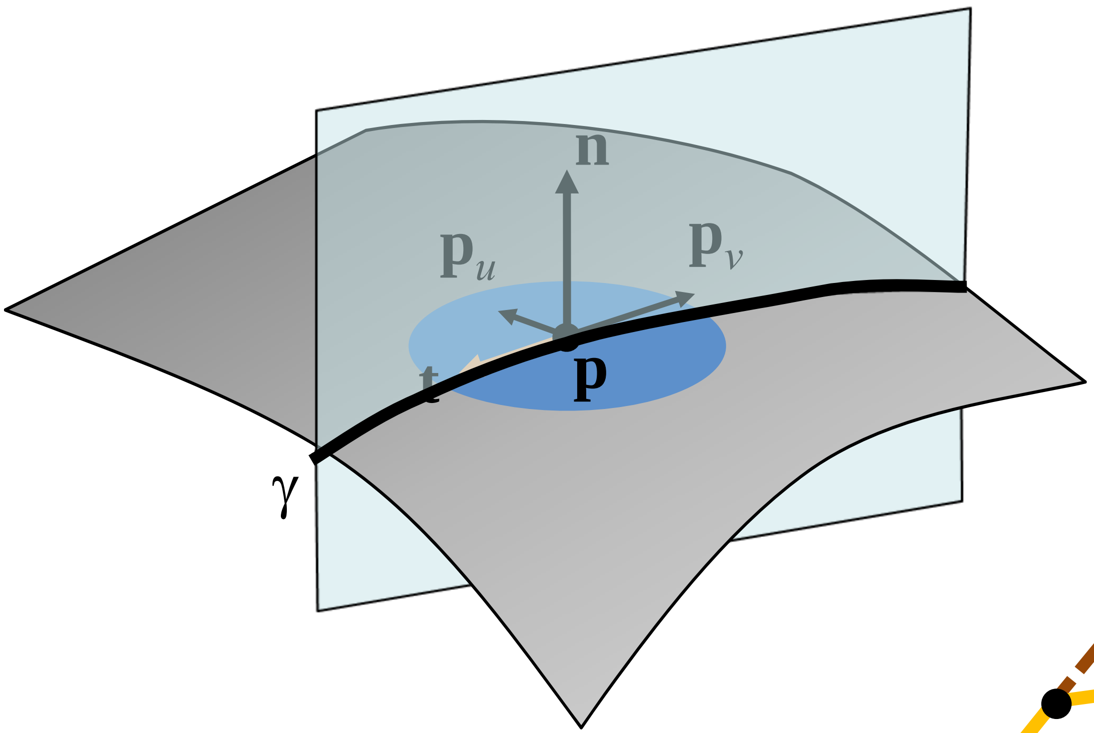
Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(\mathbf{p}))$$



Curvature on a curve: the rate of change in normal

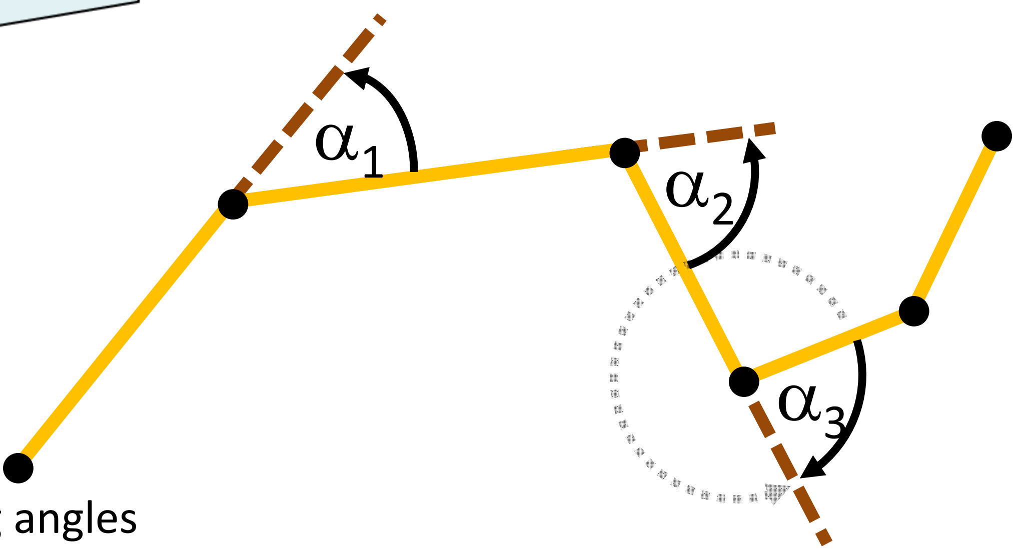
Normal Curvature



The curve γ is the intersection of the surface with the plane through $\mathbf{n}$ and $\mathbf{t}$.

Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(\mathbf{p}))$$



Discrete curvature: turning angles

Surface Curvatures

- Principal curvatures
  - Maximal curvature $\kappa_1 = \kappa_{\max} = \max_{\varphi} \kappa_n(\varphi)$
  - Minimal curvature $\kappa_2 = \kappa_{\min} = \min_{\varphi} \kappa_n(\varphi)$
- Mean curvature

$$H = \frac{\kappa_1 + \kappa_2}{2} = \frac{1}{2\pi} \int_0^{2\pi} \kappa_n(\varphi)\, \mathrm{d}\varphi$$

- Gaussian curvature

$$K = \kappa_1 \cdot \kappa_2$$

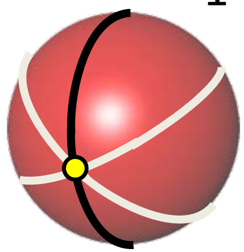
Classification

Local surface shape by curvatures

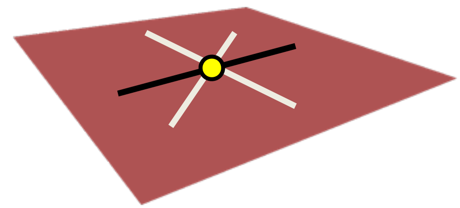
$K > 0,\ \kappa_1 = \kappa_2$

$K = 0$

Isotropic:
all directions are principal directions



spherical (umbilical)



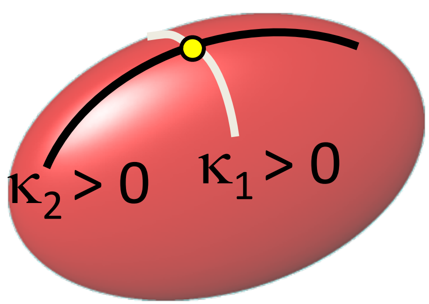
planar

$K > 0$

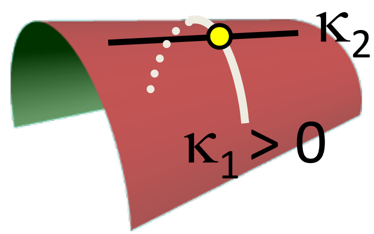
$K = 0$

$K < 0$

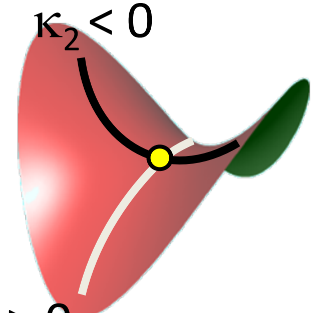
Anisotropic:
2 distinct principal directions



elliptic



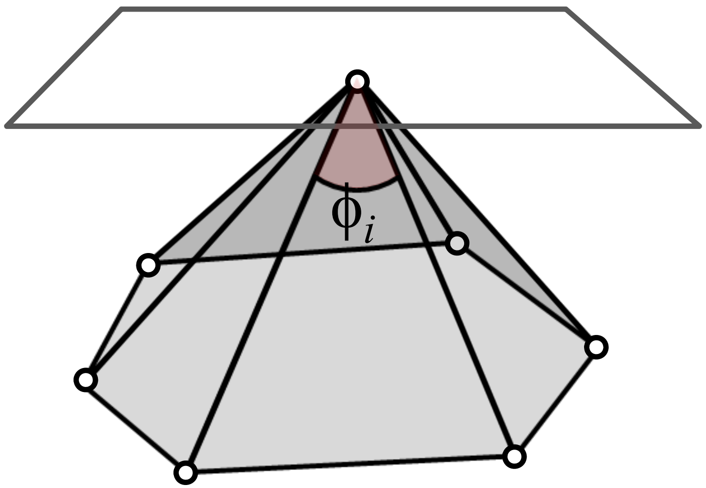
parabolic

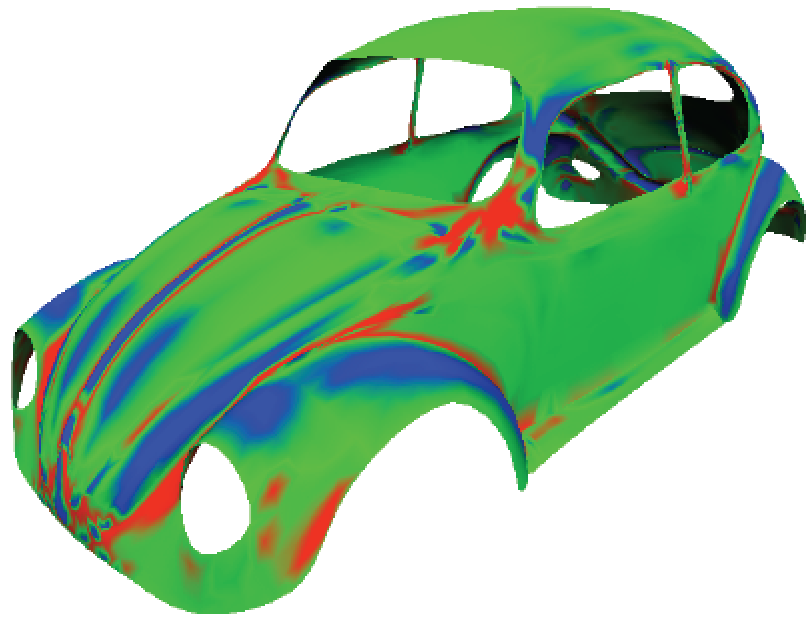


hyperbolic

Discrete Gaussian Curvature

- Angle deficit





$$K = 2\pi - \sum_{i=1}^{|N(\mathbf{v})|} \phi_i$$

Mean Curvature

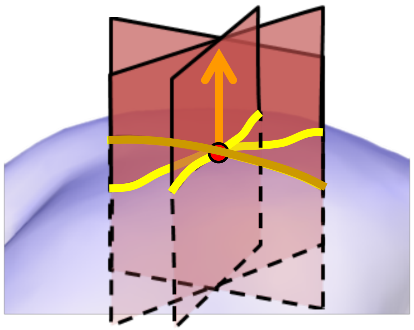
$$H = \frac{1}{2\pi}\int_0^{2\pi} \kappa(\varphi)\,\mathrm{d}\varphi$$

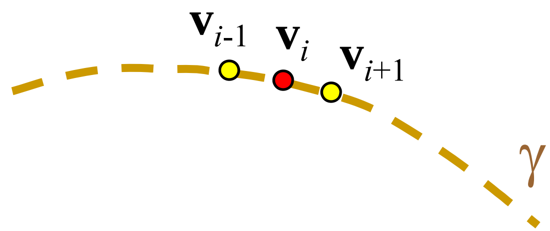
Can define through the Laplace-Beltrami operator

$$\boxed{\Delta_M \mathbf{p} = -H\mathbf{n}}$$

Discrete Mean Curvature

- Intuition:





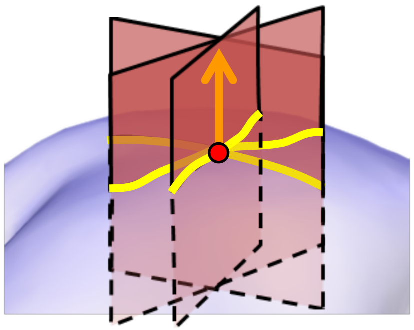
$$H = \frac{1}{2\pi}\int_0^{2\pi} \kappa(\varphi)\,\mathrm{d}\varphi$$

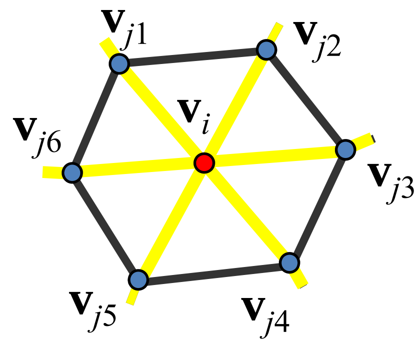
$$\kappa\mathbf{n} = \gamma''$$

$$\gamma'' \approx \frac{1}{t}\left(\left(\mathbf{v}_i - \mathbf{v}_{i-1}\right) - \left(\mathbf{v}_{i+1} - \mathbf{v}_i\right)\right) = -\frac{1}{t}\left(\mathbf{v}_{i-1} + \mathbf{v}_{i+1} - 2\mathbf{v}_i\right)$$

Discrete Laplace-Beltrami

- Intuition for uniform discretization





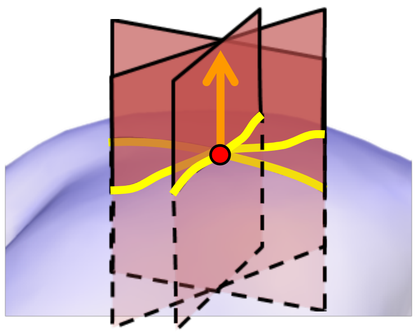
$$H = \frac{1}{2\pi}\int_0^{2\pi} \kappa(\varphi)\,\mathrm{d}\varphi$$

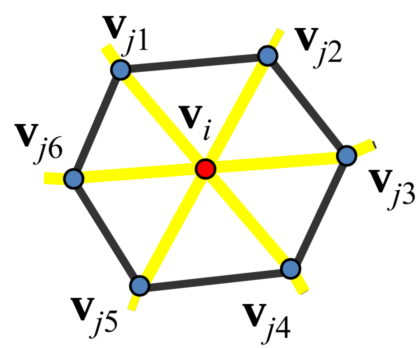
$$\begin{aligned}
&\mathbf{v}_{j1} + \mathbf{v}_{j4} - 2\mathbf{v}_i \quad + \\
&\mathbf{v}_{j2} + \mathbf{v}_{j5} - 2\mathbf{v}_i \quad + \\
&\mathbf{v}_{j3} + \mathbf{v}_{j6} - 2\mathbf{v}_i \quad =
\end{aligned}$$

$$6\,L(\mathbf{v}_i) = \sum_{\mathrm{k}=1}^{6} \mathbf{v}_{jk} - 6\mathbf{v}_i \;\approx\; -6\,H\mathbf{n}$$

Discrete Laplace-Beltrami

- Intuition for uniform discretization





$$H = \frac{1}{2\pi}\int_0^{2\pi} \kappa(\varphi)\,\mathrm{d}\varphi$$

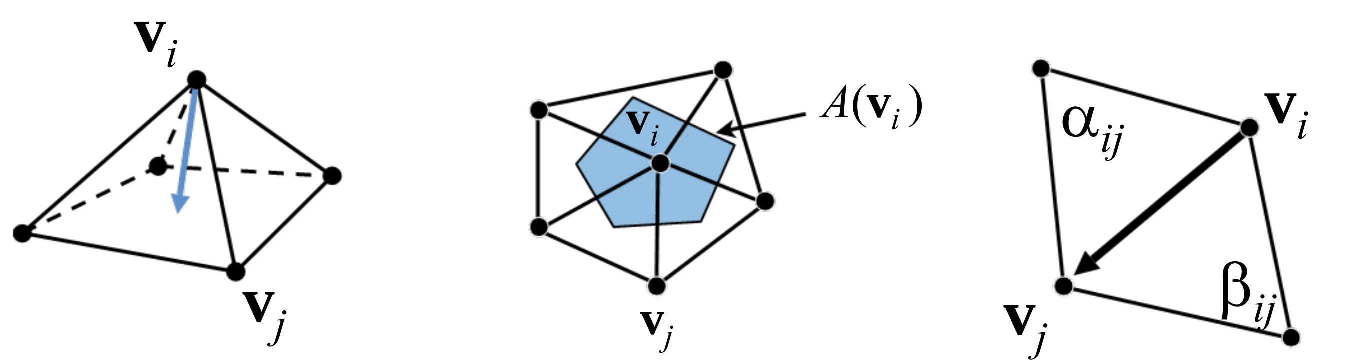
$$\begin{aligned}
&\mathbf{v}_{j1} + \mathbf{v}_{j4} - 2\mathbf{v}_i \quad + \\
&\mathbf{v}_{j2} + \mathbf{v}_{j5} - 2\mathbf{v}_i \quad + \\
&\mathbf{v}_{j3} + \mathbf{v}_{j6} - 2\mathbf{v}_i \quad =
\end{aligned}$$

$$\boxed{L(\mathbf{v}_i) = \frac{1}{6}\left(\sum_{\mathrm{k}=1}^{6} \mathbf{v}_{jk} - 6\mathbf{v}_i\right) \approx -H\mathbf{n}}$$

Discrete Laplace-Beltrami

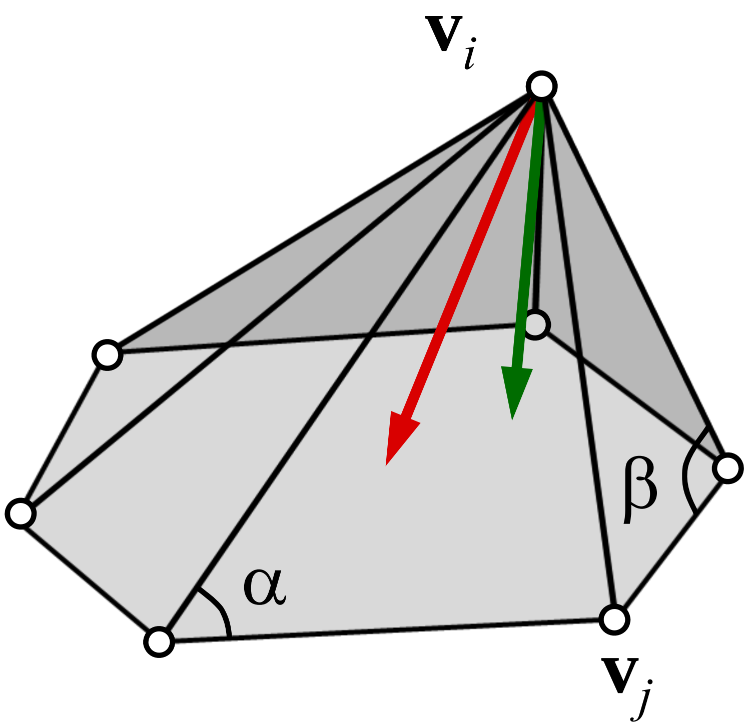
- Cotangent formula – to compensate for triangle shape irregularity

$$L_c(\mathbf{v}_i) = \frac{1}{2A(\mathbf{v}_i)} \sum_{\mathbf{v}_j \in N_1(\mathbf{v}_i)} \left(\cot\alpha_{ij} + \cot\beta_{ij}\right)\left(\mathbf{v}_j - \mathbf{v}_i\right)$$



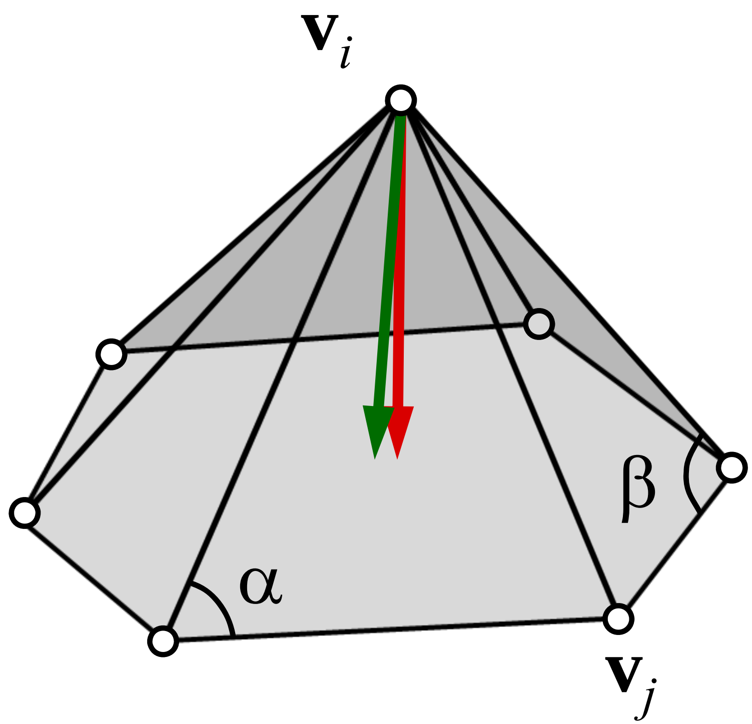
Discrete Laplace-Beltrami

- When the edge lengths are equal, the uniform and the cotangent Laplacians coincide



Discrete Laplace-Beltrami

- When the edge lengths are equal, the uniform and the cotangent Laplacians coincide



Linear Surface-Based Deformation

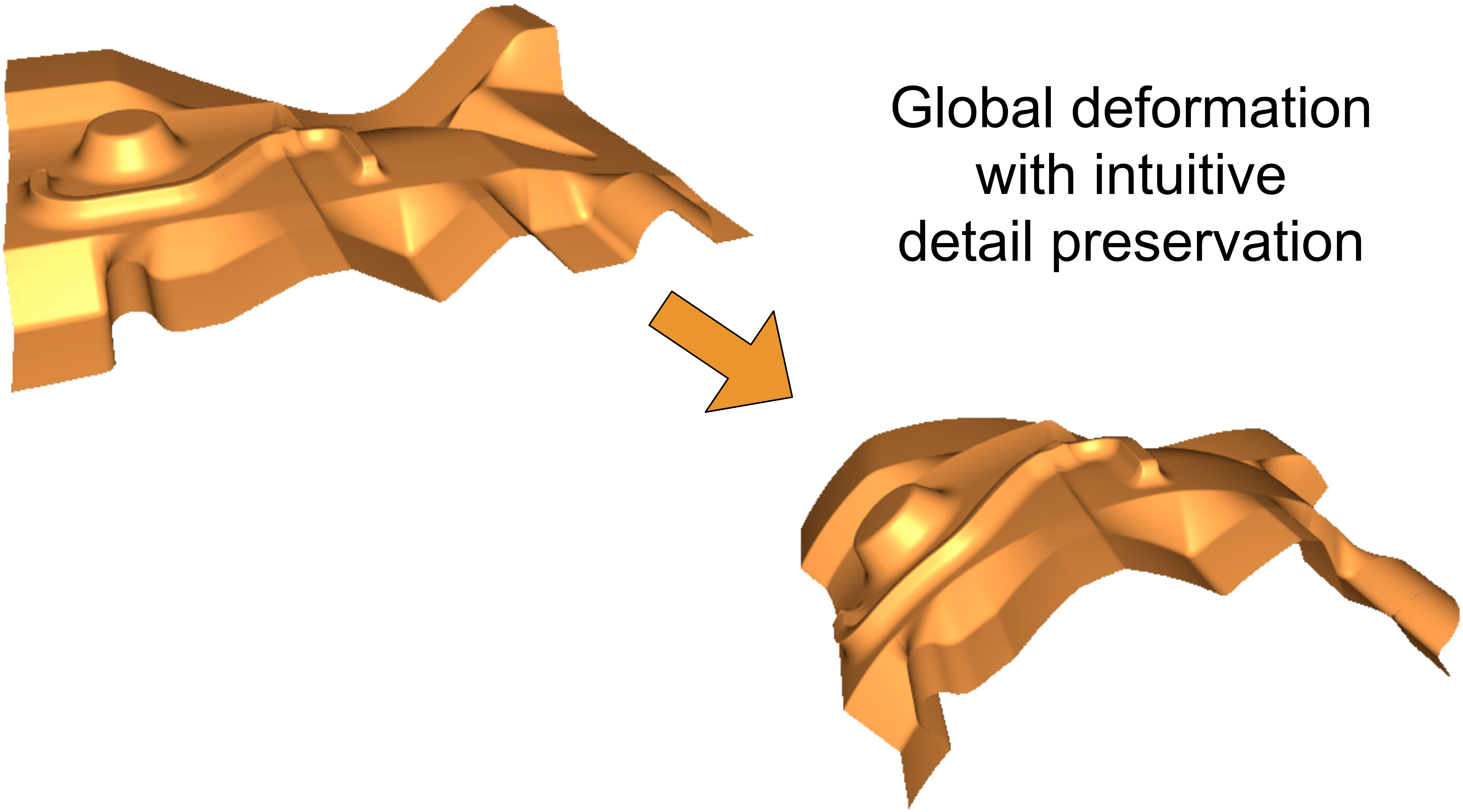
Prof. Dr. Mario Botsch

Computer Graphics & Geometry Processing
Bielefeld University

Universität Bielefeld

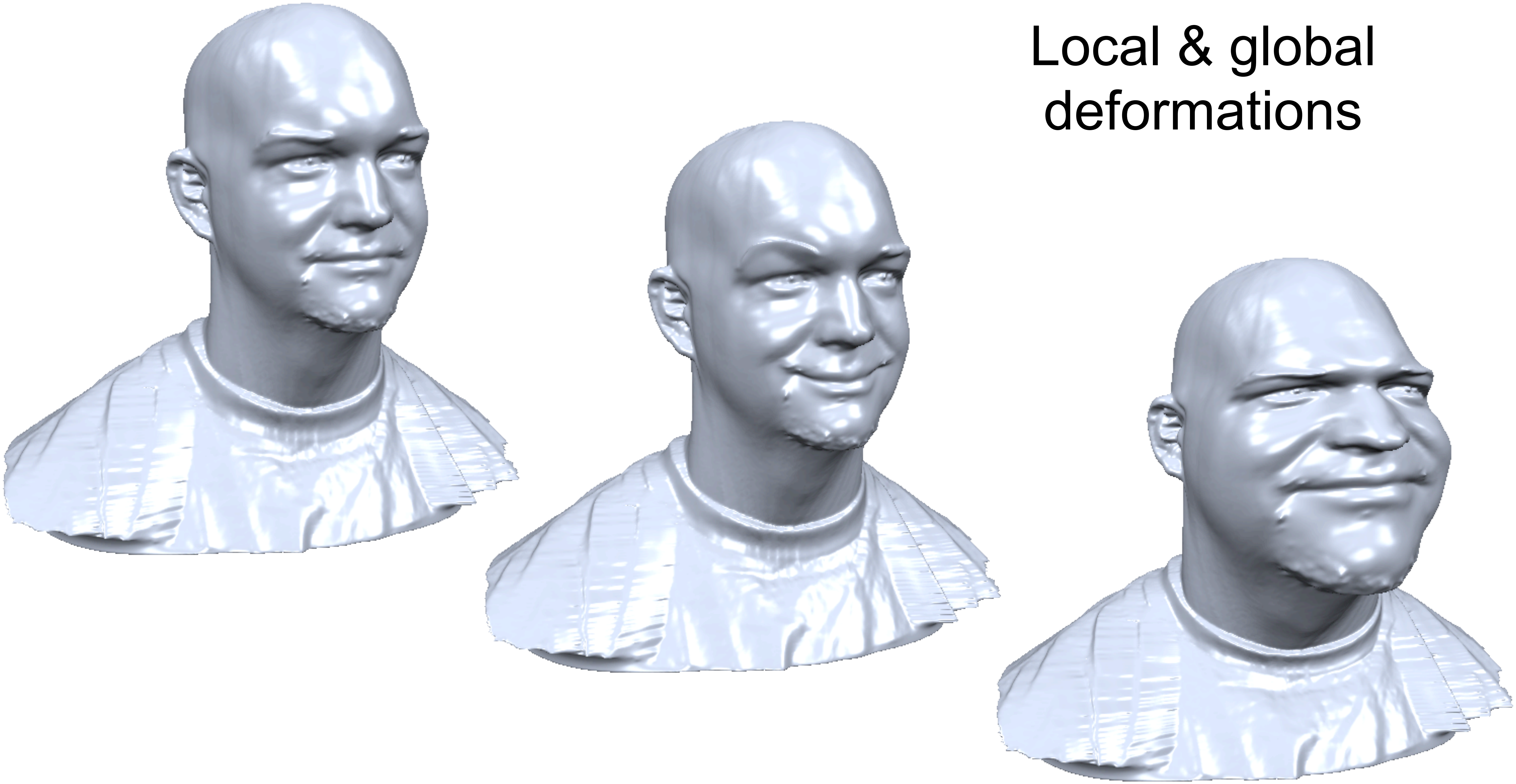
CITEC

Mesh Deformation



Global deformation
with intuitive
detail preservation

Mesh Deformation



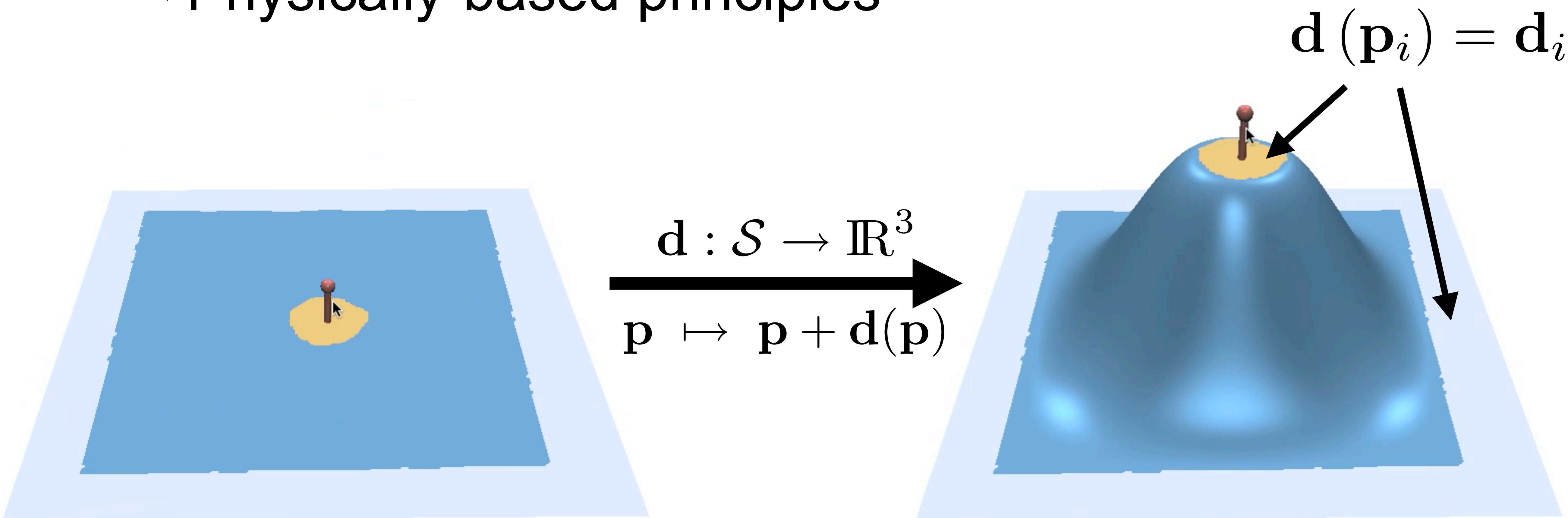
Local & global deformations

Linear Surface-Based Deformation

- **Shell-Based Deformation**
- Multiresolution Deformation
- Differential Coordinates

Modeling Metaphor

- Mesh deformation by displacement function **d**
  - – Interpolate prescribed constraints
  - – Smooth, intuitive deformation
  - ➡ Physically-based principles



Shell Deformation Energy

- **Stretching**
  - Change of local distances
  - Captured by 1st fundamental form

$$\int_\Omega k_s \left\| \mathbf{I} - \bar{\mathbf{I}} \right\|^2$$

$$\mathbf{I} = \begin{bmatrix} \mathbf{x}_u^T \mathbf{x}_u & \mathbf{x}_u^T \mathbf{x}_v \\ \mathbf{x}_v^T \mathbf{x}_u & \mathbf{x}_v^T \mathbf{x}_v \end{bmatrix}$$

- **Bending**
  - Change of local curvature
  - Captured by 2nd fundamental form

$$\int_\Omega k_b \left\| \mathbf{II} - \bar{\mathbf{II}} \right\|^2$$

$$\mathbf{II} = \begin{bmatrix} \mathbf{x}_{uu}^T \mathbf{n} & \mathbf{x}_{uv}^T \mathbf{n} \\ \mathbf{x}_{vu}^T \mathbf{n} & \mathbf{x}_{vv}^T \mathbf{n} \end{bmatrix}$$

- Stretching & bending is sufficient
  - Differetial geometry: “1st and 2nd fundamental forms determine a surface up to rigid motion.”

Physically-Based Deformation

- Nonlinear stretching & bending energies

$$\int_{\Omega} k_s \underbrace{\boxed{\left\|\mathbf{I} - \mathbf{I}'\right\|^2}}_{\text{stretching}} + k_b \underbrace{\boxed{\left\|\mathbf{II} - \mathbf{II}'\right\|^2}}_{\text{bending}} \,\mathrm{d}u\mathrm{d}v$$

- Linearize terms → Quadratic energy

$$\int_{\Omega} k_s \underbrace{\boxed{\left(\left\|\mathbf{d}_u\right\|^2 + \left\|\mathbf{d}_v\right\|^2\right)}}_{\text{stretching}} + k_b \underbrace{\boxed{\left(\left\|\mathbf{d}_{uu}\right\|^2 + 2\left\|\mathbf{d}_{uv}\right\|^2 + \left\|\mathbf{d}_{vv}\right\|^2\right)}}_{\text{bending}} \mathrm{d}u\mathrm{d}v$$

Physically-Based Deformation

- Minimize linearized bending energy

$$E(\mathbf{d}) \; = \; \int_{\mathcal{S}} \|\mathbf{d}_{uu}\|^2 + 2\,\|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2 \,\mathrm{d}u\mathrm{d}v \;\; \rightarrow \;\; \min$$

$$f\,(x) \rightarrow \min$$

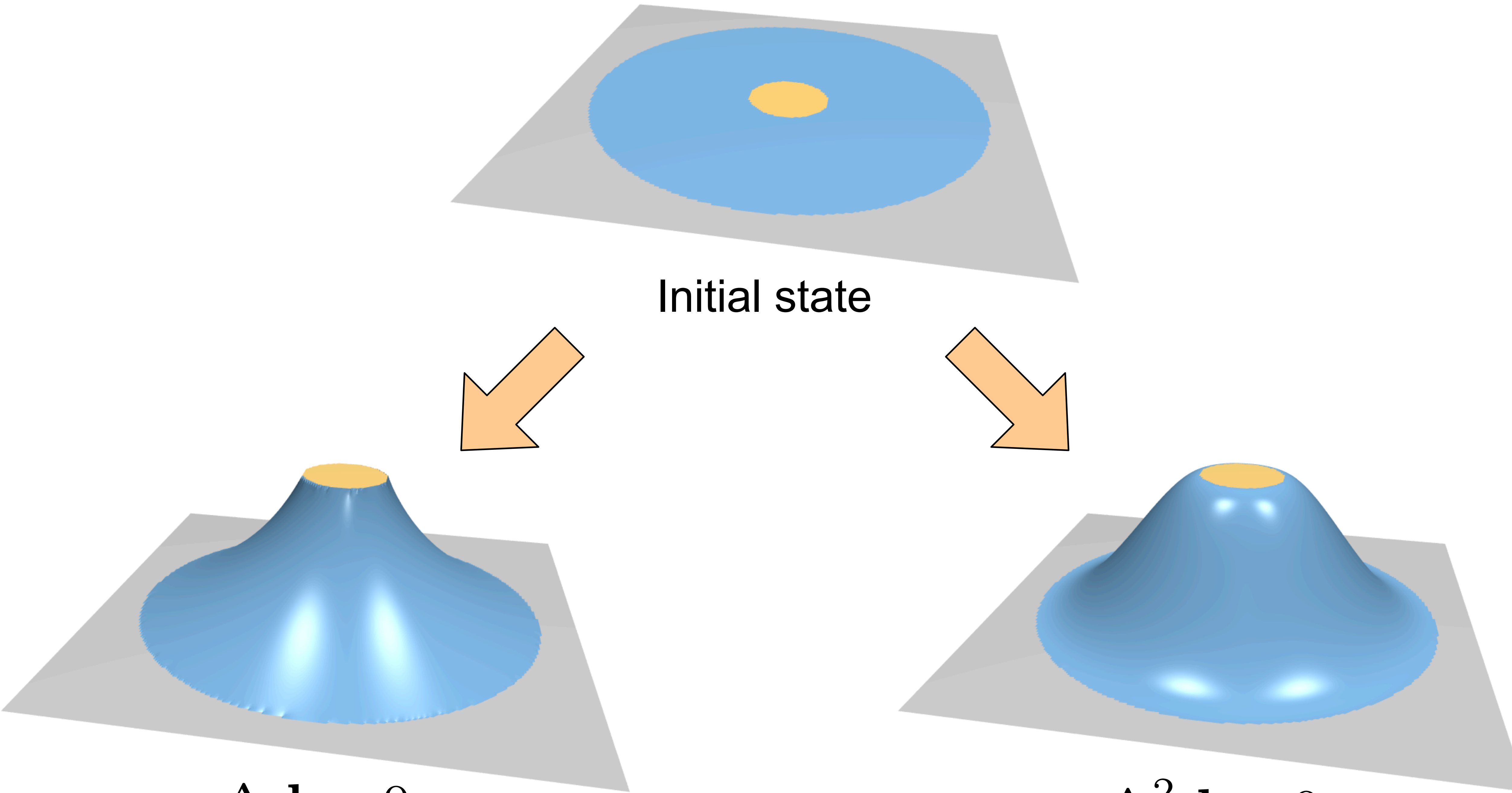
- Variational calculus → Euler-Lagrange PDE

$$\Delta^2\mathbf{d} \; := \; \mathbf{d}_{uuuu} + 2\mathbf{d}_{uuvv} + \mathbf{d}_{vvvv} \; = \; 0$$

$$f'(x) = 0$$

➡ “Best” deformation that satisfies constraints

Deformation Energies



$$\Delta \mathbf{d} = 0$$

(Membrane)

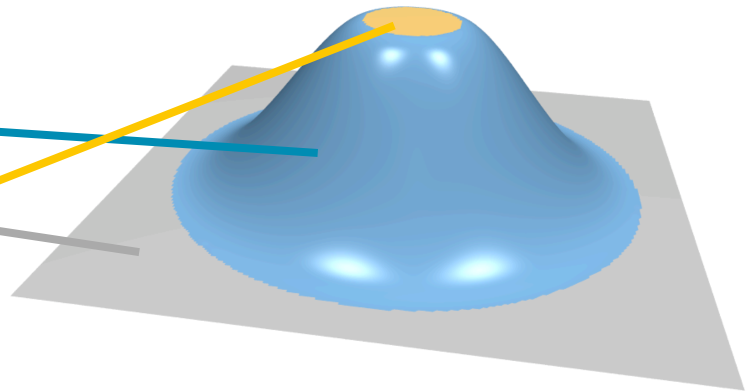
$$\Delta^2 \mathbf{d} = 0$$

(Thin plate)

PDE Discretization

- Euler-Lagrange PDE

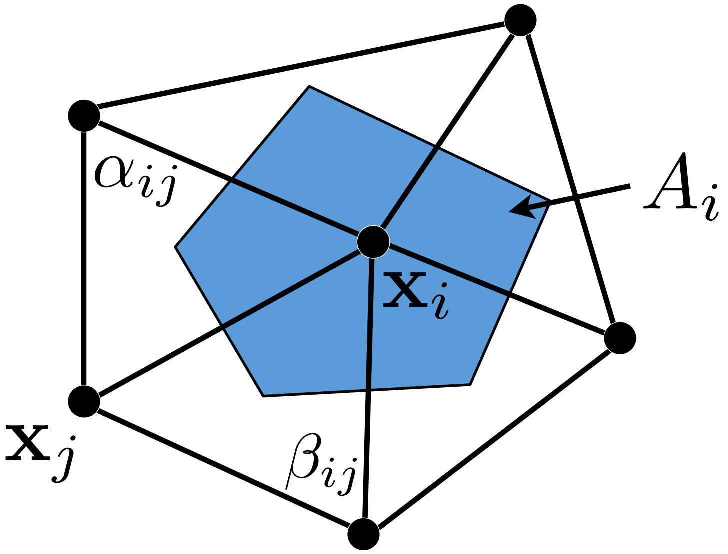
$$\Delta^2 \mathbf{d} = \mathbf{0}$$
$$\mathbf{d} = \mathbf{0}$$
$$\mathbf{d} = \delta \mathbf{h}$$



- Laplace discretization

$$\Delta \mathbf{d}_i = \frac{1}{2A_i} \sum_{j \in \mathcal{N}_i} (\cot \alpha_{ij} + \cot \beta_{ij})(\mathbf{d}_j - \mathbf{d}_i)$$

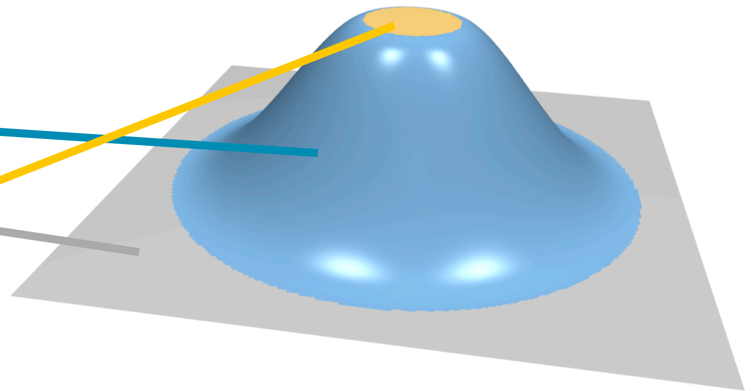
$$\Delta^2 \mathbf{d}_i = \Delta(\Delta \mathbf{d}_i)$$



Linear System

- Sparse linear system (19 nz/row)

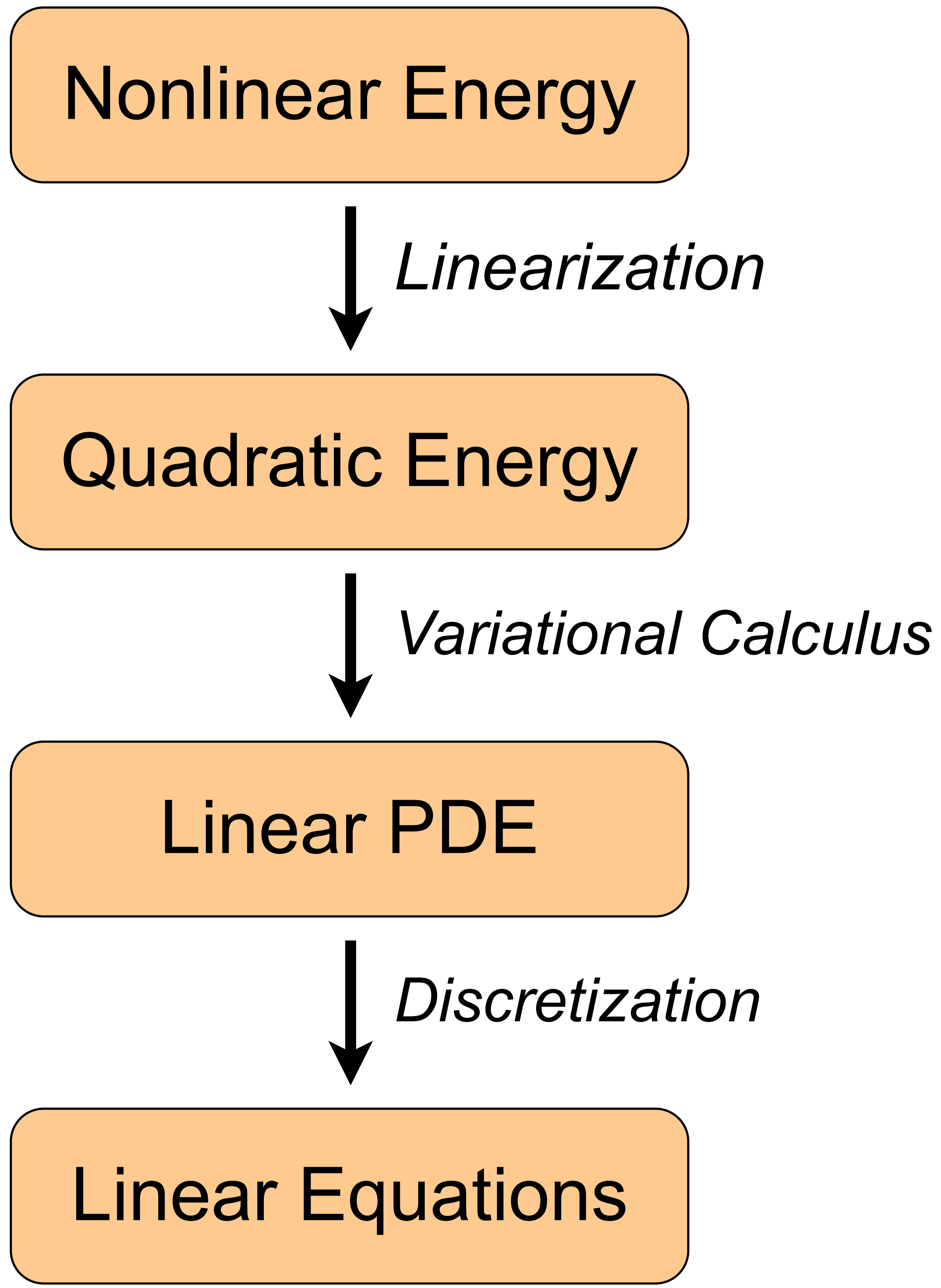
$$\begin{pmatrix} & \Delta^2 & \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \vdots \\ \mathbf{d}_i \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \delta\mathbf{h}_i \end{pmatrix}$$



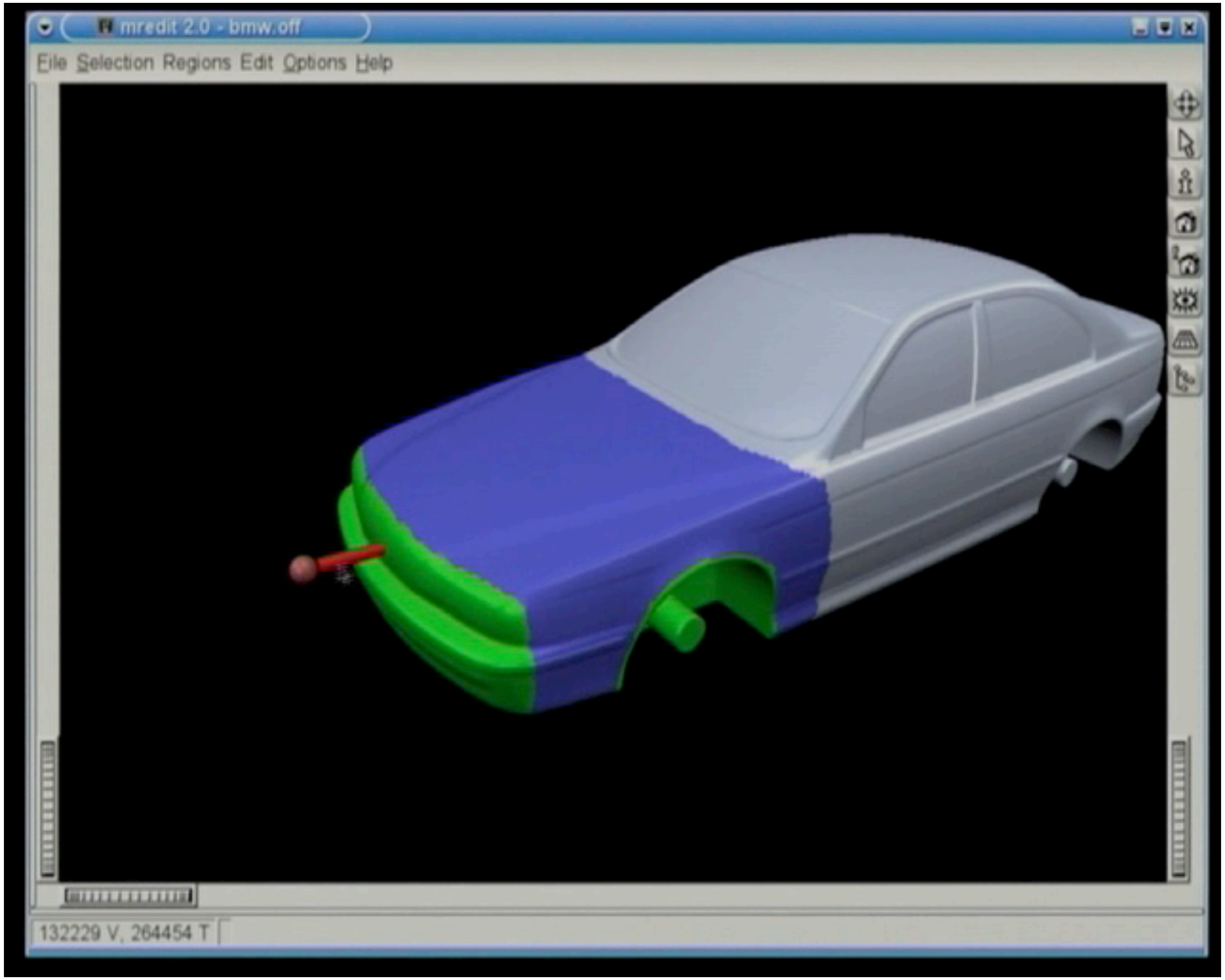
  - Turn into symmetric positive definite system

- Solve this system *each frame*
  - Use efficient linear solvers !!!
  - Sparse Cholesky factorization
  - **See course notes for details**

Derivation Steps

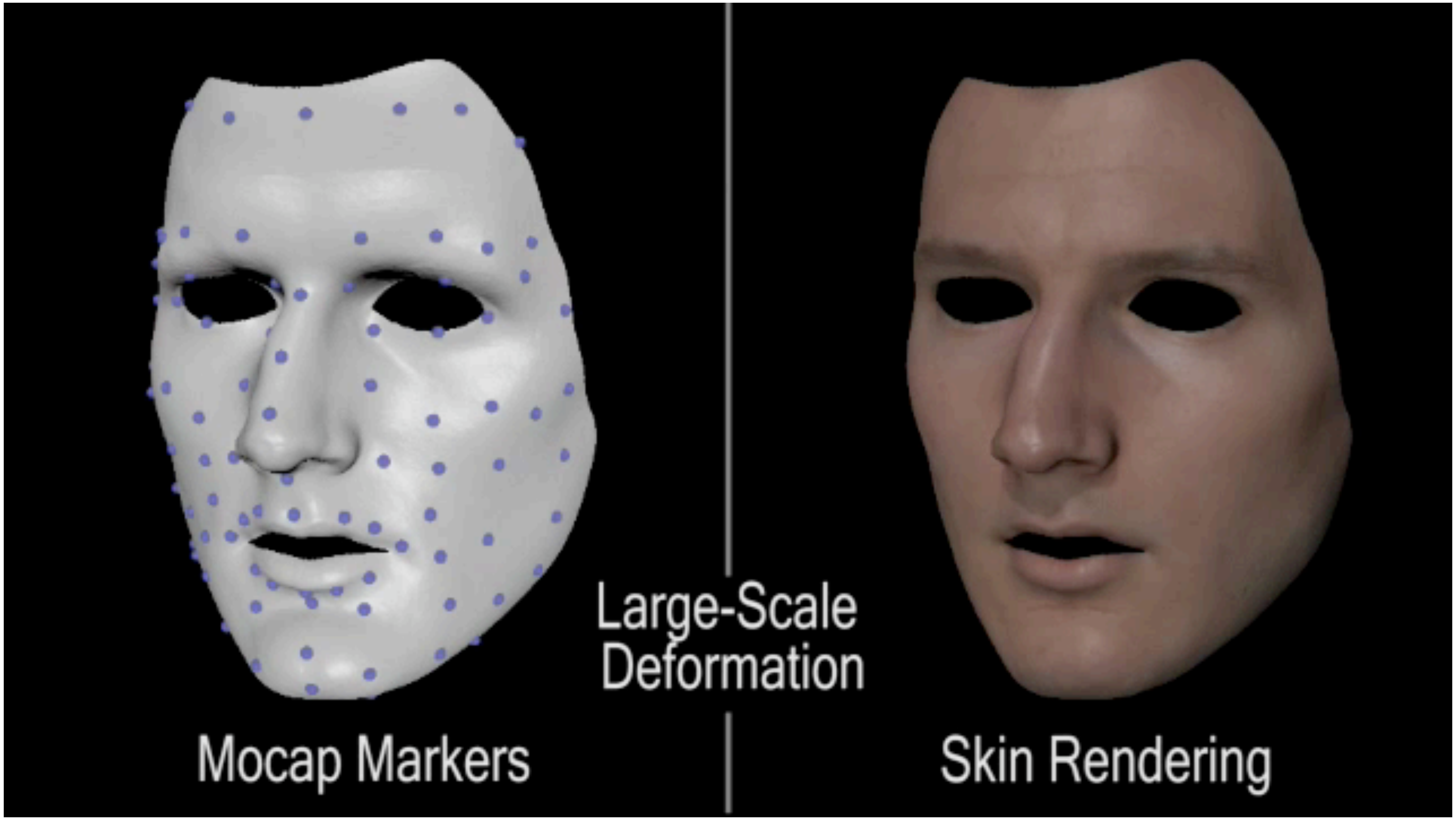


CAD-Like Deformation



[Botsch & Kobbelt, SIGGRAPH 04]

Face Animation



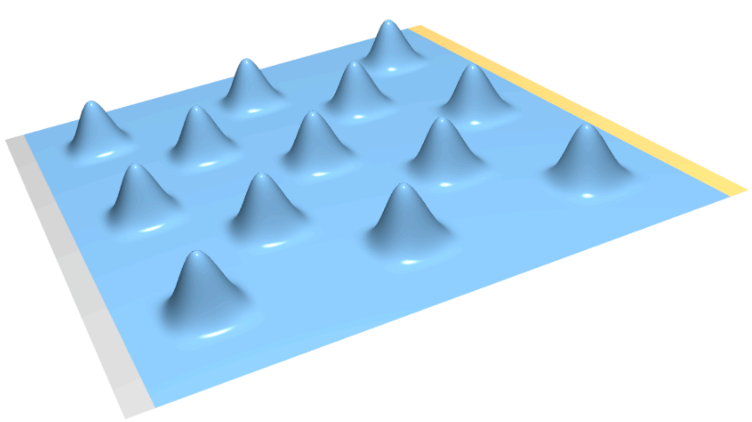
[Bickel et al, SCA 08]

Linear Surface-Based Deformation

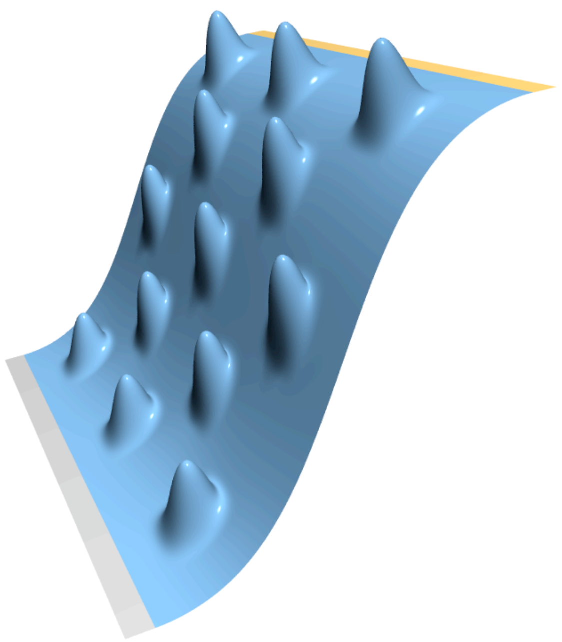
- Shell-Based Deformation
- **Multiresolution Deformation**
- Differential Coordinates

Multiresolution Modeling

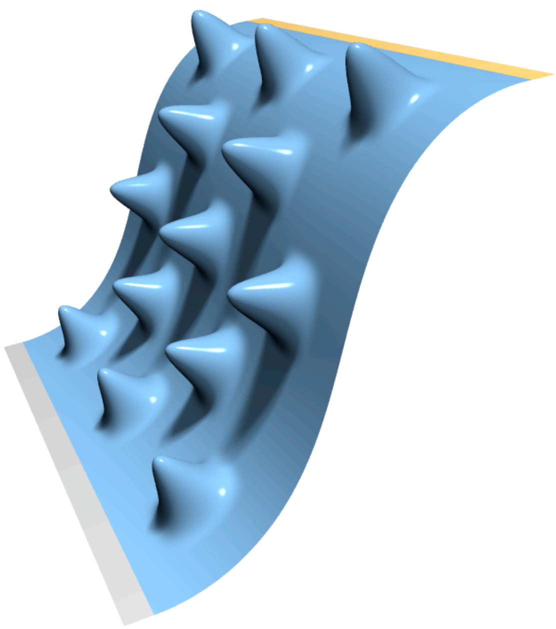
- Even pure translations induce local rotations!
  ➡ Inherently non-linear coupling
- Alternative approach
  – Linear deformation + multi-scale decomposition...



Original

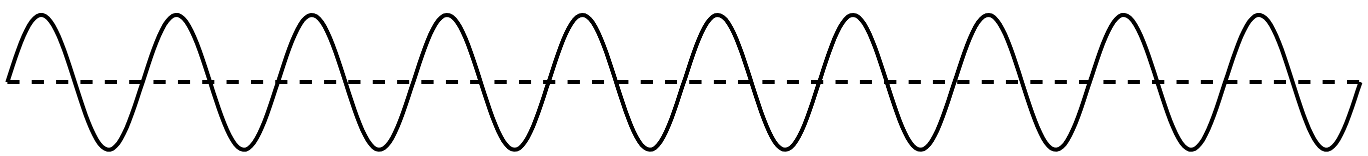


Linear



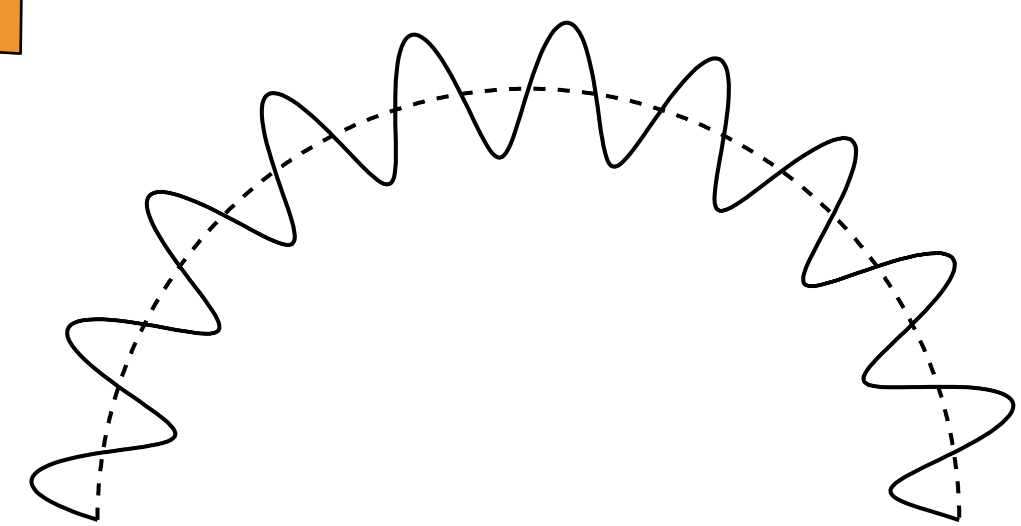
Nonlinear

Multiresolution Editing



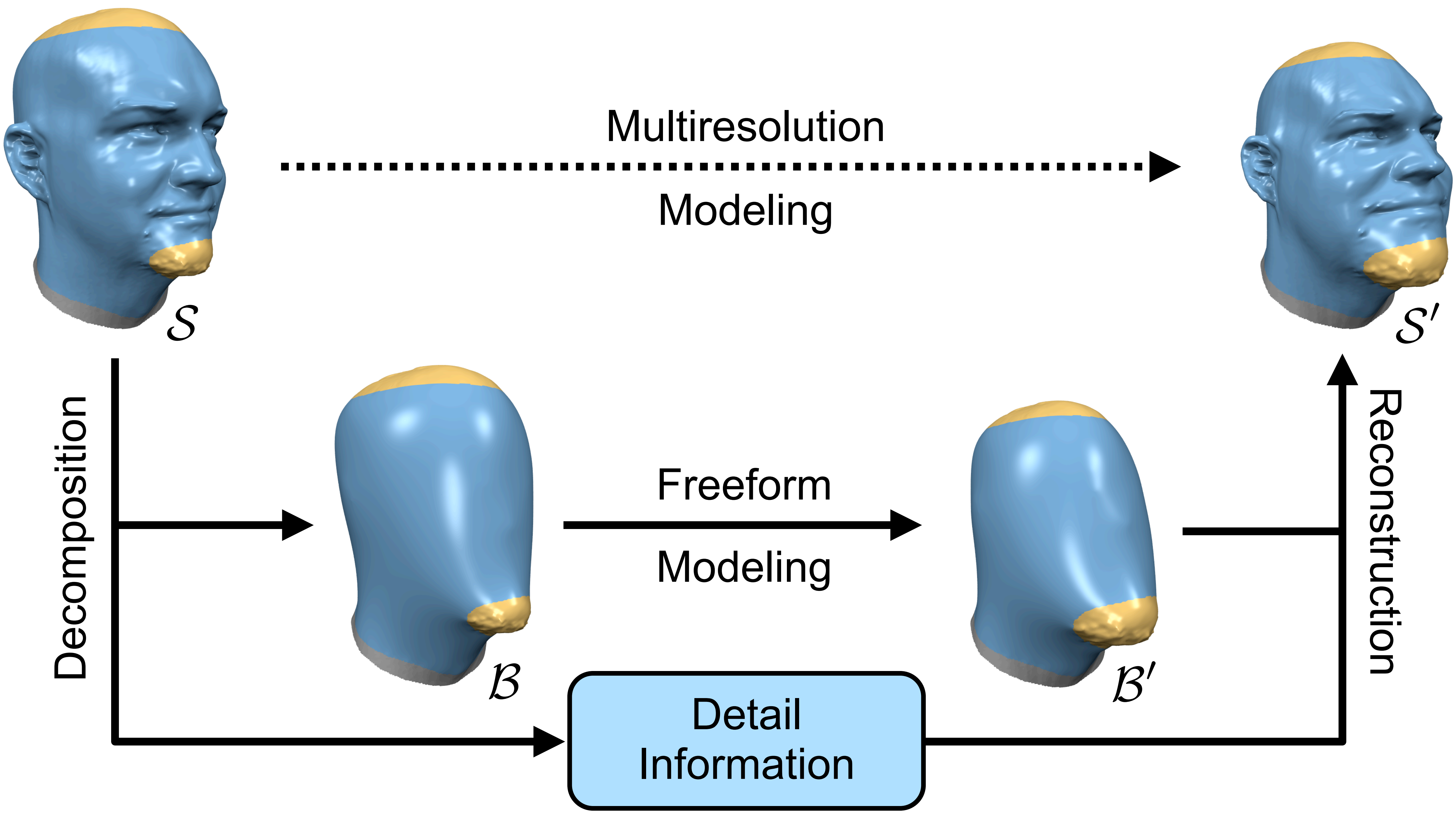
Frequency decomposition

Change low frequencies

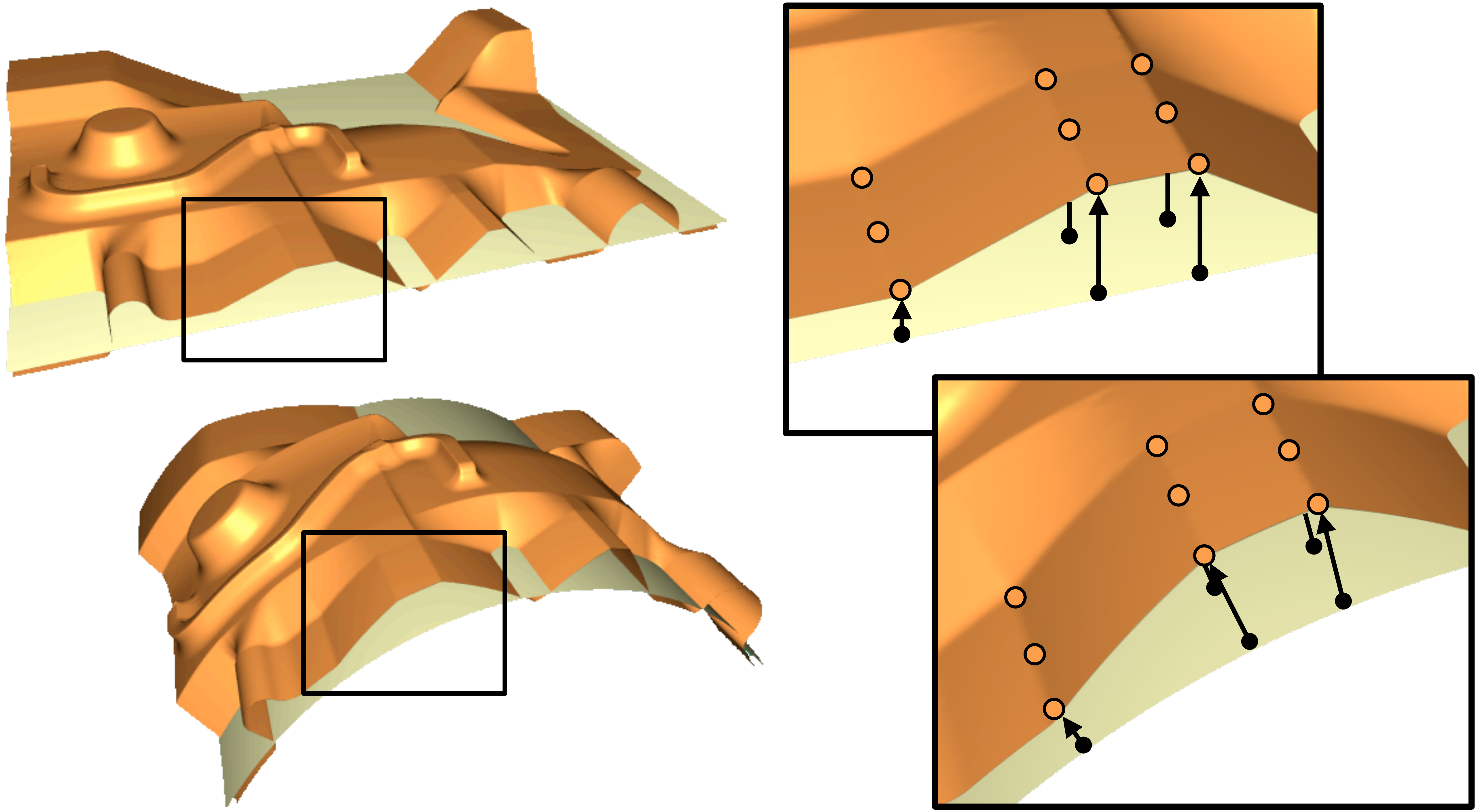


Add high frequency details, stored in local frames

Multiresolution Editing

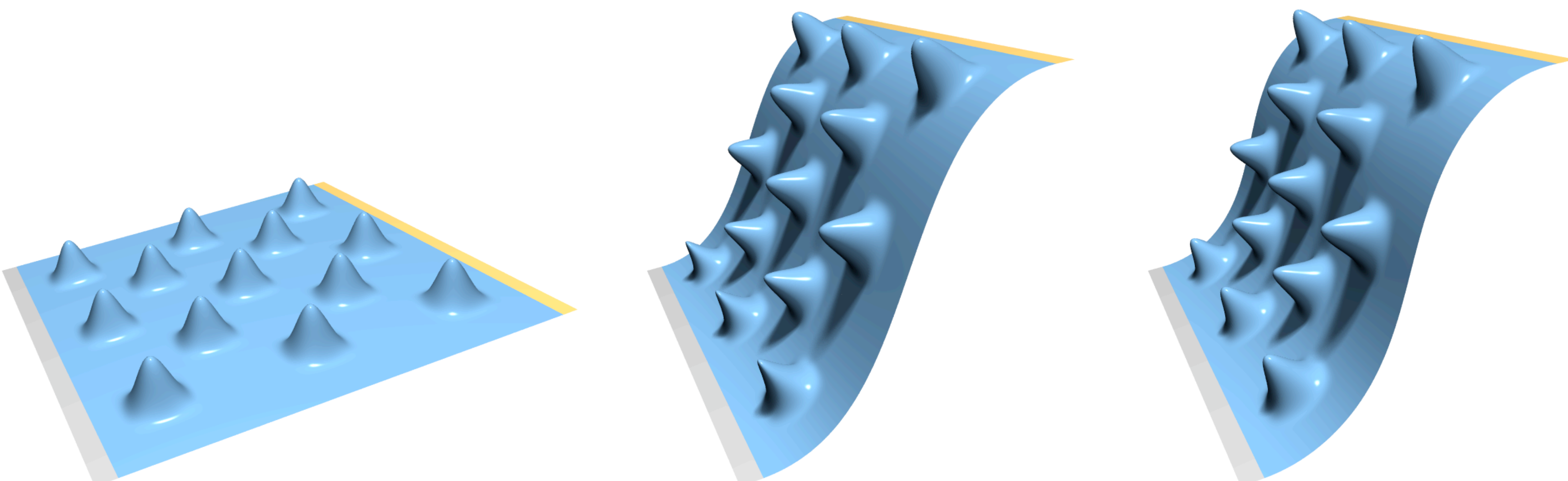


Normal Displacements



Limitations

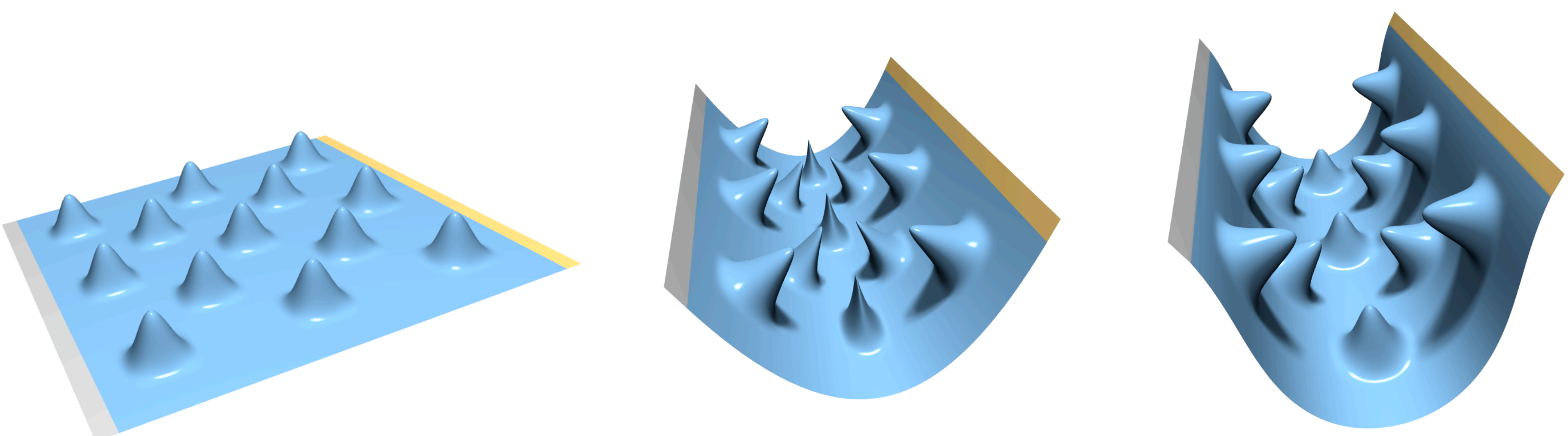
- Neighboring displacements are not coupled
  - Surface bending changes their angle
  - Leads to volume changes or self-intersections



Original Normal Displ. Nonlinear

Limitations

- Neighboring displacements are not coupled
  - Surface bending changes their angle
  - Leads to volume changes or self-intersections



Original Normal Displ. Nonlinear

Limitations

- Neighboring displacements are not coupled
  - Surface bending changes their angle
  - Leads to volume changes or self-intersections
  - **See course notes for some other techniques...**
- Multiresolution hierarchy difficult to compute
  - Complex topology
  - Complex geometry
  - Might require more hierarchy levels

Linear Surface-Based Deformation

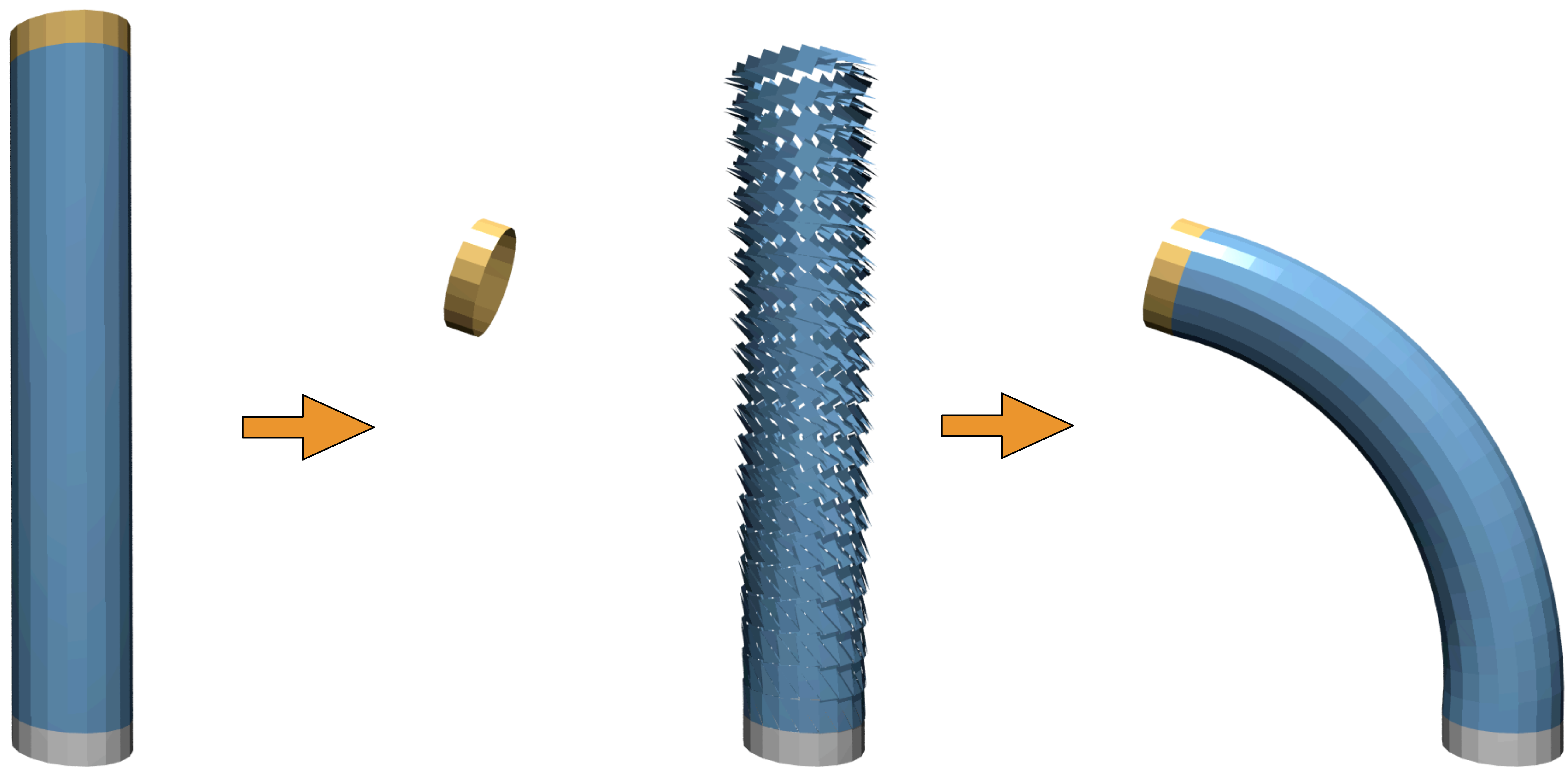
- Shell-Based Deformation
- Multiresolution Deformation
- **Differential Coordinates**

Differential Coordinates

1. Manipulate *differential coordinates* instead of *spatial* coordinates
   - Gradients, Laplacians, local frames
   - Intuition: Close connection to surface normal

2. Find mesh with desired differential coords
   - Cannot be solved exactly
   - Formulate as energy minimization

Differential Coordinates



Original | Rotated Diff-Coords | Reconstructed Mesh

Differential Coordinates

- **Which differential coordinate $\boldsymbol{\delta}_i$?**
  - Gradients
  - Laplacians
  - ...
- How to get local transformations $T_i(\boldsymbol{\delta}_i)$?
  - Smooth propagation
  - Implicit optimization
  - ...

Gradient-Based Editing

- Manipulate gradient of a function (e.g. a surface)

$$\mathbf{g} = \nabla \mathbf{f} \qquad \mathbf{g} \mapsto \mathbf{T}(\mathbf{g})$$

- Find function **f**' whose gradient is (close to) **g**'

$$\mathbf{f}' = \underset{\mathbf{f}}{\operatorname{argmin}} \int_{\Omega} \left\| \nabla \mathbf{f} - \mathbf{T}(\mathbf{g}) \right\|^2 \mathrm{d}u \mathrm{d}v$$

- Variational calculus $\rightarrow$ Euler-Lagrange PDE

$$\Delta \mathbf{f}' = \operatorname{div} \mathbf{T}(\mathbf{g})$$

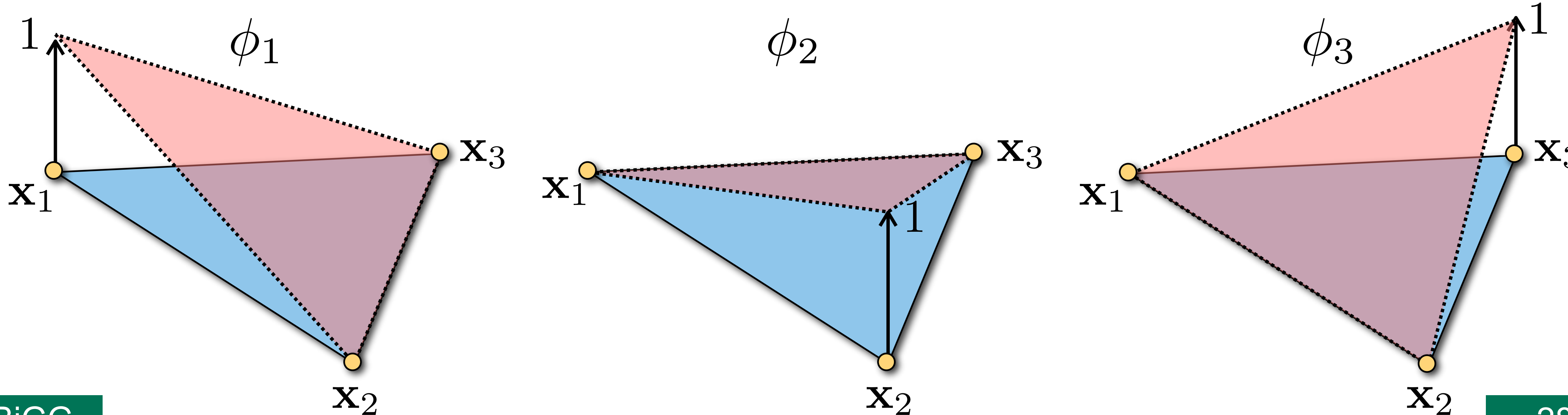
Gradient-Based Editing

- Consider piecewise linear ***coordinate function***

$$\mathbf{p}(u,v) = \sum_{v_i} \mathbf{p}_i \cdot \phi_i(u,v)$$

- Its gradient is

$$\nabla \mathbf{p}(u,v) = \sum_{v_i} \mathbf{p}_i \cdot \nabla \phi_i(u,v)$$



Gradient-Based Editing

- Consider piecewise linear ***coordinate function***

$$\mathbf{p}(u,v) = \sum_{v_i} \mathbf{p}_i \cdot \phi_i(u,v)$$

- Its gradient is

$$\nabla\mathbf{p}(u,v) = \sum_{v_i} \mathbf{p}_i \cdot \nabla\phi_i(u,v)$$

- It is constant per triangle

$$\nabla\mathbf{p}\big|_{f_j} \;=:\; \mathbf{g}_j \in \mathbb{R}^{3\times 3}$$

Gradient-Based Editing

- Gradient of coordinate function $\mathbf{p}$

$$\begin{pmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_F \end{pmatrix} = \underbrace{\mathbf{G}}_{(3F \times V)} \begin{pmatrix} \mathbf{p}_1^T \\ \vdots \\ \mathbf{p}_V^T \end{pmatrix}$$

- Manipulate per-face gradients

$$\mathbf{g}_j \mapsto \mathbf{T}_j(\mathbf{g}_j)$$

Gradient-Based Editing

- Reconstruct mesh from new gradients
  - Overdetermined $(3F \times V)$ system
  - Weighted least squares system
  - ➡Linear Poisson system $\Delta \mathbf{p}' = \operatorname{div} \mathbf{T}(\mathbf{g})$

$$\underbrace{\boxed{\mathbf{G}^T \mathbf{D} \mathbf{G}}}_{\operatorname{div}\nabla = \Delta} \cdot \begin{pmatrix} {\mathbf{p}'_1}^T \\ \vdots \\ {\mathbf{p}'_V}^T \end{pmatrix} = \underbrace{\boxed{\mathbf{G}^T \mathbf{D}}}_{\operatorname{div}} \cdot \begin{pmatrix} \mathbf{T}_1(\mathbf{g}_1) \\ \vdots \\ \mathbf{T}_F(\mathbf{g}_F) \end{pmatrix}$$

Laplacian-Based Editing

- Manipulate Laplacians field of a surface

$$\mathbf{l} = \Delta(\mathbf{p}) \ , \quad \mathbf{l} \mapsto \mathbf{T}(\mathbf{l})$$

- Find surface whose Laplacian is (close to) **δ**'

$$\mathbf{p}' = \underset{\mathbf{p}}{\operatorname{argmin}} \int_{\Omega} \|\Delta\mathbf{p} - \mathbf{T}(\mathbf{l})\|^2 \, \mathrm{d}u\mathrm{d}v$$

- Variational calculus yields Euler-Lagrange PDE

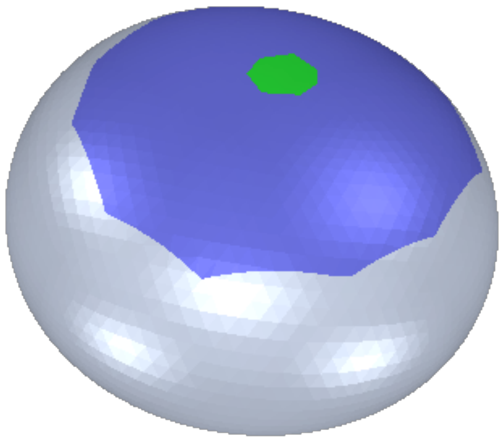
$$\Delta^2 \mathbf{p}' = \Delta\mathbf{T}(\mathbf{l})$$

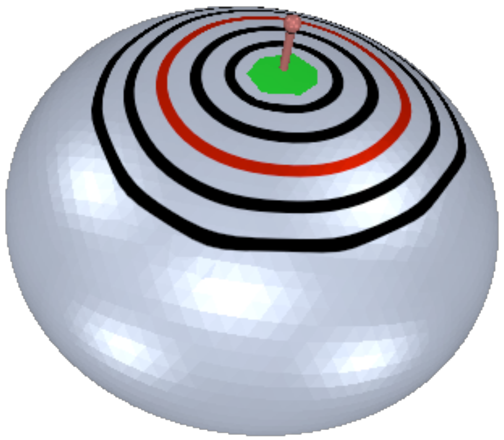
Differential Coordinates

- Which differential coordinate $\boldsymbol{\delta}_i$?
  - Gradients
  - Laplacians
  - ...
- **How to get local transformations $T_i(\boldsymbol{\delta}_i)$?**
  - Smooth propagation
  - Implicit optimization
  - ...

Smooth Propagation

1. Compute handle's deformation gradient
2. Extract rotation and scale/shear components
3. Propagate damped rotations over ROI

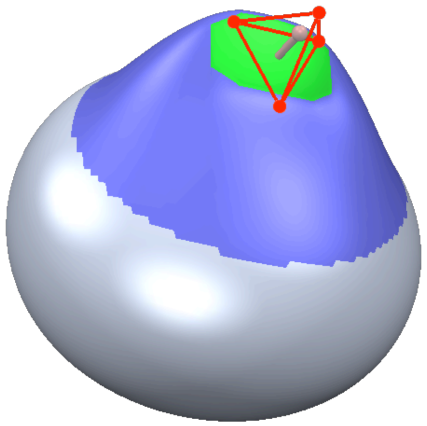




Deformation Gradient

- Handle has been transformed *affinely*

$$\mathbf{T}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{t}$$



- Deformation gradient is

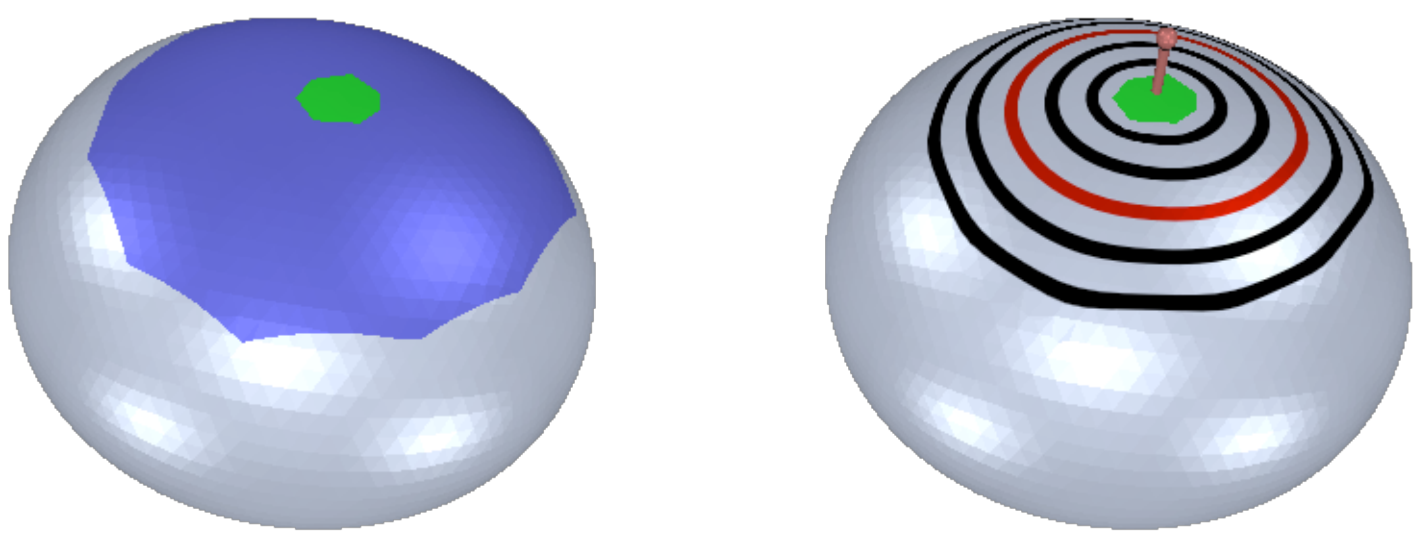
$$\nabla\mathbf{T}(\mathbf{x}) = \mathbf{A}$$

- Extract rotation $\mathbf{R}$ and scale/shear $\mathbf{S}$

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \quad \Rightarrow \quad \mathbf{R} = \mathbf{U}\mathbf{V}^T,\ \mathbf{S} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^T$$

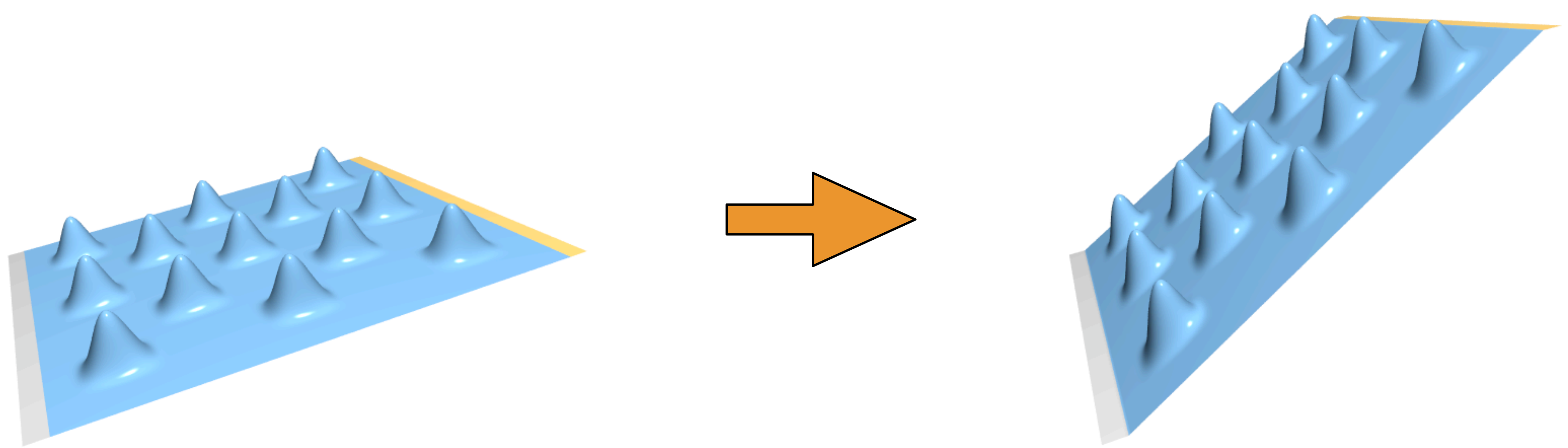
Smooth Propagation

- Construct smooth scalar field [0,1]
  - $s(\mathbf{x})=1$: Full deformation (handle)
  - $s(\mathbf{x})=0$: No deformation (fixed part)
  - $s(\mathbf{x})\in(0,1)$: Damp handle transformation (in between)



Limitations

- Differential coordinates work well for **rotations**
  - Represented by deformation gradient
- **Translations** don't change deformation gradient
  - Translations don't change differetial coordinates
  - *"Translation insensitivity"*



Implicit Optimization

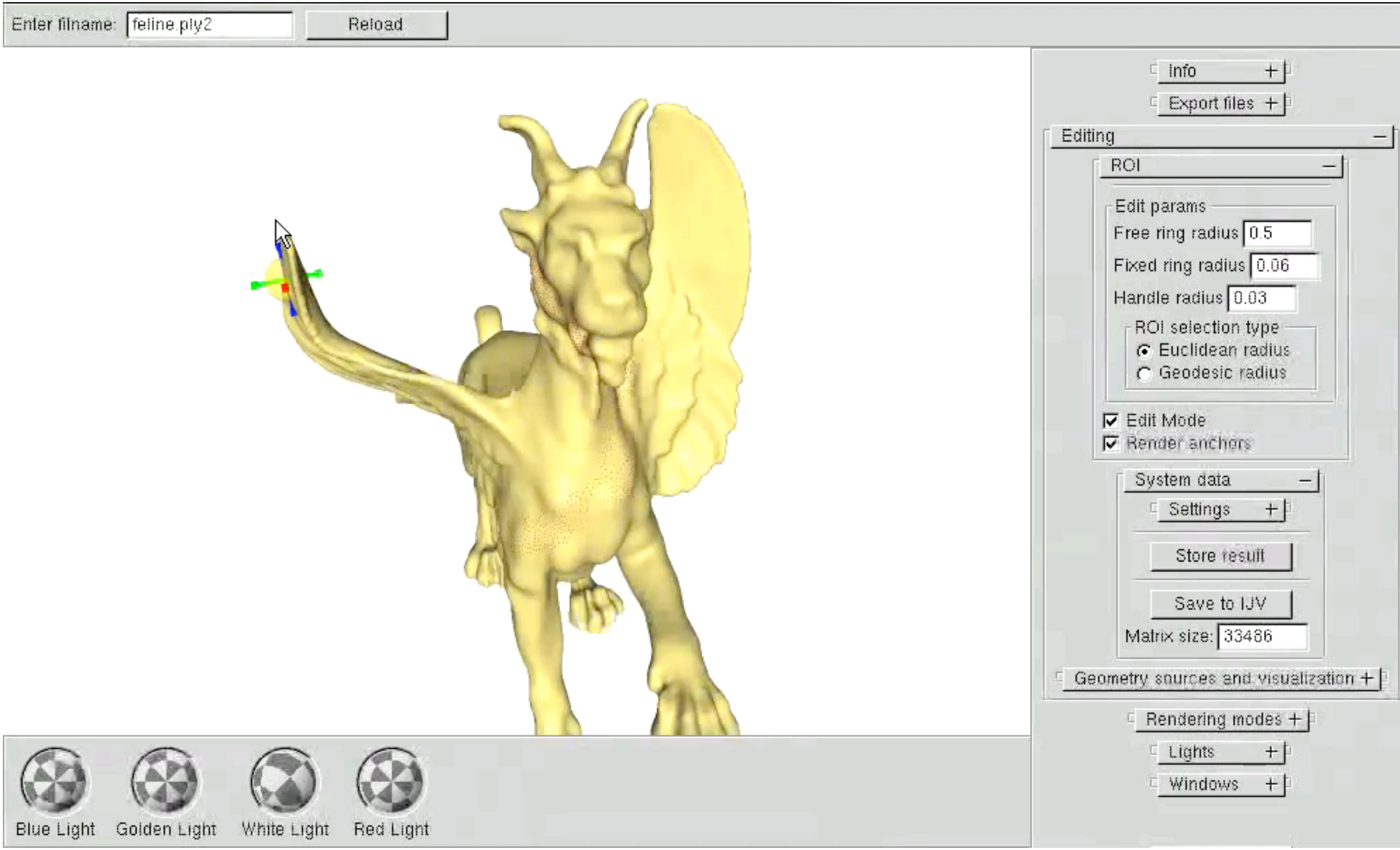
- Optimize for positions $\mathbf{p}_i$' & transformations $\mathbf{T}_i$

$$\Delta^2 \begin{pmatrix} \vdots \\ \mathbf{p}'_i \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ \Delta \mathbf{T}_i(\mathbf{l}_i) \\ \vdots \end{pmatrix} \quad \leftrightarrow \quad \mathbf{T}_i(\mathbf{p}_i - \mathbf{p}_j) = \mathbf{p}'_i - \mathbf{p}'_j$$

- Linearize rotation/scale $\rightarrow$ one linear system

$$\mathbf{T}_i = \begin{pmatrix} s & -r_3 & r_2 \\ r_3 & s & -r_1 \\ -r_2 & r_1 & s \end{pmatrix}$$

Laplacian Surface Editing



Connection to Shells?

- Neglect local transformations $\mathbf{T}_i$ for a moment...

$$\int \|\Delta \mathbf{p}' - \mathbf{l}\|^2 \to \min \longrightarrow \Delta^2 \mathbf{p}' = \Delta \mathbf{l}$$

$$\Big\downarrow \begin{matrix} \mathbf{p}' = \mathbf{p} + \mathbf{d} \\ \mathbf{l} = \Delta \mathbf{p} \end{matrix}$$

$$\Delta^2(\mathbf{p} + \mathbf{d}) = \Delta^2 \mathbf{p}$$

$$\Big\downarrow$$

$$\int \|\mathbf{d}_{uu}\|^2 + 2\|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2 \to \min \longleftarrow \Delta^2 \mathbf{d} = 0$$

- Basic formulations equivalent!
- Differ in detail preservation
  - Rotation of Laplacians
  - Multi-scale decomposition

Linear Surface-Based Deformation

- Shell-Based Deformation
- Multiresolution Deformation
- Differential Coordinates

Eurographics 2009

Interactive Shape Modeling and Deformation

T3: Half-Day Tutorial

Linear Space Deformations

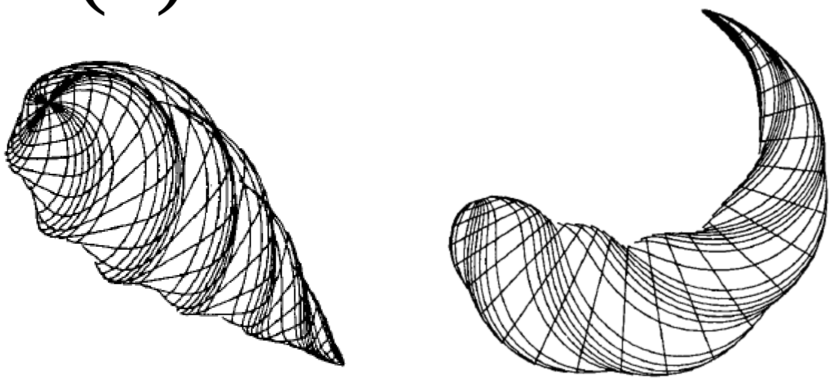
Space Deformation

- 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

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

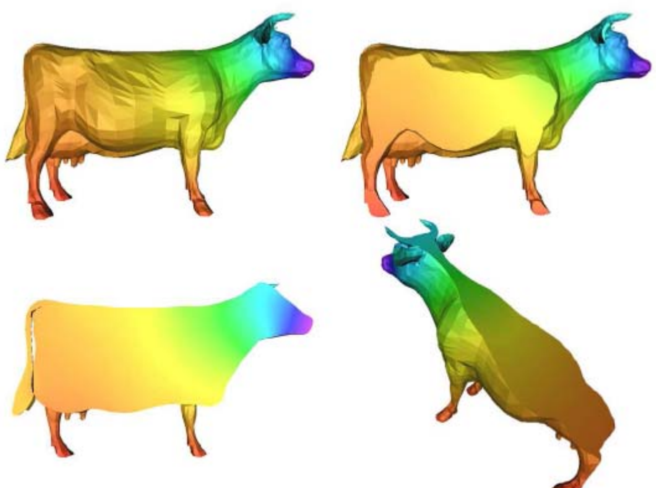


Twist warp
Global and local deformation of solids
[A. Barr, SIGGRAPH 84]

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 $B_i(\mathbf{x}) : \mathrm{R}^3 \to \mathrm{R}$

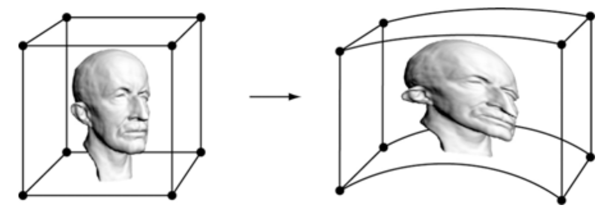
$$\mathbf{d}(\mathbf{x}) = \sum_{i=1}^{k} \mathbf{d}_i B_i(\mathbf{x})$$



- Basis functions should be smooth for aesthetic results

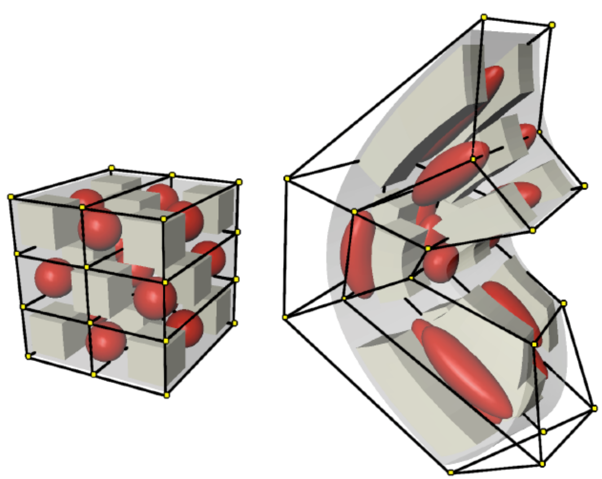
Trivariate Tensor Product Bases

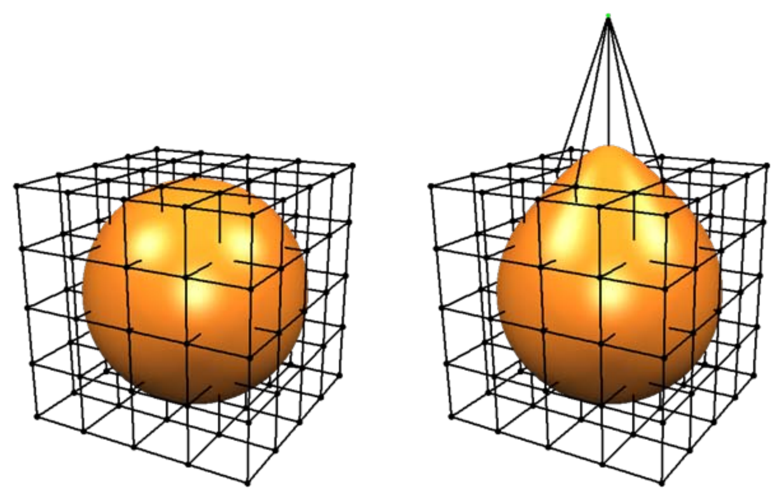
[Sederberg and Parry 86]



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

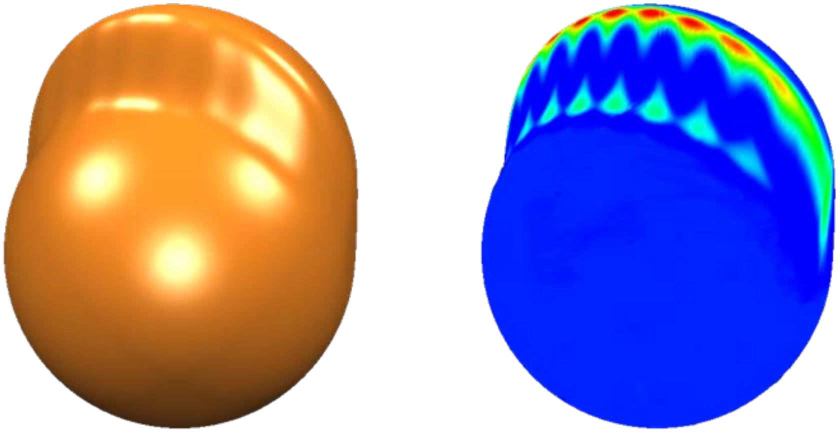
$$\mathbf{d}(x,y,z) = \sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n}\mathbf{d}_{ijk}\, N_i(x) N_j(y) N_i(z)$$





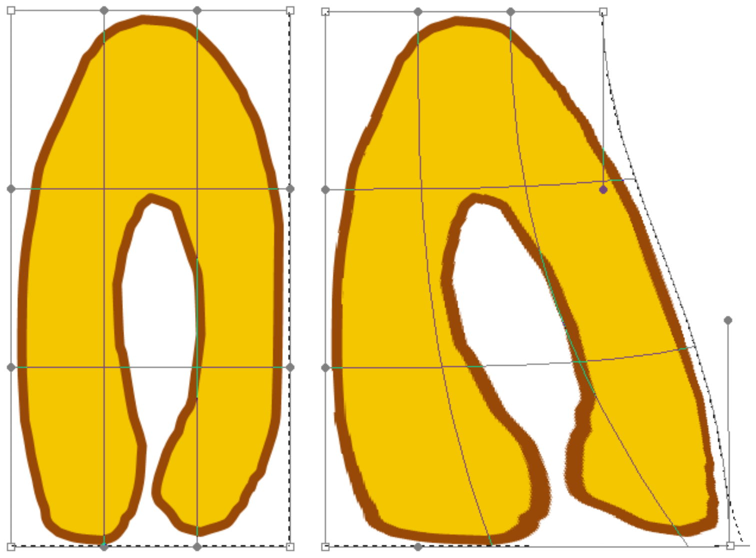
Trivariate Tensor Product Bases

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



Lattice as Control Object

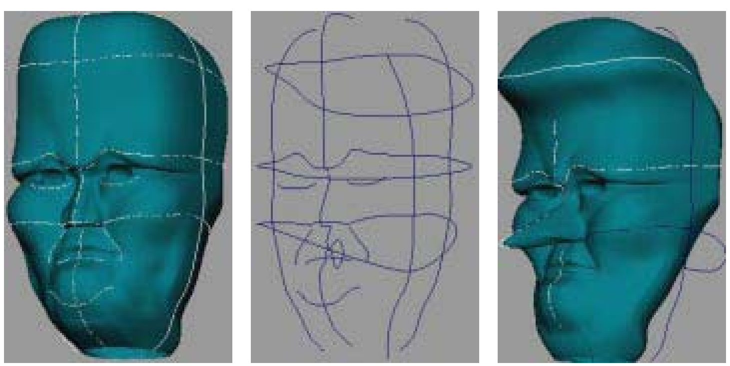
- Difficult to manipulate
- The control object is not related to the shape of the edited object
- Part of the shape in close Euclidean distance always deform similarly, even if geodesically far

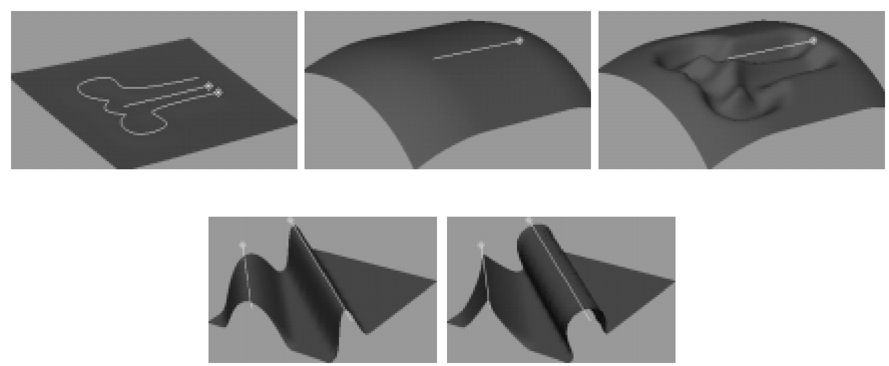


Wires

[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

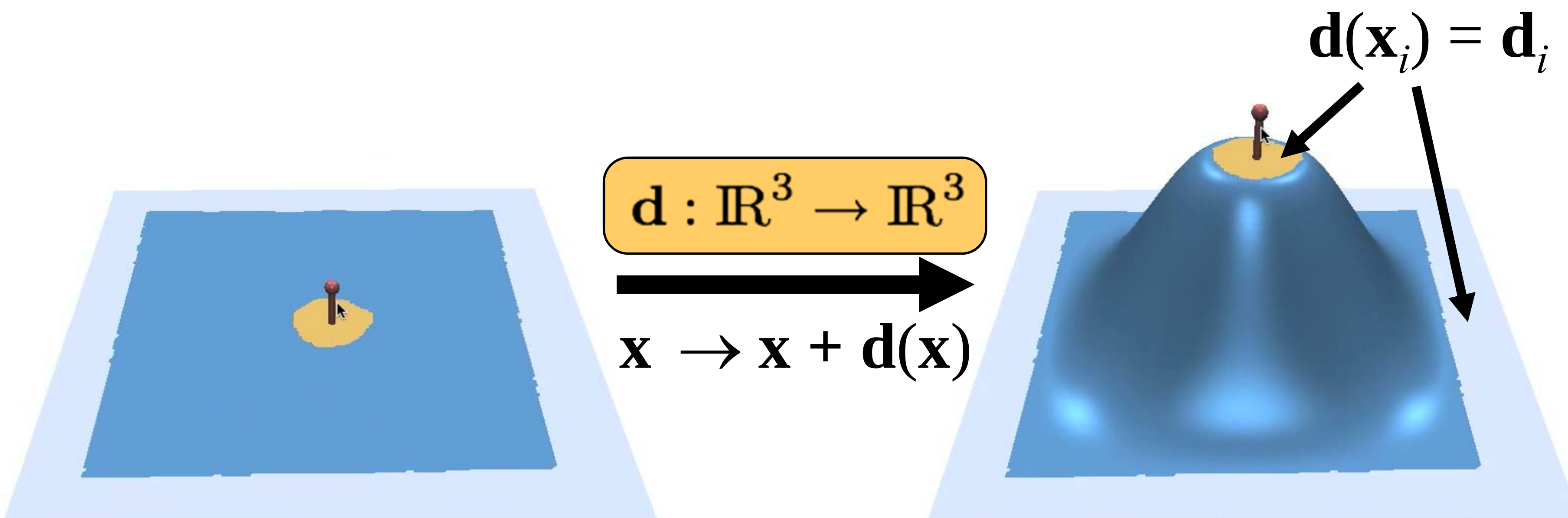




Handle Metaphor

[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

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

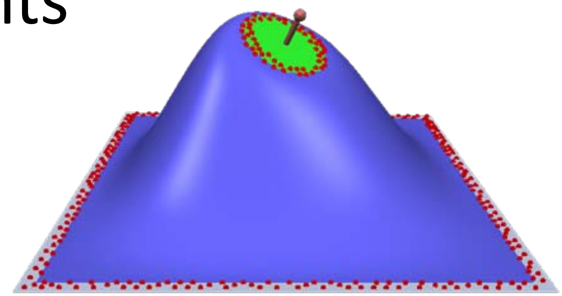
RBF Fitting

[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 $\mathbf{p}$



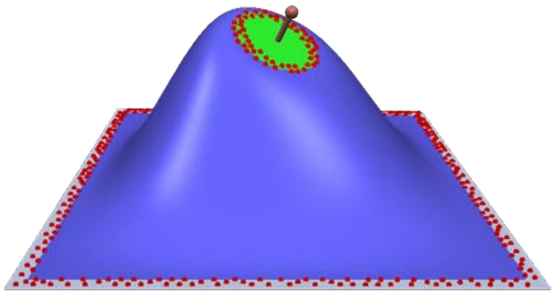
RBF Fitting

[RBF, Botsch and Kobbelt 05]

- Represent deformation by RBFs

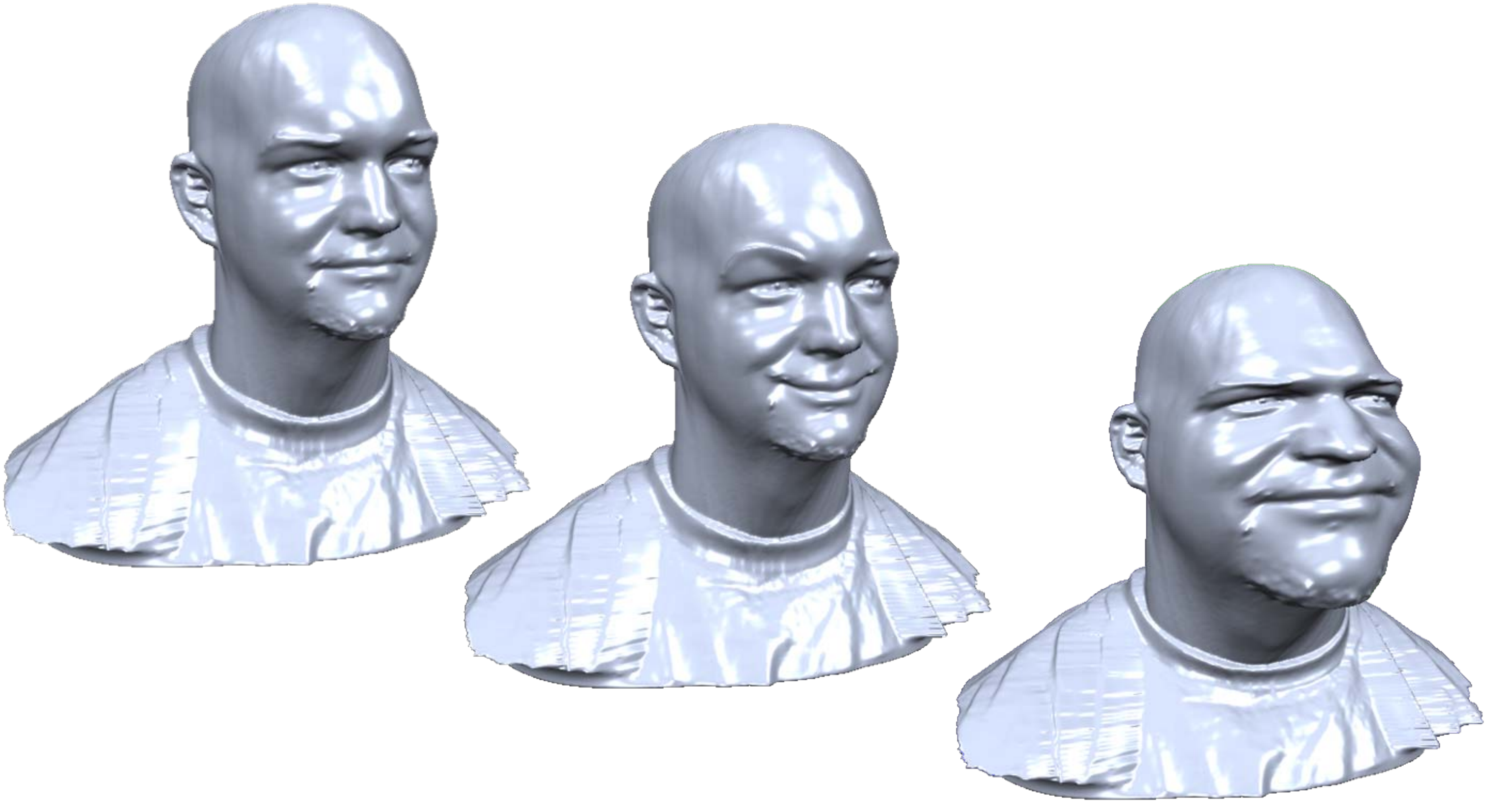
$$\mathbf{d}(\mathbf{x}) = \sum_j \mathbf{w}_j \cdot \varphi(\| \mathbf{c}_j - \mathbf{x} \|) + \mathbf{p}(\mathbf{x})$$

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



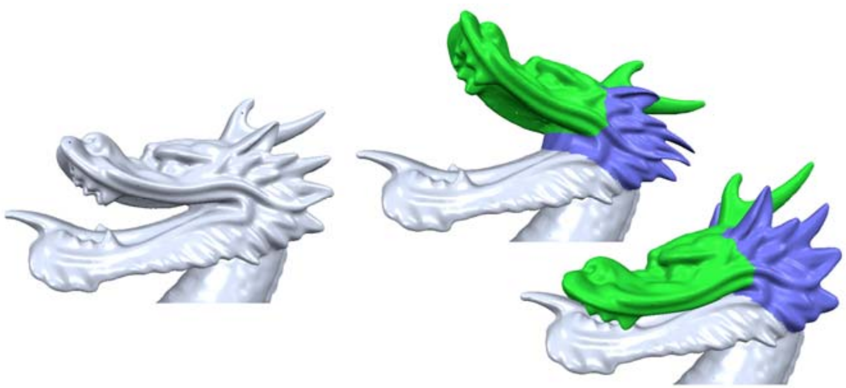
Local & Global Deformations

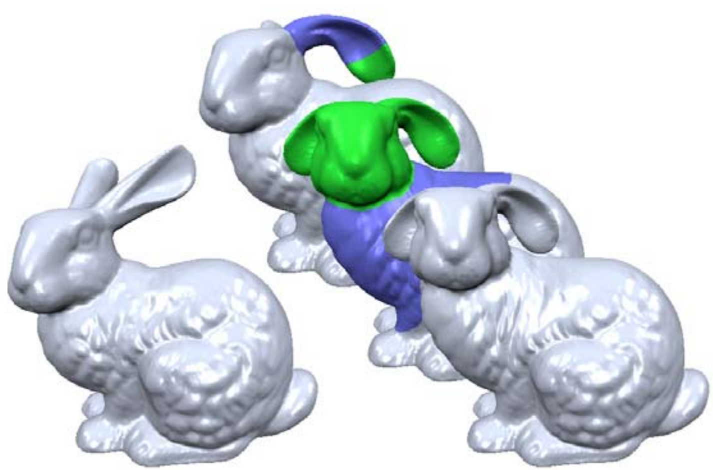
[RBF, Botsch and Kobbelt 05]

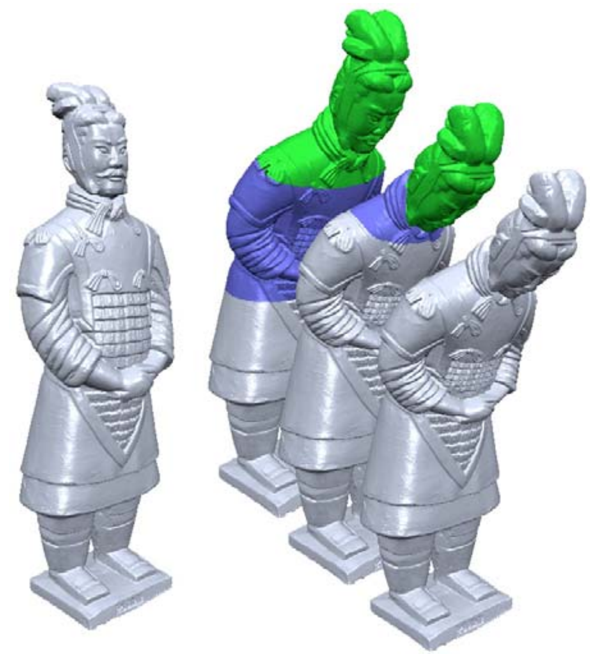


Local & Global Deformations

[RBF, Botsch and Kobbelt 05]





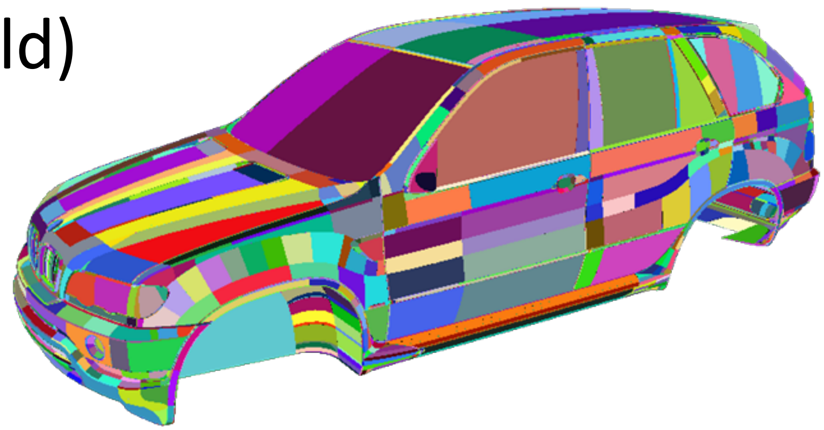


1M vertices
movie

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

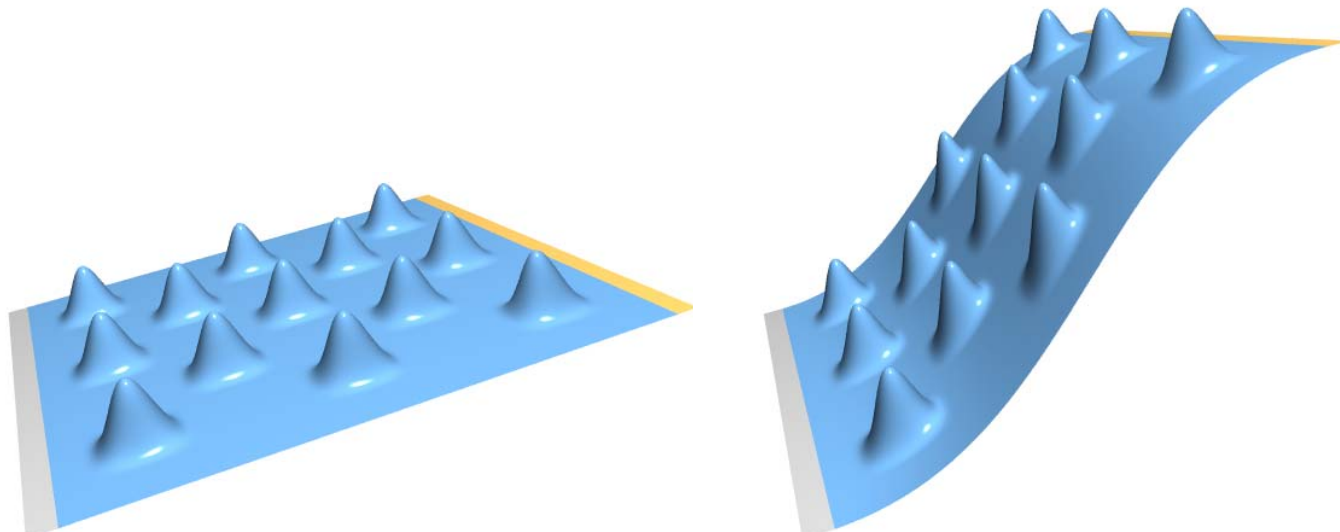
$$Jac(\mathbf{F}) = \begin{vmatrix} \frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} & \frac{\partial F}{\partial z} \\ \frac{\partial G}{\partial x} & \frac{\partial G}{\partial y} & \frac{\partial G}{\partial z} \\ \frac{\partial H}{\partial x} & \frac{\partial H}{\partial y} & \frac{\partial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

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

Space Deformations

Summary so far

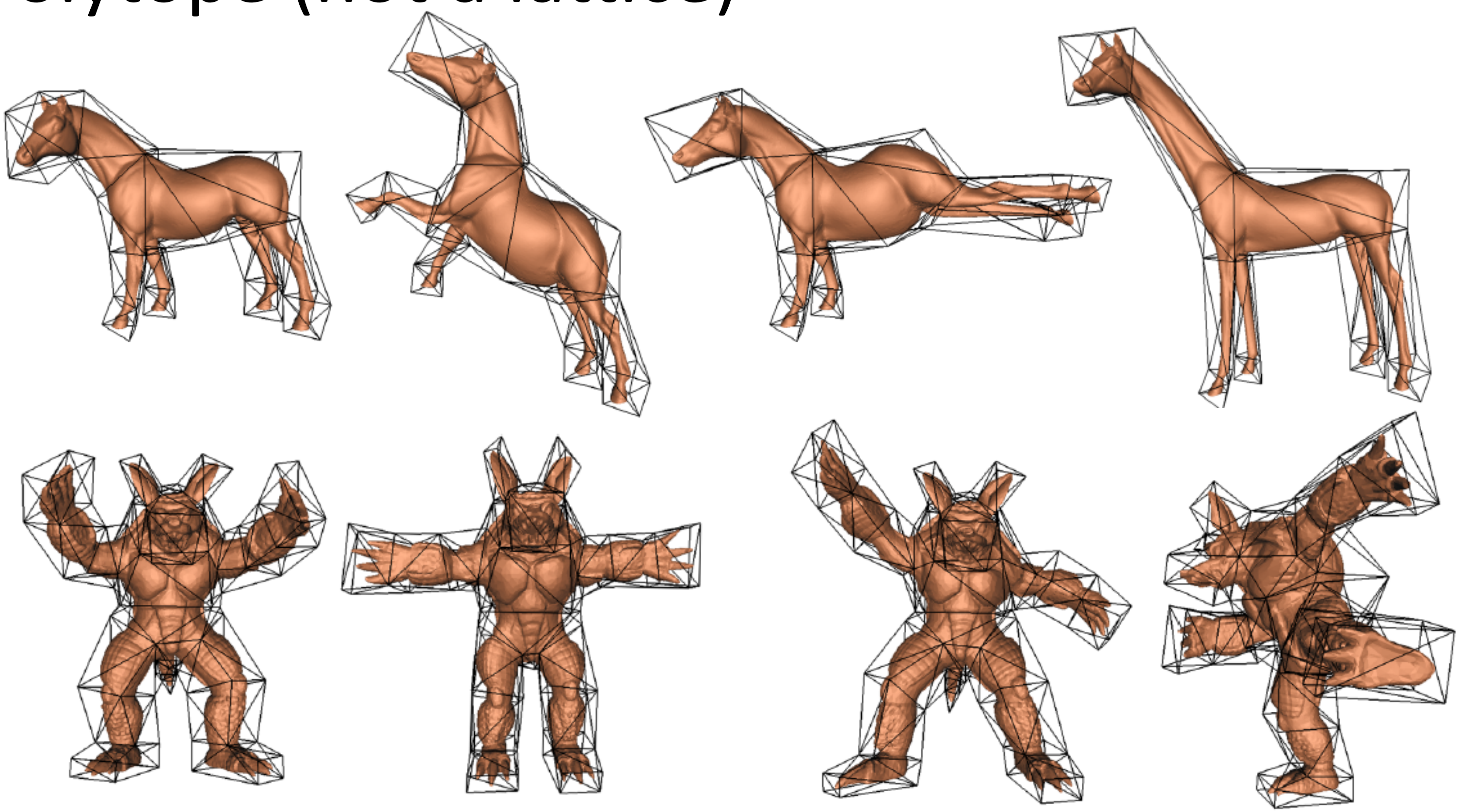
- The deformation is only loosely aware of the shape that is being edited
- Small Euclidean distance $\rightarrow$ similar deformation
- Local surface detail may be distorted



Cage-based Deformations

[Ju et al. 2005]

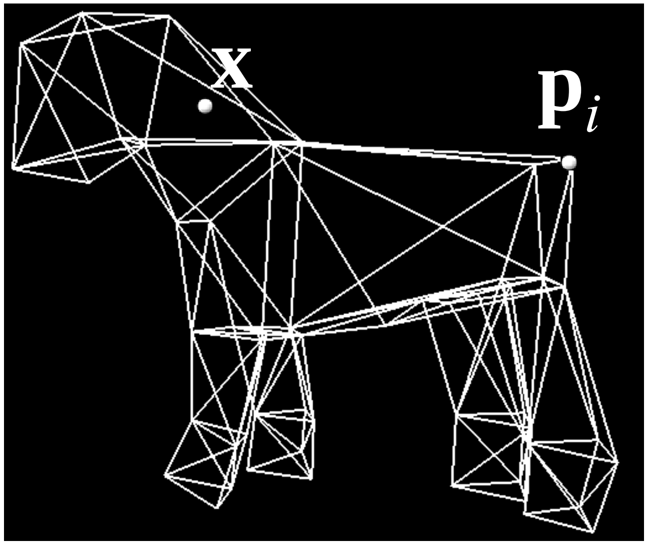
- Cage = crude version of the input shape
- Polytope (not a lattice)



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

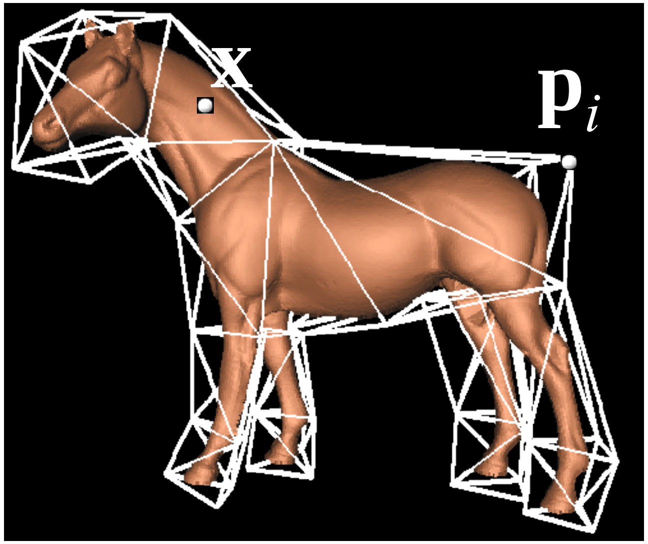


$$\mathbf{x} = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}_i$$

Cage-based Deformations

[Ju et al. 2005]

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

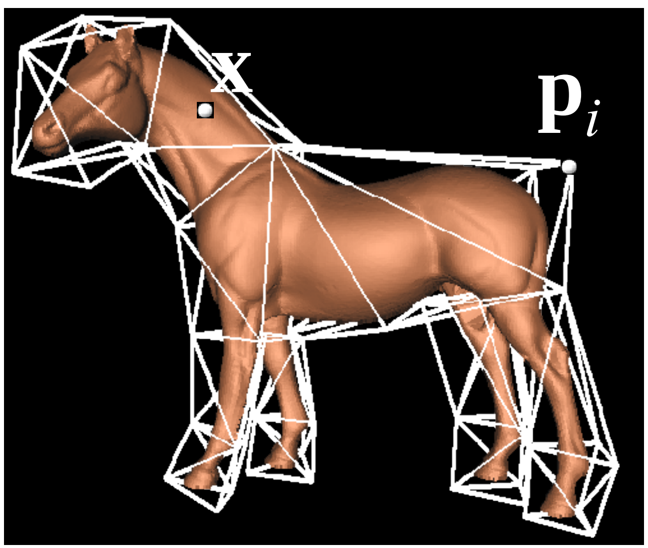


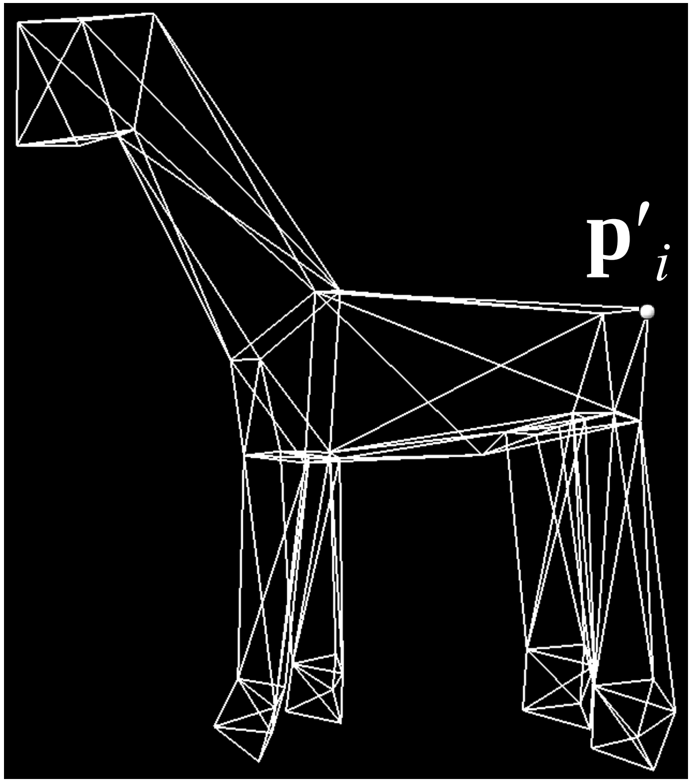
$$\mathbf{x} = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}_i$$

Cage-based Deformations

[Ju et al. 2005]

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



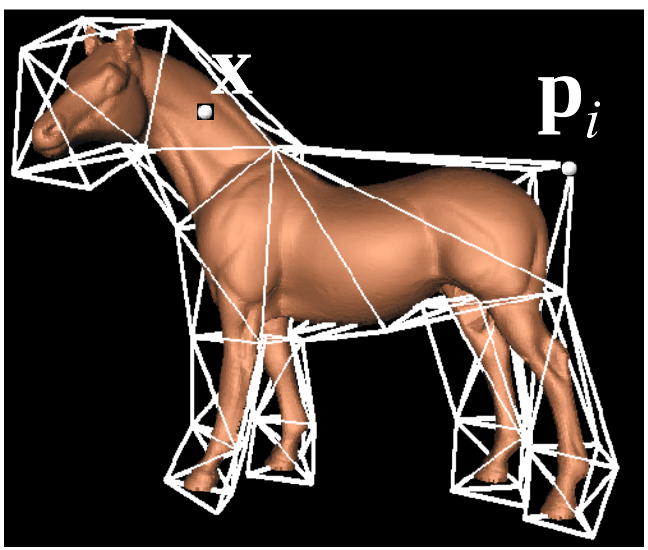


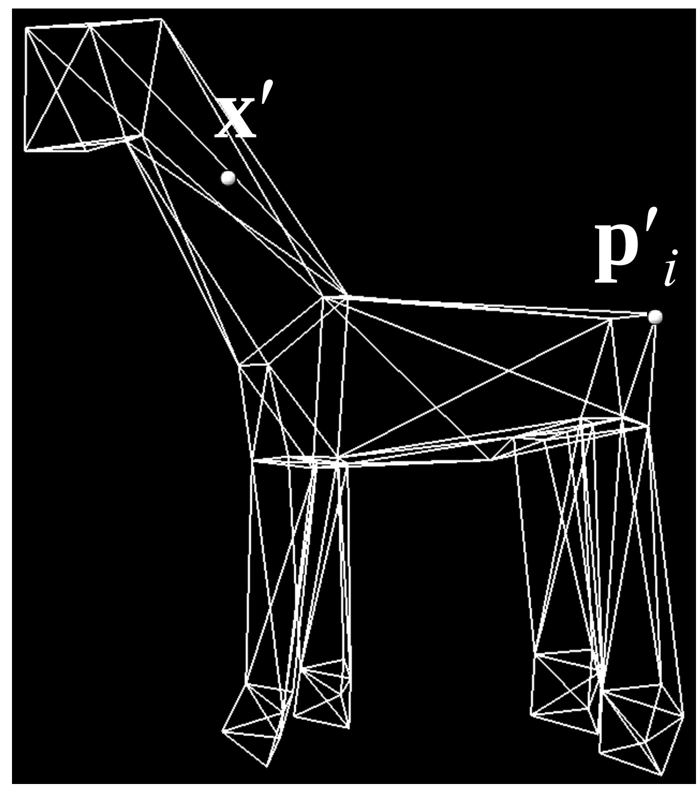
Cage-based Deformations

[Ju et al. 2005]

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

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



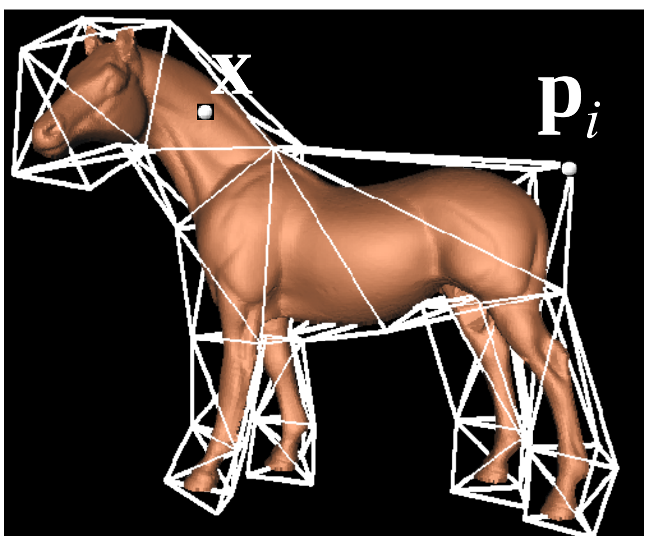


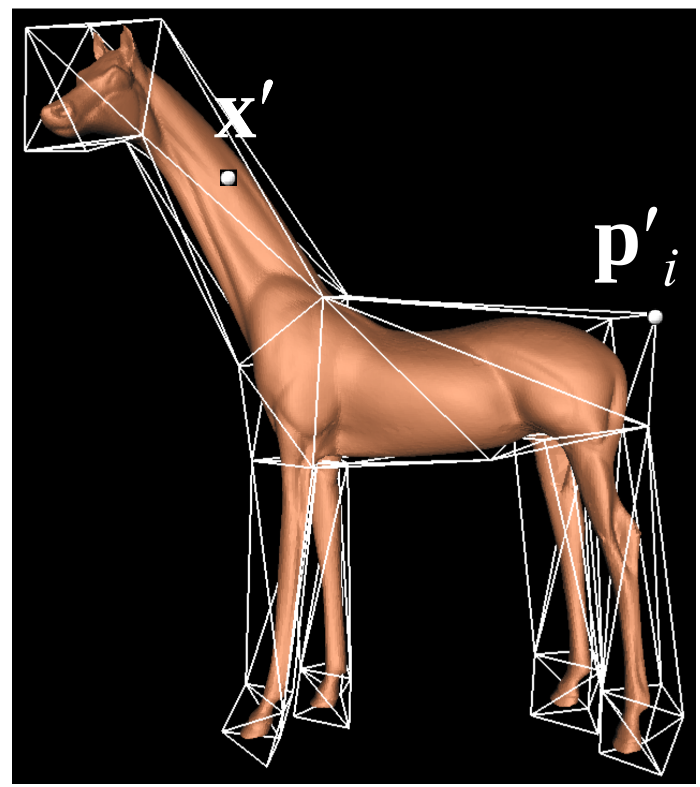
Cage-based Deformations

[Ju et al. 2005]

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

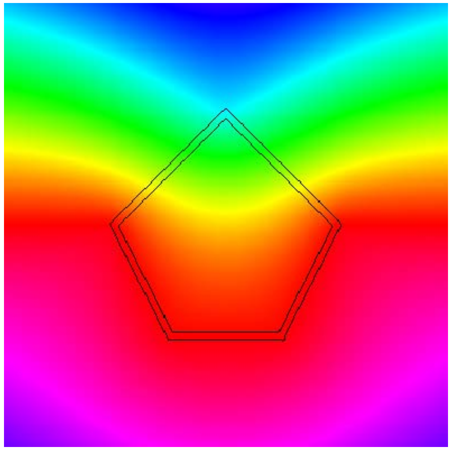
$$\mathbf{x}' = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}_i'$$

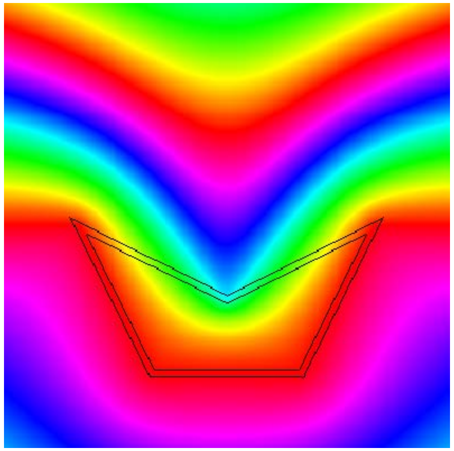


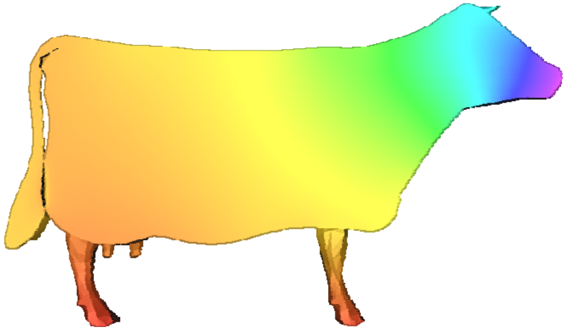


Coordinate Functions

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

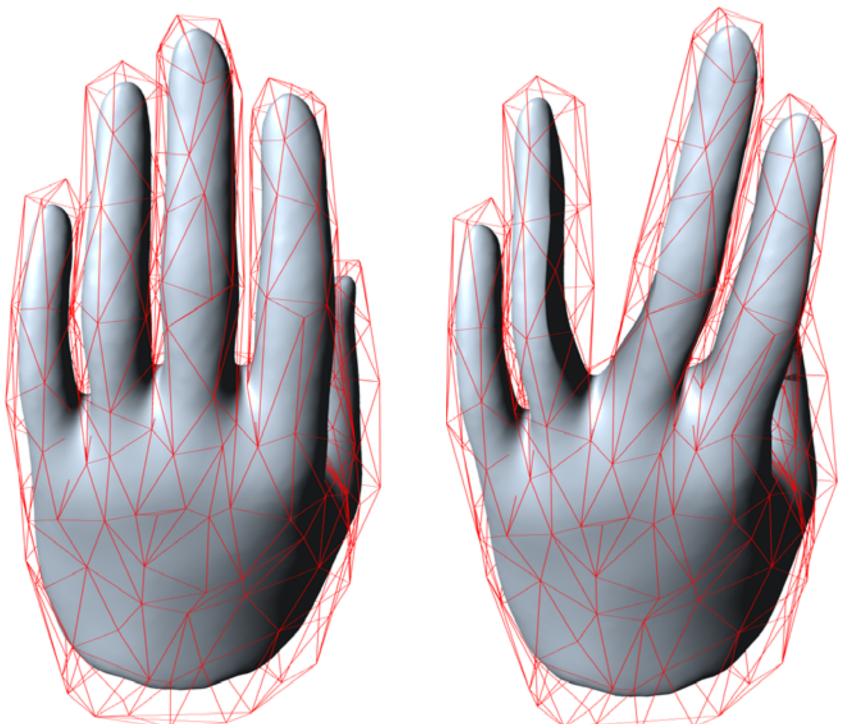






Coordinate Functions

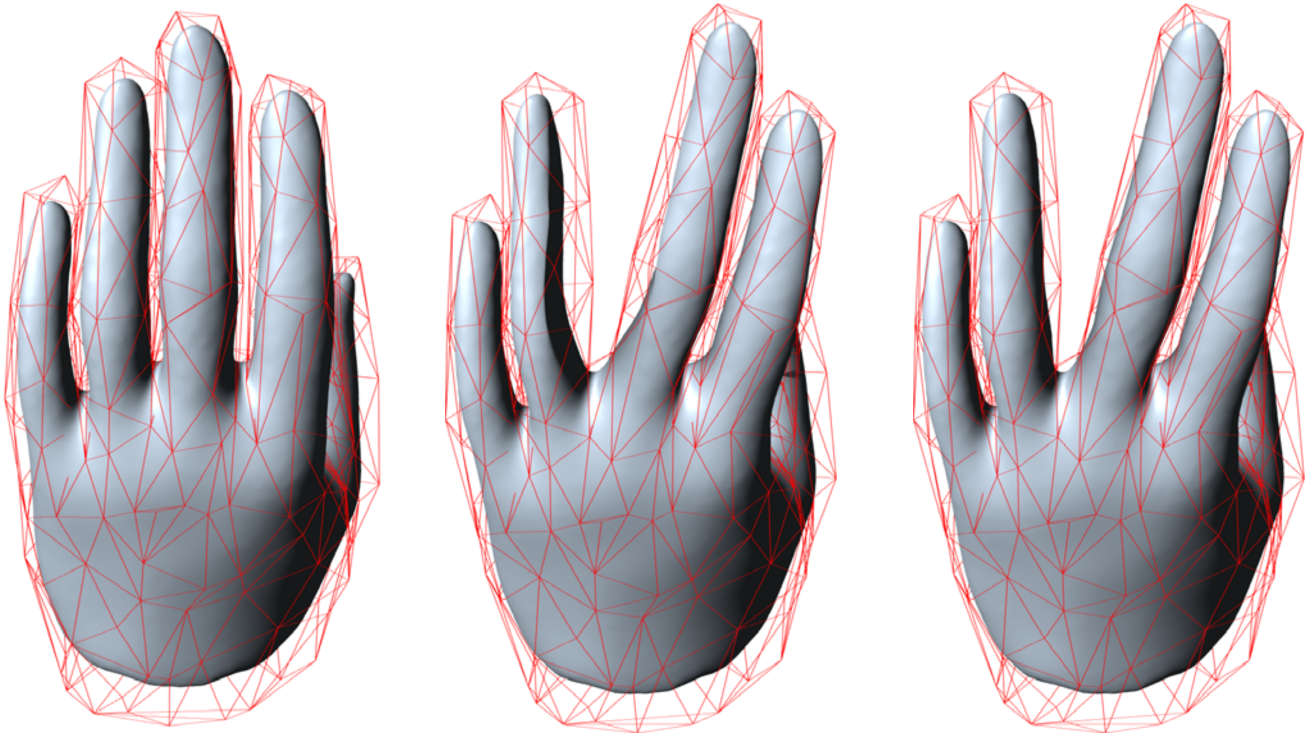
- Mean-value coordinates (Floater, Ju et al. 2005)
  - Not necessarily positive on non-convex domains



MVC

Coordinate Functions

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



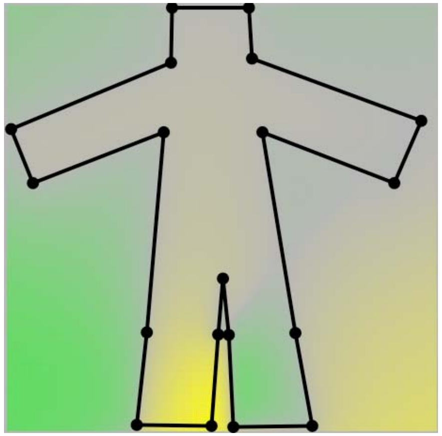
MVC PMVC

Coordinate Functions

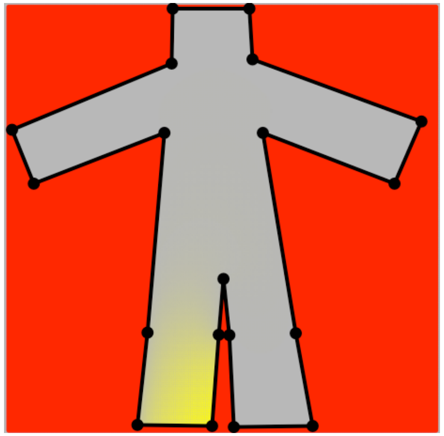
- Harmonic coordinates (Joshi et al. 2007)
  - Harmonic functions $h_i(\mathbf{x})$ for each cage vertex $\mathbf{p}_i$
  - Solve

$$\Delta h = 0$$

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



MVC



HC

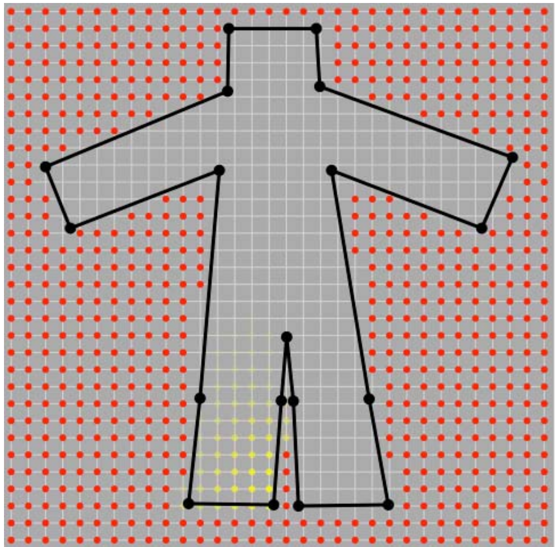
Coordinate Functions

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

$$\Delta h = 0$$

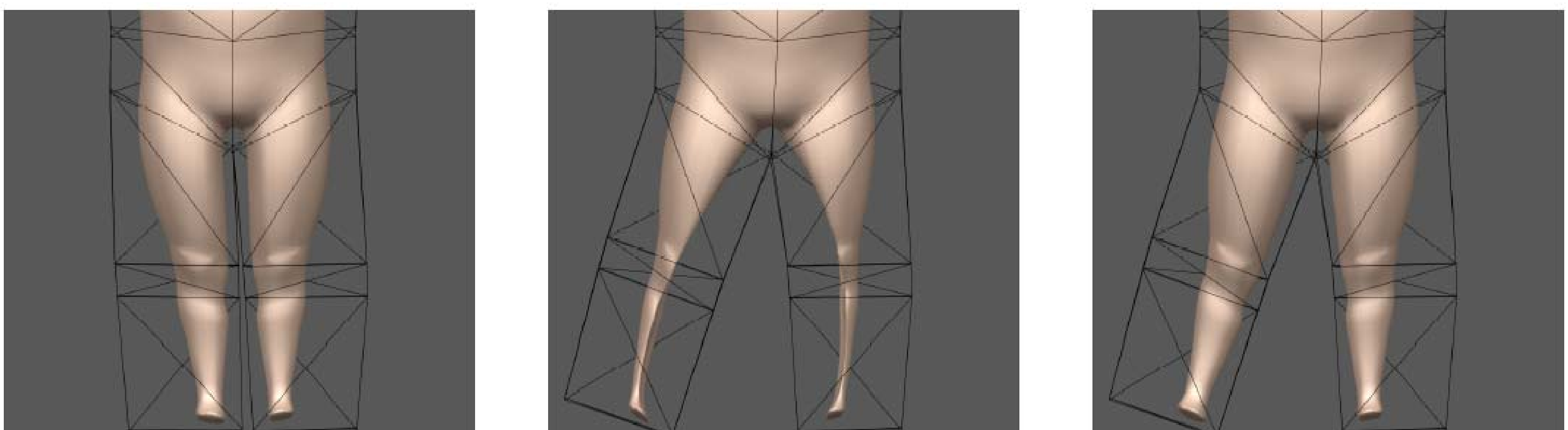
    subject to: h_i linear on the boundary s.t. $h_i(\mathbf{p}_i) = \delta_{ij}$

  - Volumetric Laplace equation
  - Discretization, no closed-form



Coordinate Functions

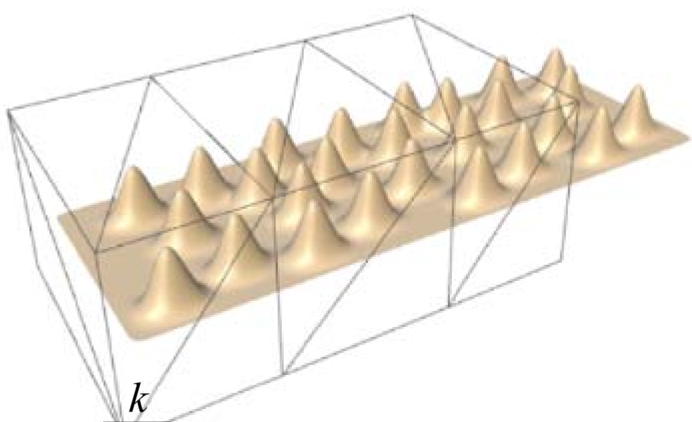
- Harmonic coordinates (Joshi et al. 2007)



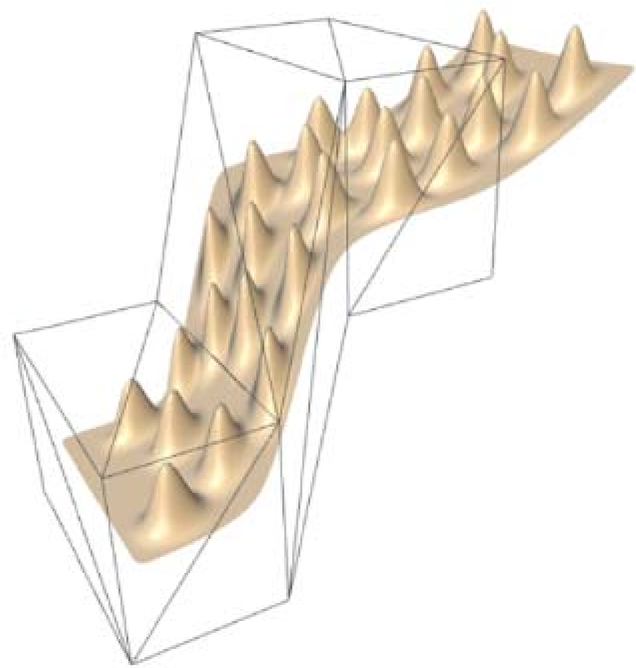
MVC HC

Coordinate Functions

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

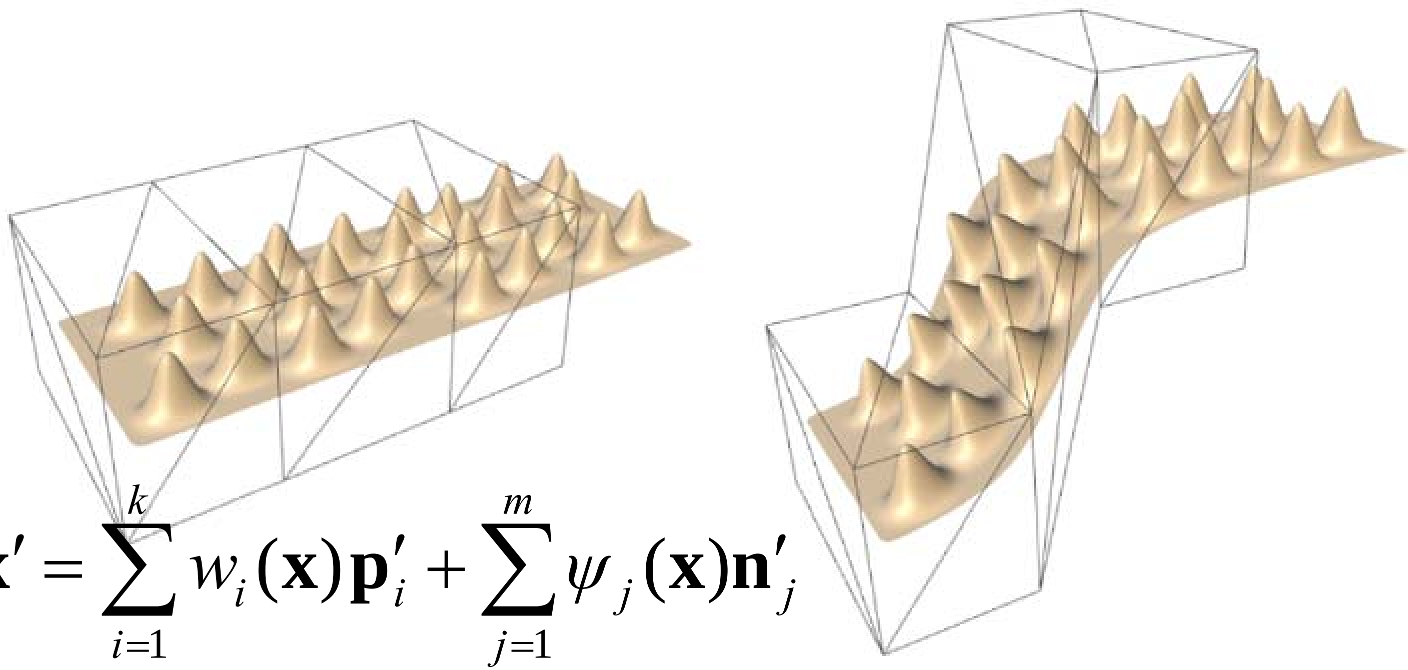


$$\mathbf{x}' = \sum_{i=1}^{k} w_i(\mathbf{x})\mathbf{p}_i'$$



Coordinate Functions

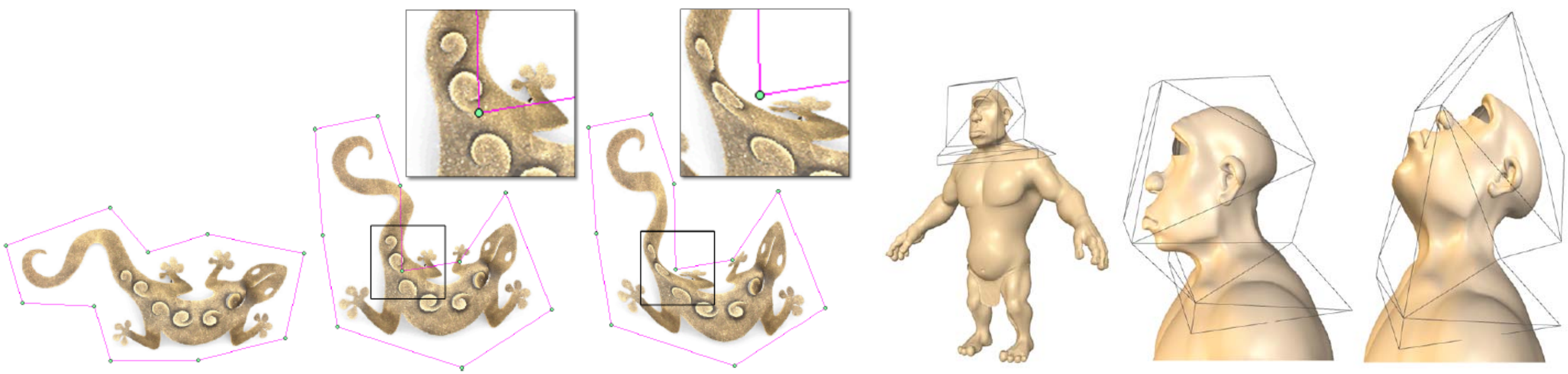
- Green coordinates (Lipman et al. 2008)
- Correction: Make the coordinates depend on the cage faces as well



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

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**

Coffee/Tea Break

Resume at 11:00

Summary of Linear Methods

Prof. Dr. Mario Botsch

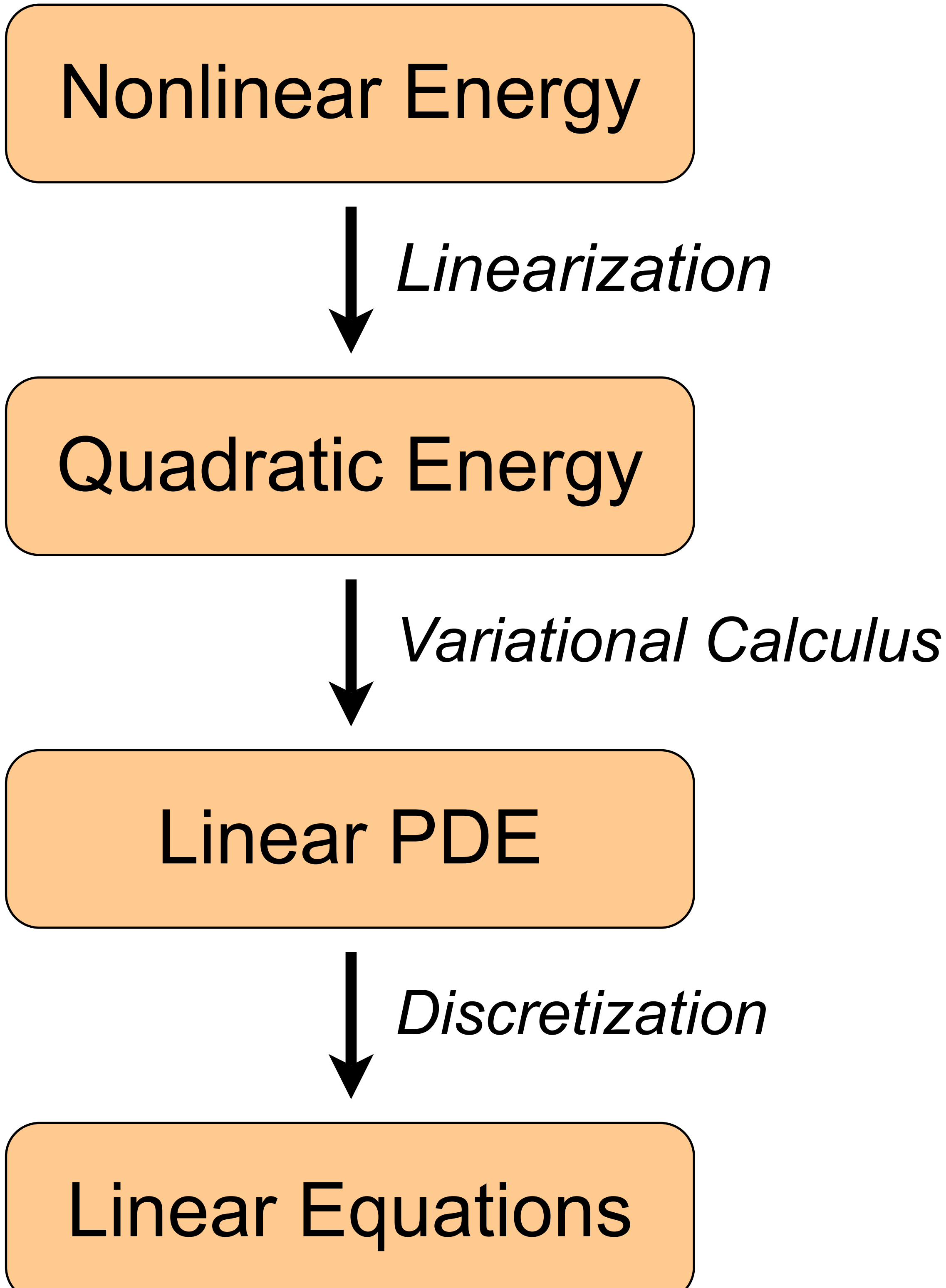
Computer Graphics & Geometry Processing

Bielefeld University

Universität Bielefeld

CITEC

Linear Approaches



Linear Approaches

- Resulting linear systems
  - Shell-based $\Delta^2 \mathbf{d} = \mathbf{0}$
  - Gradient-based $\Delta \mathbf{p} = \nabla \cdot \mathbf{T}(\mathbf{g})$
  - Laplacian-based $\Delta^2 \mathbf{p} = \Delta\, \mathbf{T}(\mathbf{l})$
- Properties
  - Highly sparse
  - Symmetric, positive definite (*SPD*)
  - Solve for new RHS each frame!

Linear SPD Solvers

- **Dense Cholesky factorization**
  - Cubic complexity
  - High memory consumption (doesn't exploit sparsity)
- **Iterative conjugate gradients**
  - Quadratic complexity
  - Need sophisticated preconditioning
- **Multigrid solvers**
  - Linear complexity
  - But rather complicated to develop (and to use)
- **Sparse Cholesky factorization**
  - Linear complexity
  - Easy to use

Dense Cholesky Factorization

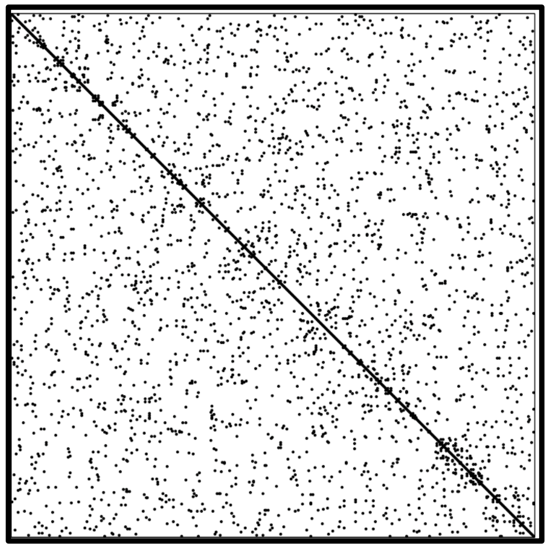
Solve $\mathbf{Ax} = \mathbf{b}$

1. Cholesky factorization $\mathbf{A} = \mathbf{LL}^T$
2. Solve system $\mathbf{y} = \mathbf{L}^{-1}\mathbf{b}, \quad \mathbf{x} = \mathbf{L}^{-T}\mathbf{y}$

Dense Cholesky Factorization

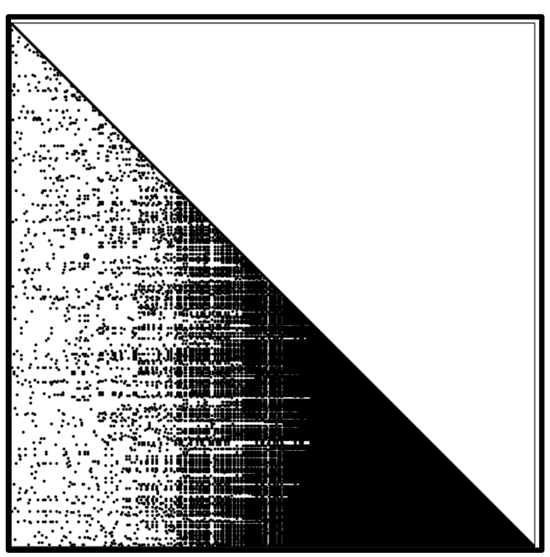
$A=LL^T$

500×500 matrix
3500 non-zeros



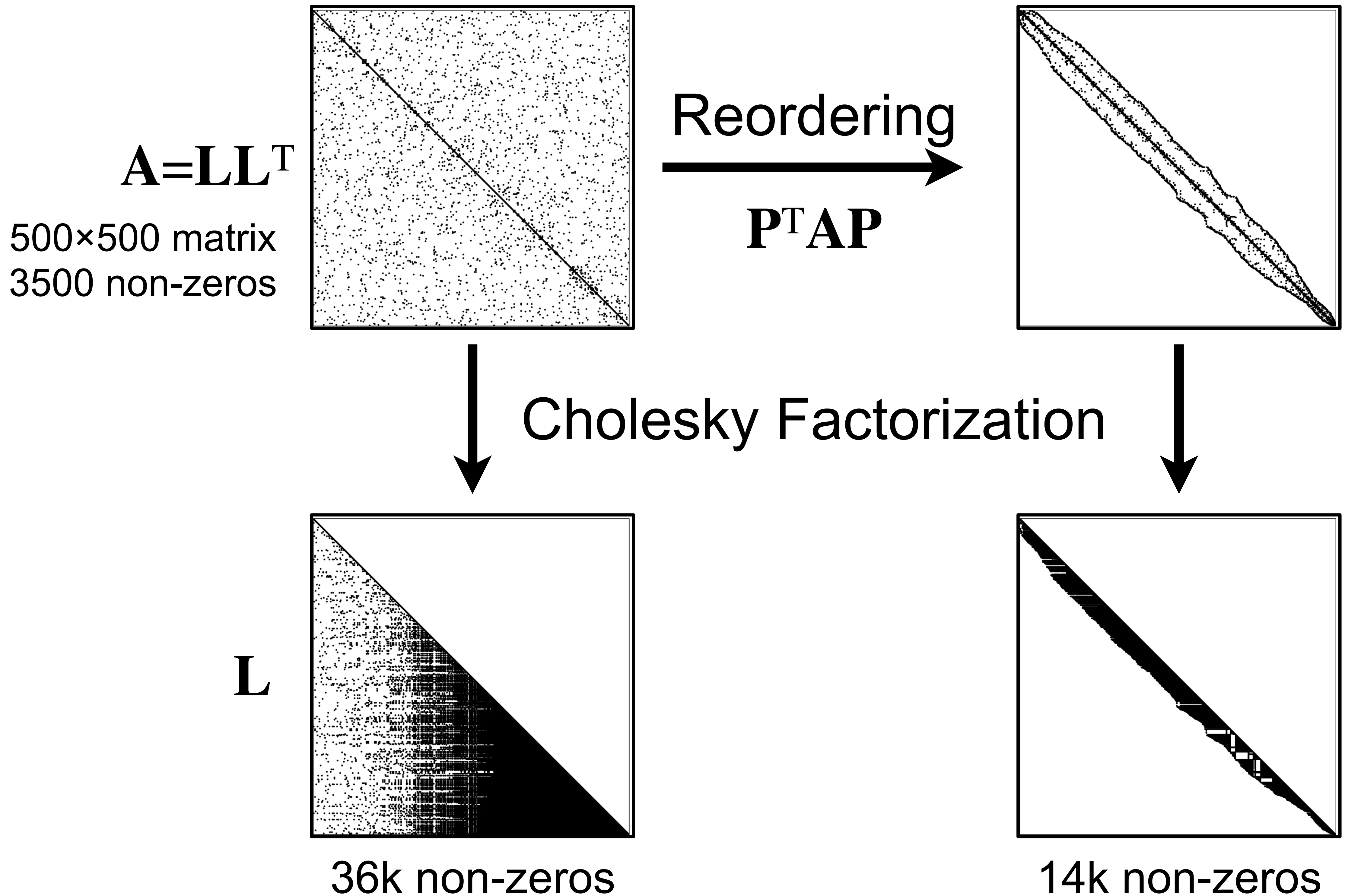
Cholesky Factorization

L

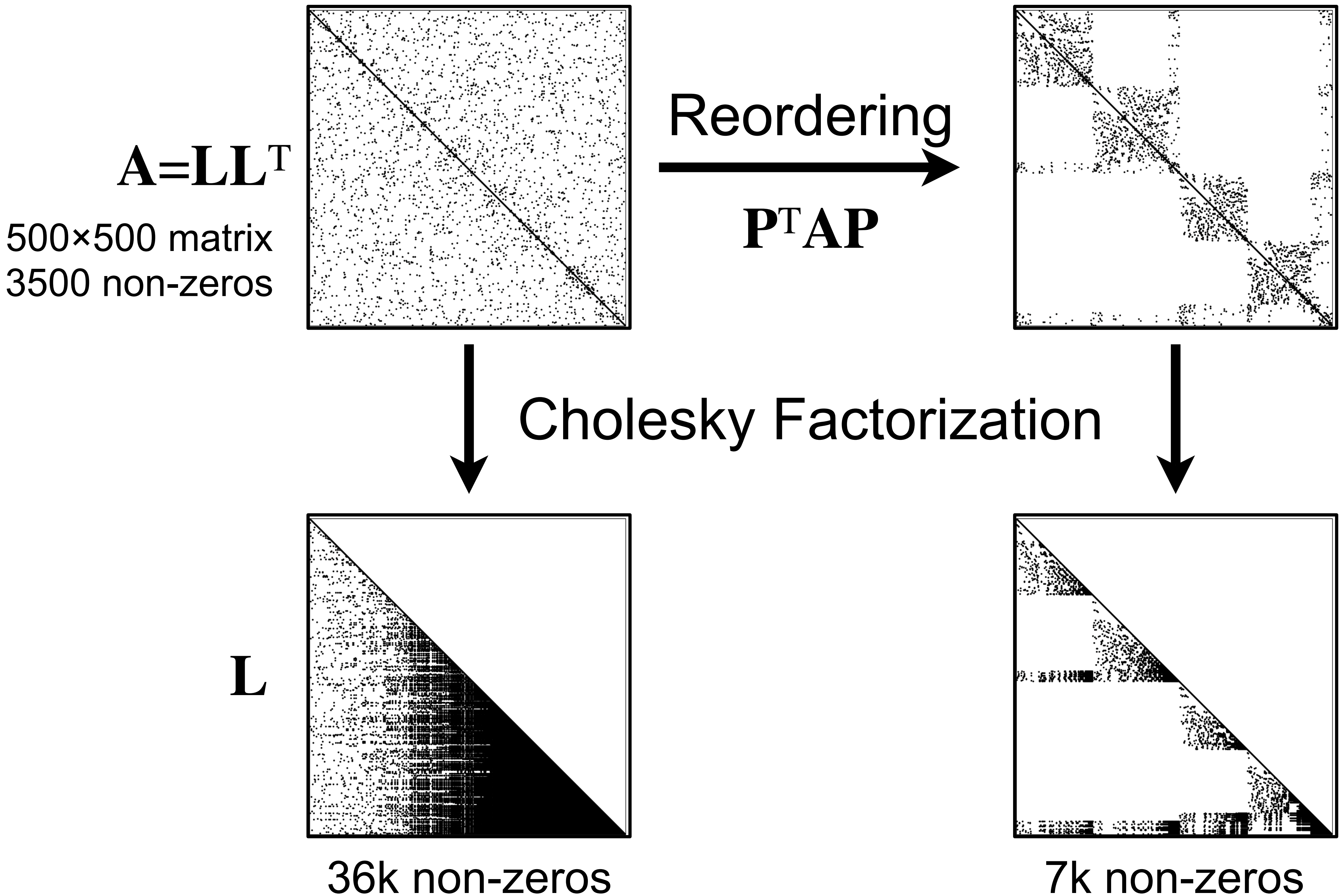


36k non-zeros

Sparse Cholesky Factorization



Sparse Cholesky Factorization



Sparse Cholesky Solver

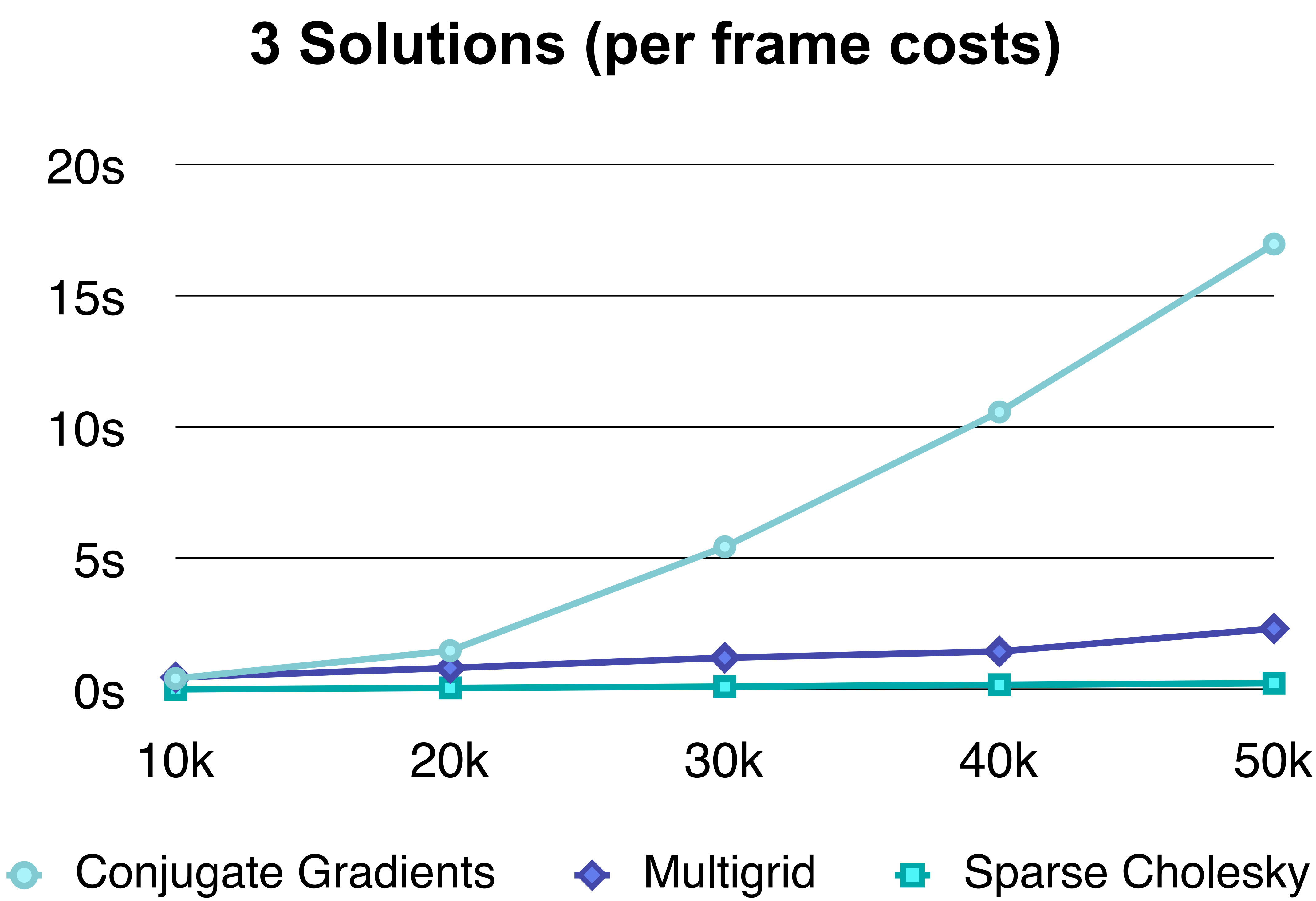
Solve $\mathbf{A}\mathbf{x} = \mathbf{b}$

Pre-computation

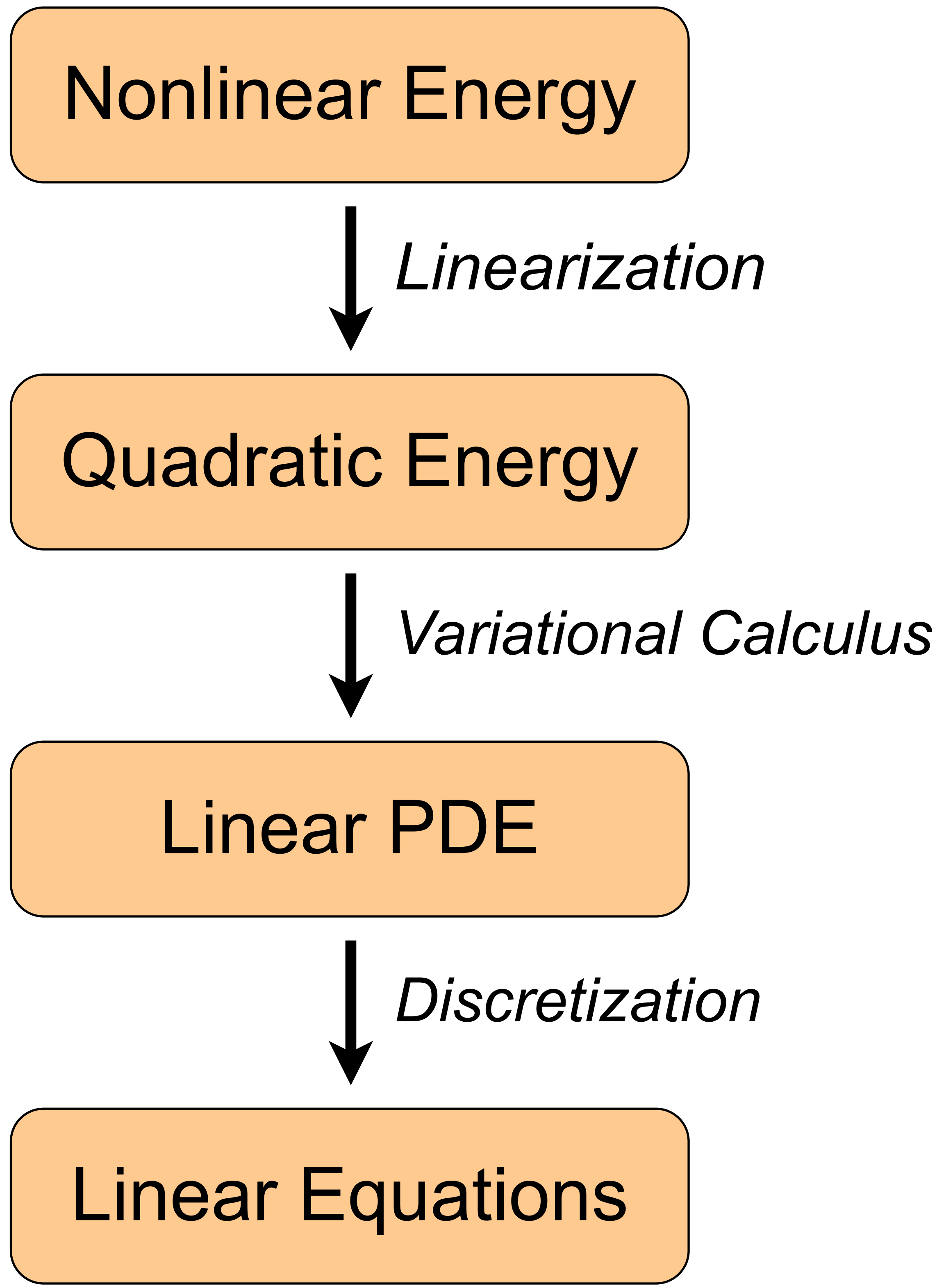
1. Matrix re-ordering $\tilde{\mathbf{A}} = \mathbf{P}^T\mathbf{A}\mathbf{P}$
2. Cholesky factorization $\tilde{\mathbf{A}} = \mathbf{L}\mathbf{L}^T$
3. Solve system $\mathbf{y} = \mathbf{L}^{-1}\mathbf{P}^T\mathbf{b}, \quad \mathbf{x} = \mathbf{P}\mathbf{L}^{-T}\mathbf{y}$

Per-frame computation

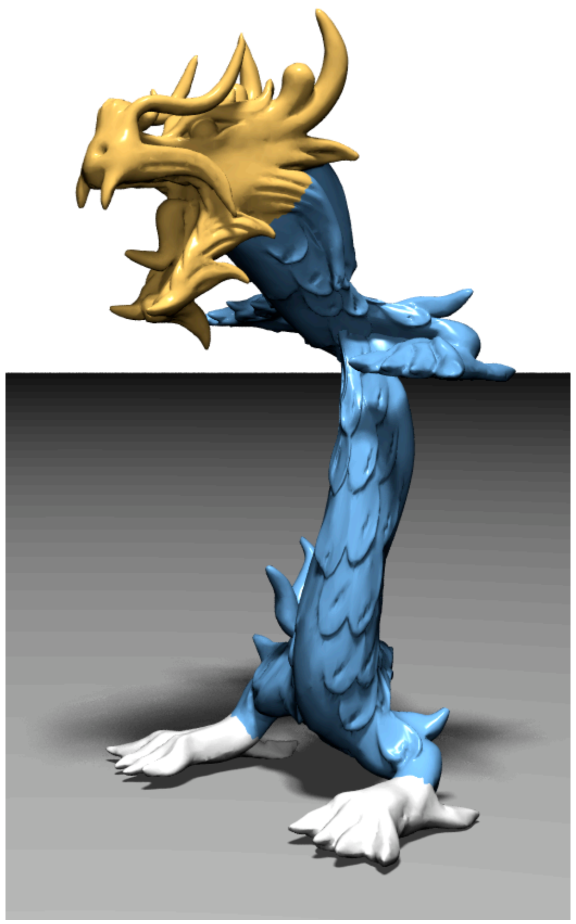
Bi-Laplace Systems



Linear Approaches



Linear vs. Nonlinear



Shell

Gradient

Nonlinear

Linear Approaches

Nonlinear Energy

↓ *Linearization* ← **causes artifacts**

Quadratic Energy

↓ *Variational Calculus*

Linear PDE

↓ *Discretization*

Linear Equations

Linearizations / Simplifications

- **Shell-based deformation**

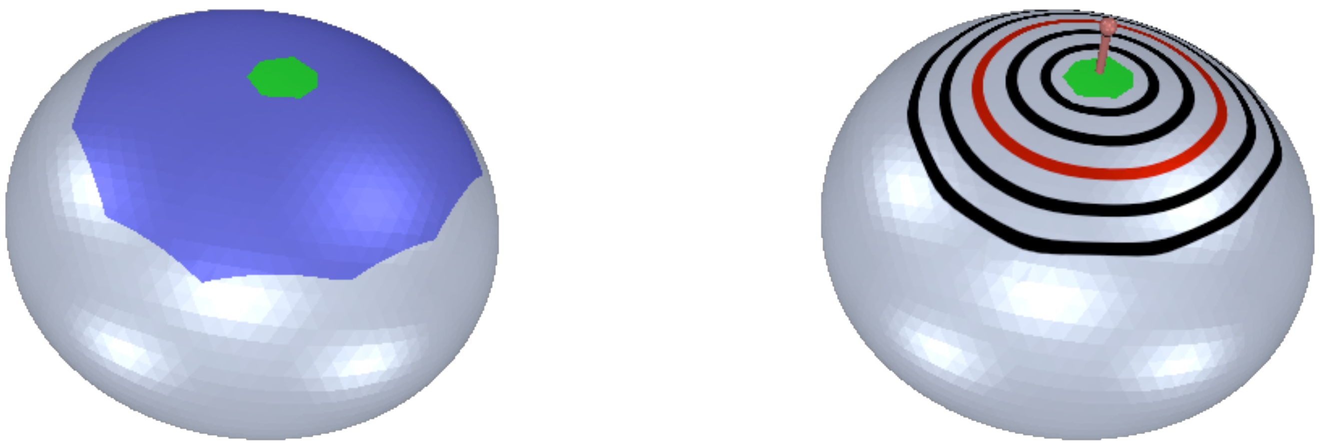
$$\int_\Omega k_s \left\|\mathbf{I} - \mathbf{I}'\right\|^2 + k_b \left\|\mathbf{II} - \mathbf{II}'\right\|^2 \, \mathrm{d}u\mathrm{d}v$$

$$\int_\Omega k_s \left(\|\mathbf{d}_u\|^2 + \|\mathbf{d}_v\|^2 \right) \; + \; k_b \left(\|\mathbf{d}_{uu}\|^2 + 2\,\|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2 \right) \mathrm{d}u\mathrm{d}v$$

Linearizations / Simplifications

- **Gradient-based editing**

$$\nabla \mathbf{T}(\mathbf{x}) = \mathbf{A}$$



Linearizations / Simplifications

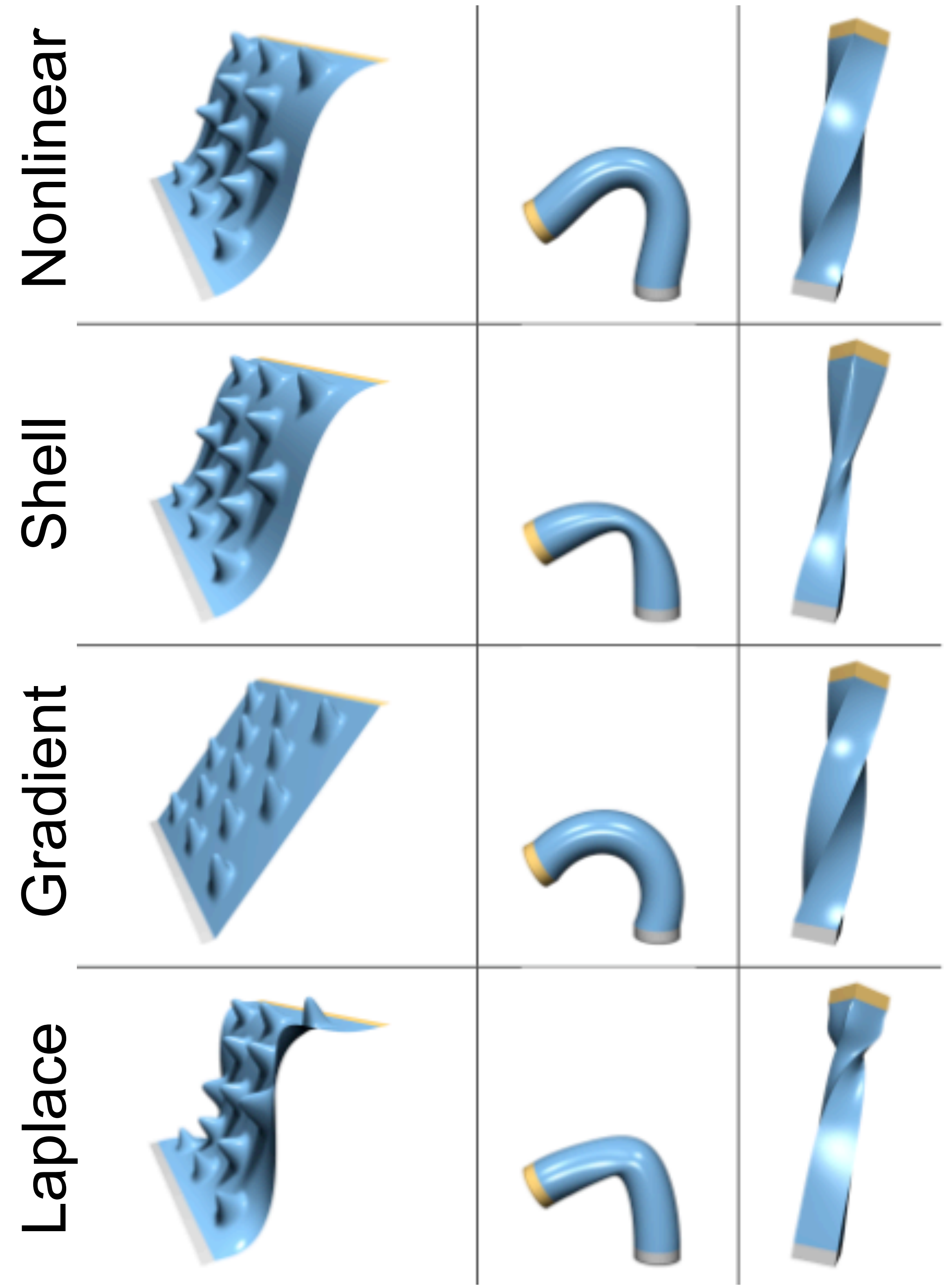
- **Laplacian surface editing**

$$\mathbf{R}\mathbf{x} \;\approx\; \mathbf{x} + (\mathbf{r} \times \mathbf{x}) \;=\; \begin{pmatrix} 1 & -r_3 & r_2 \\ r_3 & 1 & -r_1 \\ -r_2 & r_1 & 1 \end{pmatrix} \mathbf{x}$$

$$\mathbf{T}_i \;=\; \begin{pmatrix} s & -r_3 & r_2 \\ r_3 & s & -r_1 \\ -r_2 & r_1 & s \end{pmatrix}$$

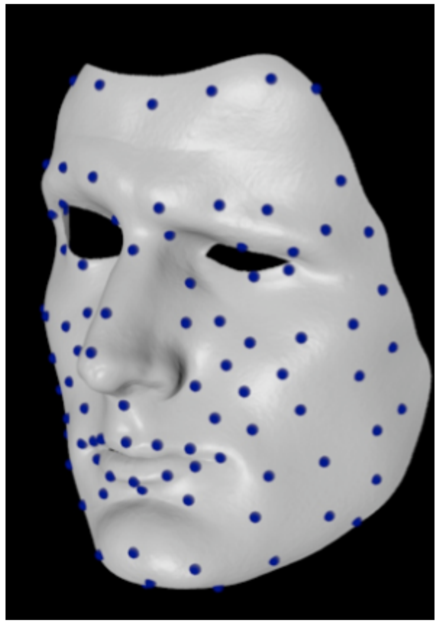
Linear vs. Nonlinear

- Analyze existing methods
  - Some work for translations
  - Some work for rotations
  - No method works for both



Linear vs. Nonlinear

- Linear approaches
  - Solve linear system each frame
  - Small deformations
  - Dense constraints



- Nonlinear approaches
  - Solve nonlinear problem each frame
  - Large deformations
  - Sparse constraints

Nonlinear Surface-Based Deformation

Prof. Dr. Mario Botsch

Computer Graphics & Geometry Processing
Bielefeld University

Universität Bielefeld

CITEC

Nonlinear Surface Deformation

- **Nonlinear Optimization**
- Shell-Based Deformation
- (Differential Coordinates)

Nonlinear Minimization

- Given a nonlinear deformation energy

$$E(\mathbf{d}) \;=\; E(\mathbf{d}_1, \ldots, \mathbf{d}_n)$$

find the displacement $\mathbf{d}(\mathbf{x})$ that minimizes $E(\mathbf{d})$, while satisfying the modeling constraints.

- Typically $E(\mathbf{d})$ stays the same, but the modeling constraints change each frame.

Gradient Descent

- Start with initial guess $\mathbf{d}_0$
- Iterate until convergence
  - Find descent direction $\mathbf{h} = -\nabla E(\mathbf{d})$
  - Find step size λ
  - Update $\mathbf{d} = \mathbf{d} + \lambda\mathbf{h}$
- Properties
  + Easy to implement, guaranteed convergence
  – Slow convergence

Newton's Method

- Start with initial guess $\mathbf{d}_0$
- Iterate until convergence
  - Find descent direction as $\mathbf{H}(\mathbf{d})\,\mathbf{h} = -\nabla E(\mathbf{d})$
  - Find step size λ
  - Update $\mathbf{d} = \mathbf{d} + \lambda\mathbf{h}$
- Properties
  - + Fast convergence if close to minimum
  - – Needs pos. def. $\mathbf{H}$, needs 2nd derivatives for $\mathbf{H}$

Nonlinear Least Squares

Given a nonlinear vector-valued error function

$$\mathbf{e}(\mathbf{d}_1, \ldots, \mathbf{d}_n) = \begin{pmatrix} e_1(\mathbf{d}_1, \ldots, \mathbf{d}_n) \\ \vdots \\ e_m(\mathbf{d}_1, \ldots, \mathbf{d}_n) \end{pmatrix}$$

find the displacement $\mathbf{d}(\mathbf{x})$ that minimizes the nonlinear least squares error

$$E(\mathbf{d}_1, \ldots, \mathbf{d}_n) = \frac{1}{2} \|\mathbf{e}(\mathbf{d}_1, \ldots, \mathbf{d}_n)\|^2$$

Gauss-Newton Method

- Start with initial guess $\mathbf{d}_0$
- Iterate until convergence
  - Find descent direction as $(\mathbf{J}(\mathbf{d})^T \mathbf{J}(\mathbf{d}))\, \mathbf{h} = -\mathbf{J}(\mathbf{d})^T \mathbf{e}$
  - Find step size λ
  - Update $\mathbf{d} = \mathbf{d} + \lambda \mathbf{h}$
- Properties
  + Fast convergence if close to minimum
  + Needs full-rank $\mathbf{J}(\mathbf{d})$, needs 1st derivatives for $\mathbf{J}(\mathbf{d})$

Nonlinear Optimization

- Has to solve a linear system each frame
  - Matrix changes in each iteration!
  - Factorize matrix each time
- Numerically more complex
  - No guaranteed convergence
  - Might need several iterations
  - Converges to closest local minimum

➡ Spend more time on fancy solvers...

Nonlinear Surface Deformation

- Nonlinear Optimization
- **Shell-Based Deformation**
- (Differential Coordinates)

Shell-Based Deformation

- **Discrete Shells**

  [Grinspun et al, SCA 2003]

- Rigid Cells

  [Botsch et al, SGP 2006]

- As-Rigid-As-Possible Modeling

  [Sorkine & Alexa, SGP 2007]

Discrete Shells

- Main idea
  - Don’t discretize continuous energy
  - Define **discrete** energy instead
  - Leads to simpler (still nonlinear) formulation
- Discrete energy
  - How to measure stretching on meshes?
  - How to measure bending on meshes?

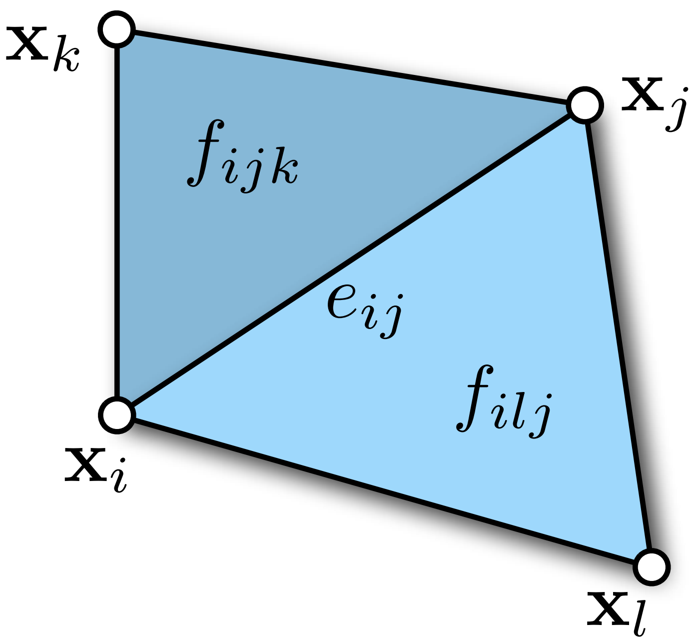
Discrete Shell Energy

- **Stretching**: Change of edge lengths

$$\sum_{e_{ij} \in E} \lambda_{ij} \left(|e_{ij}| - |\bar{e}_{ij}| \right)^2$$

- **Stretching**: Change of triangle areas

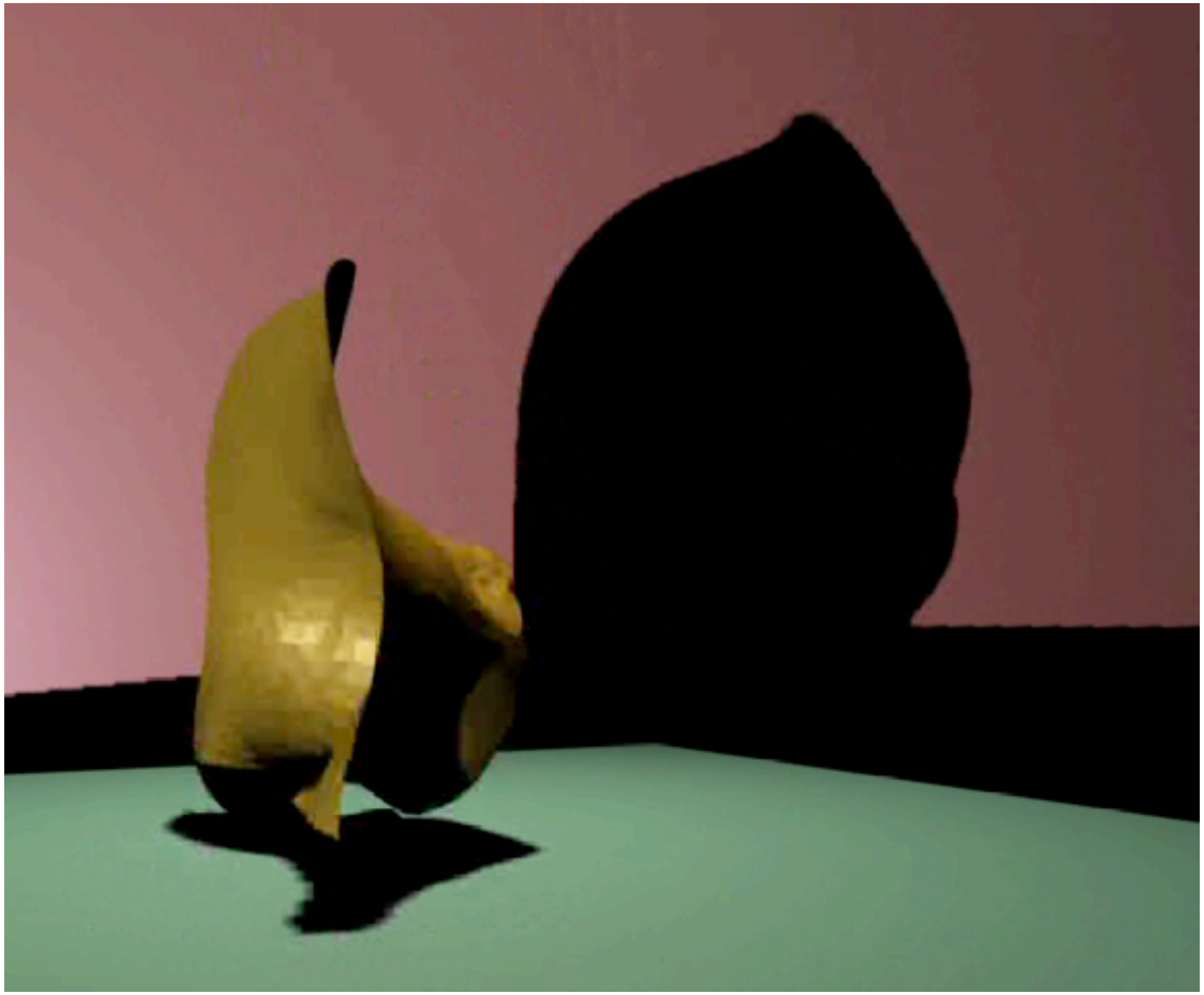
$$\sum_{f_{ijk} \in F} \lambda_{ijk} \left(|f_{ijk}| - |\bar{f}_{ijk}| \right)^2$$



- **Bending**: Change of dihedral angles

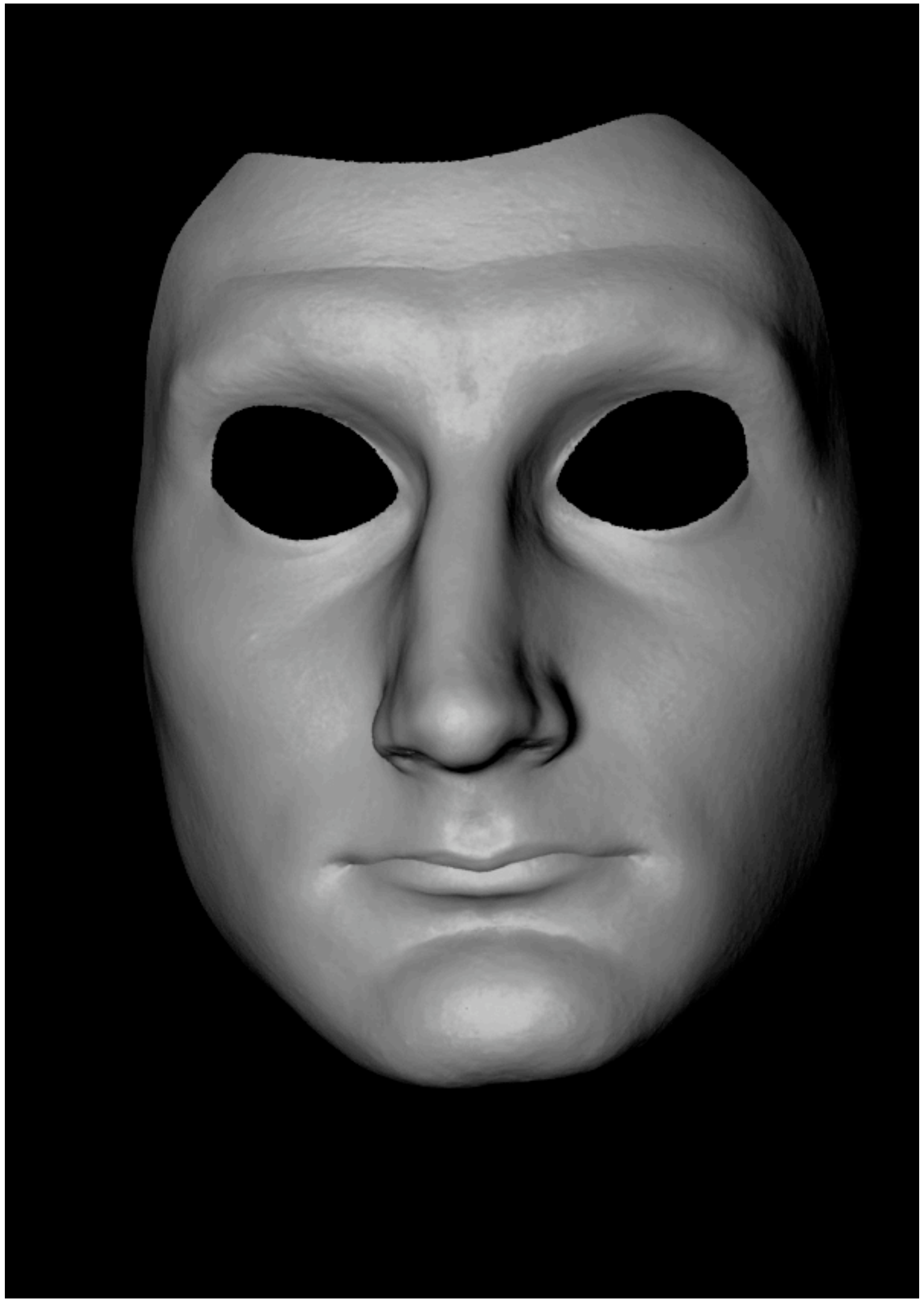
$$\sum_{e_{ij} \in E} \mu_{ij} \left(\theta_{ij} - \bar{\theta}_{ij} \right)^2$$

Discrete Shells



[Grinspun 2003]

Realistic Face Animations



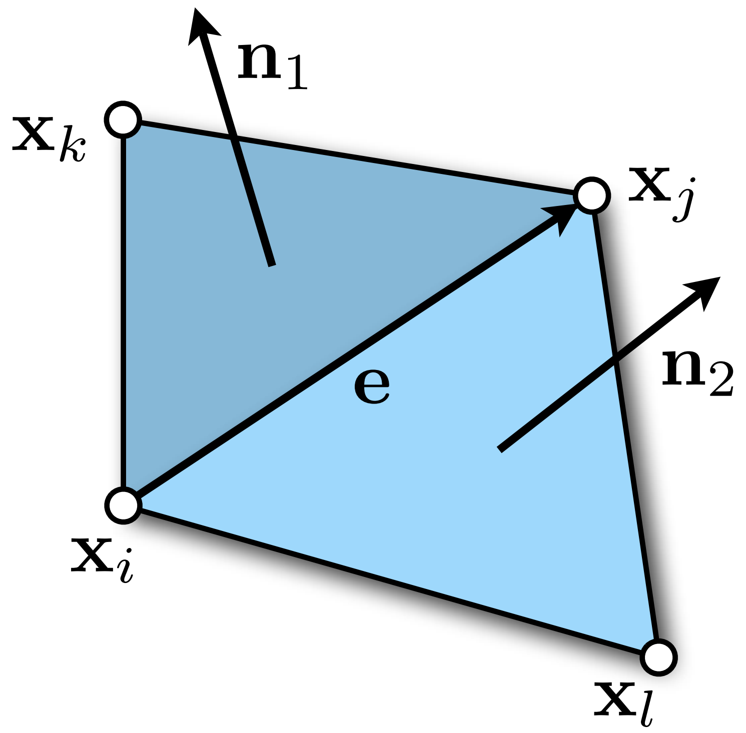
Linear model

Nonlinear model

Discrete Energy Gradients

- Gradients of edge length

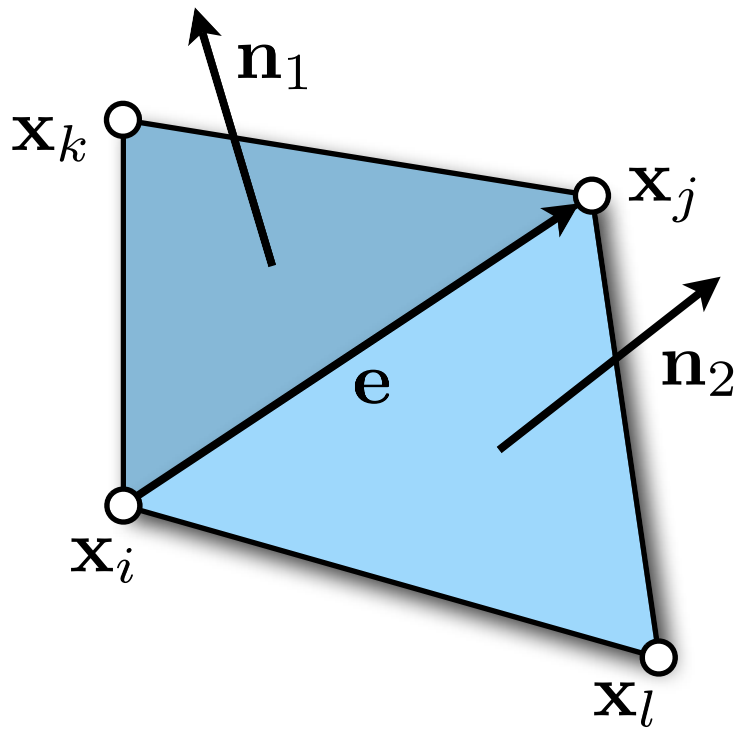
$$
\begin{aligned}
|e_{ij}| &= \|\mathbf{x}_j - \mathbf{x}_i\| \\
\frac{\partial |e_{ij}|}{\partial \mathbf{x}_i} &= \frac{-\mathbf{e}}{\|\mathbf{e}\|} \\
\frac{\partial |e_{ij}|}{\partial \mathbf{x}_j} &= \frac{\mathbf{e}}{\|\mathbf{e}\|}
\end{aligned}
$$



Discrete Energy Gradients

- Gradients of triangle area

$$
\begin{aligned}
|f_{ijk}| &= \frac{1}{2}\left\|\mathbf{n}_1\right\| \\
\frac{\partial |f_{ijk}|}{\partial \mathbf{x}_i} &= \frac{\mathbf{n}_1 \times (\mathbf{x}_k - \mathbf{x}_j)}{2\left\|\mathbf{n}_1\right\|} \\
\frac{\partial |f_{ijk}|}{\partial \mathbf{x}_j} &= \frac{\mathbf{n}_1 \times (\mathbf{x}_i - \mathbf{x}_k)}{2\left\|\mathbf{n}_1\right\|} \\
\frac{\partial |f_{ijk}|}{\partial \mathbf{x}_k} &= \frac{\mathbf{n}_1 \times (\mathbf{x}_j - \mathbf{x}_i)}{2\left\|\mathbf{n}_1\right\|}
\end{aligned}
$$



Discrete Energy Gradients

- Gradients of dihedral angle

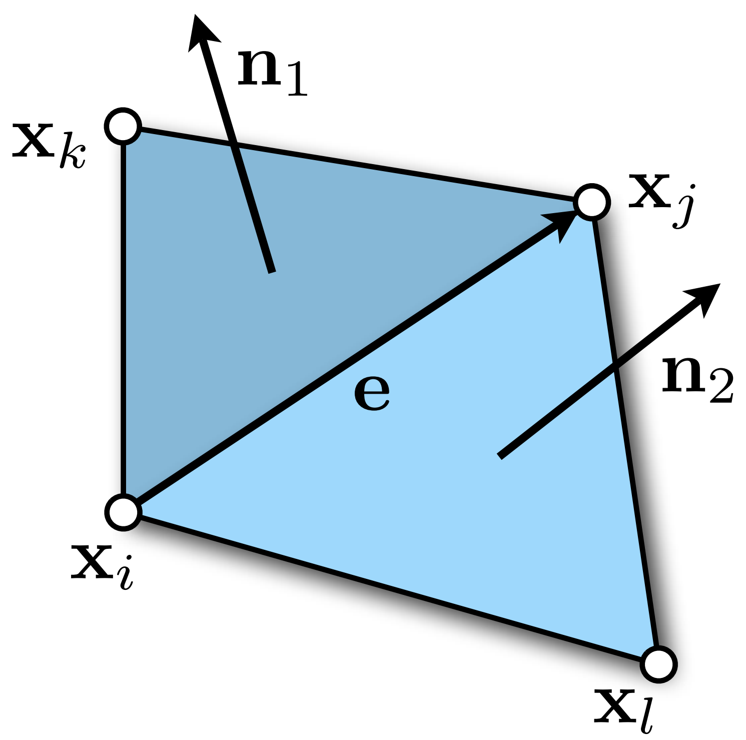
$$\theta = \operatorname{atan}\left(\frac{\sin\theta}{\cos\theta}\right) = \operatorname{atan}\left(\frac{(\mathbf{n}_1 \times \mathbf{n}_2)^T \mathbf{e}}{\mathbf{n}_1^T \mathbf{n}_2 \cdot \|\mathbf{e}\|}\right)$$

$$\frac{\partial\theta}{\partial\mathbf{x}_i} = \frac{(\mathbf{x}_k - \mathbf{x}_j)^T \mathbf{e}}{\|\mathbf{e}\|} \cdot \frac{-\mathbf{n}_1}{\|\mathbf{n}_1\|^2} + \frac{(\mathbf{x}_l - \mathbf{x}_j)^T \mathbf{e}}{\|\mathbf{e}\|} \cdot \frac{-\mathbf{n}_2}{\|\mathbf{n}_2\|^2}$$

$$\frac{\partial\theta}{\partial\mathbf{x}_j} = \frac{(\mathbf{x}_i - \mathbf{x}_k)^T \mathbf{e}}{\|\mathbf{e}\|} \cdot \frac{-\mathbf{n}_1}{\|\mathbf{n}_1\|^2} + \frac{(\mathbf{x}_i - \mathbf{x}_l)^T \mathbf{e}}{\|\mathbf{e}\|} \cdot \frac{-\mathbf{n}_2}{\|\mathbf{n}_2\|^2}$$

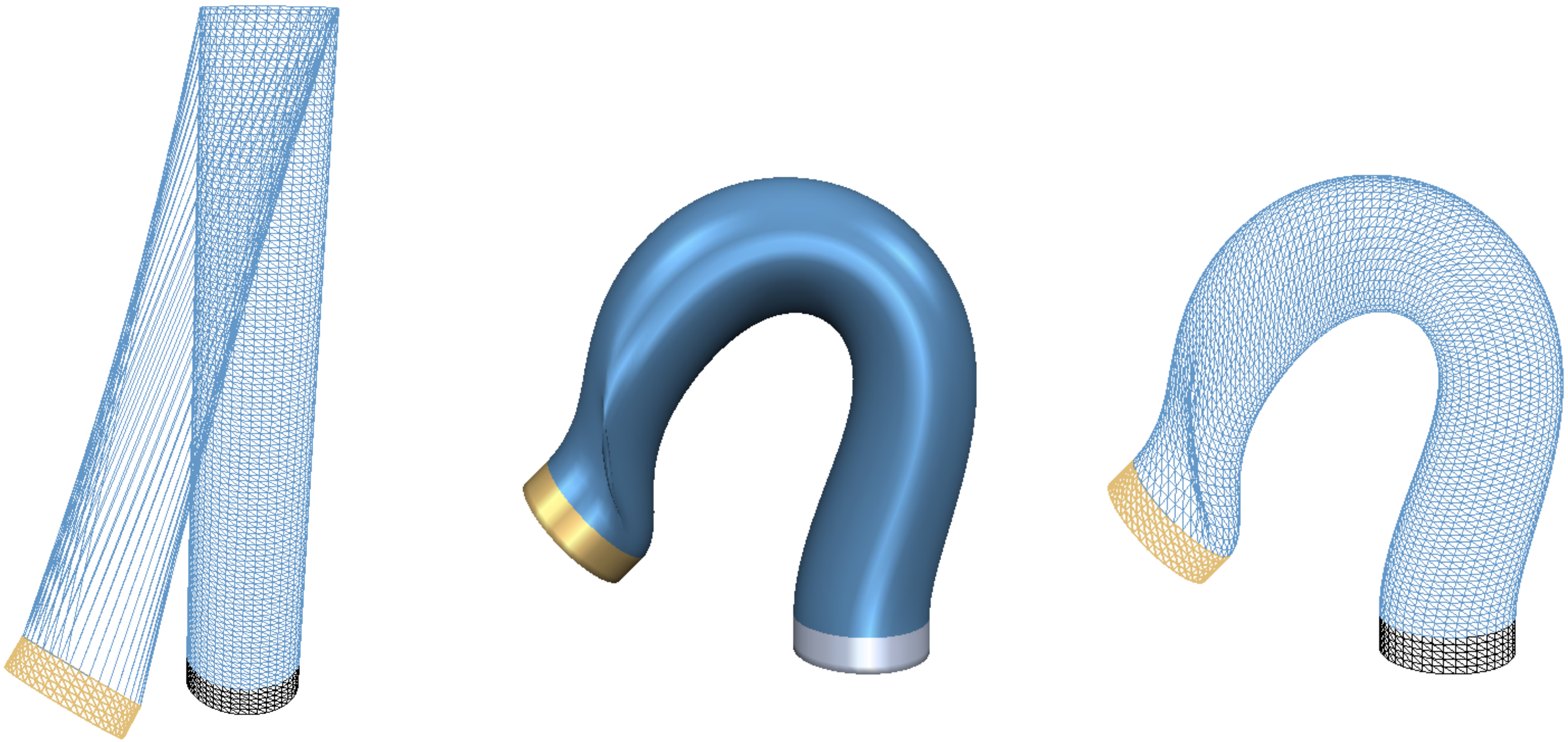
$$\frac{\partial\theta}{\partial\mathbf{x}_k} = \|\mathbf{e}\| \cdot \frac{-\mathbf{n}_1}{\|\mathbf{n}_1\|^2}$$

$$\frac{\partial\theta}{\partial\mathbf{x}_l} = \|\mathbf{e}\| \cdot \frac{-\mathbf{n}_2}{\|\mathbf{n}_2\|^2}$$



Discrete Shell Editing

- Problems with large deformation
  - Bad initial state causes numerical problems



Shell-Based Deformation

- Discrete Shells
  [Grinspun et al, SCA 2003]
- **Rigid Cells**
  [Botsch et al, SGP 2006]
- As-Rigid-As-Possible Modeling
  [Sorkine & Alexa, SGP 2007]

Nonlinear Shape Deformation

- *Nonlinear* editing too instable?
- <u>*Physically plausible*</u> vs. physically correct

➡ Trade physical correctness for

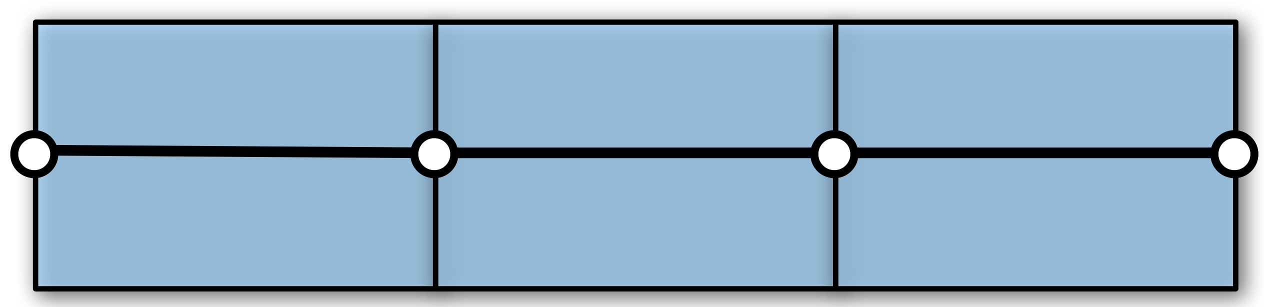
  – Computational efficiency
  – Numerical robustness

Elastically Connected Rigid Cells

- Qualitatively emulate thin-shell behavior
- Thin volumetric layer around center surface
- Extrude polygonal cell C_i per mesh face

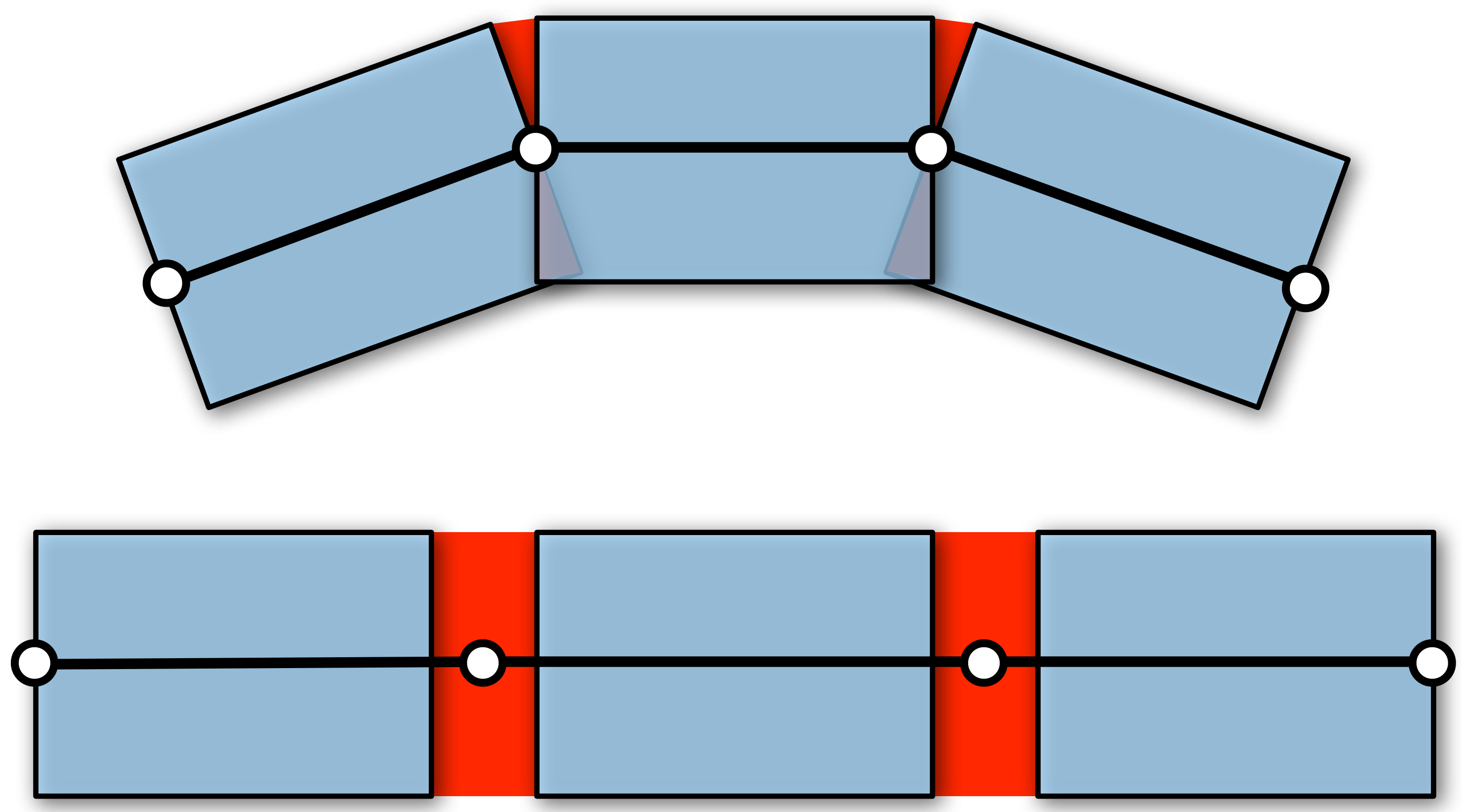
Elastically Connected Rigid Cells

- Aim for robustness
  - Prevent cells from degenerating
  - ➡ Keep cells *rigid*



Elastically Connected Rigid Cells

- Connect cells along their faces
  - Nonlinear elastic energy
  - Measures bending, stretching, twisting, ...

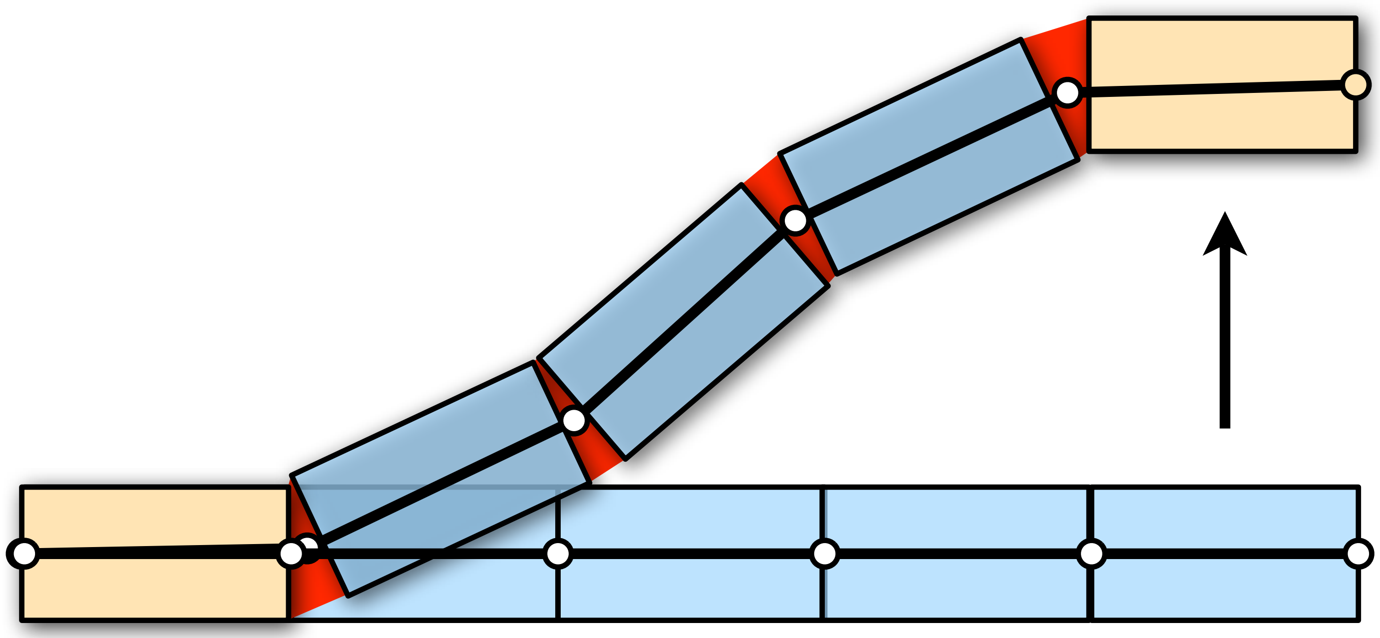


Cell-Based Surface Deformation

1. Prescribes position/orientation for cells

2. Find optimal rigid motions per cell

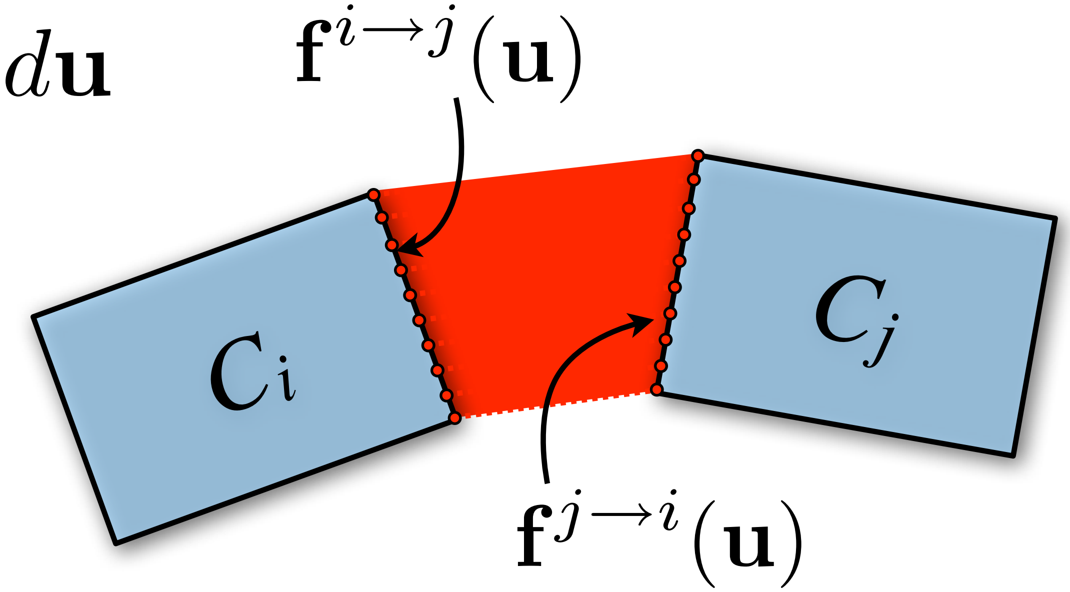
3. Update vertices by average cell transformations



Elastically Connected Rigid Cells

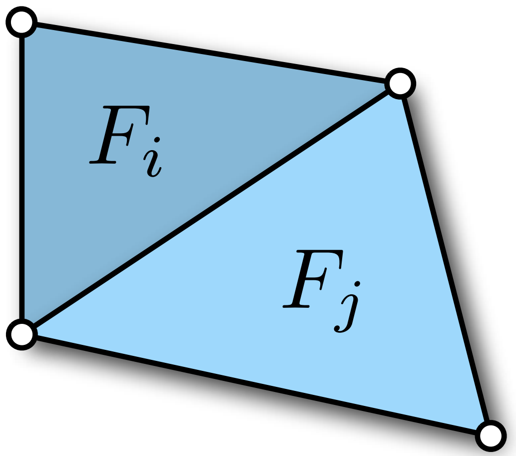
- Pairwise energy

$$E_{ij} = \int_{[0,1]^2} \left\| \mathbf{f}^{\,i\to j}(\mathbf{u}) - \mathbf{f}^{\,j\to i}(\mathbf{u}) \right\|^2 d\mathbf{u}$$



- Global energy

$$E = \sum_{\{i,j\}} w_{ij} \cdot E_{ij} \quad , \qquad w_{ij} = \frac{\|\mathbf{e}_{ij}\|^2}{|F_i| + |F_j|}$$



Nonlinear Minimization

- Find ***rigid*** motion $\mathbf{T}_i$ per cell C_i

$$\min_{\{\mathbf{T}_i\}} \sum_{\{i,j\}} w_{ij} \int_{[0,1]^2} \left\| \mathbf{T}_i\left(\mathbf{f}^{\,i\to j}(\mathbf{u})\right) - \mathbf{T}_j\left(\mathbf{f}^{\,j\to i}(\mathbf{u})\right) \right\|^2 \mathrm{d}\mathbf{u}$$

- Generalized global *shape matching* problem
  - Robust geometric optimization
  - Nonlinear Newton-type minimization
  - Hierarchical multi-grid solver

Newton-Type Iteration

1. Linearization of rigid motions

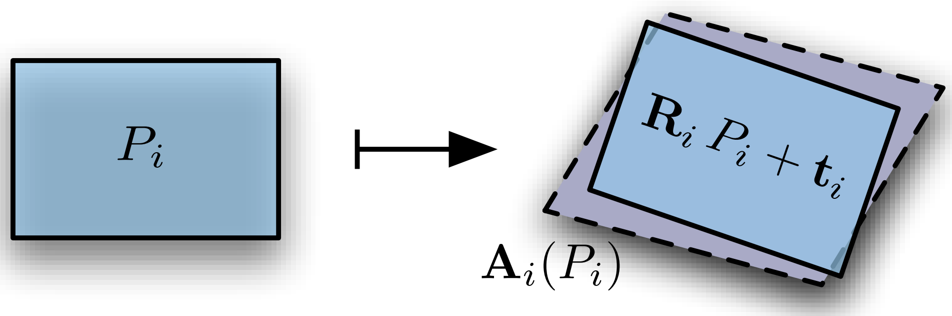
$$\mathbf{R}_i\,\mathbf{x} + \mathbf{t}_i \;\approx\; \mathbf{x} + (\omega_i \times \mathbf{x}) + \mathbf{v}_i \;=:\; \mathbf{A}_i\,\mathbf{x}$$

2. Quadratic optimization of velocities

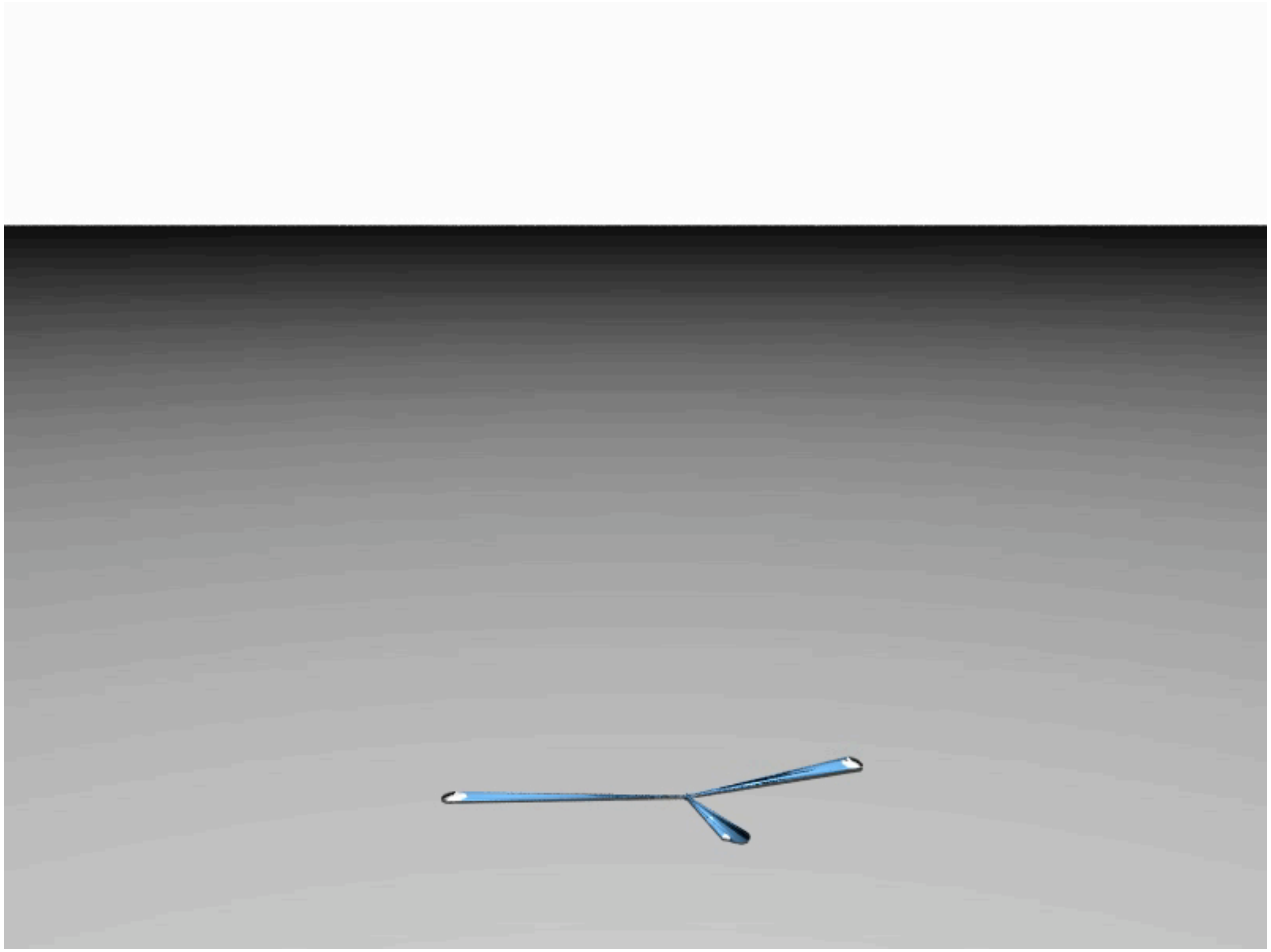
$$\min_{\{\mathbf{v}_i,\,\omega_i\}} \sum_{\{i,j\}} w_{ij} \int_{[0,1]^2} \left\| \mathbf{A}_i\left(\mathbf{f}^{\,i\to j}(\mathbf{u})\right) - \mathbf{A}_j\left(\mathbf{f}^{\,j\to i}(\mathbf{u})\right) \right\|^2 \mathrm{d}\mathbf{u}$$

3. Project $\mathbf{A}_i$ onto rigid motion manifold

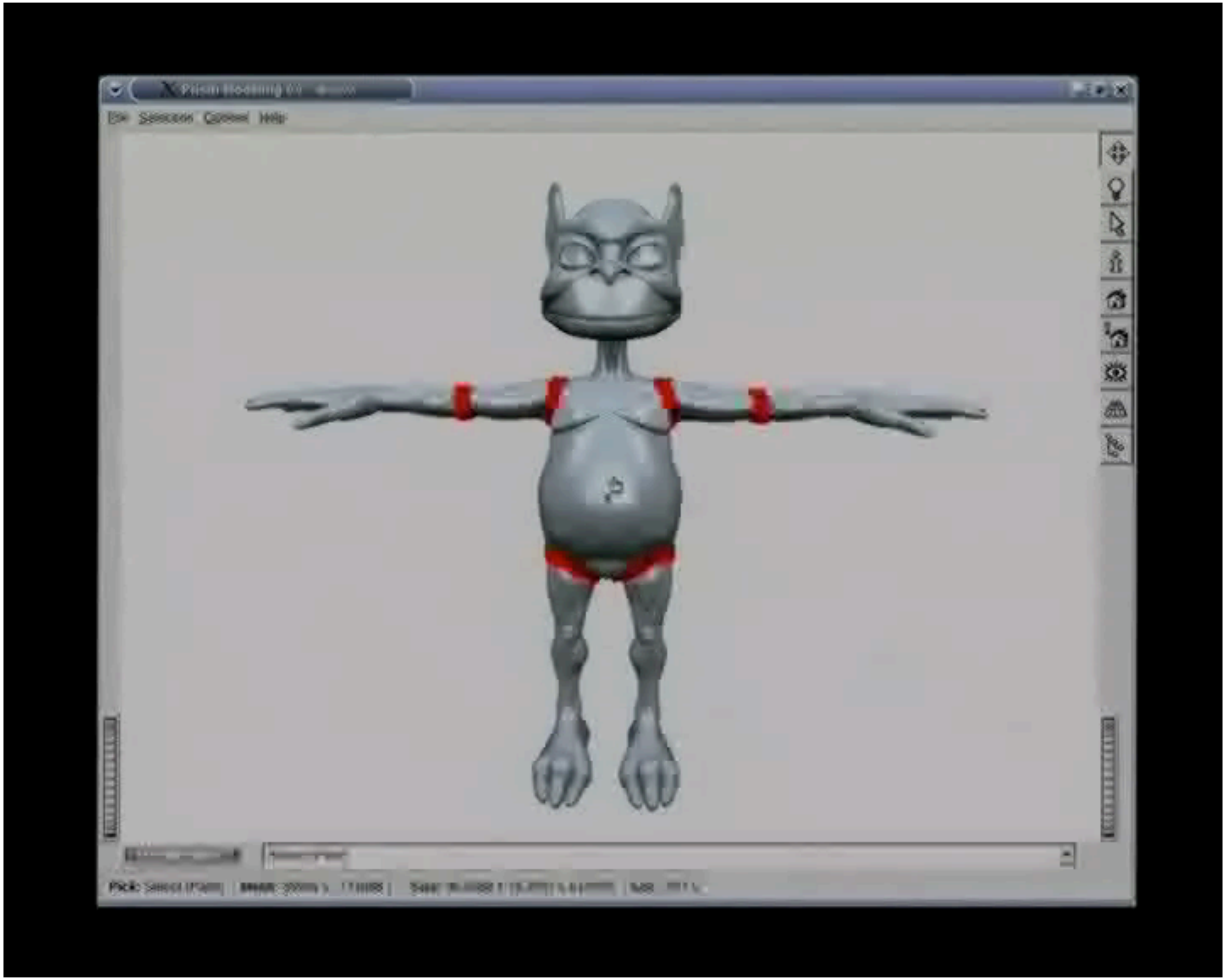
➡ Local shape matching



Robustness

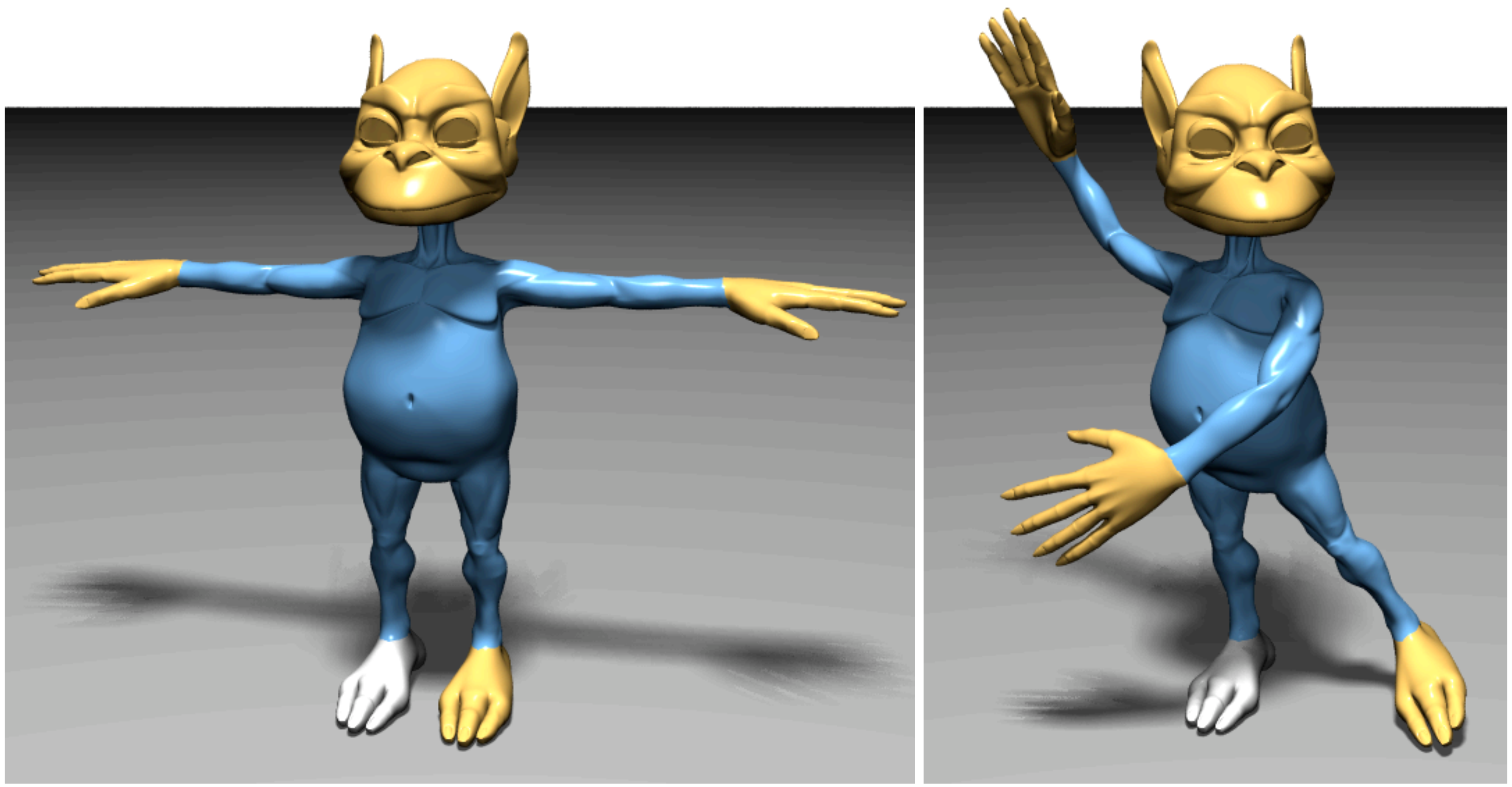


Character Posing



Goblin Posing

- Intuitive large scale deformations
- Whole session < 5 min

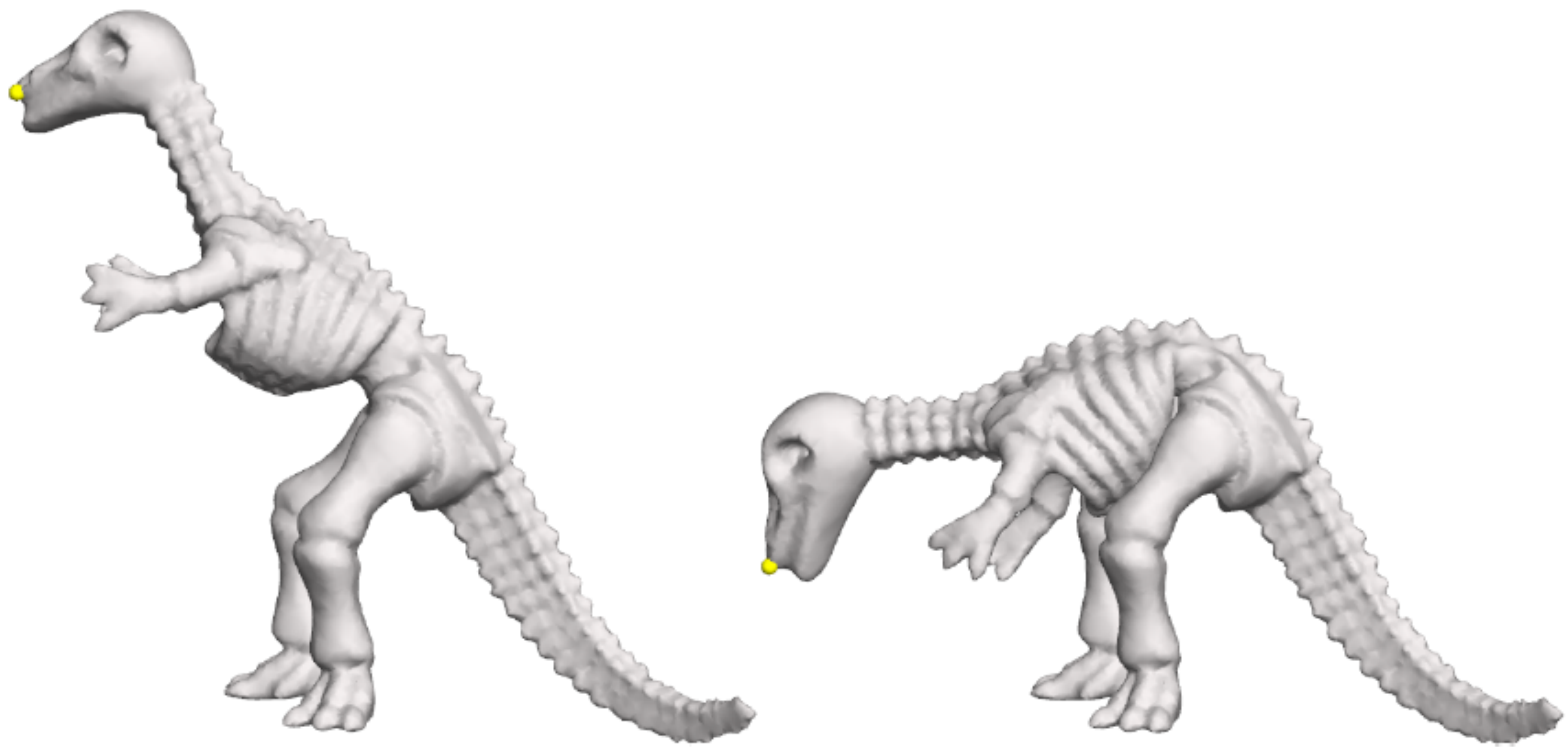


Shell-Based Deformation

- Discrete Shells
  [Grinspun et al, SCA 2003]
- Rigid Cells
  [Botsch et al, SGP 2006]
- **As-Rigid-As-Possible Modeling**
  [Sorkine & Alexa, SGP 2007]

Surface Deformation

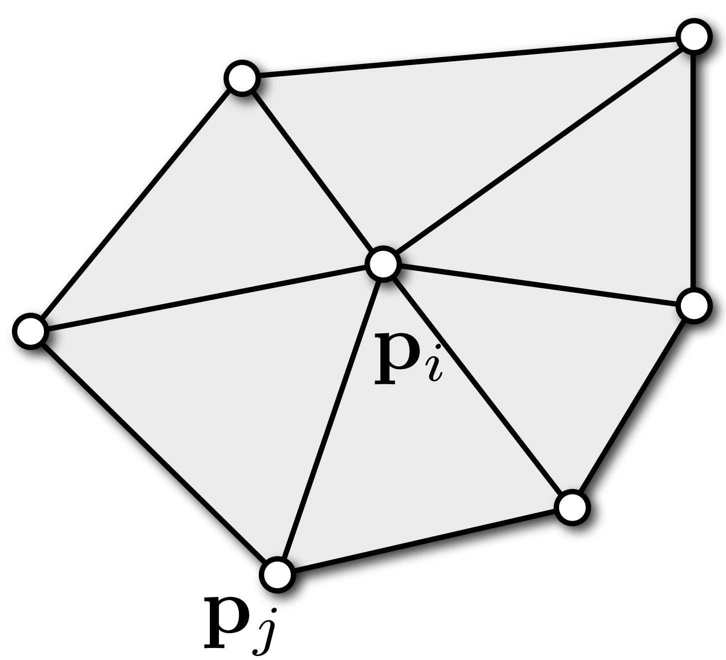
- Smooth large scale deformation
- Local as-rigid-as-possible behavior
  - Preserves small-scale details



Cell Deformation Energy

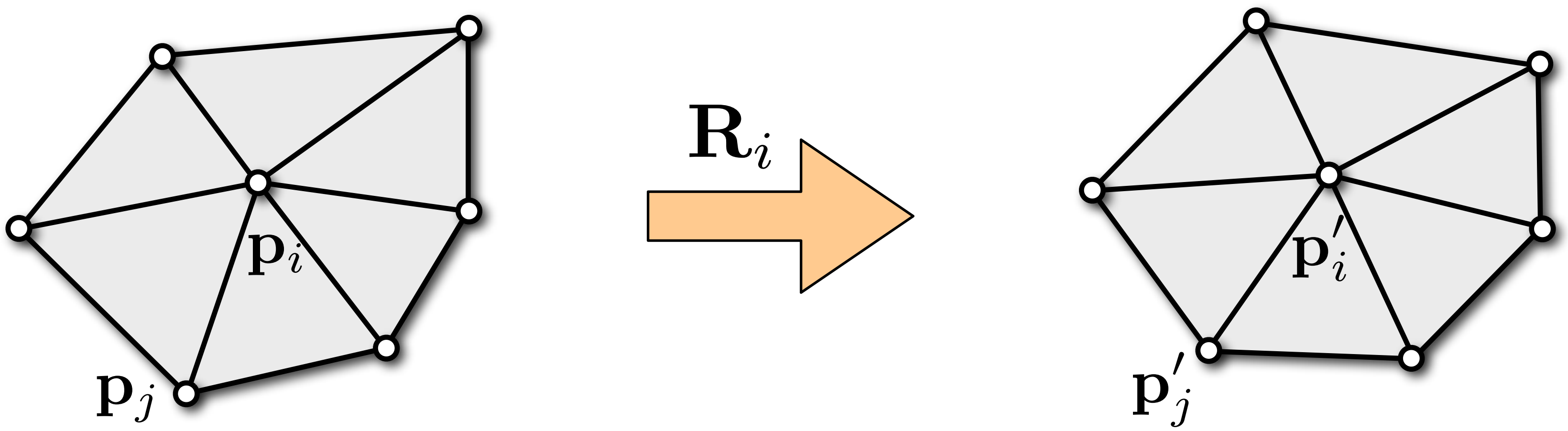
- Vertex neighborhoods should deform rigidly

$$\sum_{j \in N(i)} \left\| \left(\mathbf{p}'_j - \mathbf{p}'_i \right) - \mathbf{R}_i \left(\mathbf{p}_j - \mathbf{p}_i \right) \right\|^2 \rightarrow \min$$



Cell Deformation Energy

- If $\mathbf{p}, \mathbf{p}'$ are known then $\mathbf{R}_i$ is uniquely defined



- *Shape matching* problem
  - Build covariance matrix $\mathbf{S} = \mathbf{P}\mathbf{P}'^{\mathrm{T}}$
  - SVD: $\mathbf{S} = \mathbf{U}\mathbf{\Sigma}\mathbf{W}^{\mathrm{T}}$
  - Extract rotation $\mathbf{R}_i = \mathbf{U}\mathbf{W}^{\mathrm{T}}$

Total Deformation Energy

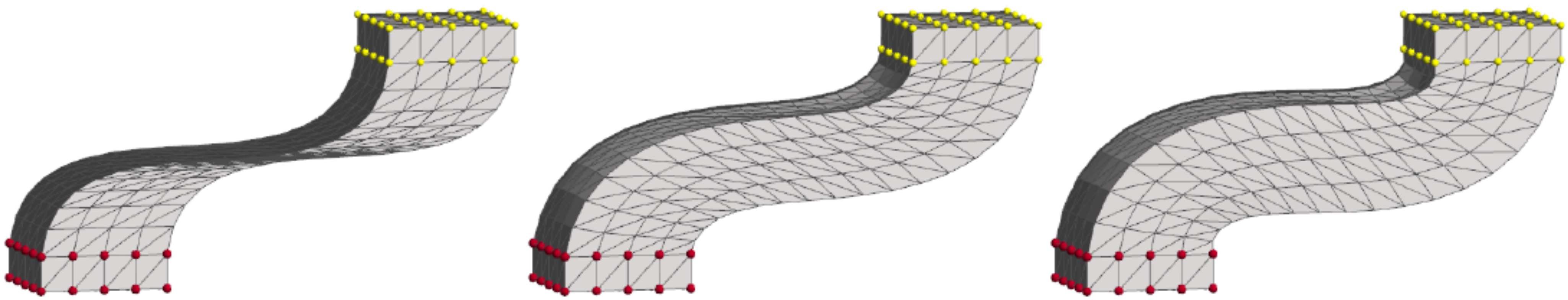
- Sum over all vertex

$$\min_{\mathbf{p}'} \sum_{i=1}^{n} \sum_{j \in N(i)} \left\| \left(\mathbf{p}'_j - \mathbf{p}'_i\right) - \mathbf{R}_i \left(\mathbf{p}_j - \mathbf{p}_i\right) \right\|^2$$

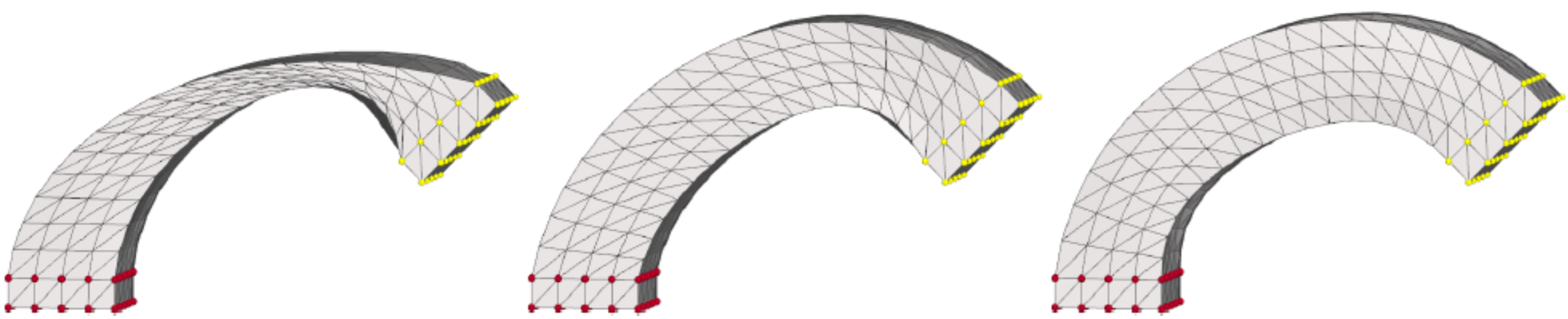
- Treat $\mathbf{p}'$ and $\mathbf{R}_i$ as separate variables
- Allows for alternating optimization
  - Fix $\mathbf{p}'$, find $\mathbf{R}_i$: Local shape matching per cell
  - Fix $\mathbf{R}_i$, find $\mathbf{p}'$: Solve Laplacian system

As-Rigid-As-Possible Modeling

- Start from naïve Laplacian editing as initial guess

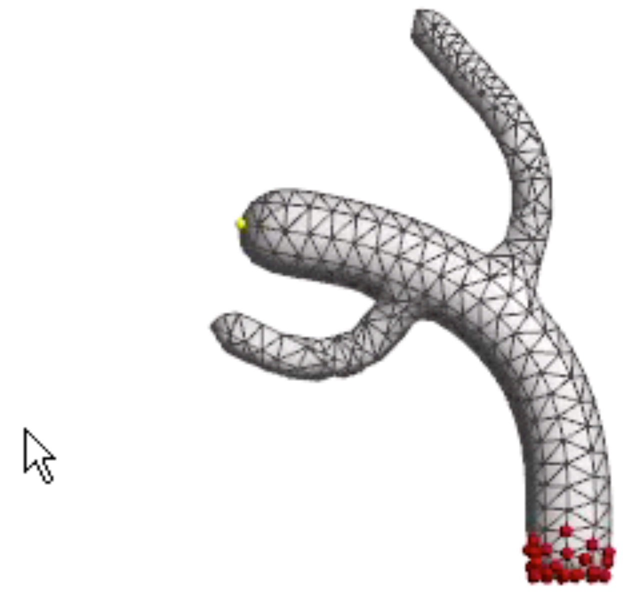


initial guess | 1 iteration | 2 iterations



initial guess | 1 iterations | 4 iterations

As-Rigid-As-Possible Modeling



Shell-Based Deformation

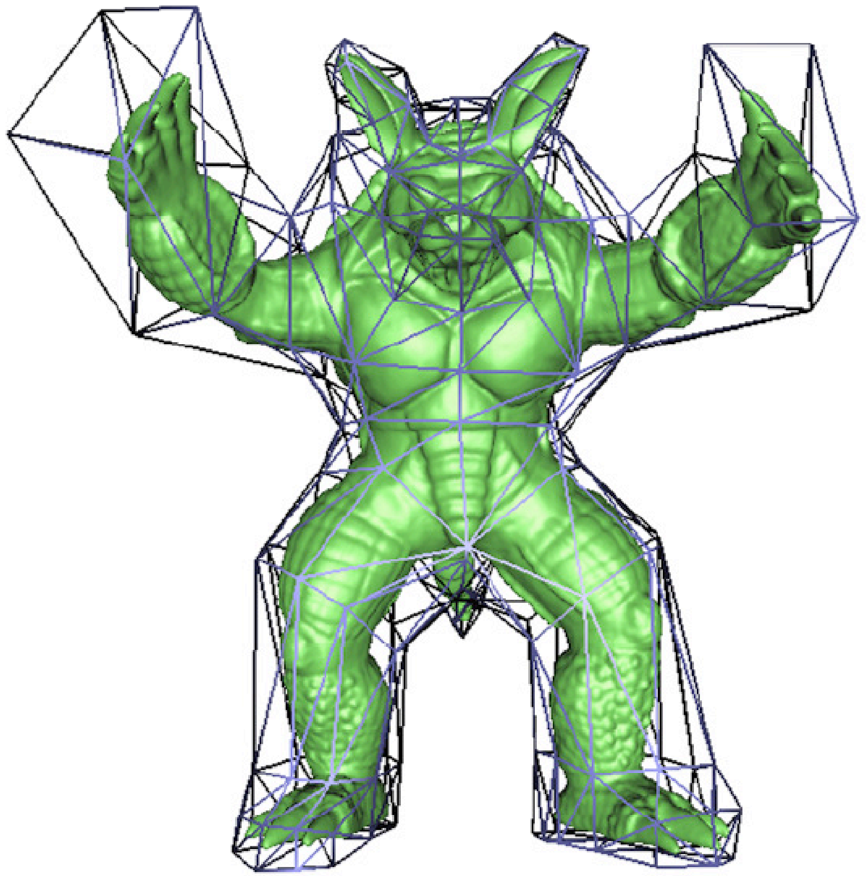
- Discrete Shells
  [Grinspun et al, SCA 2003]
- Rigid Cells
  [Botsch et al, SGP 2006]
- As-Rigid-As-Possible Modeling
  [Sorkine & Alexa, SGP 2007]

Nonlinear Surface Deformation

- Limitations of Linear Methods
- Shell-Based Deformation
- **(Differential Coordinates)**

Subspace Gradient Deformation

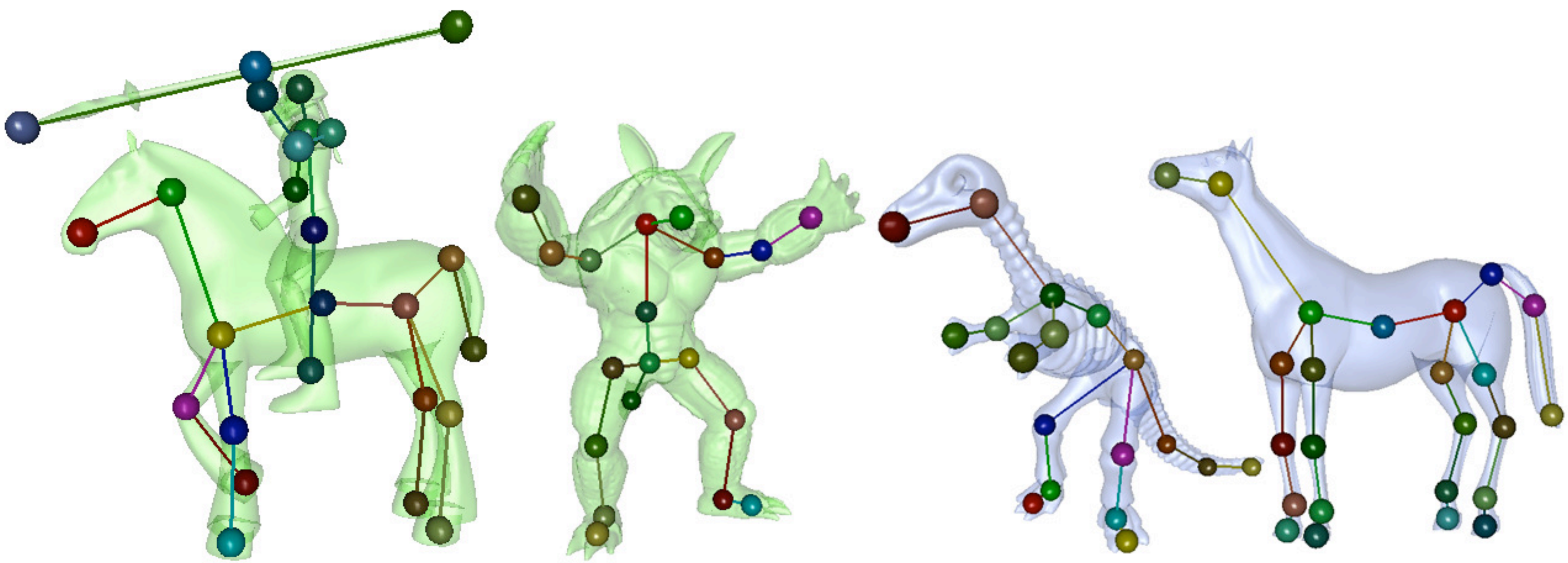
- Nonlinear Laplacian coordinates
- Least squares solution on coarse cage subspace



[Huang et al, SIGGRAPH 06]

Mesh Puppetry

- Skeletons and Laplacian coordinates
- Cascading optimization



[Shi et al, SIGGRAPH 07]

Nonlinear Surface Deformation

- Limitations of Linear Methods
- Shell-Based Deformation
- (Differential Coordinates)

Eurographics 2009

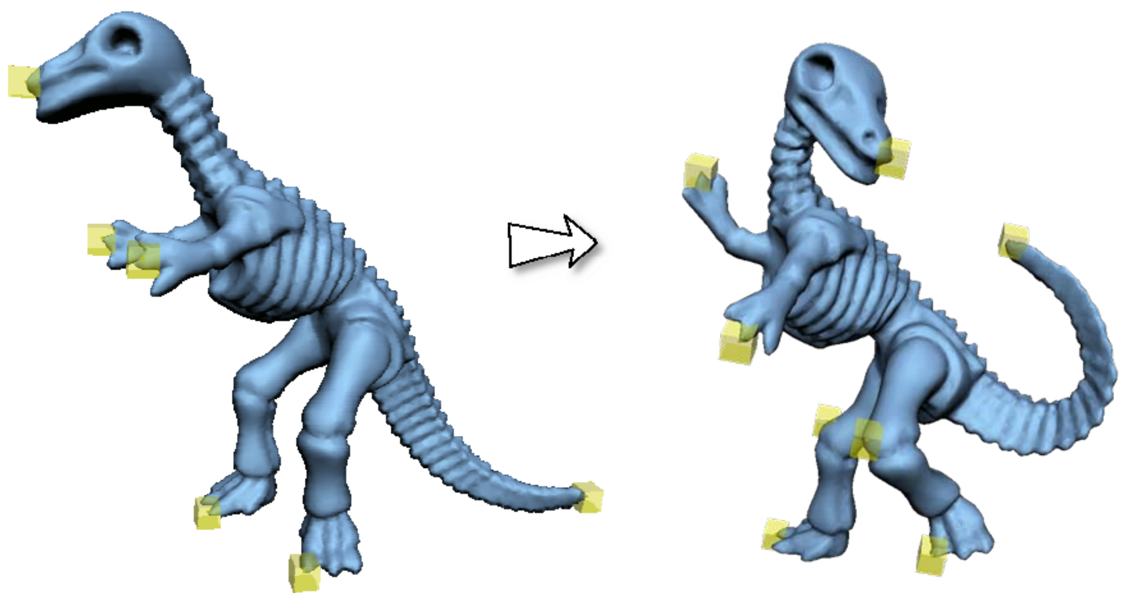
Interactive Shape Modeling and Deformation

T3: Half-Day Tutorial

Nonlinear Space Deformations

Nonlinear Space Deformations

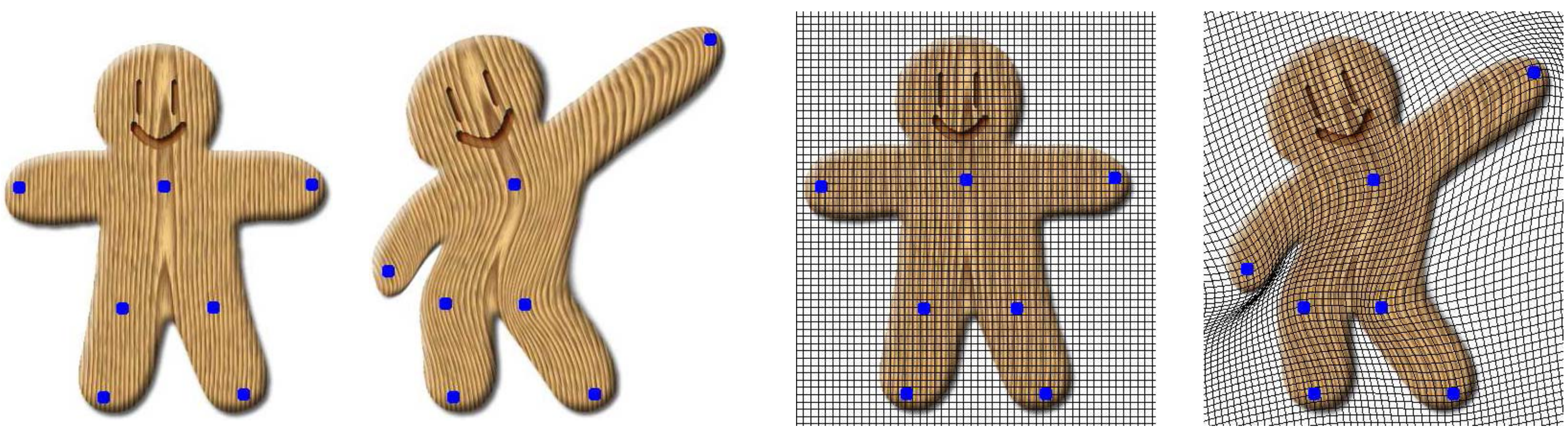
- Involve nonlinear optimization
- Enjoy the advantages of space warps
- Additionally, have shape-preserving properties



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

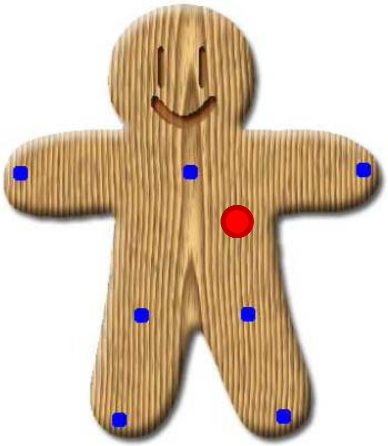
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

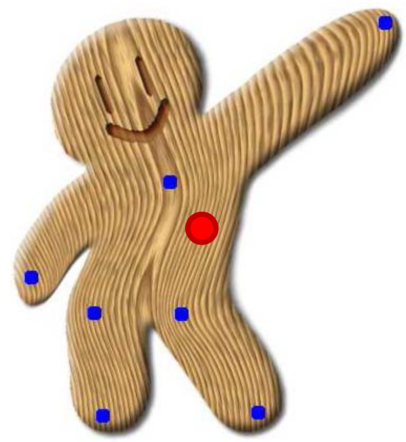
- Attach an affine transformation to each point $\mathbf{x} \in \mathrm{R}^3$:

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

- 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]

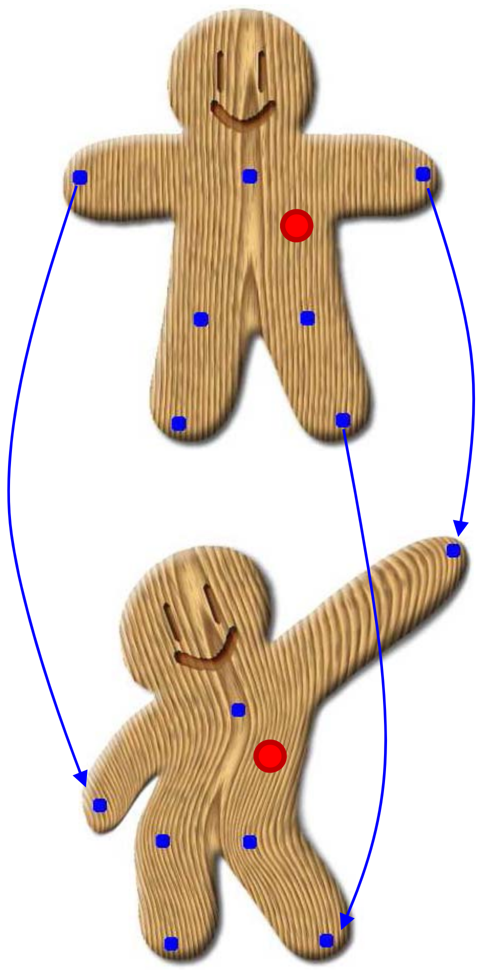
- Handles $\mathbf{p}_i$ are displaced to $\mathbf{q}_i$
- The local transformation at $\mathbf{x}$:

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

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

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

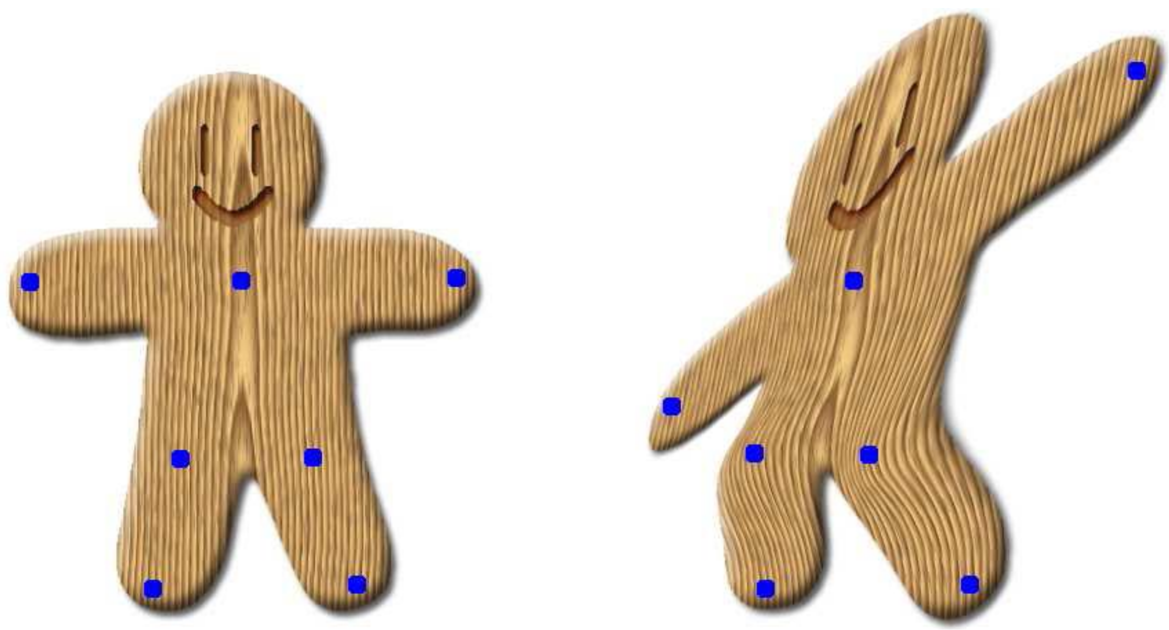
$$w_i(\mathbf{x}) = \left\| \mathbf{p}_i - \mathbf{x} \right\|^{-2\alpha}$$



As-Rigid-As-Possible Deformation

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

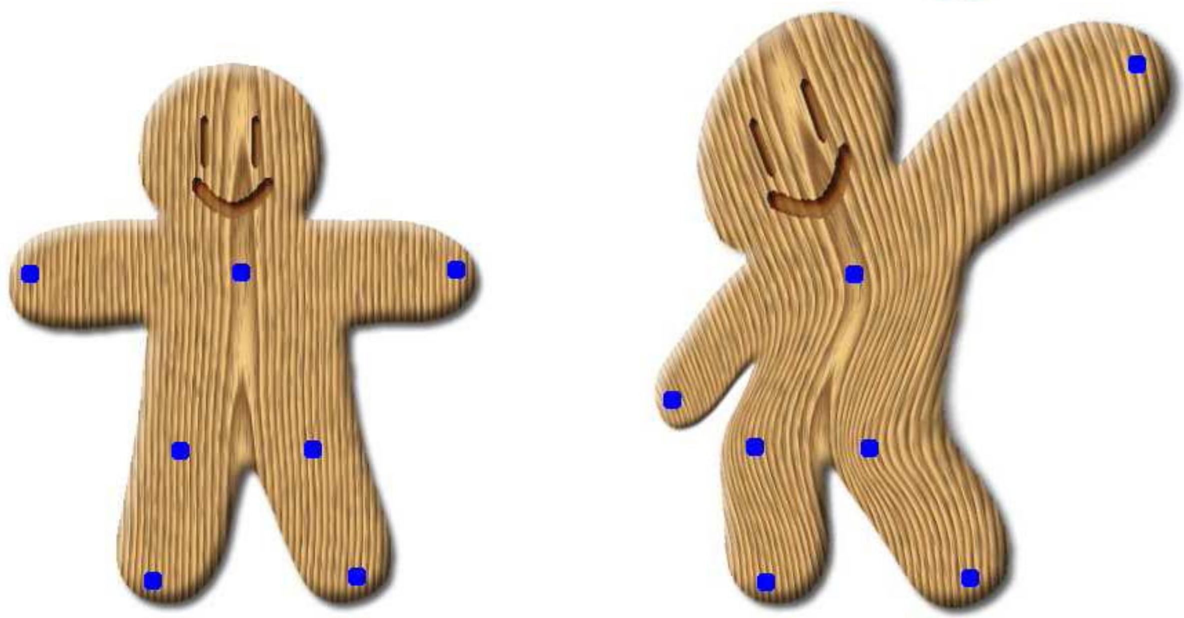
- No additional restriction on $\mathrm{A}_{\mathbf{x}}(\cdot)$ – affine local transformations



As-Rigid-As-Possible Deformation

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

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

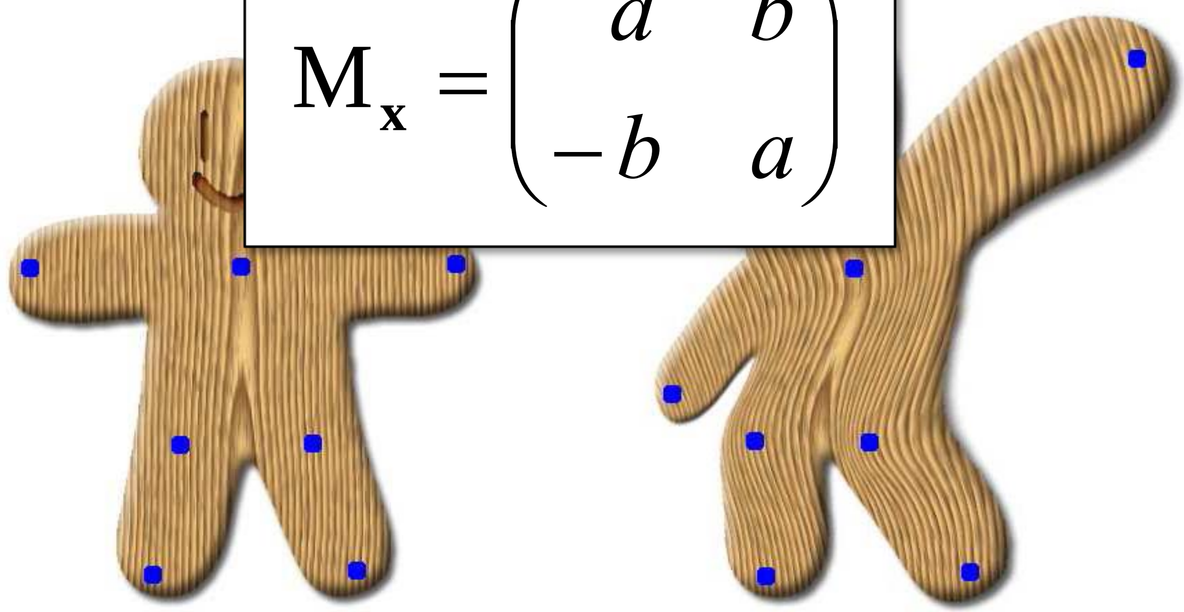


As-Rigid-As-Possible Deformation

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

- Restrict $A_{\mathbf{x}}(\cdot)$ to similarity

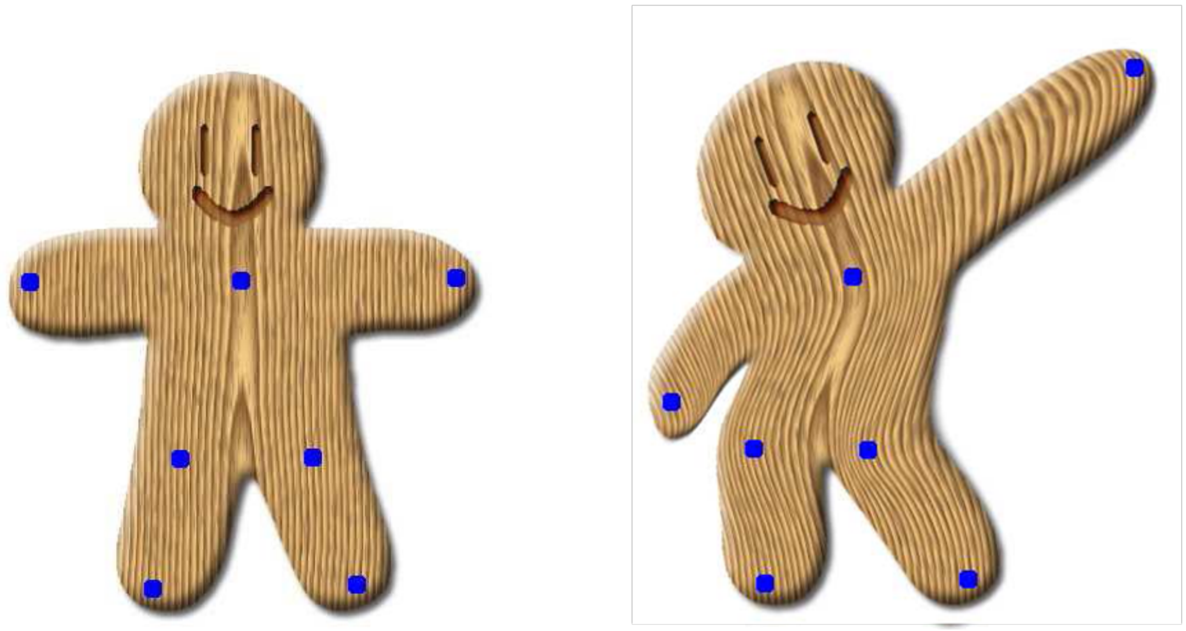
$$M_{\mathbf{x}} = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$$



As-Rigid-As-Possible Deformation

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

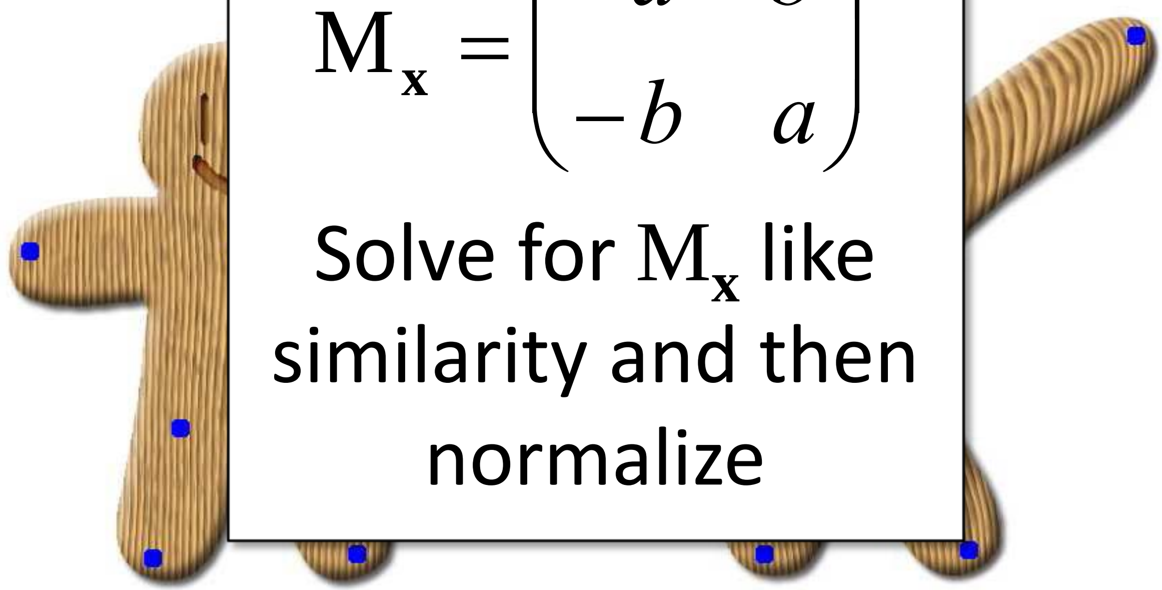
- Restrict $\mathrm{A}_{\mathbf{x}}(\cdot)$ to rigid



As-Rigid-As-Possible Deformation

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

- Restrict $A_{\mathbf{x}}(\cdot)$ to rigid



$$M_{\mathbf{x}} = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$$

Solve for $M_{\mathbf{x}}$ 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

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

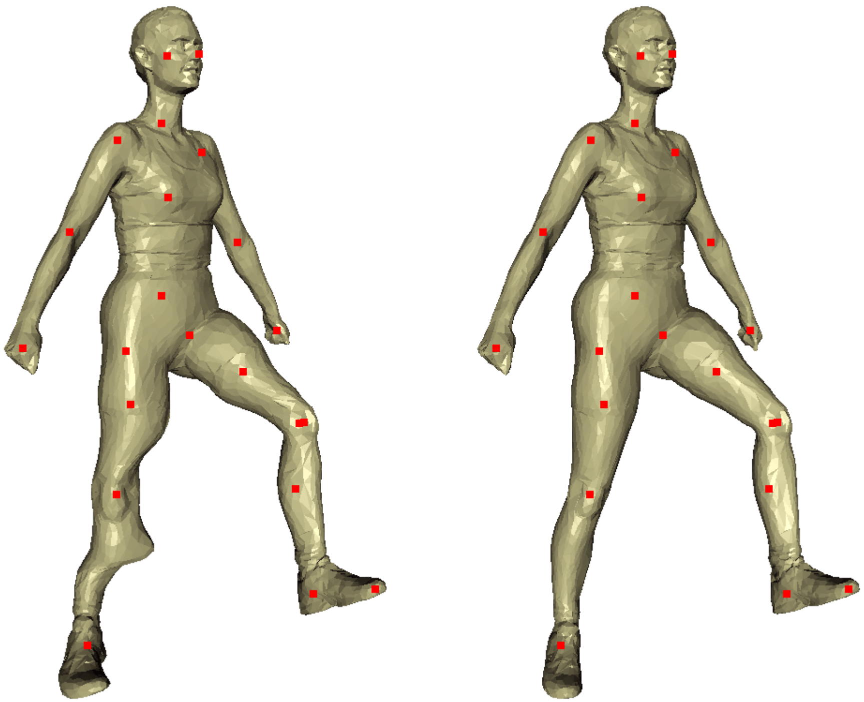
by polar decomposition of the 3×3 covariance matrix

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"

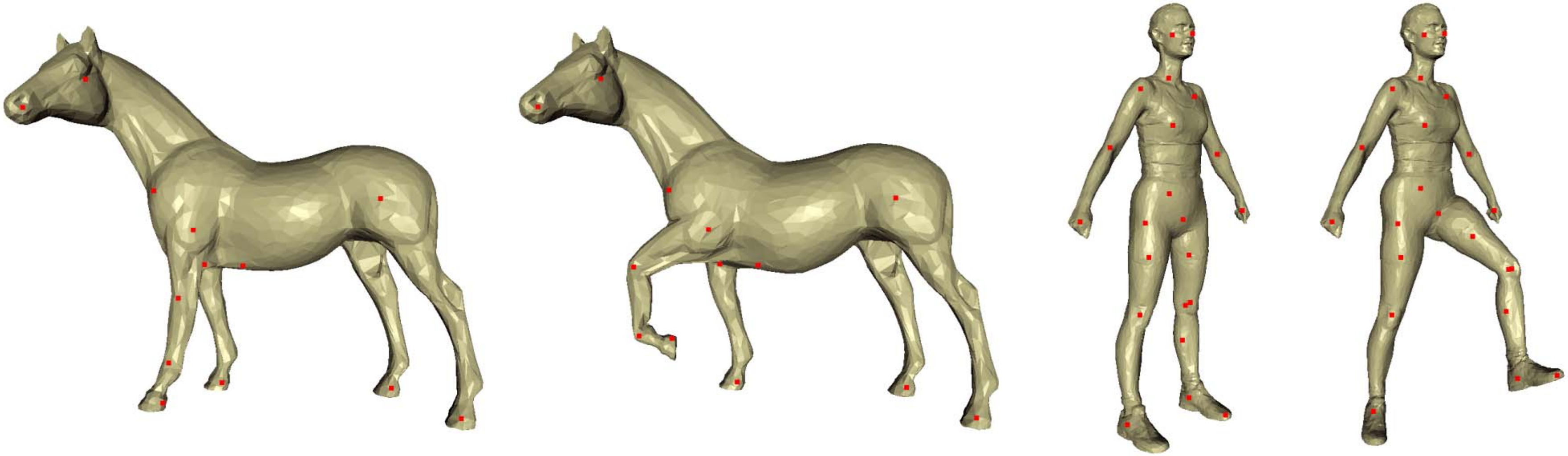
$$w_i(\mathbf{x}) = d(\mathbf{p}_i\,,\,\mathbf{x})^{-2\alpha}$$

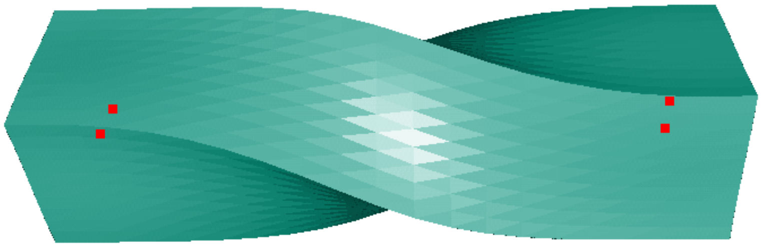


As-Rigid-As-Possible Deformation

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

- More results

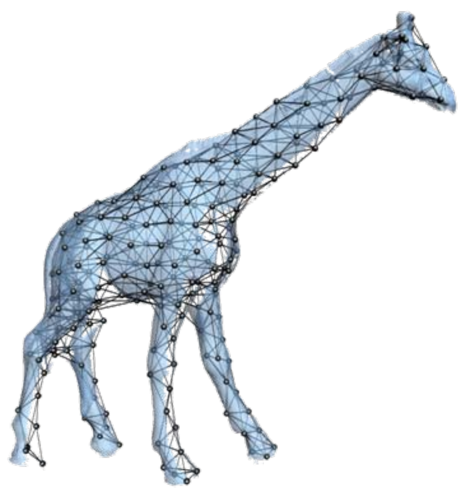




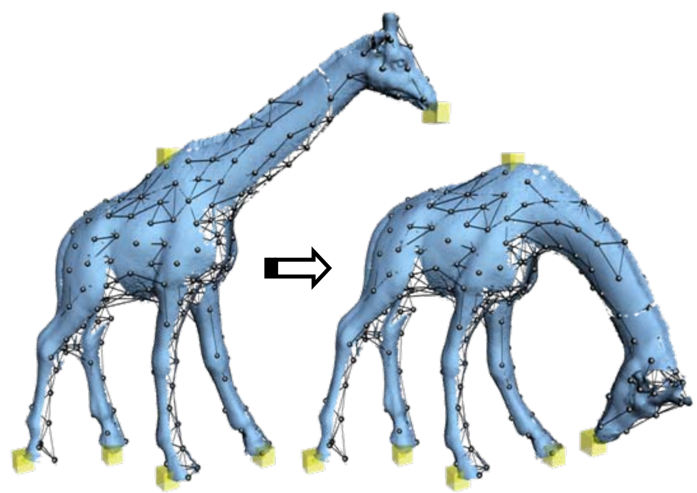
As-Rigid-As-Possible Deformation

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



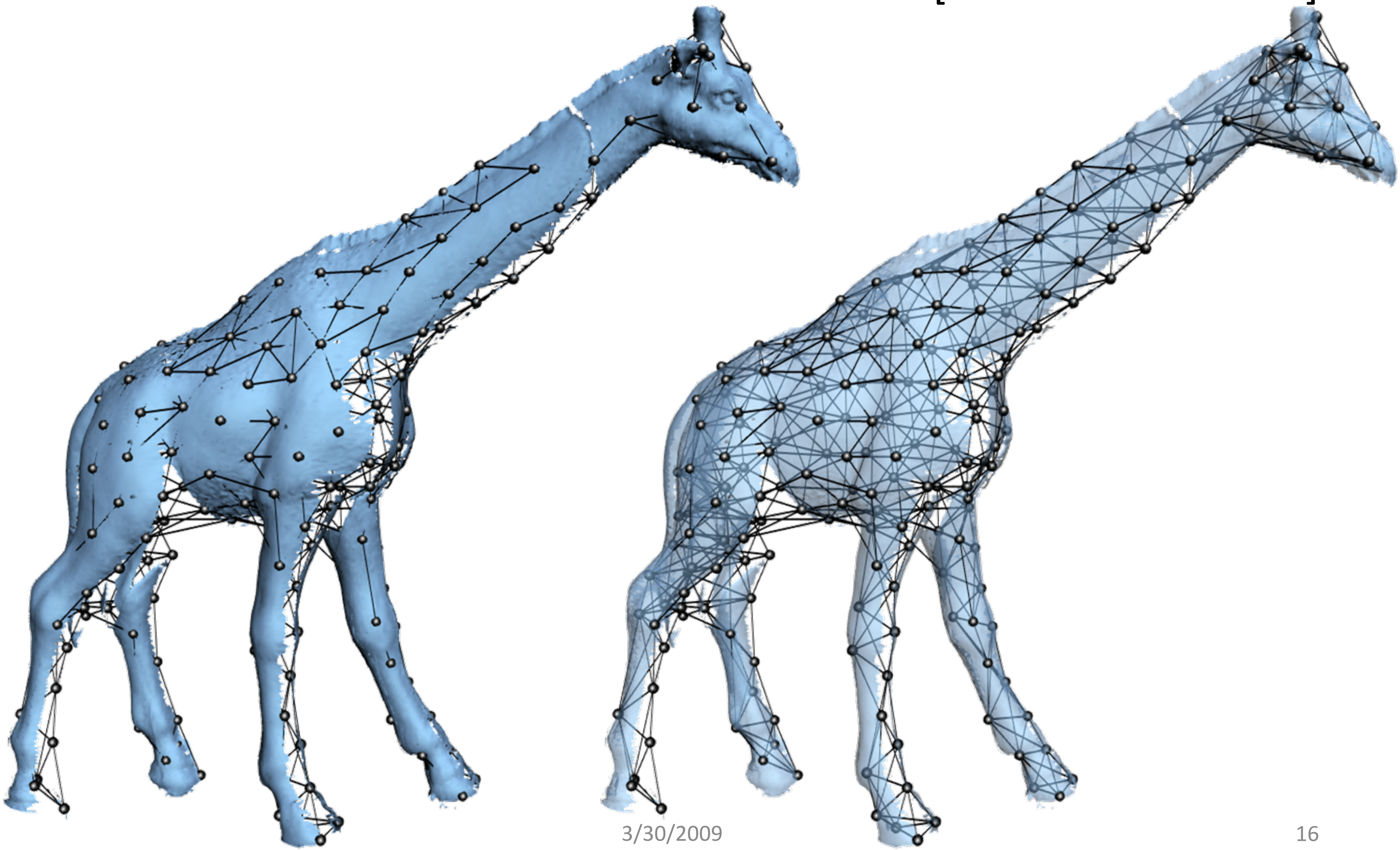
Deformation Graph



Optimization Procedure

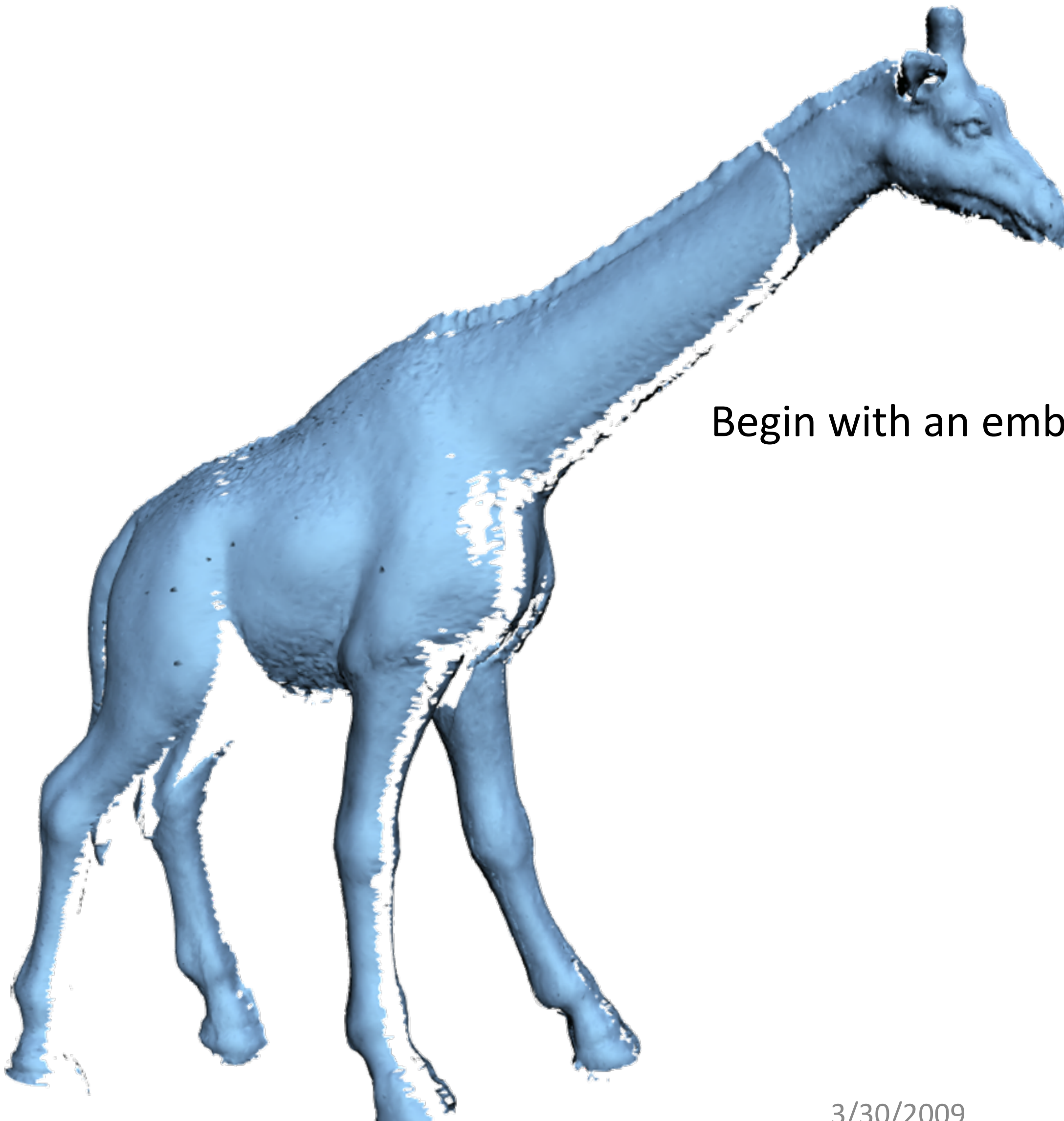
Deformation Graph

[Sumner et al. 2007]



Deformation Graph

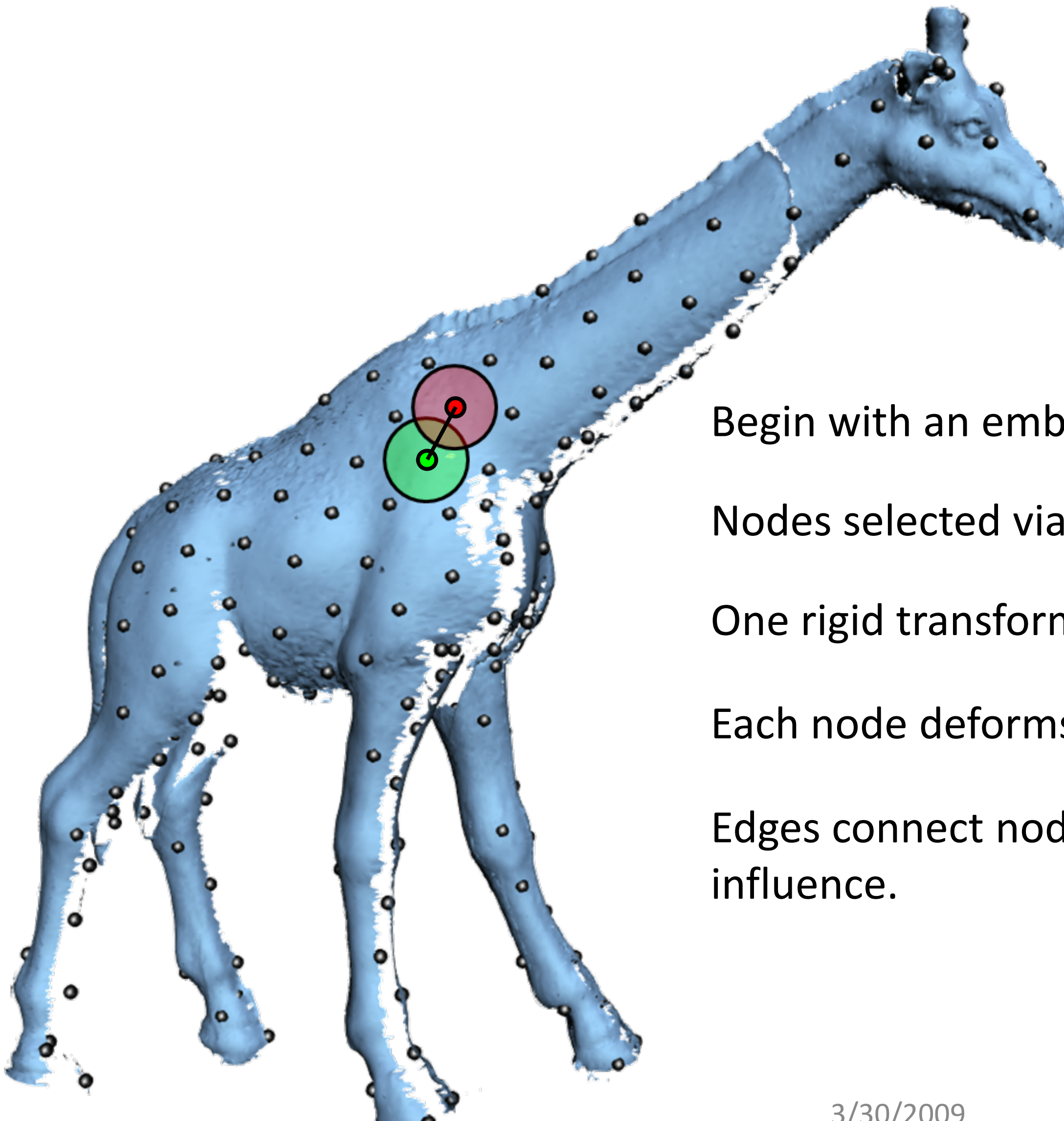
[Sumner et al. 2007]



Begin with an embedded object.

Deformation Graph

[Sumner et al. 2007]



Begin with an embedded object.

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

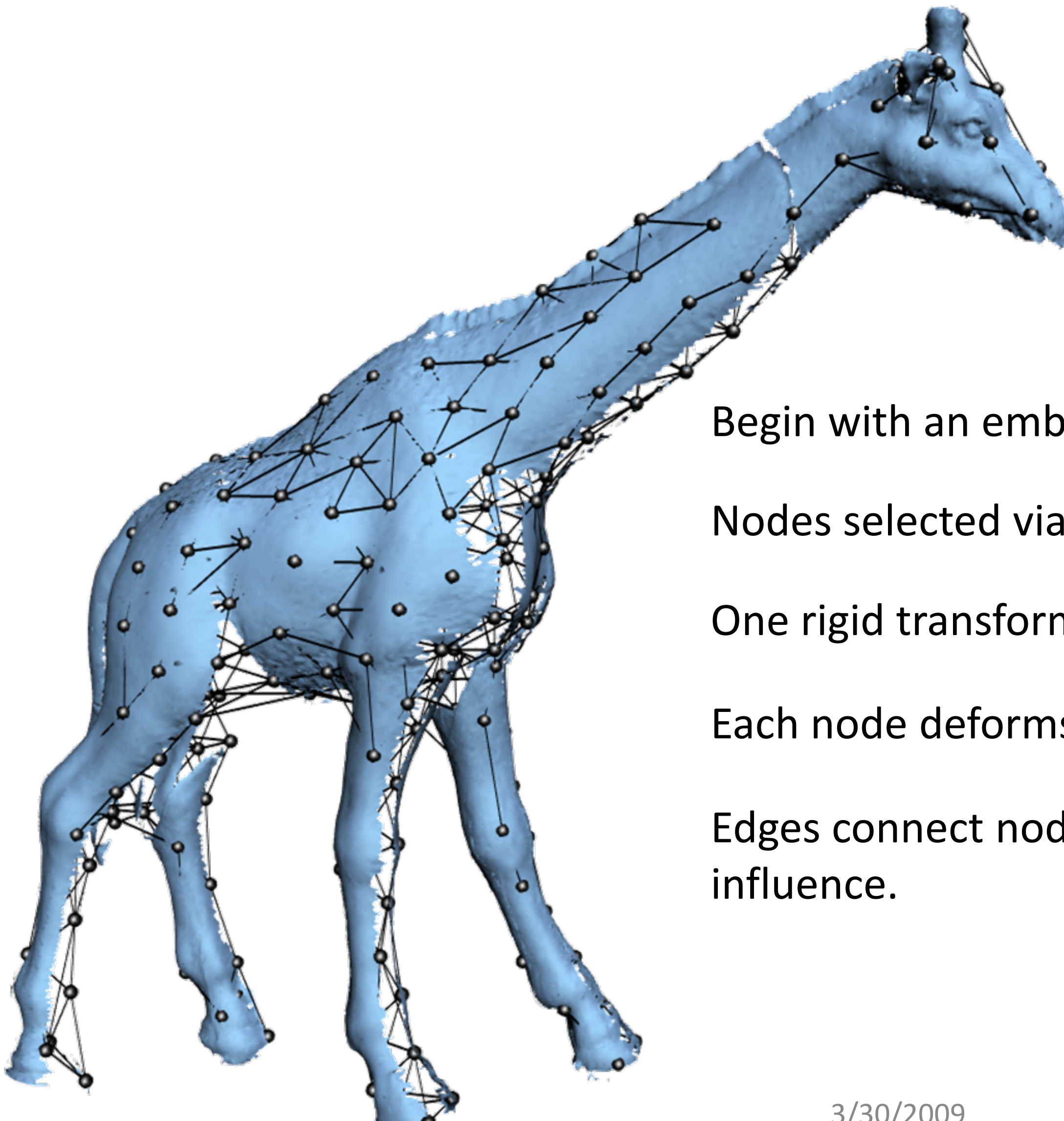
One rigid transformation for each node: R_j , $\mathbf{t}_j$

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 $\mathbf{g}_j$

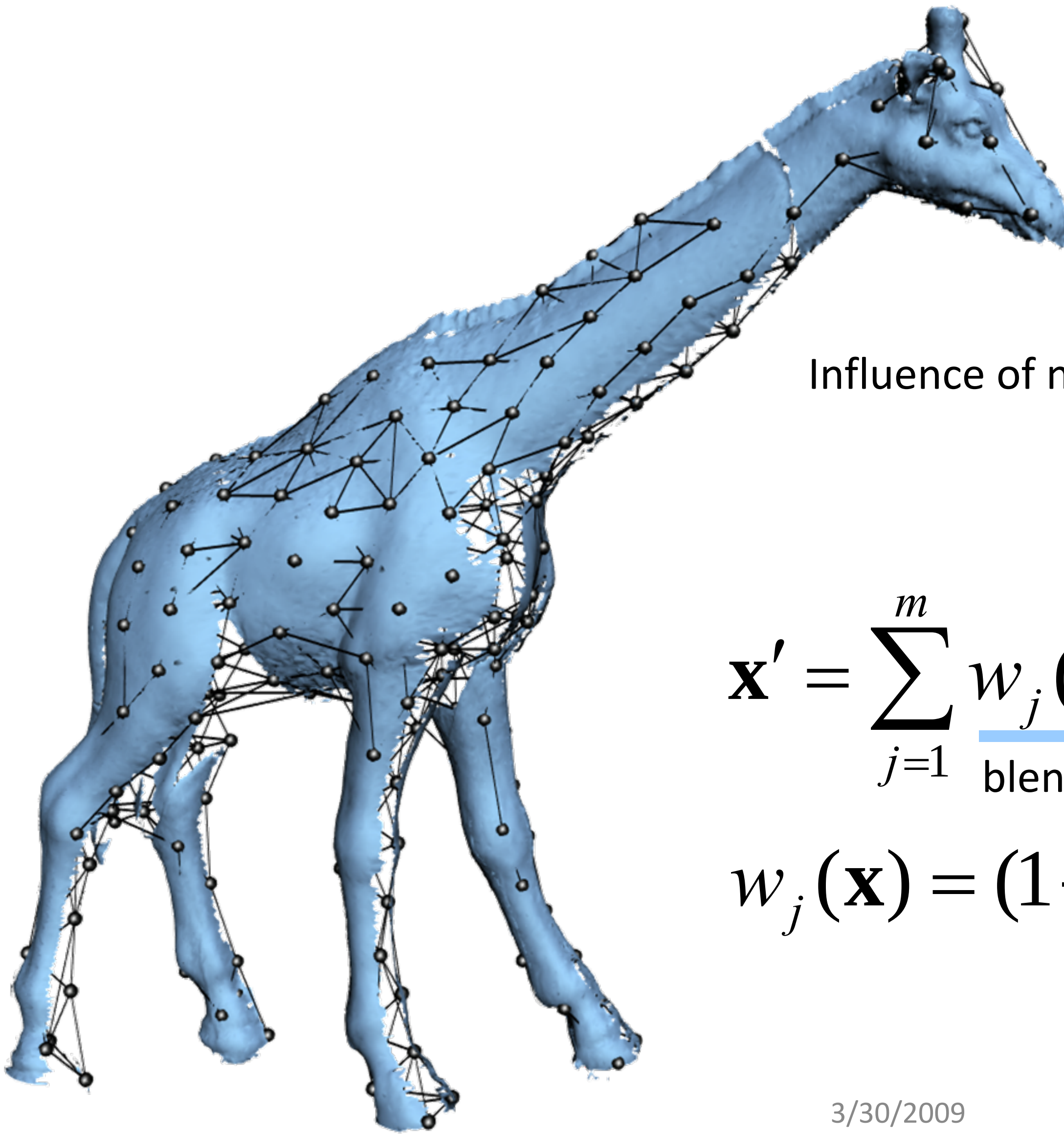
One rigid transformation for each node: R_j , $\mathbf{t}_j$

Each node deforms nearby space.

Edges connect nodes of overlapping influence.

Deformation Graph

[Sumner et al. 2007]



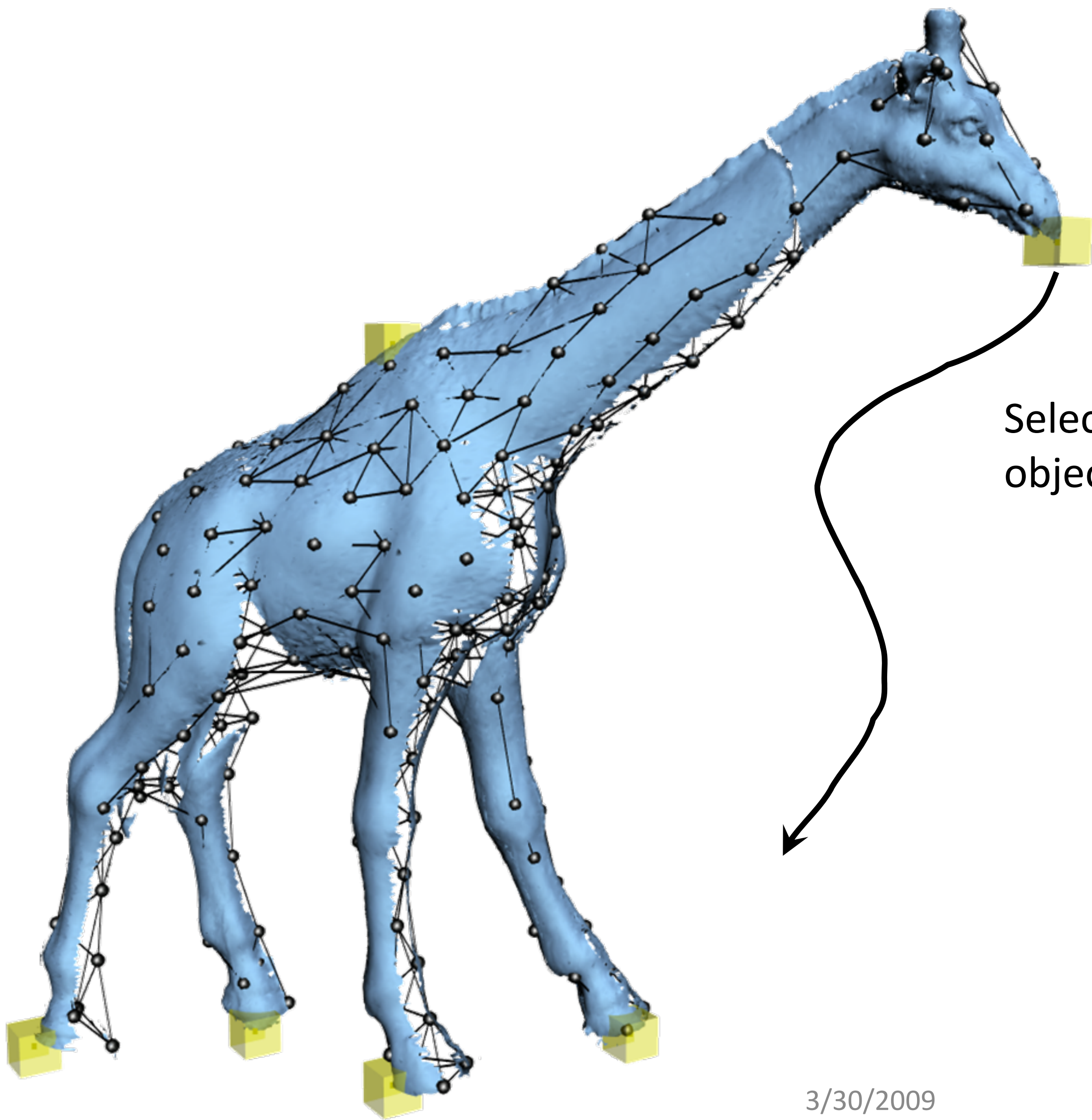
Influence of nearby transformations is blended.

$$\mathbf{x}' = \sum_{j=1}^{m} \underbrace{w_j(\mathbf{x})}_{\text{blending weights}} \Big[\overbrace{\mathrm{R}_j(\mathbf{x} - \mathbf{g}_j) + \mathbf{g}_j + \mathbf{t}_j}^{\text{point } \mathbf{x} \text{ transformed by node } j} \Big]$$

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

Optimization

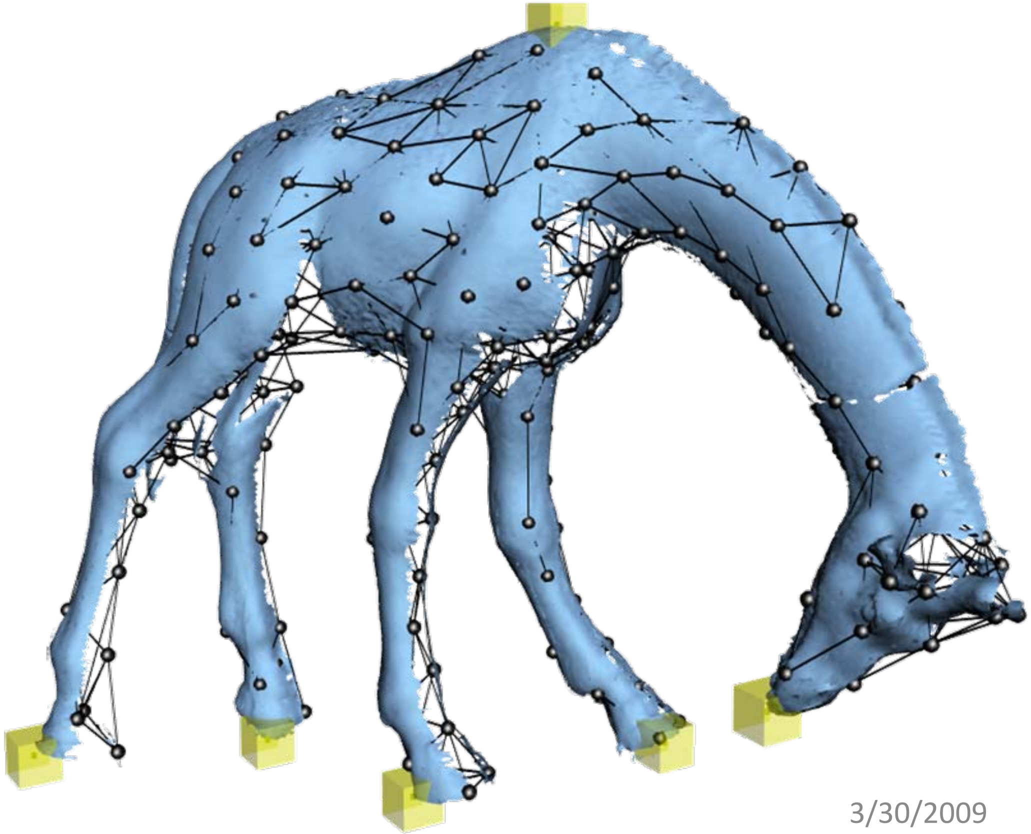
[Sumner et al. 2007]



Select & drag vertices of embedded object.

Optimization

[Sumner et al. 2007]



Select & drag vertices of embedded object.

Optimization finds

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

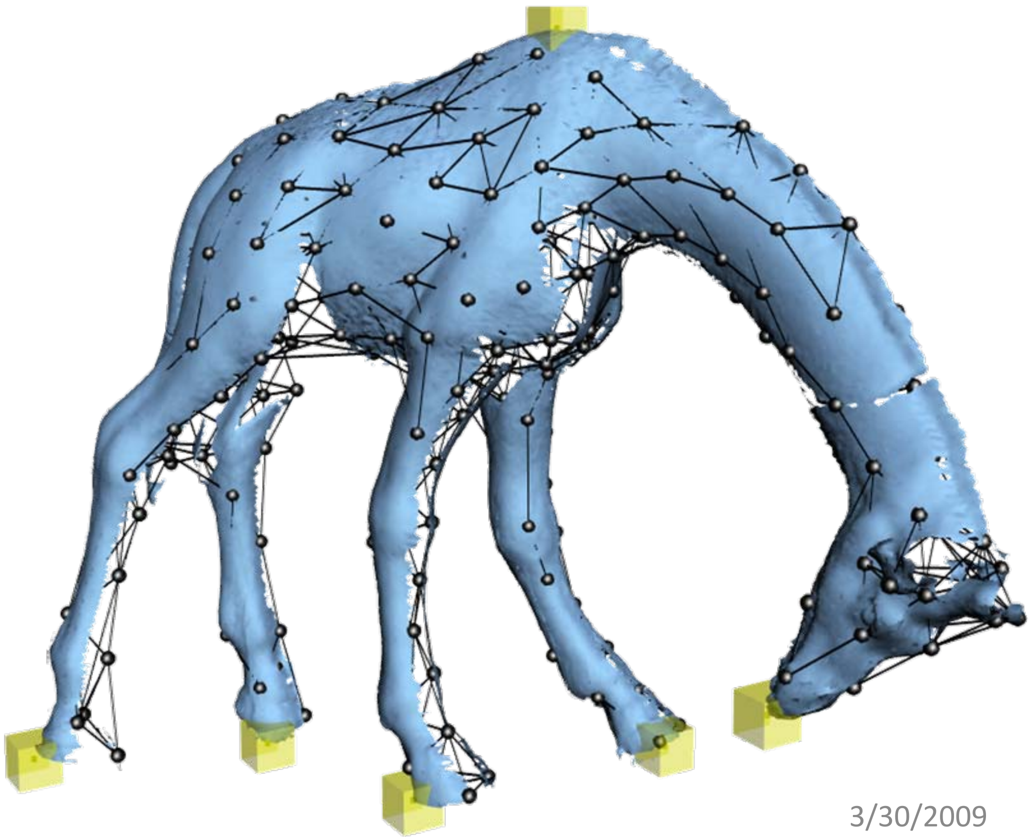
$$\min_{\mathbf{R}_1, \mathbf{t}_1, \ldots, \mathbf{R}_m, \mathbf{t}_m} \quad w_{\mathrm{rot}} \mathrm{E}_{\mathrm{rot}} + w_{\mathrm{reg}} \mathrm{E}_{\mathrm{reg}} + w_{\mathrm{con}} \mathrm{E}_{\mathrm{con}}$$

Graph parameters

Rotation term

Regularization term

Constraint term



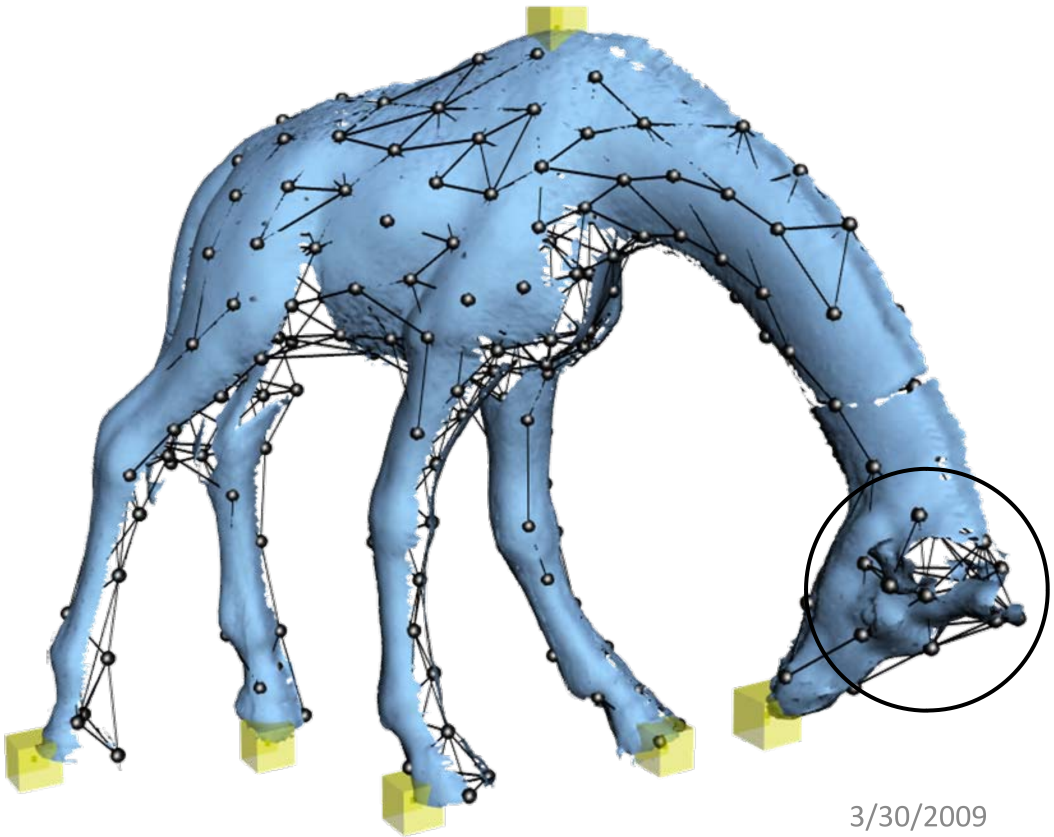
Select & drag vertices of embedded object.

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

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

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

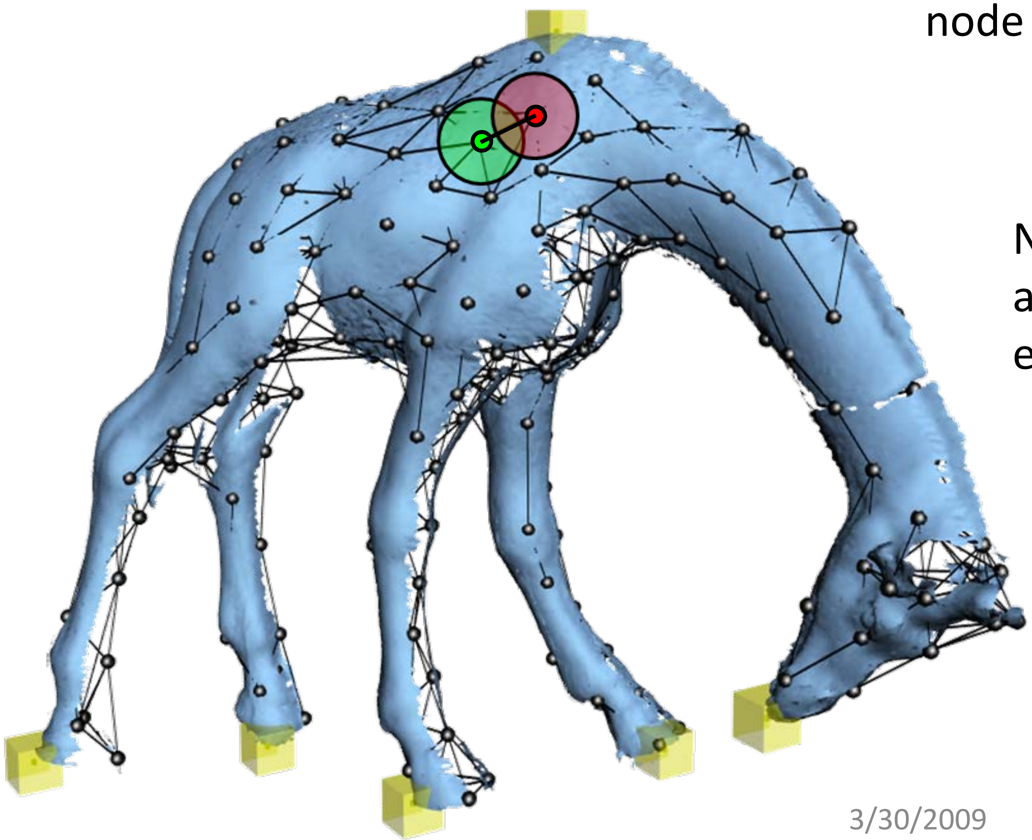
$$\mathrm{E}_{\mathrm{rot}} = \sum_{j=1}^{m} \mathrm{Rot}(\mathbf{R}_j)$$



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

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

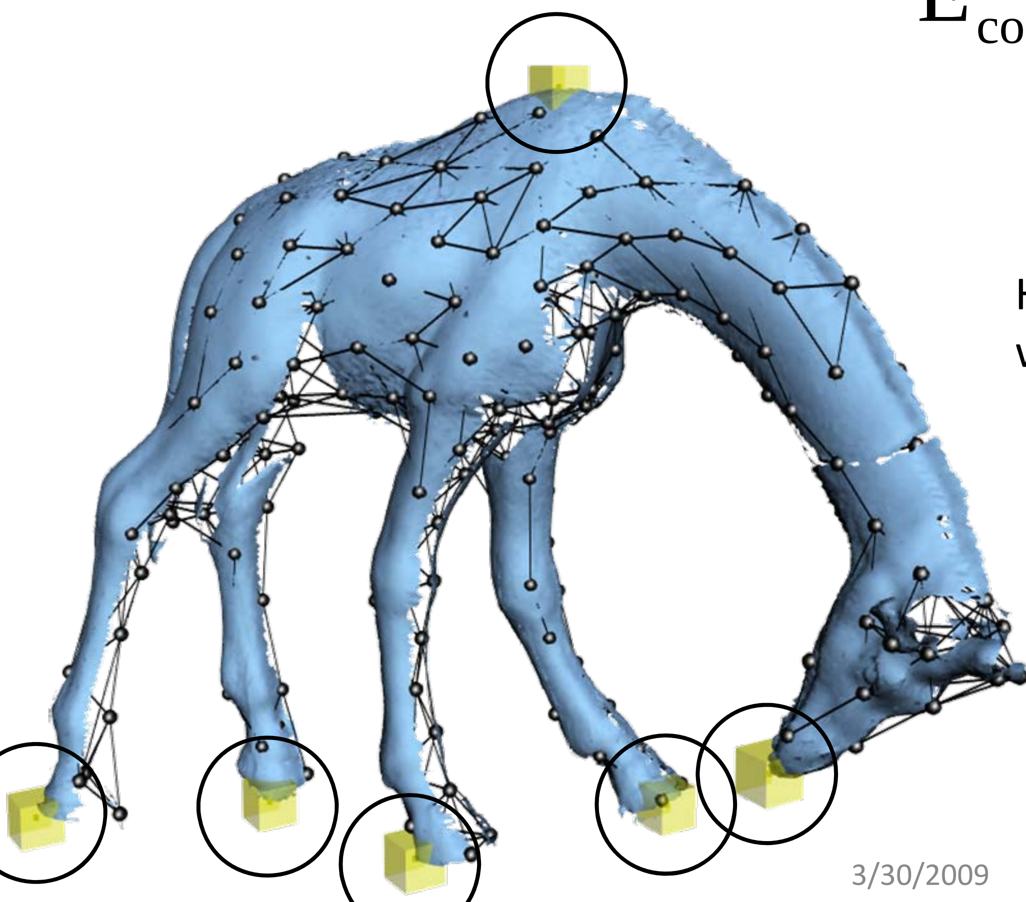
$$\mathrm{E}_{\mathrm{reg}} = \sum_{j=1}^{m} \sum_{k\in \mathrm{N}(j)} \alpha_{jk} \left\| \underbrace{\mathbf{R}_j(\mathbf{g}_k - \mathbf{g}_j) + \mathbf{g}_j + \mathbf{t}_j}_{\text{where node } j \text{ thinks node } k \text{ should go}} - \underbrace{(\mathbf{g}_k + \mathbf{t}_k)}_{\text{where node } k \text{ actually goes}} \right\|_2^2$$



Neighboring nodes should agree on where they transform each other.

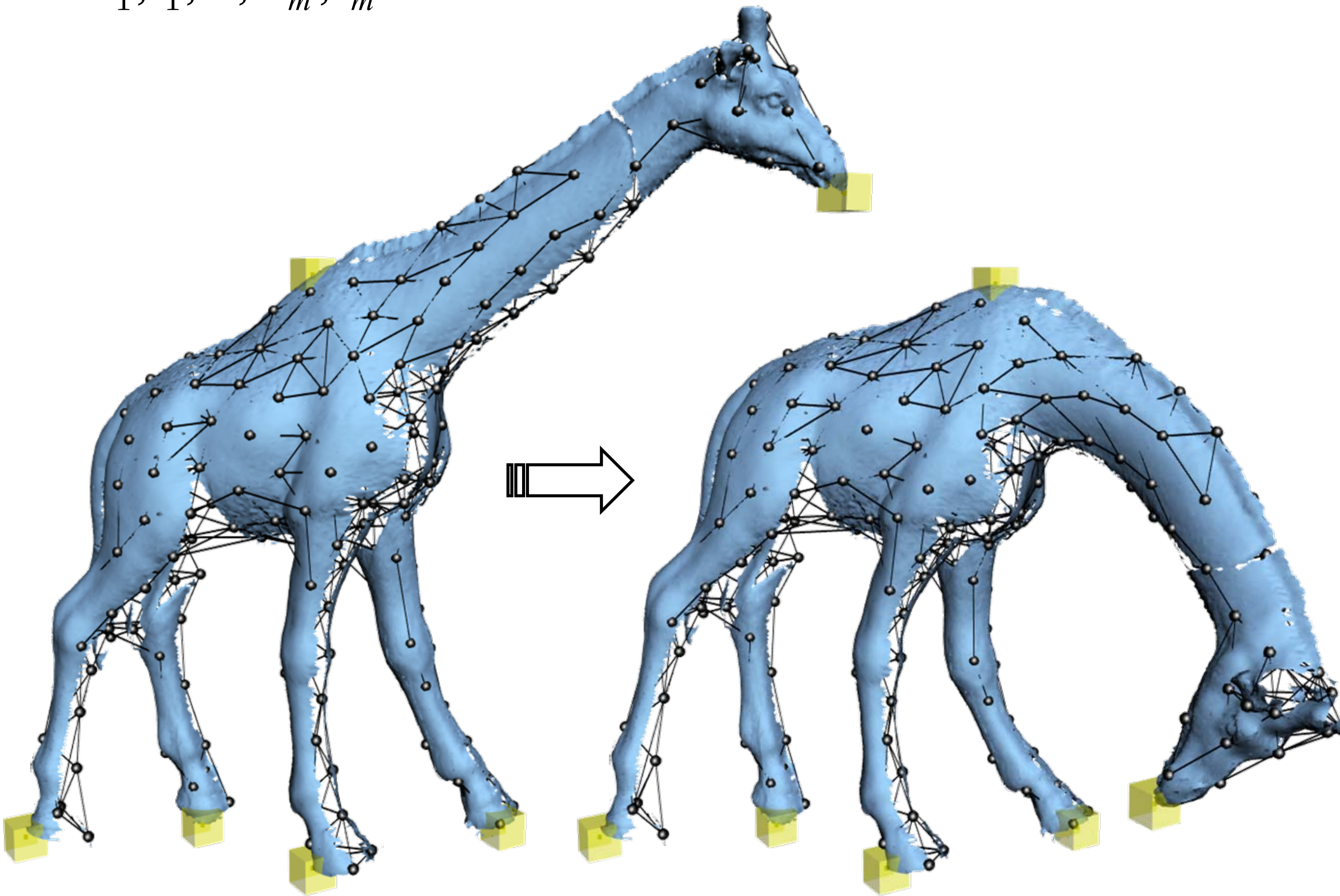
$$\min_{\mathbf{R}_1,\mathbf{t}_1,\ldots,\mathbf{R}_m,\mathbf{t}_m} \quad w_{\text{rot}}\mathrm{E}_{\text{rot}} + w_{\text{reg}}\mathrm{E}_{\text{reg}} + w_{\text{con}}\mathrm{E}_{\text{con}}$$

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



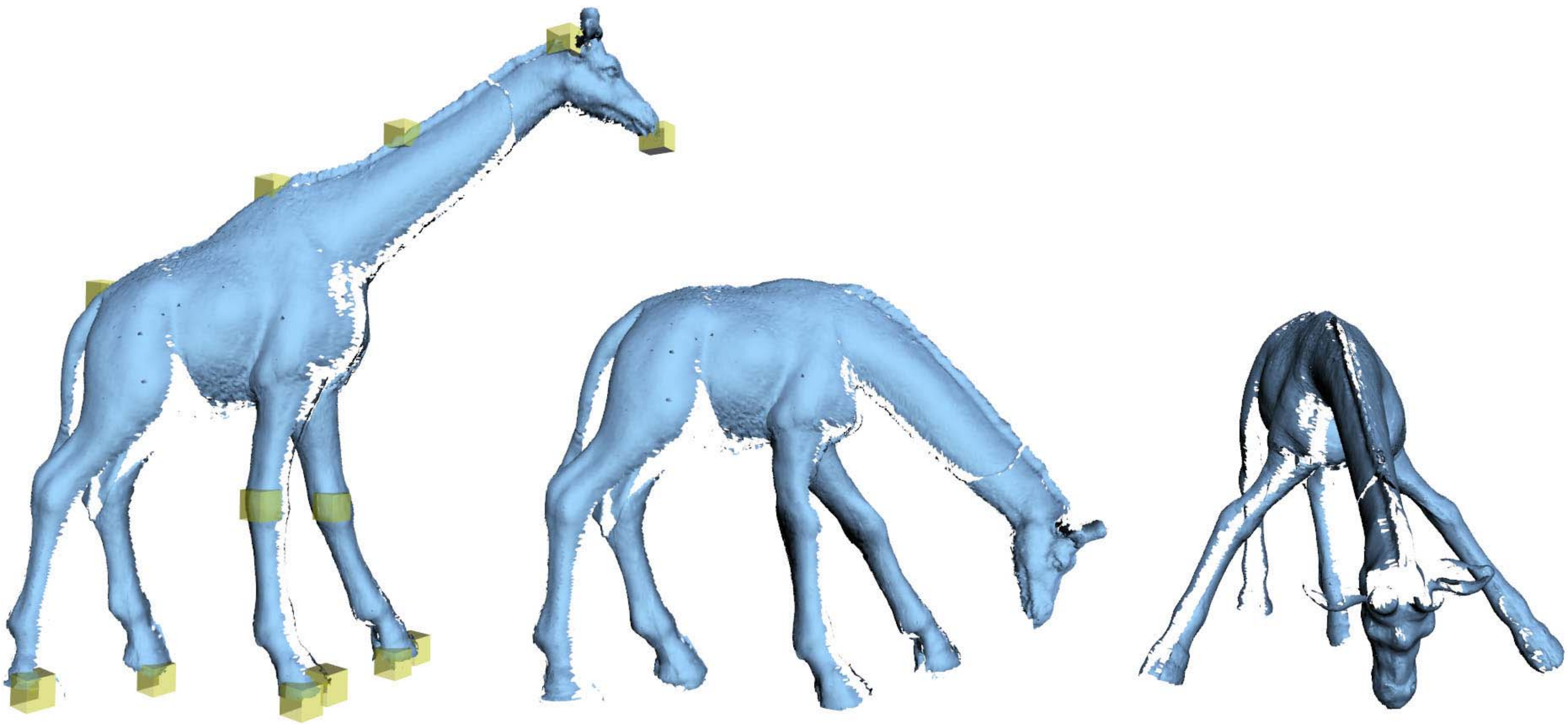
Handle vertices should go where the user puts them.

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



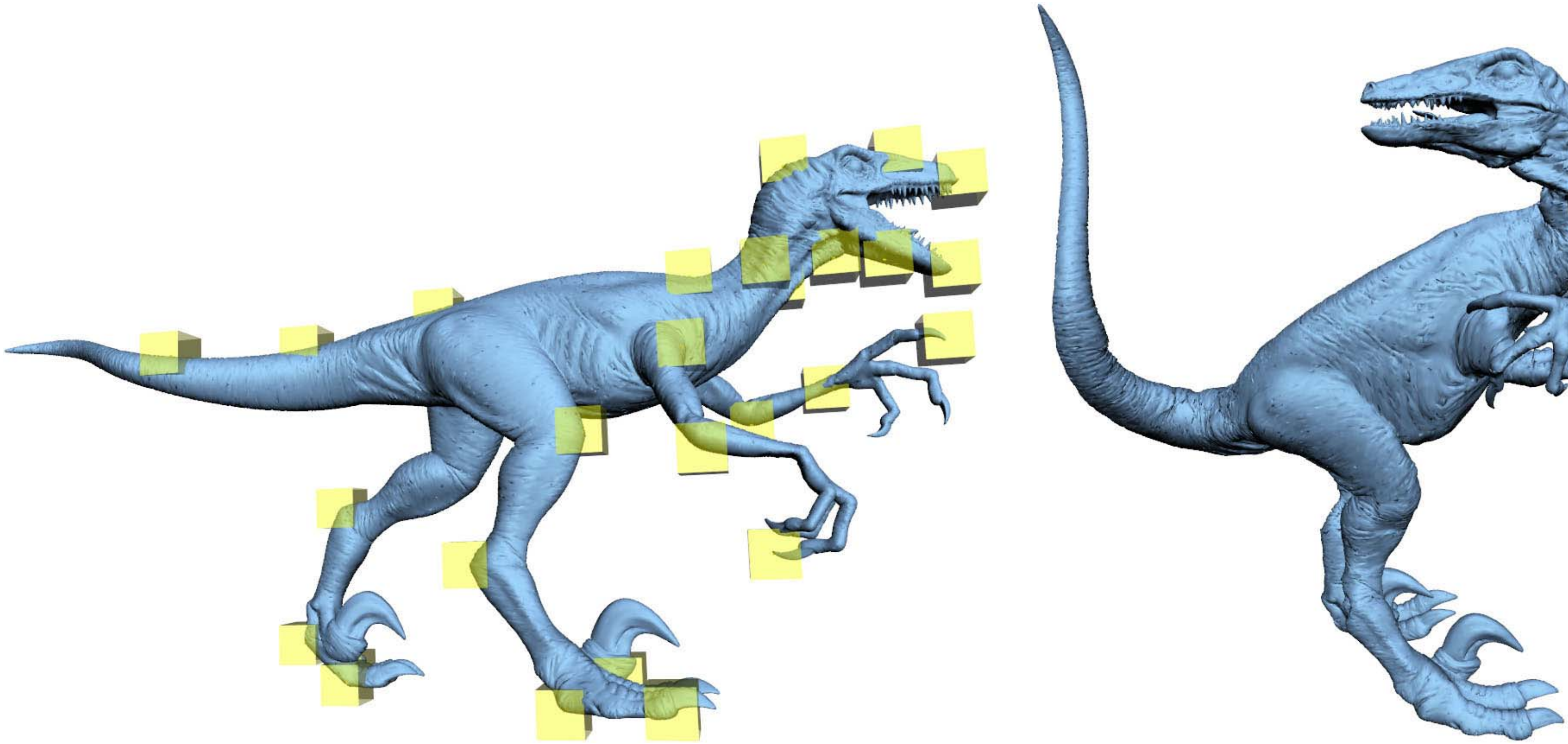
Results: Polygon Soup

[Sumner et al. 2007]



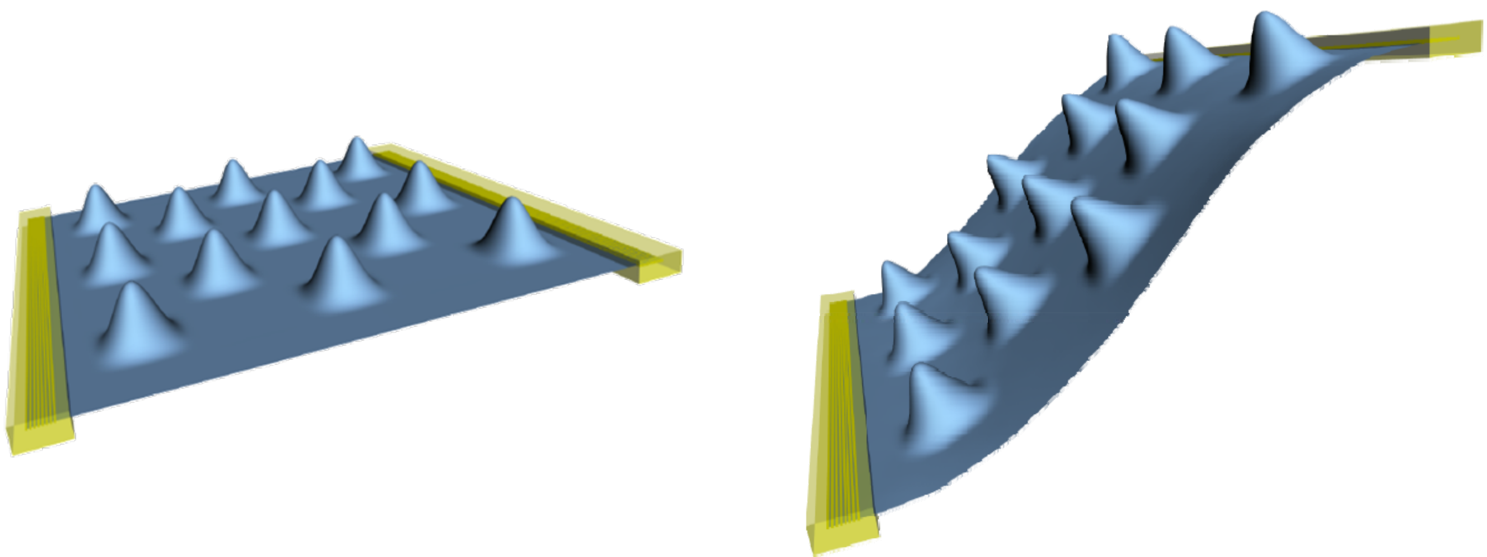
Results: Giant Mesh

[Sumner et al. 2007]



Results: Detail Preservation

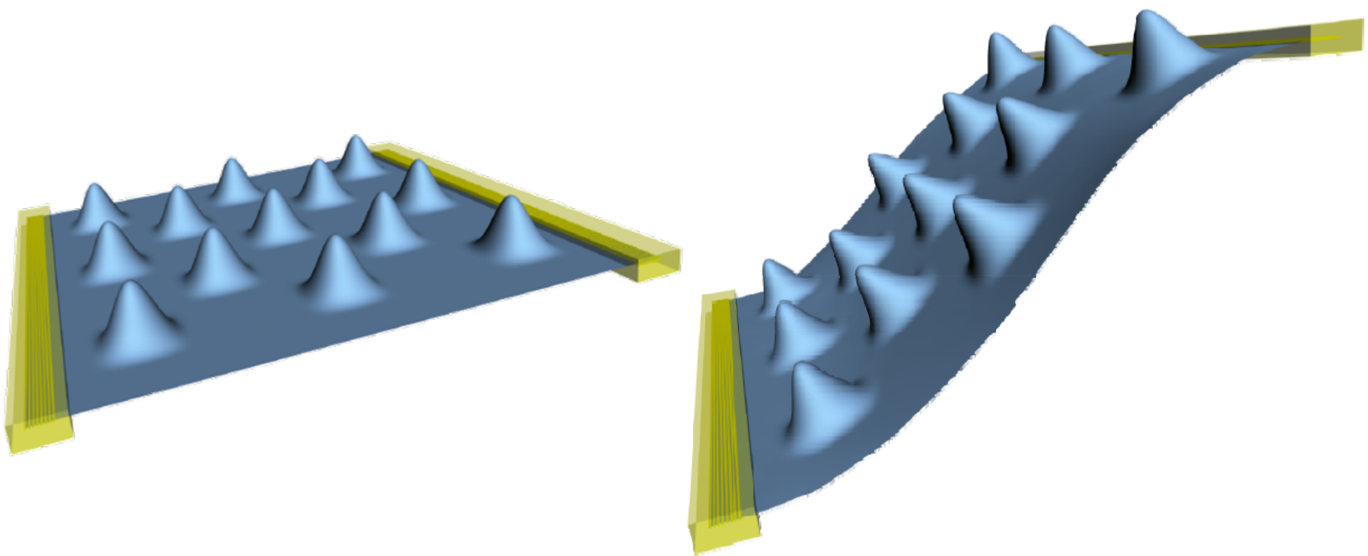
[Sumner et al. 2007]



Demo

Discussion

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



Eurographics 2009

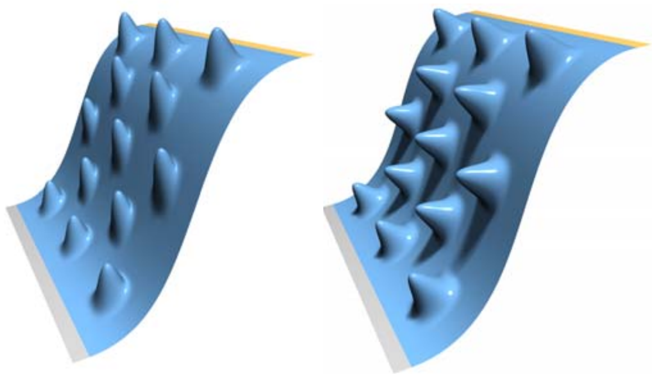
Interactive Shape Modeling and Deformation

T3: Half-Day Tutorial

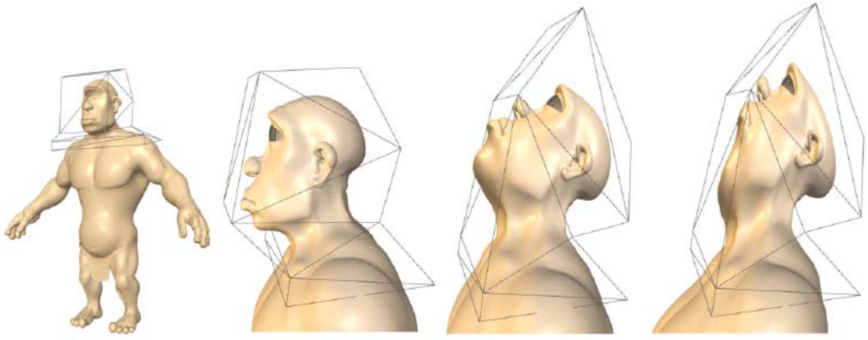
Wrap-up

Research trends

- From linear to nonlinear techniques

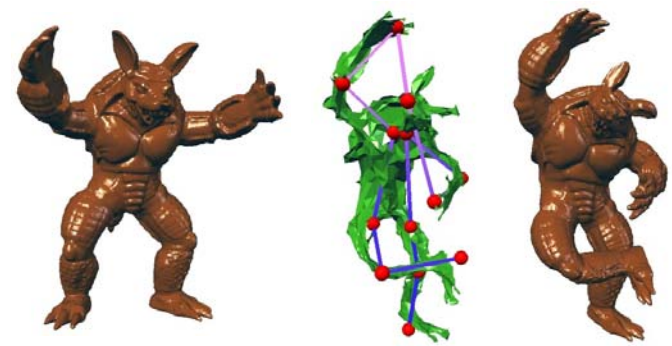


- Surface-based methods and space warps developed simultaneously



Future work?

- Higher-level editing
  - … with semantic understanding of the shape
  - … with “pseudo-physics” automatically set up from that understanding
- Hybrids between surface- and space-based methods



Acknowledgments

- The authors of the numerous papers presented in our tutorial, for their inspiring research
- Pierre Alliez, Eitan Grinspun, Tao Ju, Andrew Nealen, Mark Pauly, Scott Schaefer, Bob Sumner, Jens Vorsatz for kindly providing their slides

- Olga Sorkine's presentation was supported by an ADVANCE Women-in-Science Travel Grant funded by the NSF ADVANCE PAID award #HRD-0820202