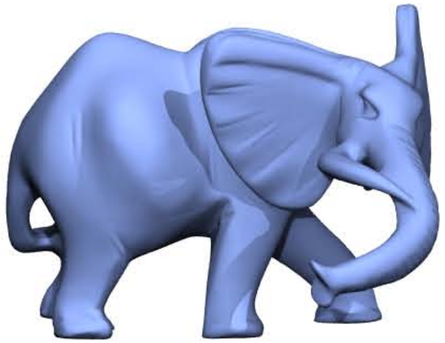
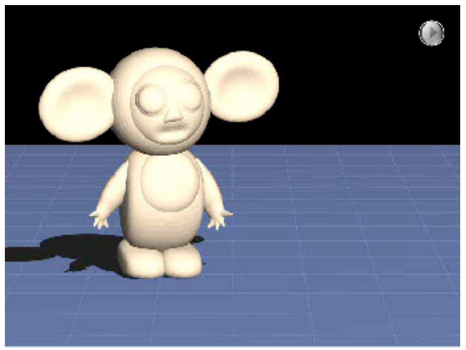
# Mesh Deformation

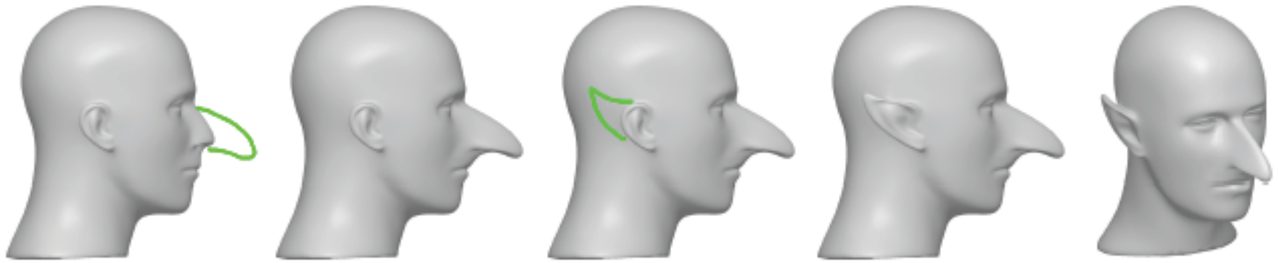


# Motivation

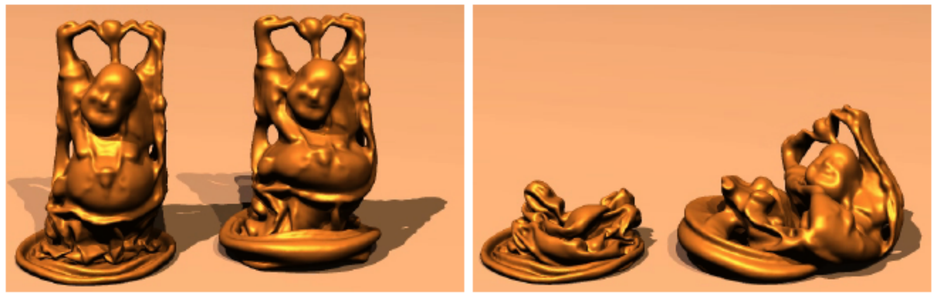
- Animation




- Editing

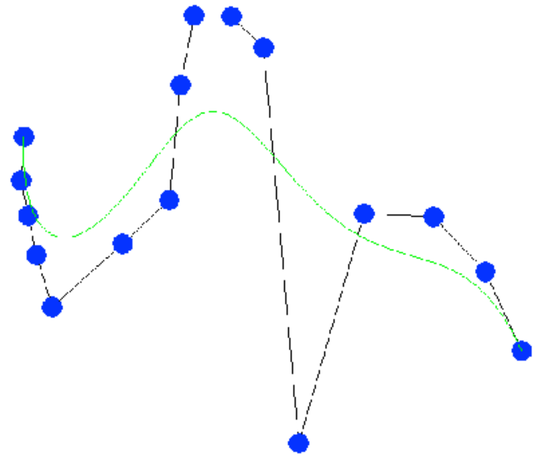


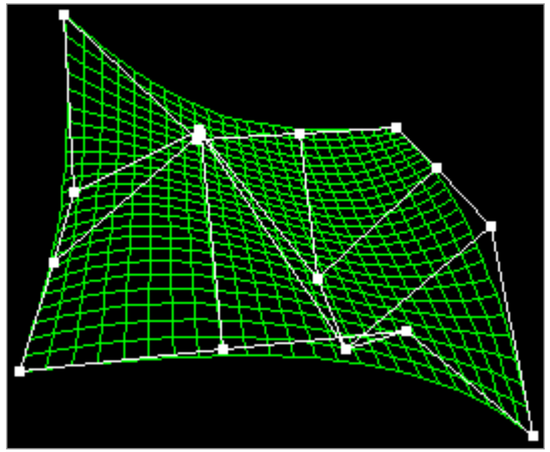
- Simulation



# Parametric Curves/Surfaces

- Deformation by control point  manipulation
- Built-in deformation mechanism
- Control structure is pre-set
  (cannot pull on arbitrary points)

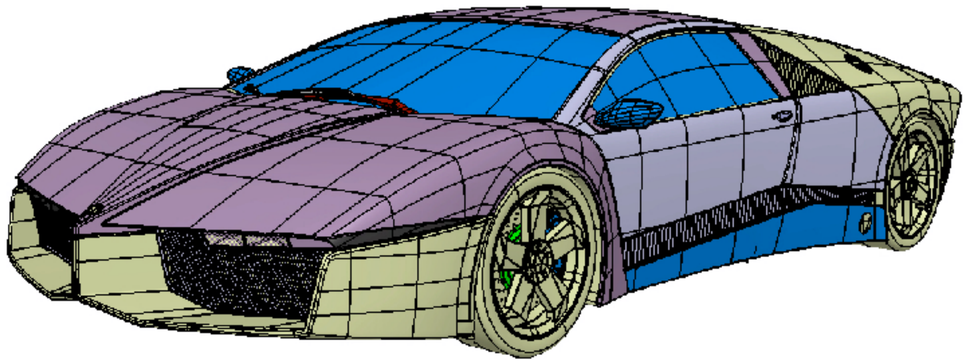


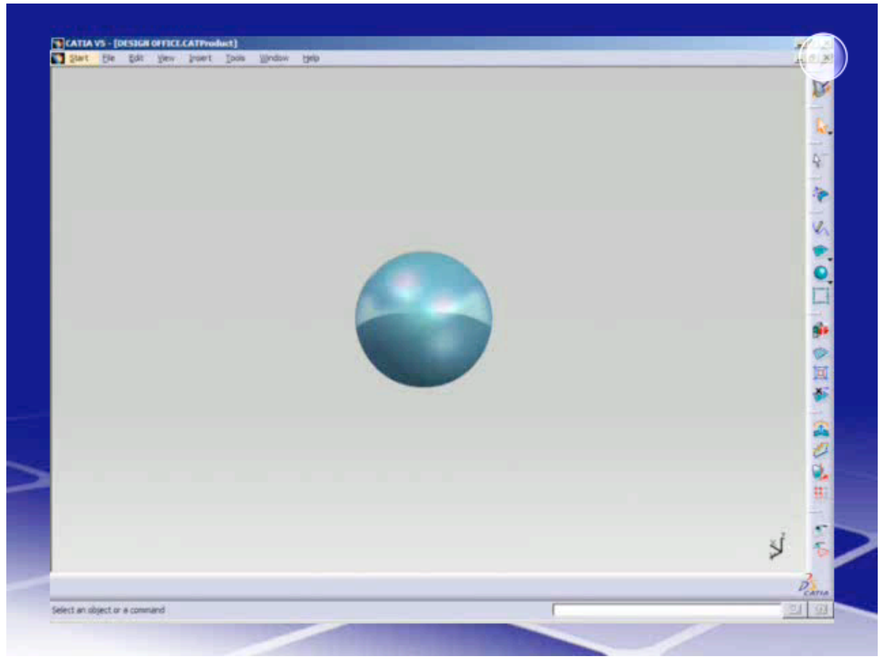


# Parametric Curves/Surfaces

- Hard to change/adapt control structure to user needs

- Hard to experiment, need a precise idea of what to model



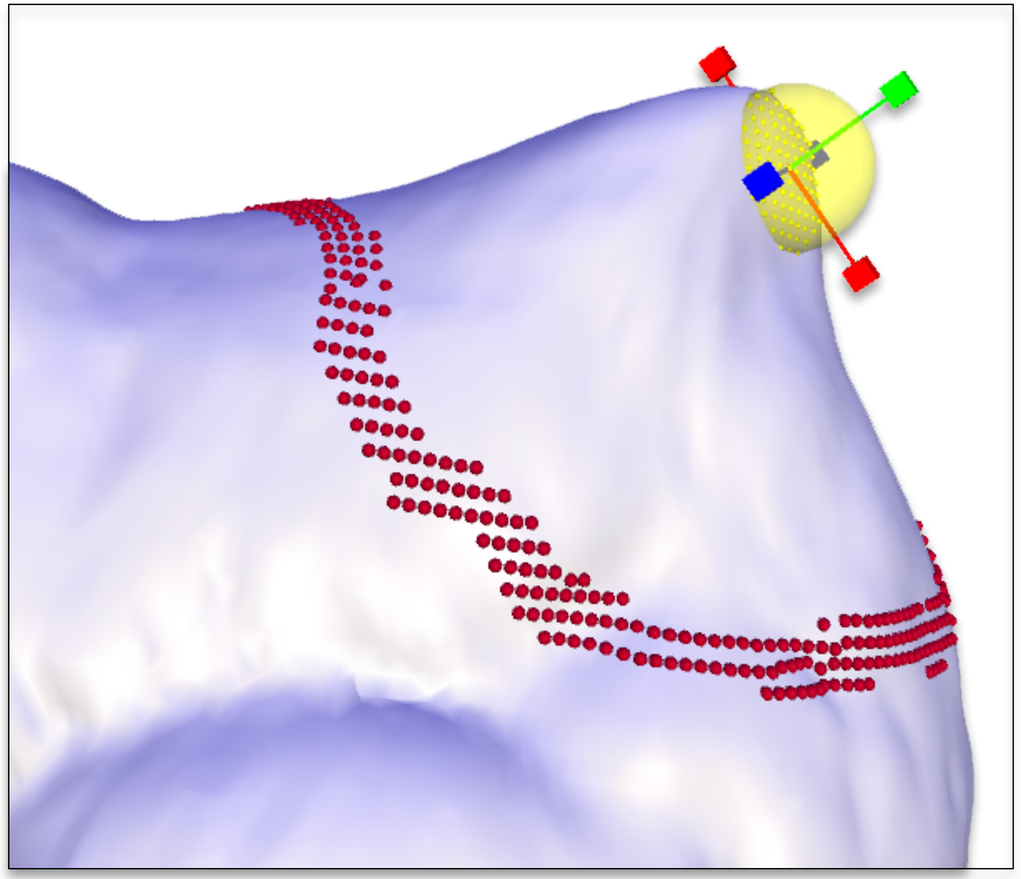



CATIA, Dassault Systemes

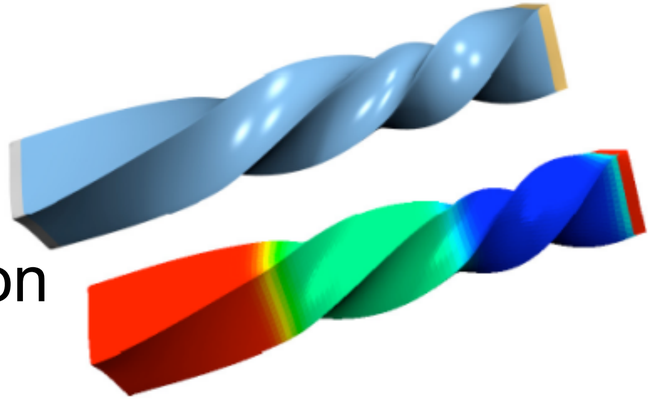
http://youtu.be/gTC5zMktMr0

# Mesh Deformation

- Naïve method: dragging single vertices
- Smarter:
  - Introduce a small set of deformation handles
    - Makes deformation/editing easier
    - Introduces a trade-off between degrees of freedom and simplicity of the deformation task
  - Create a small set of control parameters
    - Affine transformations



# Deformation: Common Paradigms

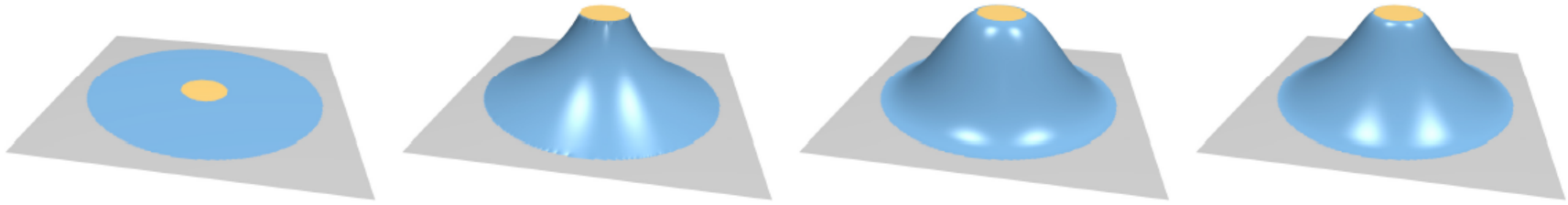


- **Surface based deformation**
  - Optimization on the surface
  - Physically motivated: variants of elastic energy minimization

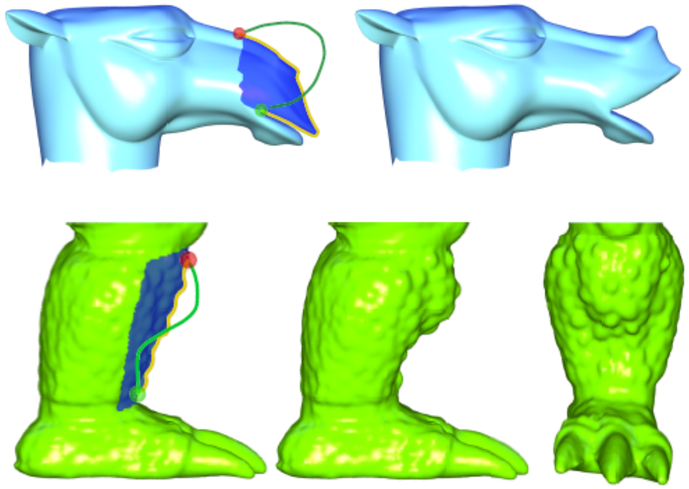
- **Space deformation**
  - Deforms 2D/3D space using a *cage*
  - Propagate deformation to all points in the space
  - Independent of shape representation

# Deformation Interfaces

- Region of interest (ROI) + affine deformation **handle** with variable boundary continuity

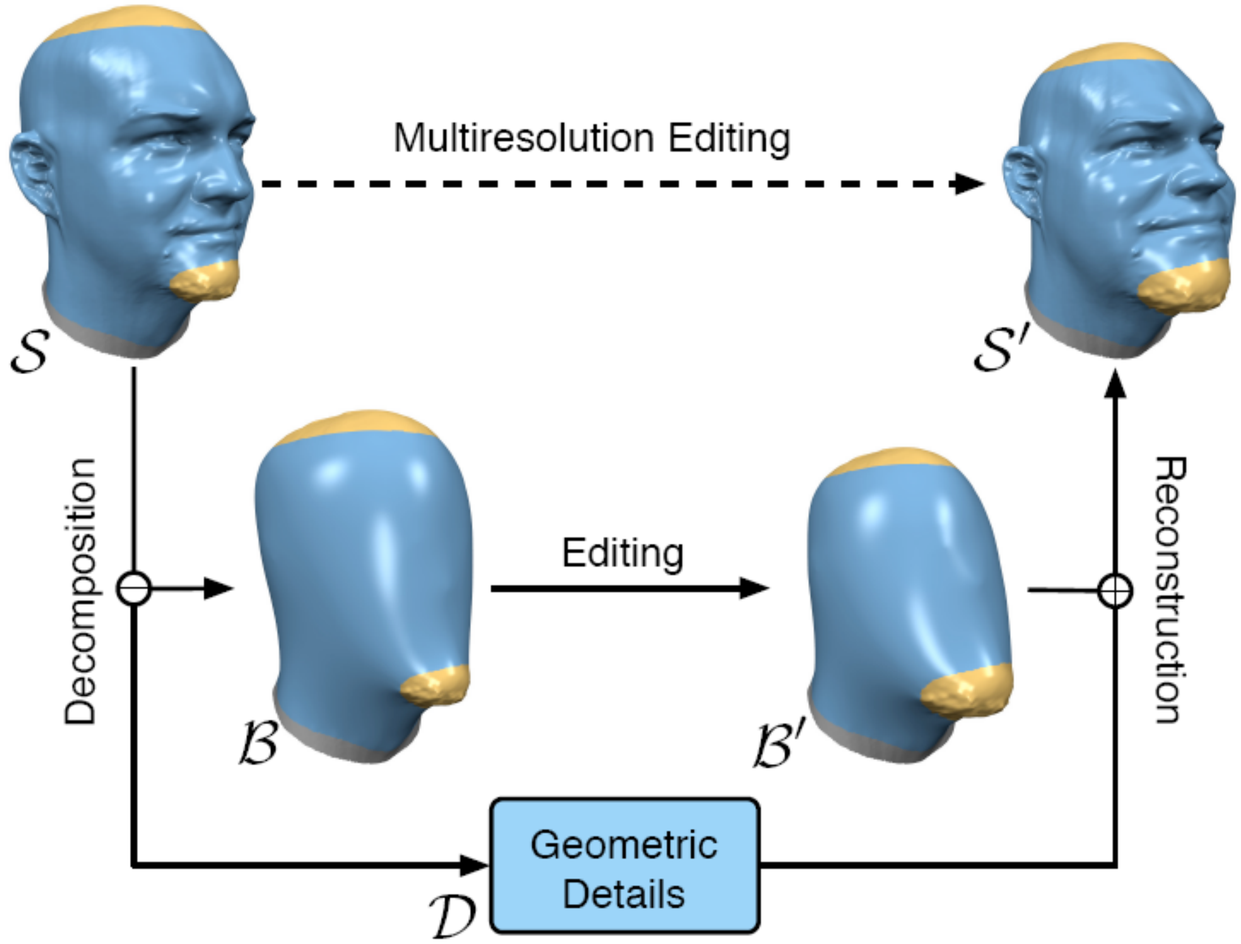


- Intuitive **sketch-based** deformation interfaces



# Scale-space Deformation

- Multi-resolution mesh editing



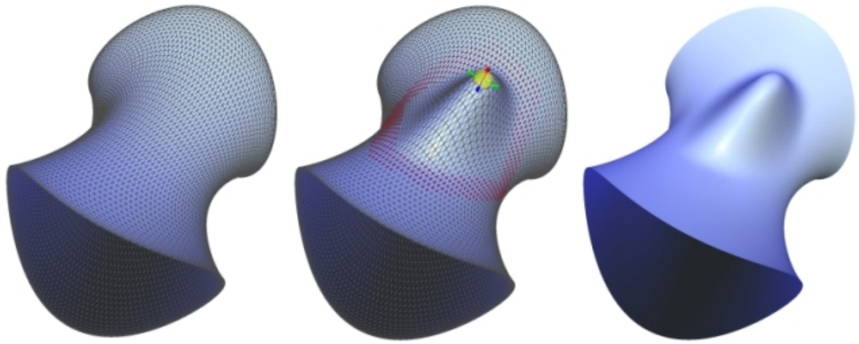
# General Framework

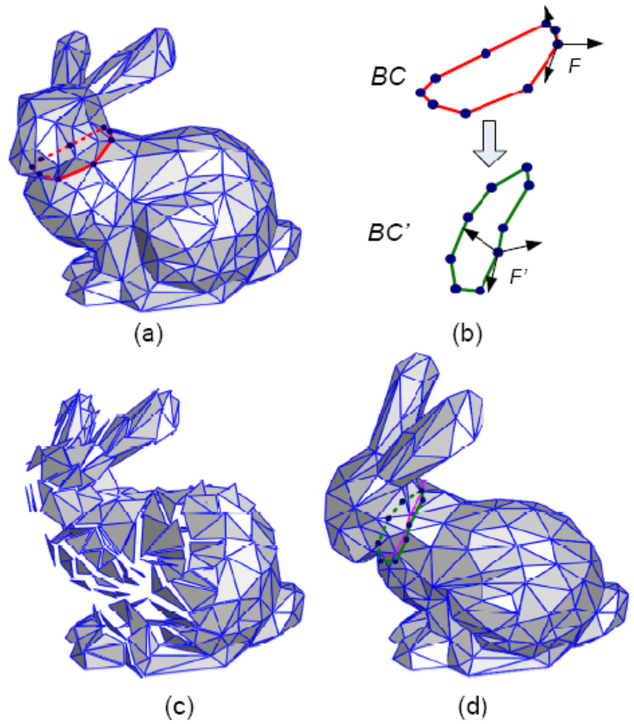
- Find a mesh that optimizes some **objective function** and satisfies **modeling constraints**

$$\mathbf{x}_{\text{def}} = \underset{\mathbf{x}'}{\operatorname{argmin}}\, E(\mathbf{x}') \quad s.t. \quad \mathbf{x}'_i = \mathbf{c}_i$$

# Surface-based Deformations: Linear Methods

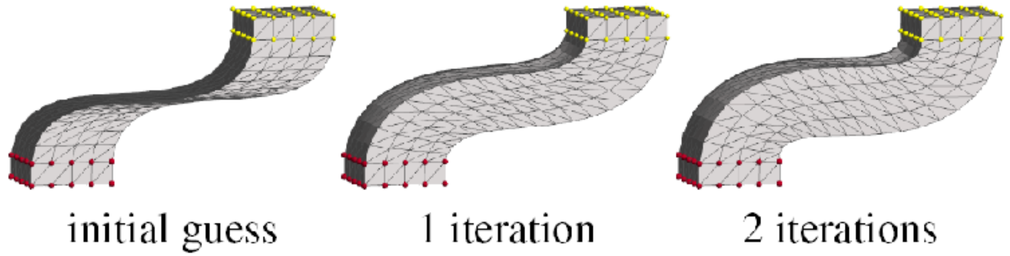
- (2D) As-rigid-as-possible shape manipulation (SIGGRAPH 2005)
- Triangle gradient methods (2004-2005)
- Laplacian surface editing (2004-2005)

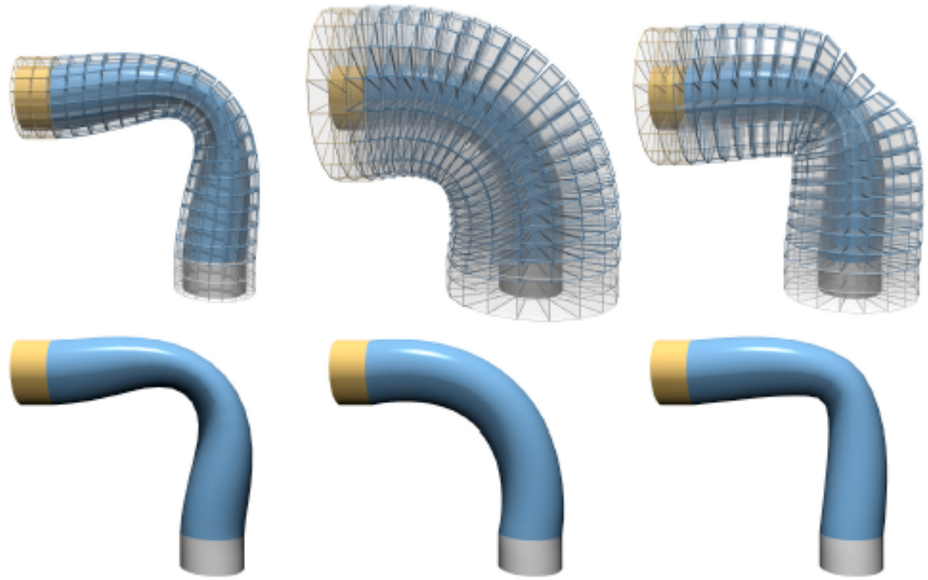




# Surface-based Deformations: Nonlinear Methods

- As rigid as possible surface modeling (SGP 2007)
- PriMo (SGP 2006)
- Mesh Puppetry (SIGGRAPH 2007)



# Summary: Surface Deformations

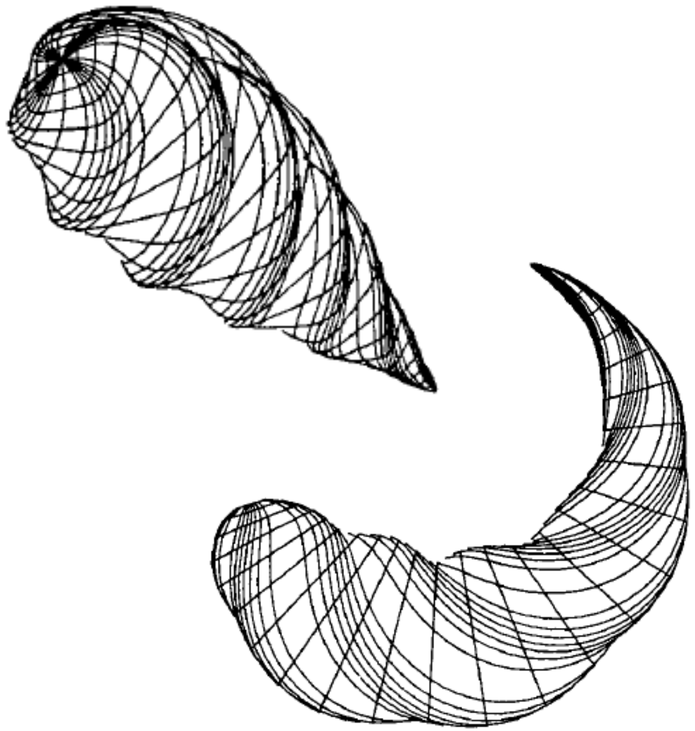
- Objective functional expressed in the mesh elements (vertices)
- Complexity depends on the mesh resolution
- Linear methods:
  - Solve a global linear system on the mesh
  - Usually suffer from some artifacts
- Nonlinear methods
  - Fewer artifacts but slower, and harder to implement

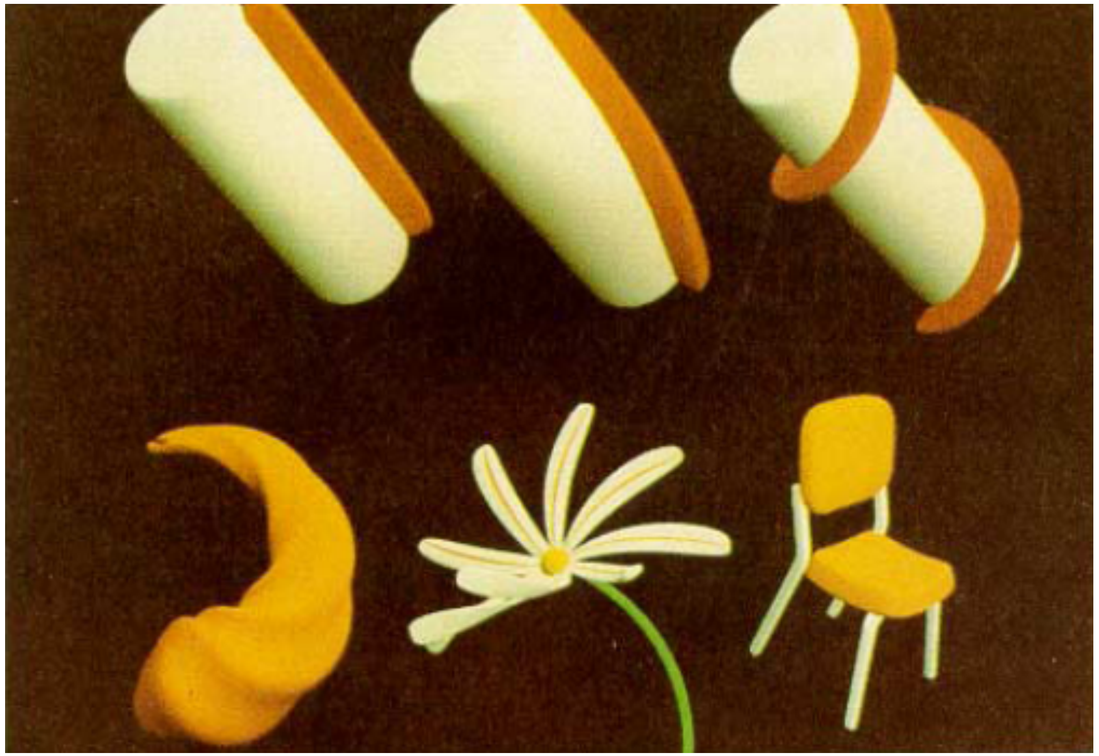
# Space Deformations

## Early seminal work in computer graphics

- Global and local deformation of solids [Barr 1984]

$$F : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$



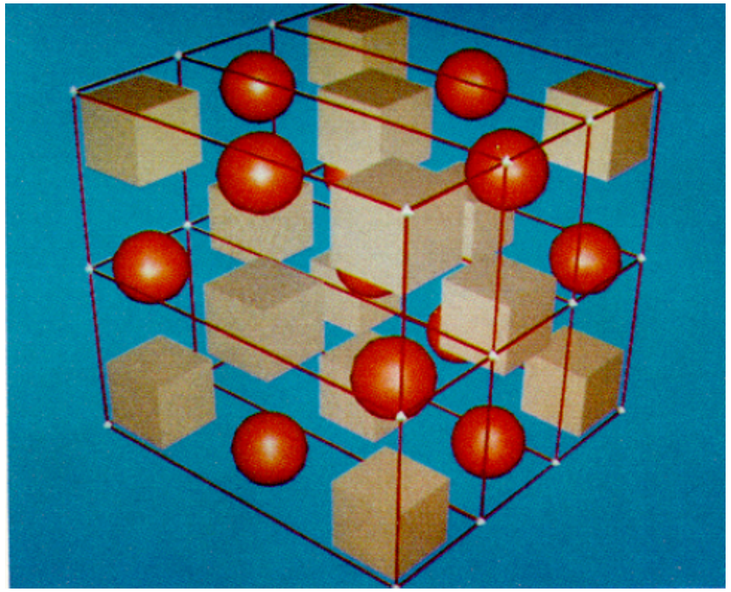


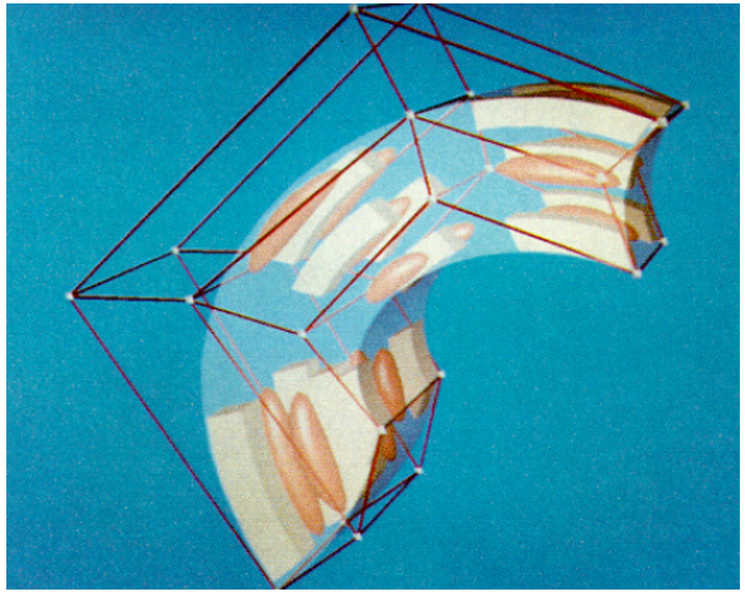
# Space Deformations

## Early seminal work in computer graphics

- Free form deformations
  [Sederberg and Parry 1986]
  - Uses trivariate tensor product polynomial basis

$$f : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$





# Space Deformations

## Early seminal work in computer graphics

- Can be designed to be volume preserving $f : \mathbb{R}^3 \rightarrow \mathbb{R}^3$


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

then the Jacobian is the determinant

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

$$\nabla \mathbf{F} := [\mathbf{F}_x \quad \mathbf{F}_y \quad \mathbf{F}_z]$$

# Space Deformations: Basic Idea

- Design a set of coordinates for all points in $R^d$ w.r.t. the “cage” vertices
  - Each point **x** can be represented as a weighted sum of cage points $\mathbf{p}_i$

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

  - When the cage changes the coords ($w_i$) stay the same, substitute the new cage geometry:

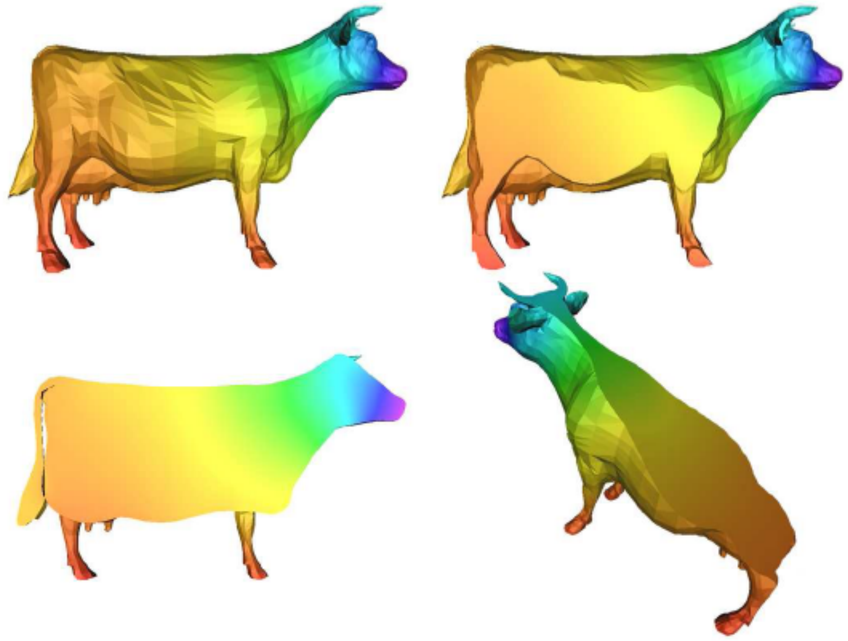
$$\mathbf{x}' = \sum_{i=1}^{k} w_i(\mathbf{x})\, \mathbf{p}'_i$$

# Space Deformations: Basic Idea

- Design a set of coordinates for all points in $R^d$ w.r.t. the “cage” vertices
  - Each point **x** can be represented as a weighted sum of cage points $\mathbf{p}_i$

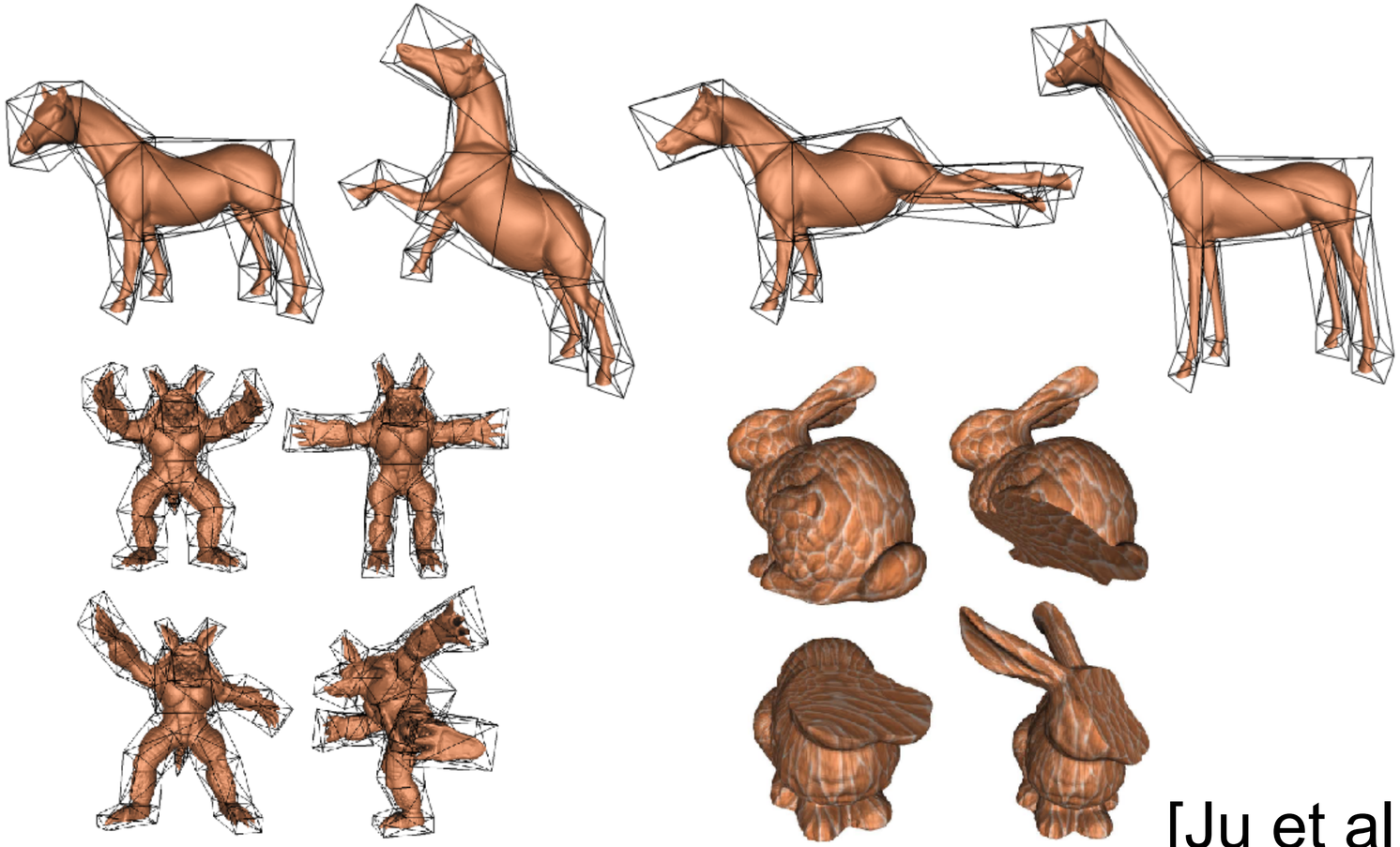
$$\mathbf{x} = \sum_{i=1}^{k} w_i(\mathbf{x})\, \mathbf{p}_i$$

  - The coordinates are smoothly varying and guarantee continuity inside the volume



# Example #1: Mean Value Coordinates

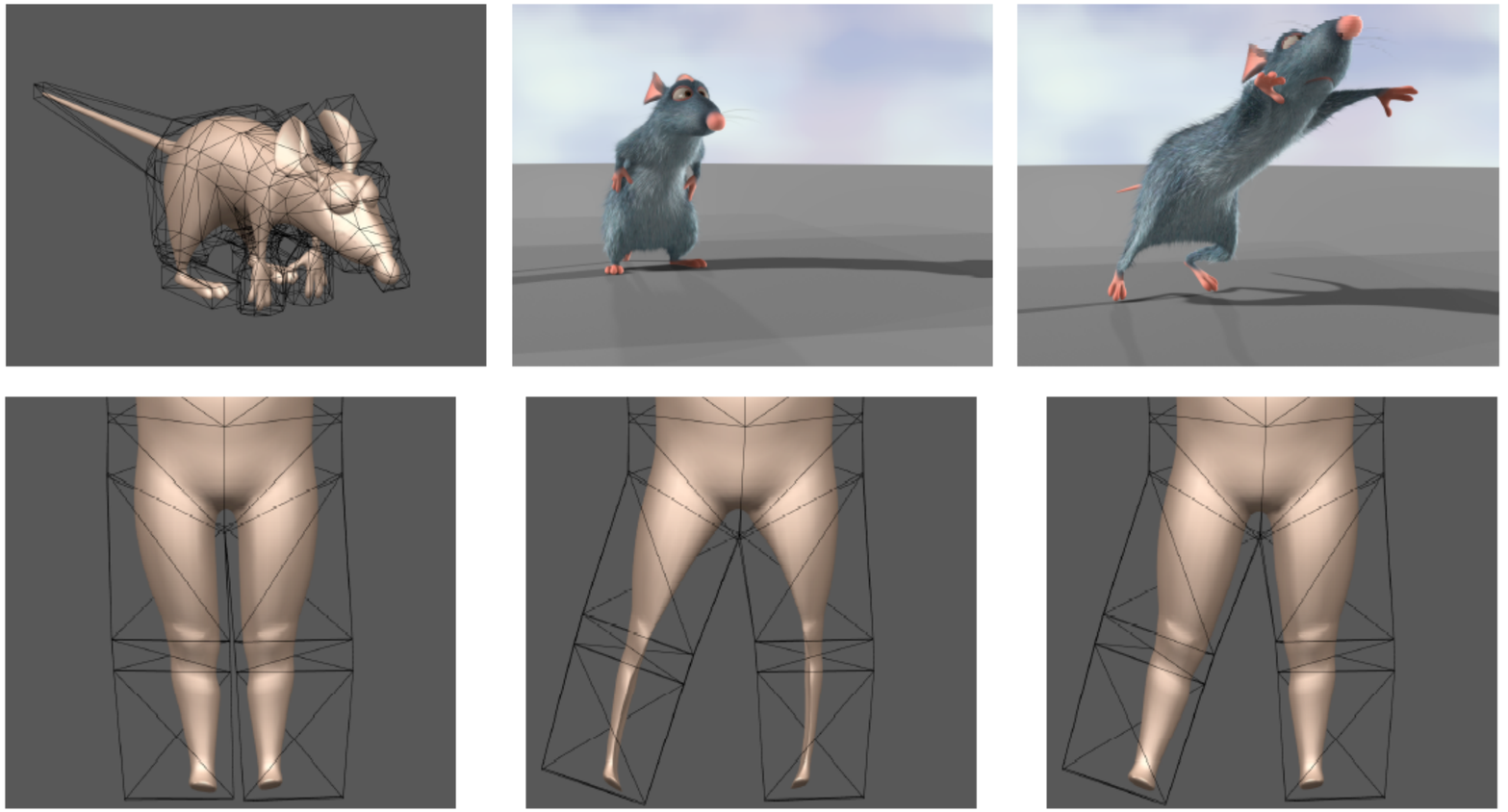
- Mean value coordinates for closed triangular mesh



[Ju et al. 2005]

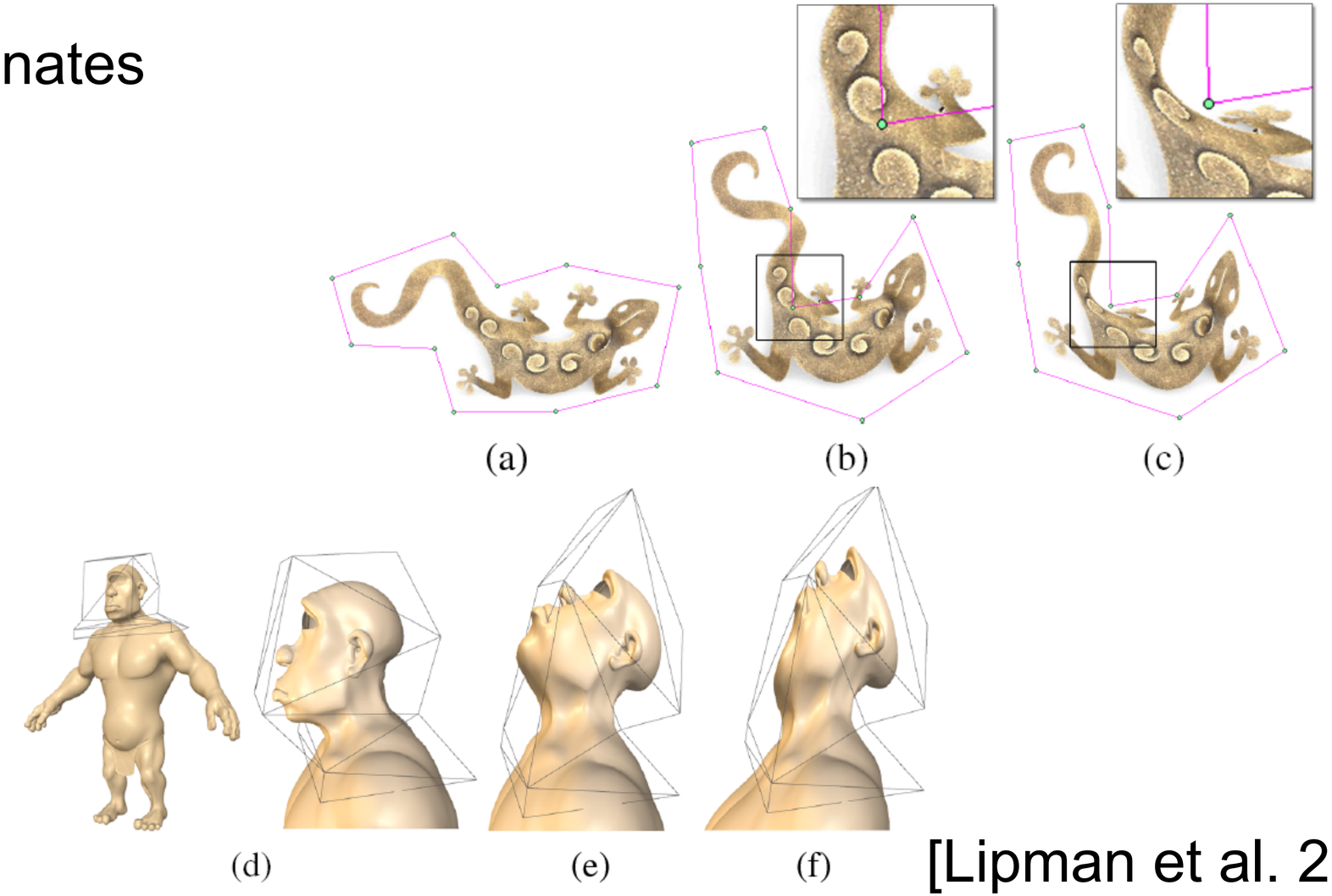
# Example #2: Harmonic Coordinates

- Harmonic coordinates [Joshi et al. 2007]



# Example #3: Green Coordinates

- Green coordinates



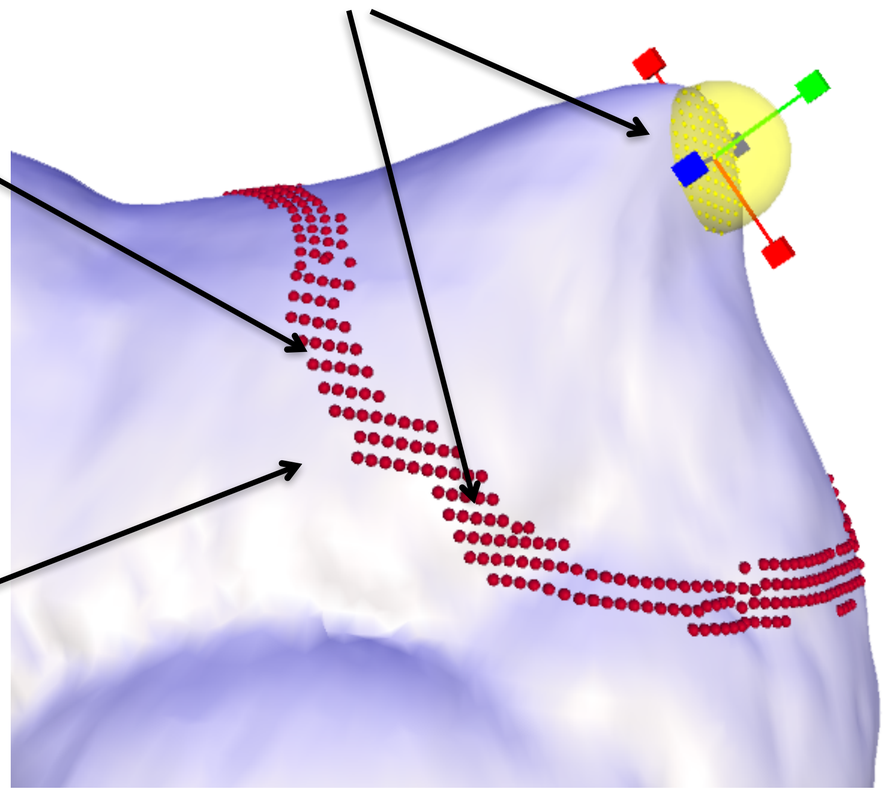

[Lipman et al. 2008]

# Summary: Space Deformations

- Complexity depends mainly on the cage;
  linear in the number of mesh elements
  - Parallel execution, GPU!
- Can handle disconnected components or
  even just point sets
- Harder to control the surface properties since the whole space is being warped
  *(lack fine grained deformation)*

# Surface-based Deformation: ROI-Handle Editing Metaphor

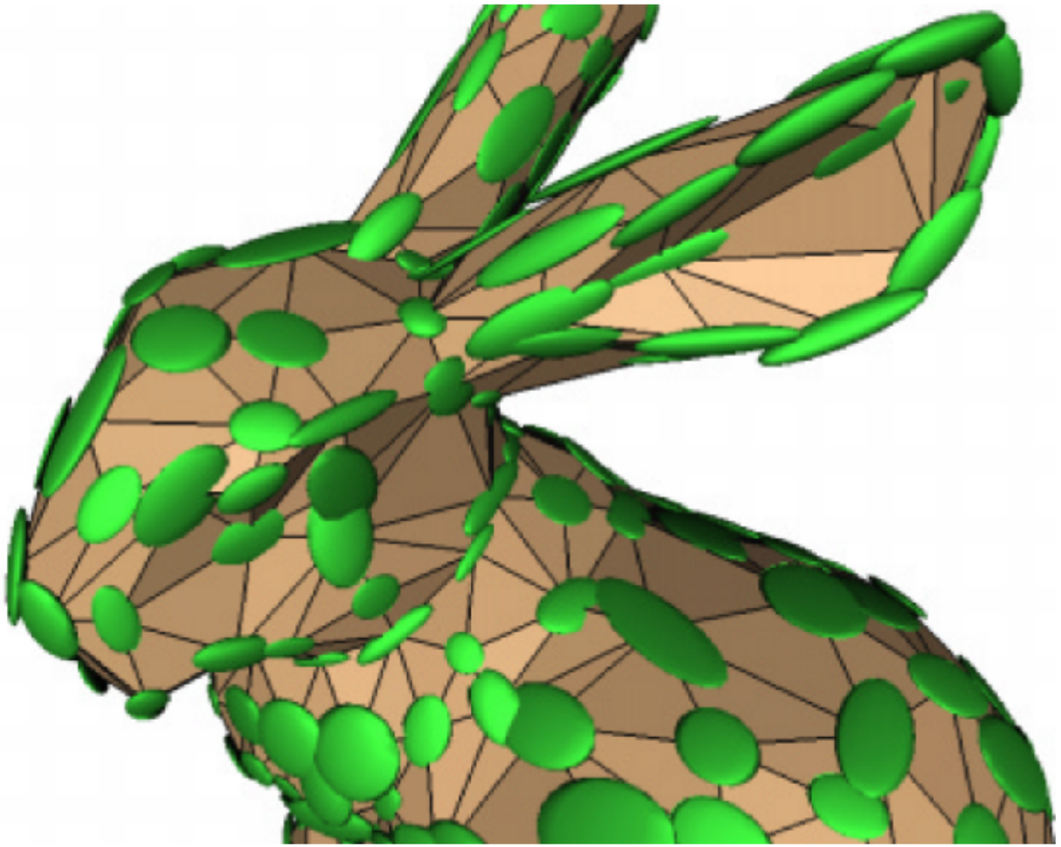
$$\mathbf{x}_{\text{def}} = \underset{\mathbf{x}'}{\operatorname{argmin}}\, E(\mathbf{x}') \quad s.t. \quad \mathbf{x}'_i = \mathbf{c}_i$$




- ROI is bounded by a belt (static anchors)
- Manipulation through handle(s) – affine transformations

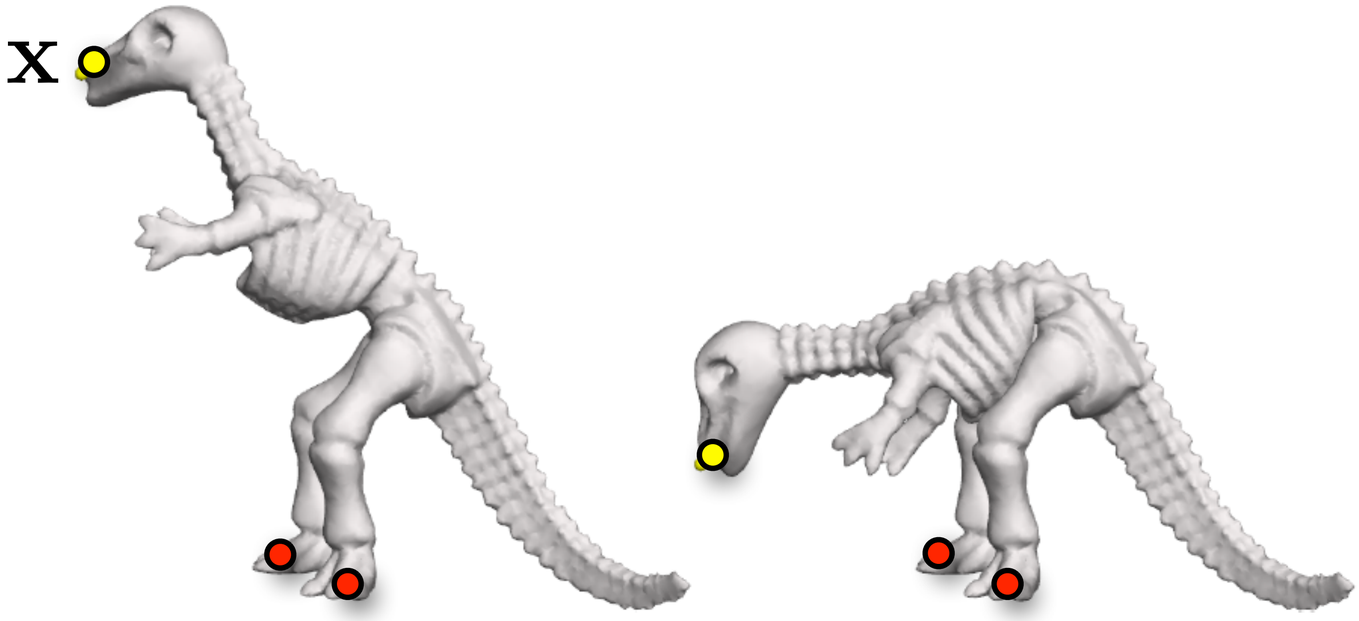
# Mesh Simplification

# Error Quadric



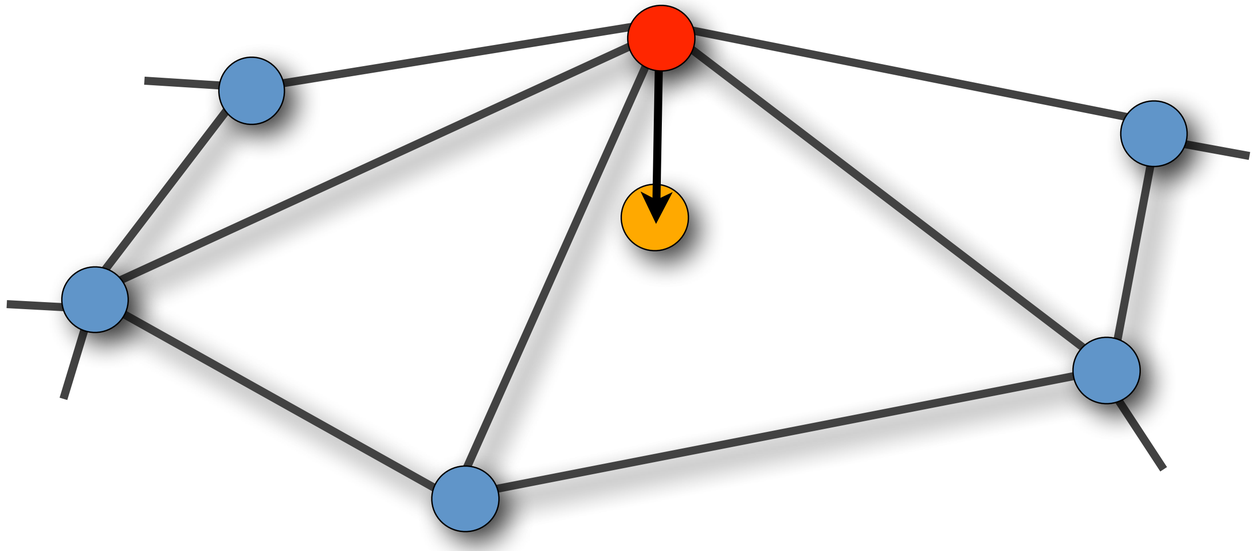
# How to Define $E(\mathbf{x}')$

- Intuitive deformations:
  - Smooth deformation on the global scale
  - Preserve local details (curvatures)
- Invariants: $E(x')$ should be zero if $x'$ is a rigid transformation of original geometry



# Recap: Differential Coordinates

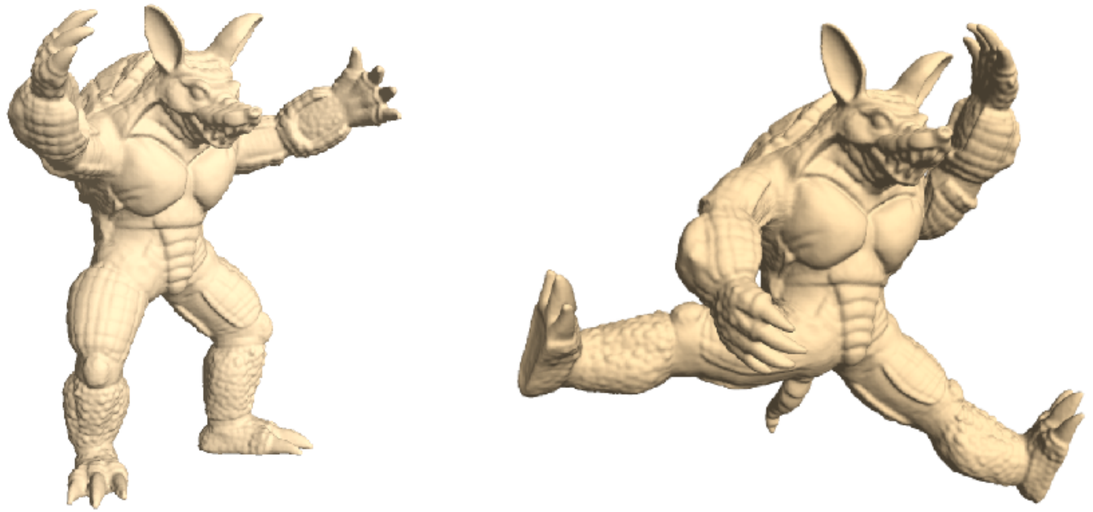
- Detail = *smooth*(surface) – surface
- Smoothing = averaging



$$\delta_i = \frac{1}{W_i} \sum_{j \in \mathcal{N}(i)} w_{ij}(\mathbf{x}_j - \mathbf{x}_i) \approx -2H_i\mathbf{n}_i$$

# Recap: Differential Coordinates

- Represent ***local detail*** at each surface point
  - Intrinsic to the shape unlike $xyz$
- Linear transition from $xyz$ to $\delta$
- Useful for operations on surfaces where surface details are important



# Simple Laplacian Editing

- Preserve mean curvature normal [≈differential coordinates] at every point in the ROI [≈ every vertex of the ROI]

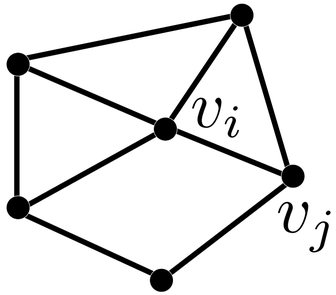
continuous: $$E(\mathcal{S}') = \int_{\mathcal{S}'} \|\Delta \mathbf{x}' - \delta\|^2 d\mathbf{x}'$$

discrete: $$E(\mathbf{x}') = \sum_{i=1}^{n} A_i \|\Delta(\mathbf{x}'_i) - \delta_i\|^2$$

# Uniform Laplace

- Uniform discretization

$$\Delta_{\text{uni}} f(v_i) := \frac{1}{|\mathcal{N}_1(v_i)|} \sum_{v_j \in \mathcal{N}_1(v_i)} (f(v_j) - f(v_i))$$

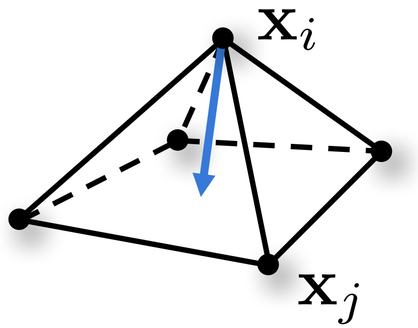



- Properties
  - depends only on connectivity
  - simple and efficient
  - bad approximation for irregular triangulations
    - can give non-zero *H* for planar meshes
    - tangential drift for mesh smoothing

# Uniform Laplace

- Uniform discretization

$$\Delta_{\text{uni}}\mathbf{x}_i \;:=\; \frac{1}{|\mathcal{N}_1(v_i)|}\sum_{v_j \in \mathcal{N}_1(v_i)} (\mathbf{x}_j - \mathbf{x}_i) \;\approx\; -2H\mathbf{n}$$

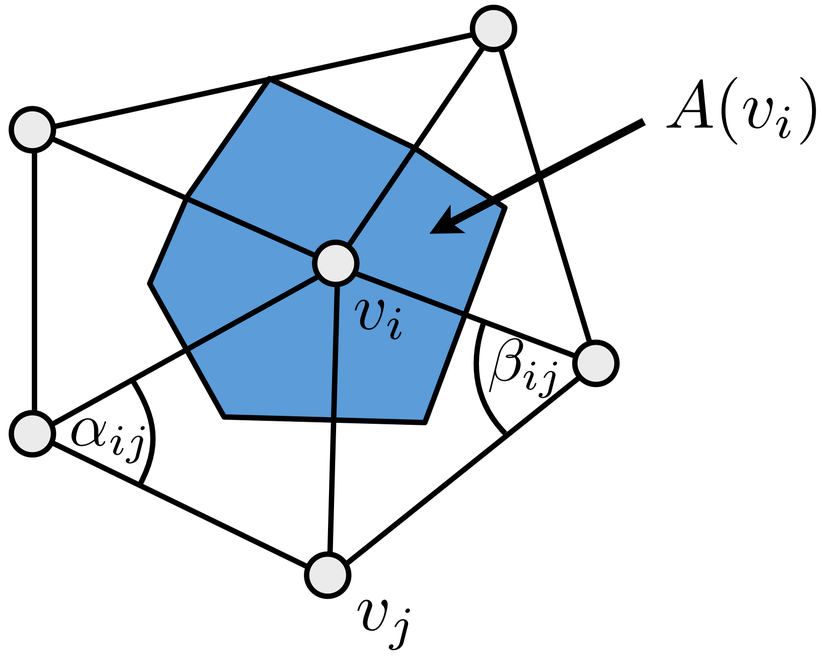



- Properties
  - depends only on connectivity
  - simple and efficient
  - bad approximation for irregular triangulations
    - can give non-zero *H* for planar meshes
    - tangential drift for mesh smoothing

# Discrete Laplace-Beltrami

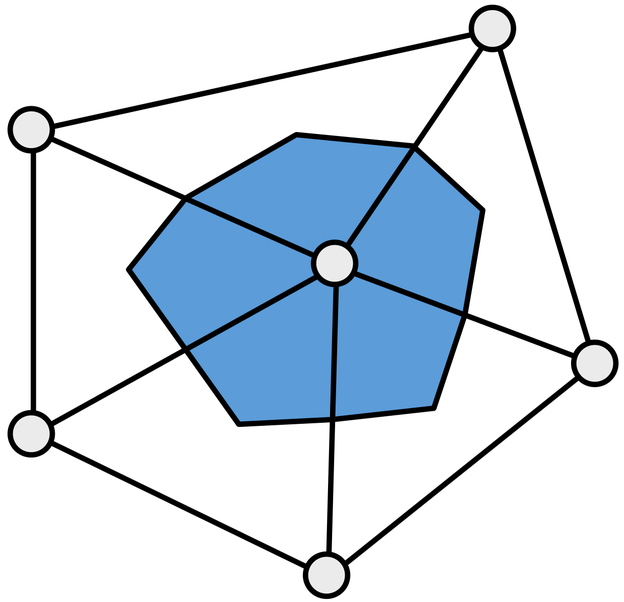
- Cotangent discretization

$$\Delta_{\mathcal{S}} f(v_i) \;:=\; \frac{1}{2A(v_i)} \sum_{v_j \in \mathcal{N}_1(v_i)} \left(\cot \alpha_{ij} + \cot \beta_{ij}\right) \left(f(v_j) - f(v_i)\right)$$



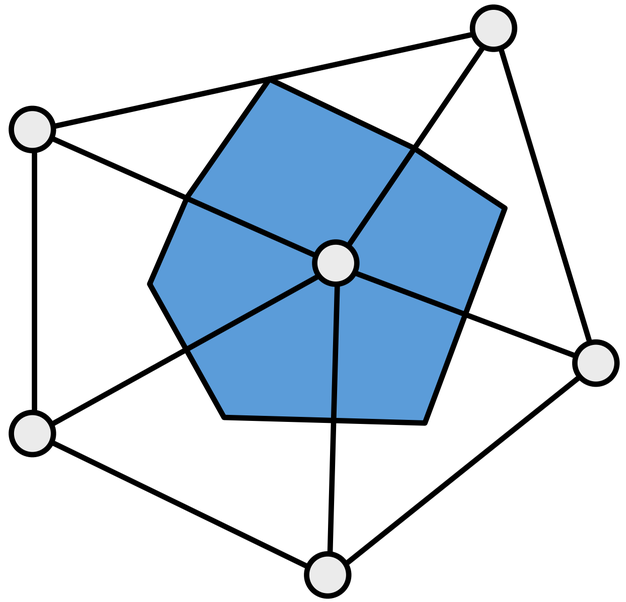
# Barycentric Cells

- Connect edge midpoints and triangle barycenters
  - Simple to compute
  - Area is 1/3 of triangle areas



# Mixed Cells

- Connect edge midpoints and
  - Circumcenters for non-obtuse triangles
  - Midpoint of opposite edge for obtuse triangles
  - Better approximation, more complex to compute...



# Discrete Laplace-Beltrami

- Cotangent discretization

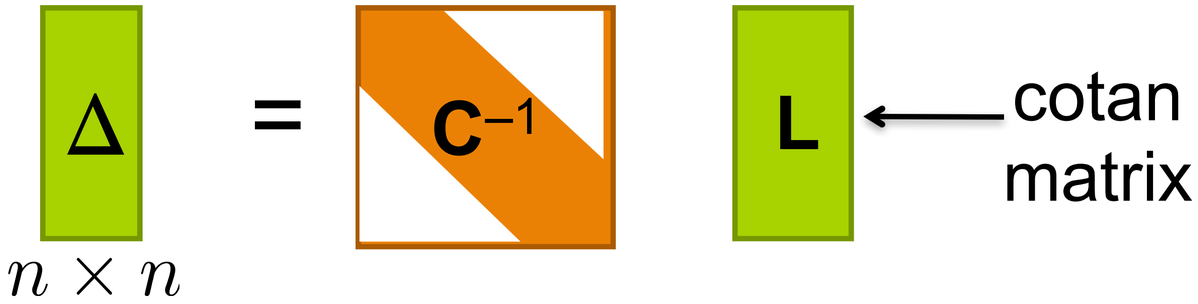
$$\Delta_{\mathcal{S}} f(v) := \frac{1}{2A(v)} \sum_{v_i \in \mathcal{N}_1(v)} (\cot \alpha_i + \cot \beta_i)\,(f(v_i) - f(v))$$

- Problems
  - weights can become negative (when?)
  - depends on triangulation
- Still the most widely used discretization

# Simplifying the Laplacian Energy

$$E(\mathbf{x}') = \sum_i A_i \|(\Delta\mathbf{x}')_i - \delta_i^{\text{org}}\|^2 = \sum_i A_i((\Delta\mathbf{x}')_i^T(\Delta\mathbf{x}')_i - 2\Delta\mathbf{x}_i'^T\delta_i^{\text{org}} + (\delta_i^{\text{org}})^T\delta_i^{\text{org}})$$

$$= (\mathbf{x}')^T\Delta^T C\Delta\mathbf{x}' - 2\mathbf{x}'^T\Delta^T C\delta^{\text{org}} + \text{constant}$$




$$\Delta^T C\Delta = (C^{-1}L)^T C(C^{-1}L) = L^T C^{-1} C C^{-1} L$$

$$= L^T C^{-1} L \longleftarrow \text{Symmetric sparse matrix!}$$

# Minimizing the Laplacian Energy

- To find the minimum, gradient = 0 and substitute the modeling constraints

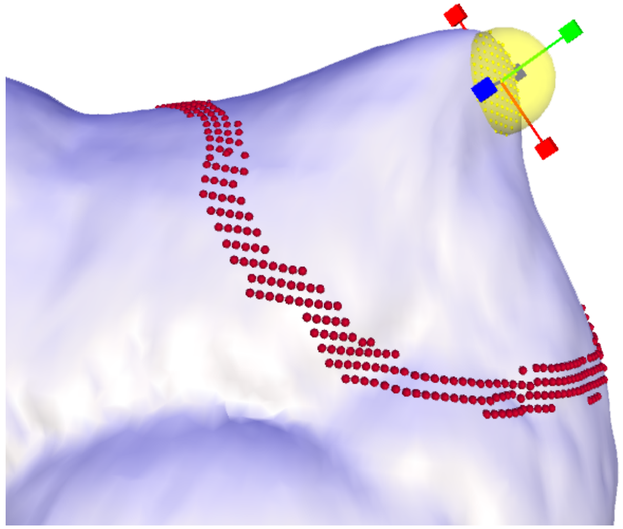
$$E(\mathbf{x}') = (\mathbf{x}')^T \Delta^T C \Delta \mathbf{x}' - 2\mathbf{x}'^T \Delta^T C \delta^{\text{org}} + \text{constant}$$

$$\frac{E(\mathbf{x}')}{\partial \mathbf{x}'} = 2LC^{-1}L\mathbf{x}' - 2L\delta = 0$$

$$\mathbf{x}'_i = \mathbf{c}_i, \;\; i \in \mathcal{C}$$

# Minimizing the Laplacian Energy

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



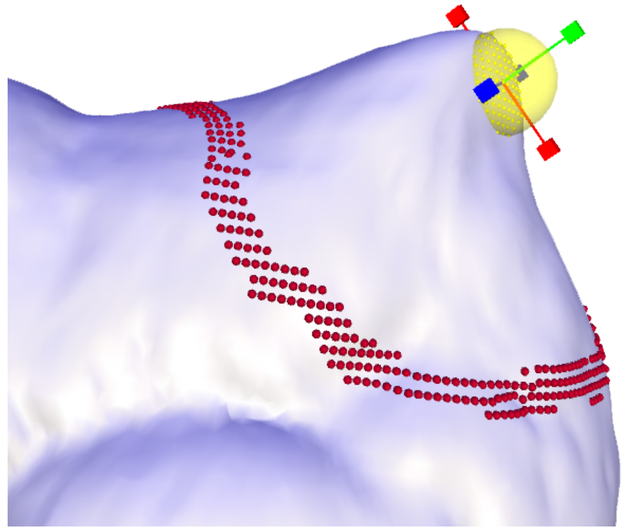
Matrix depends on the initial mesh and the indices of the constraints only.

***Matrix is fixed!***

Right-hand side contains the coordinates of the constraints (handles)

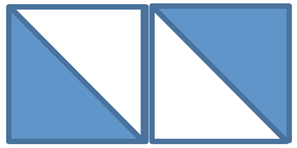
# Minimizing the Laplacian Energy

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



Sparse Cholesky decomposition:

$$A = L_{\text{chol}} L_{\text{chol}}^T$$



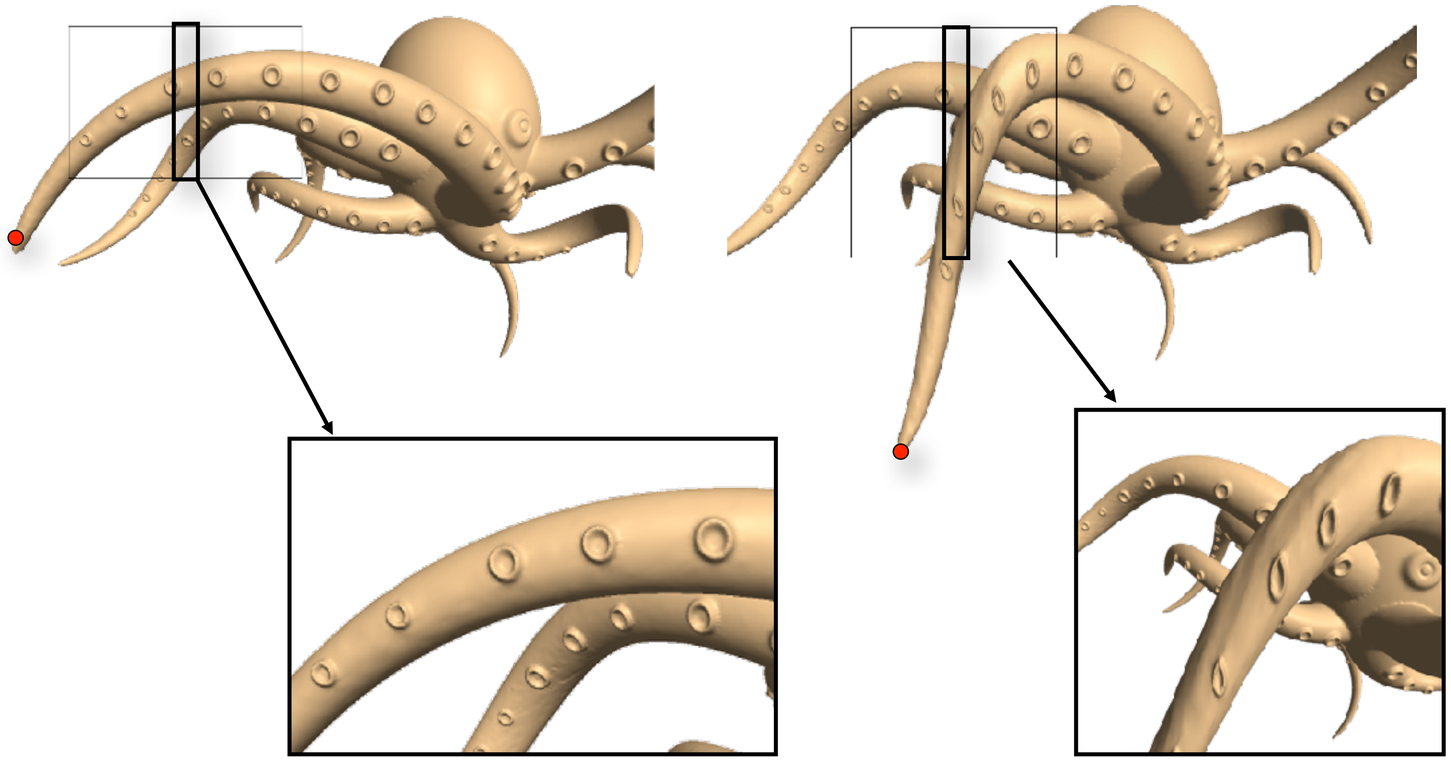
At run-time: just back-substitution!

$$L_{\text{chol}}\mathbf{y} = \mathbf{b}$$

$$L_{\text{chol}}^T\mathbf{x}' = \mathbf{y}$$

# Fundamental Problem: Invariance to Transformations

- The basic Laplacian operator is ***translation***-invariant, but not **rotation**-invariant
- E(**x**') attempts to preserve the original global orientation of the details (the normal directions)

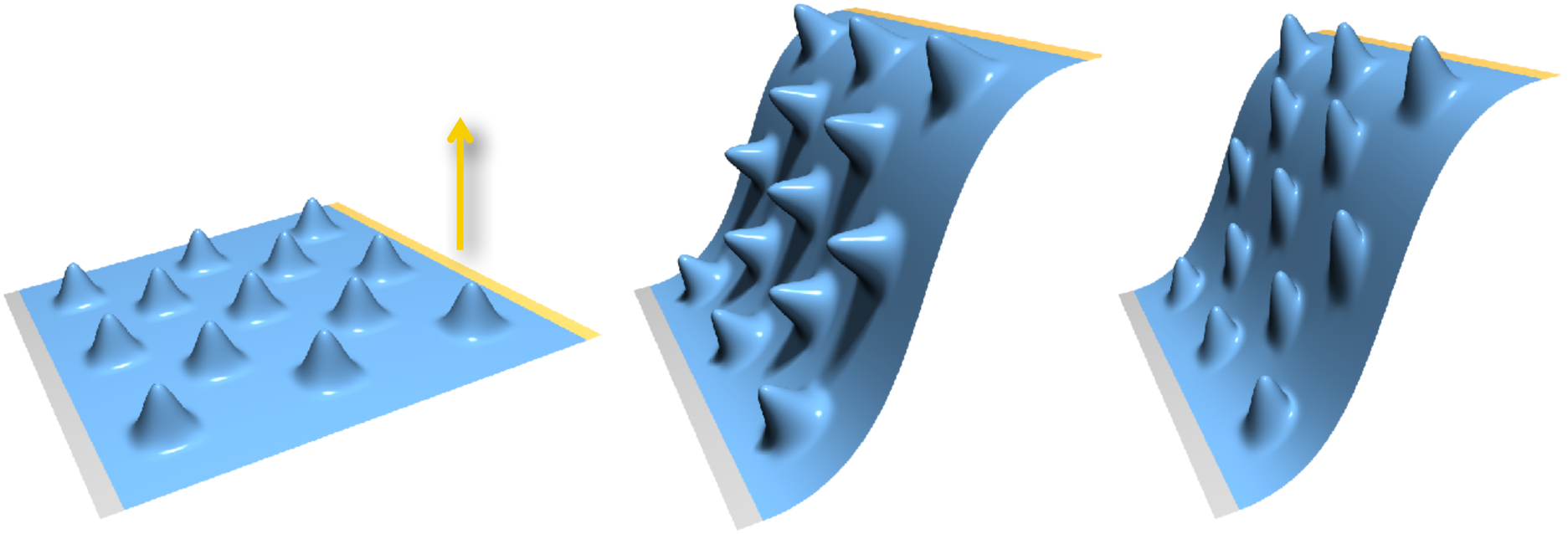


# Fundamental Problem: Invariance to Transformations

input

intuitive expectation

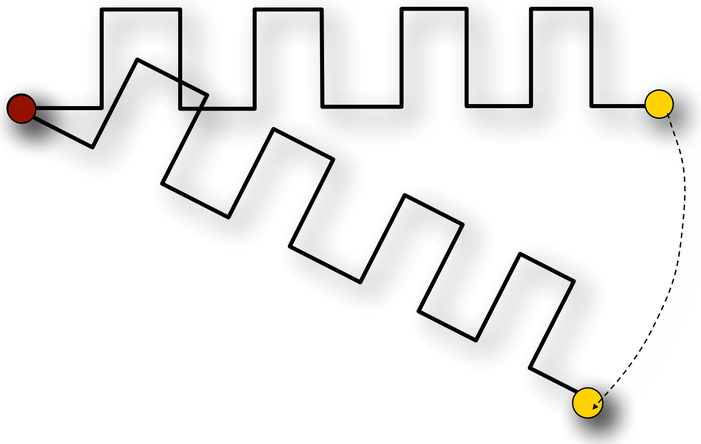
actual result



fixed vertices

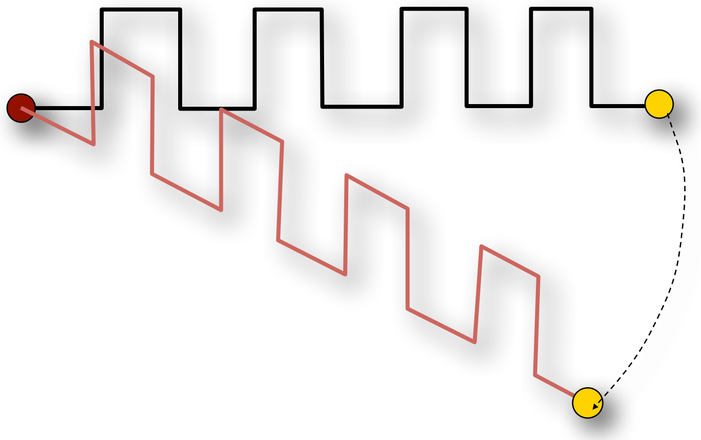
# Fundamental Problem: Invariance to Transformations

- The basic Laplacian operator is ***translation***-invariant, but not **rotation-**invariant
- E(**x**') attempts to preserve the original global orientation of the details (the normal directions)



# Fundamental Problem: Invariance to Transformations

- The basic Laplacian operator is ***translation***-invariant, but not **rotation-**invariant
- E(**x**') attempts to preserve the original global orientation of the details (the normal directions)



# Energy Functional

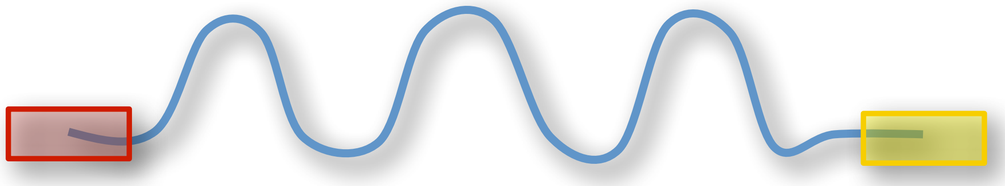
- We need a rigid-invariant energy…

$$E(\mathbf{x}') = \sum_{i=1}^{n} A_i \| \Delta(\mathbf{x}'_i) - \delta_i \|^2$$

Need to locally rotate the target mean curv. normals

# Fixing Local Rotations: Multiresolution Approach

input



$\mathcal{S}$

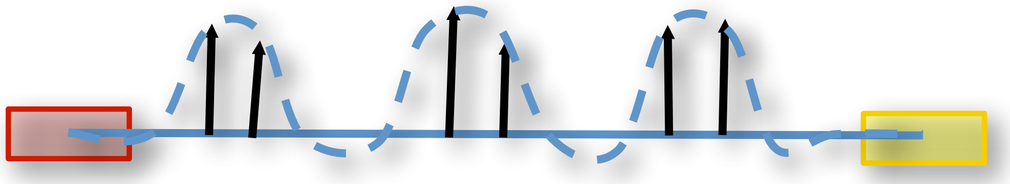
# Fixing Local Rotations: Multiresolution Approach

Smooth base surface

$\mathcal{B}$

# Fixing Local Rotations: Multiresolution Approach

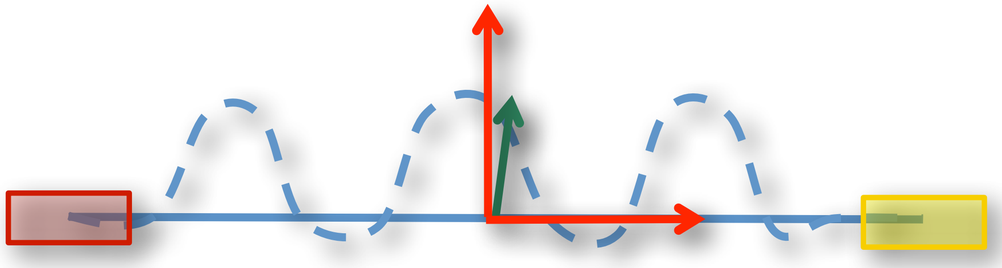
Details – displacement vectors



$$\mathcal{S} - \mathcal{B}$$

# Fixing Local Rotations: Multiresolution Approach

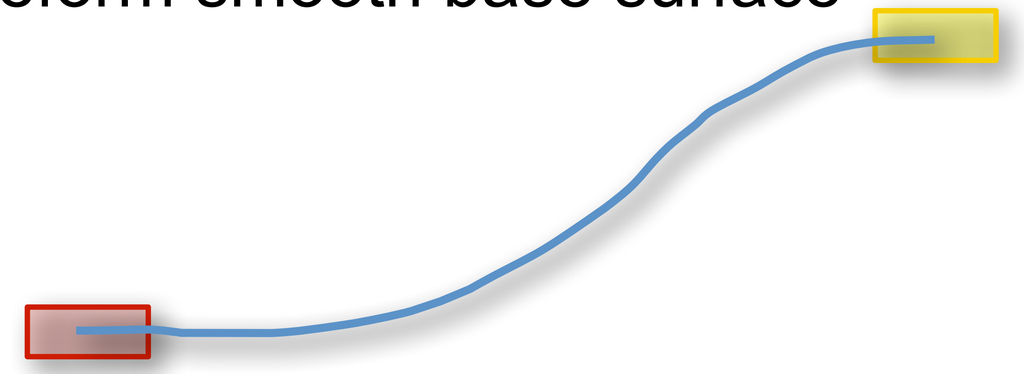
Encode details in the local frame of $B$



$$\mathbf{d}_i = a_1\mathbf{t}_i + a_2\mathbf{n}_i$$

# Fixing Local Rotations: Multiresolution Approach

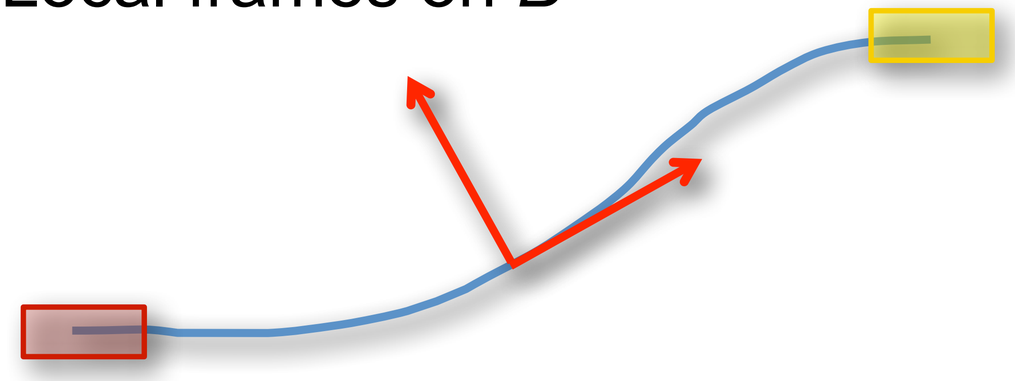
Deform smooth base surface



$\mathcal{B}'$

# Fixing Local Rotations: Multiresolution Approach

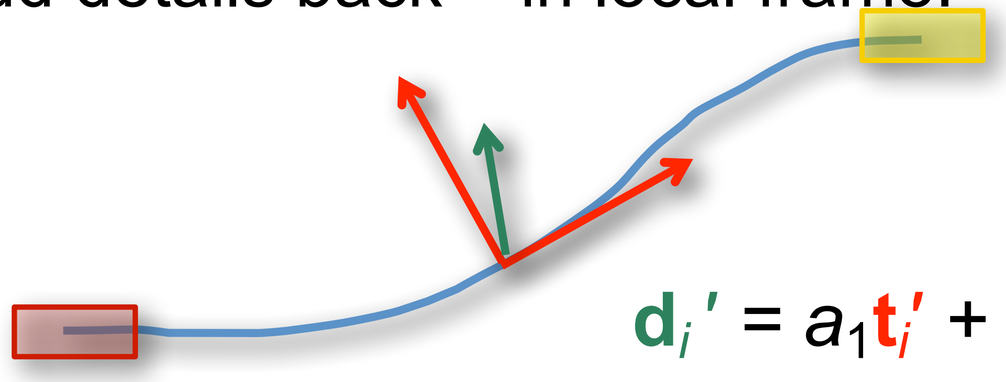
Local frames on $B'$



$\mathcal{B}'$

# Fixing Local Rotations: Multiresolution Approach

Add details back – in local frame!



$$\mathbf{d}_i' = a_1 \mathbf{t}_i' + a_2 \mathbf{n}_i'$$

$\mathcal{B}'$

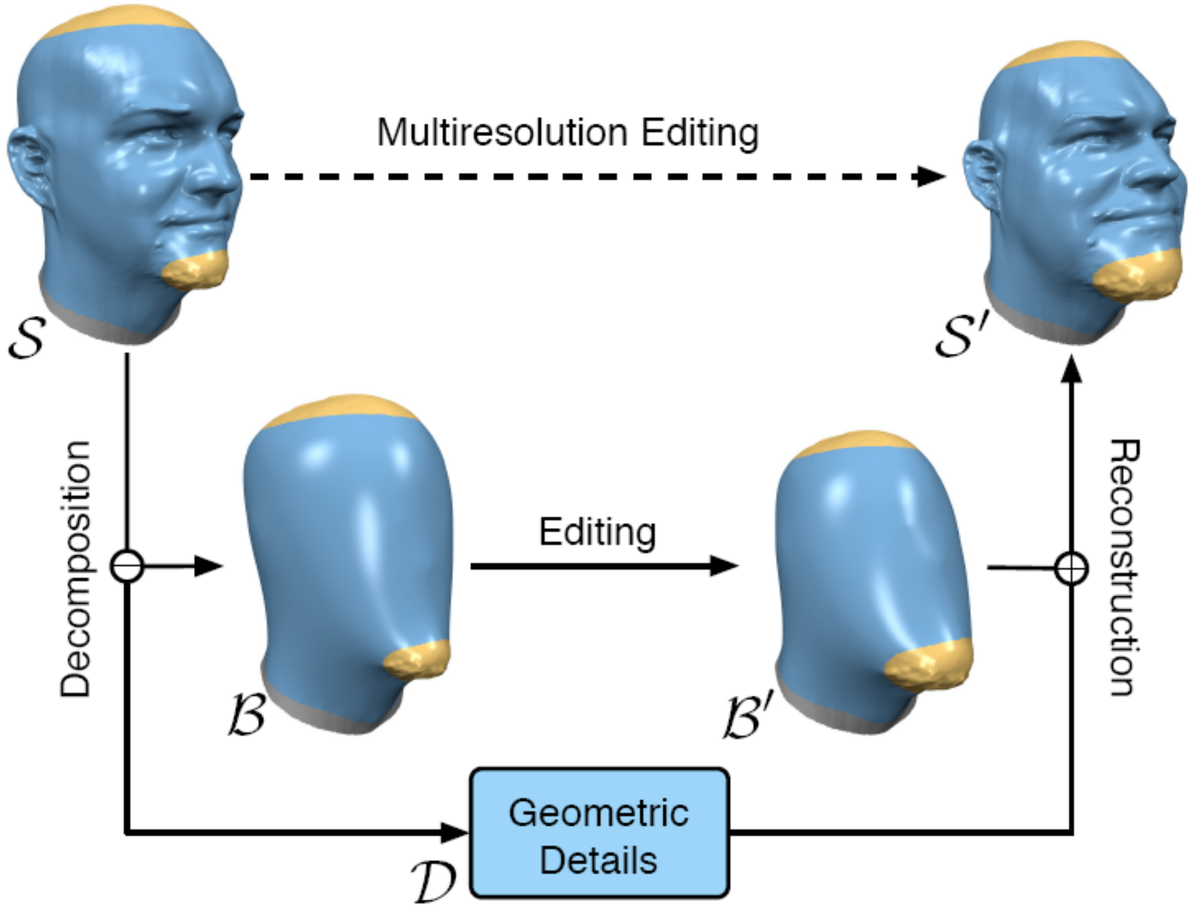
# Fixing Local Rotations: Multiresolution Approach

Displace the vertices to get the result

$\mathcal{S}'$

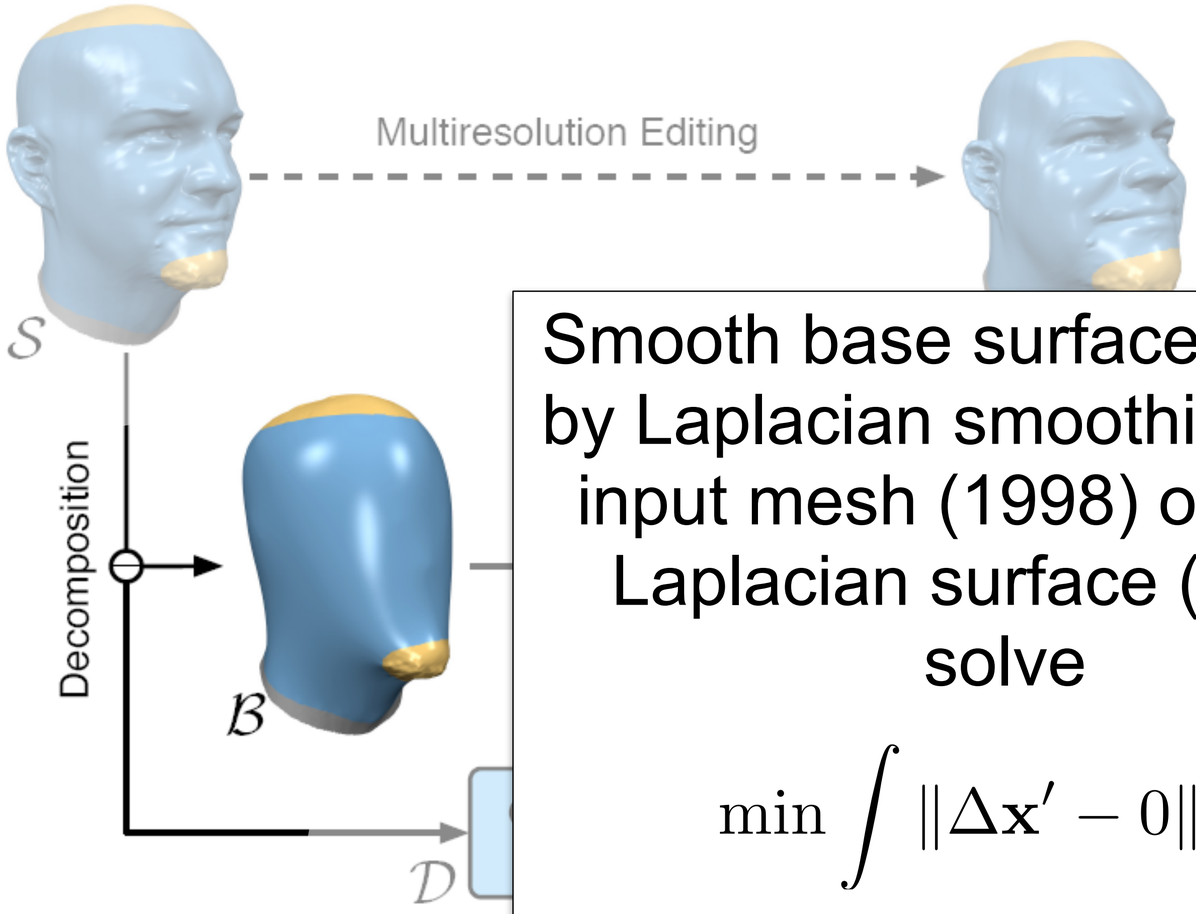
# Fixing Local Rotations: Multiresolution Approach

- Kobbelt et al. SIGGRAPH 98, Botsch and Kobbelt SIGGRAPH 2004



# Fixing Local Rotations: Multiresolution Approach

- Kobbelt et al. SIGGRAPH 98, Botsch and Kobbelt SIGGRAPH 2004



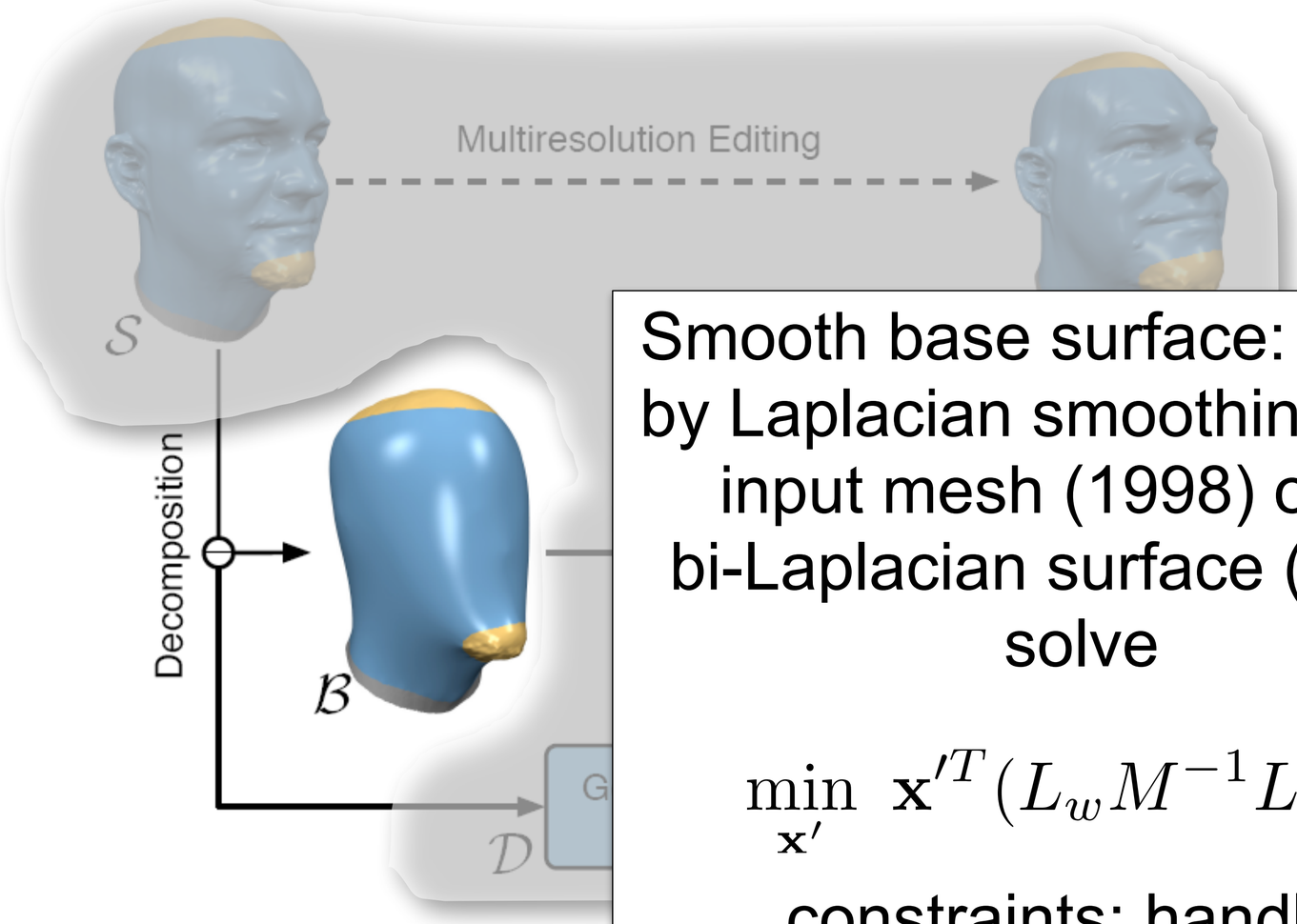

Smooth base surface: defined by Laplacian smoothing of the input mesh (1998) or the bi-Laplacian surface (2004): solve

$$\min \int \|\Delta \mathbf{x}' - 0\|^2 d\mathbf{x}'$$

constraints: handles

# Fixing Local Rotations: Multiresolution Approach

- Kobbelt et al. SIGGRAPH 98, Botsch and Kobbelt SIGGRAPH 2004



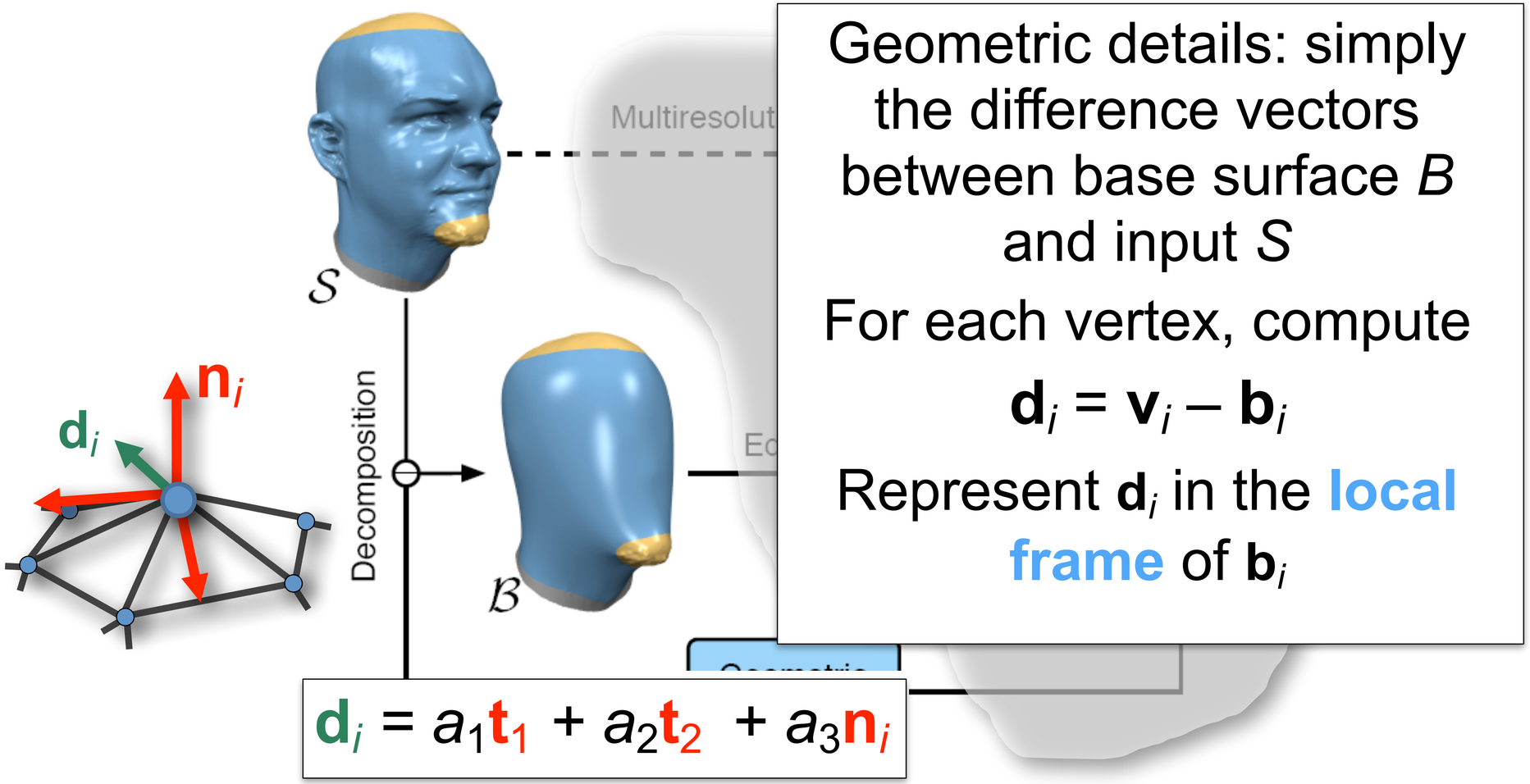

Smooth base surface: defined by Laplacian smoothing of the input mesh (1998) or the bi-Laplacian surface (2004): solve

$$\min_{\mathbf{x}'} \ \mathbf{x}'^T (L_w M^{-1} L_w)\, \mathbf{x}'$$

constraints: handles

# Fixing Local Rotations: Multiresolution Approach

- Kobbelt et al. SIGGRAPH 98, Botsch and Kobbelt SIGGRAPH 2004




Geometric details: simply the difference vectors between base surface *B* and input *S*

For each vertex, compute

$\mathbf{d}_i = \mathbf{v}_i - \mathbf{b}_i$

Represent $\mathbf{d}_i$ in the **local frame** of $\mathbf{b}_i$

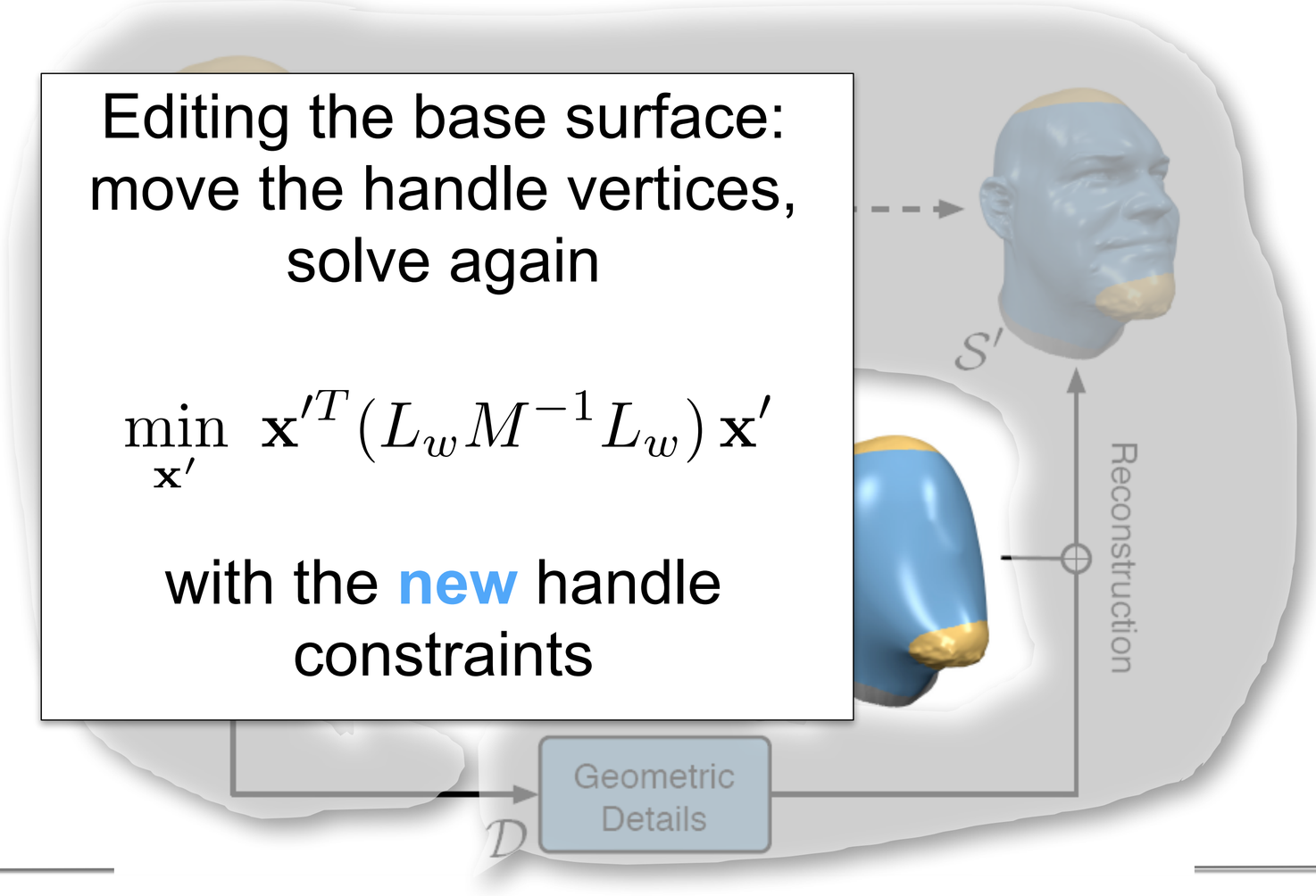
# Fixing Local Rotations: Multiresolution Approach

- Kobbelt et al. SIGGRAPH 98, Botsch and Kobbelt SIGGRAPH 2004

Editing the base surface: move the handle vertices, solve again

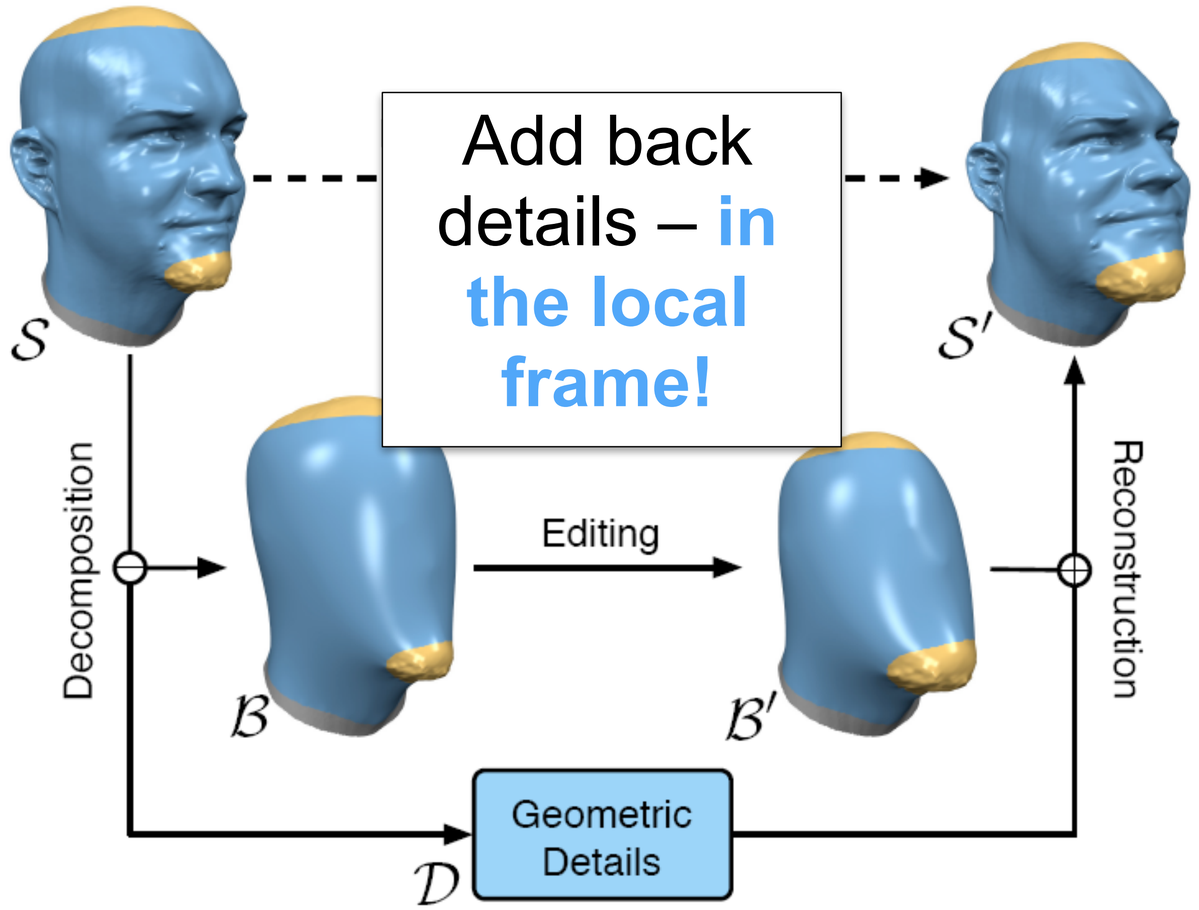
$$\min_{\mathbf{x}'} \; \mathbf{x}'^{T} \left( L_w M^{-1} L_w \right) \mathbf{x}'$$

with the **new** handle constraints



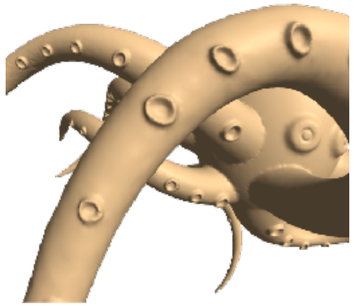
# Fixing Local Rotations: Multiresolution Approach

- Kobbelt et al. SIGGRAPH 98, Botsch and Kobbelt SIGGRAPH 2004

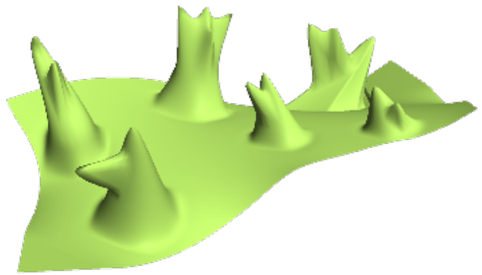


# Discussion: Multiresolution Framework

- Advantages:
  - Fast! Linear solve for the base surface deformation, and then add back displacements
  - Intuitive, easy to implement
- Problem: works only for small height fields (when details vectors are small)



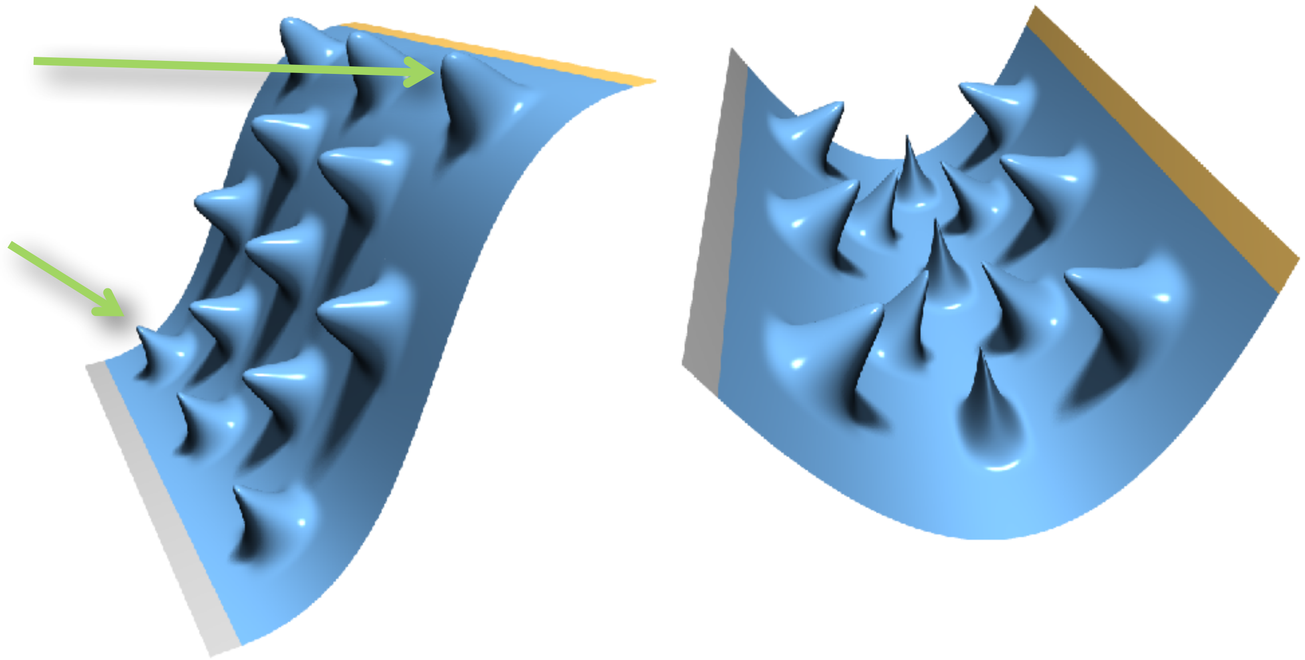
almost a height field



not a height field

# Multiresolution Framework: Discussion

- Problem: If detail vectors are too big we get overshooting and **self-intersections**, especially in concave cases



# Shell-based Deformation

- Stretching + bending

$$E(S') = \int \int_{\Omega} k_s \|\mathbf{I}'(u,v) - \mathbf{I}(u,v)\|_F^2 + k_b \|\mathbf{II}'(u,v) - \mathbf{II}(u,v)\|_F^2 dudv$$

$$E(\mathbf{d}) = \int \int_{\Omega} k_s \left( \|\mathbf{d}_u(u,v)\|^2 - \|\mathbf{d}_v(u,v)\|^2 \|_F^2 \right) + k_b \left( \|\mathbf{d}_{uu}(u,v)\|^2 + 2\|\mathbf{d}_{uv}(u,v)\|^2 + \|\mathbf{d}_{vv}(u,v)\|^2 \right) dudv$$

$$-k_s \Delta \mathbf{d} + k_b \Delta^2 \mathbf{d} = 0$$

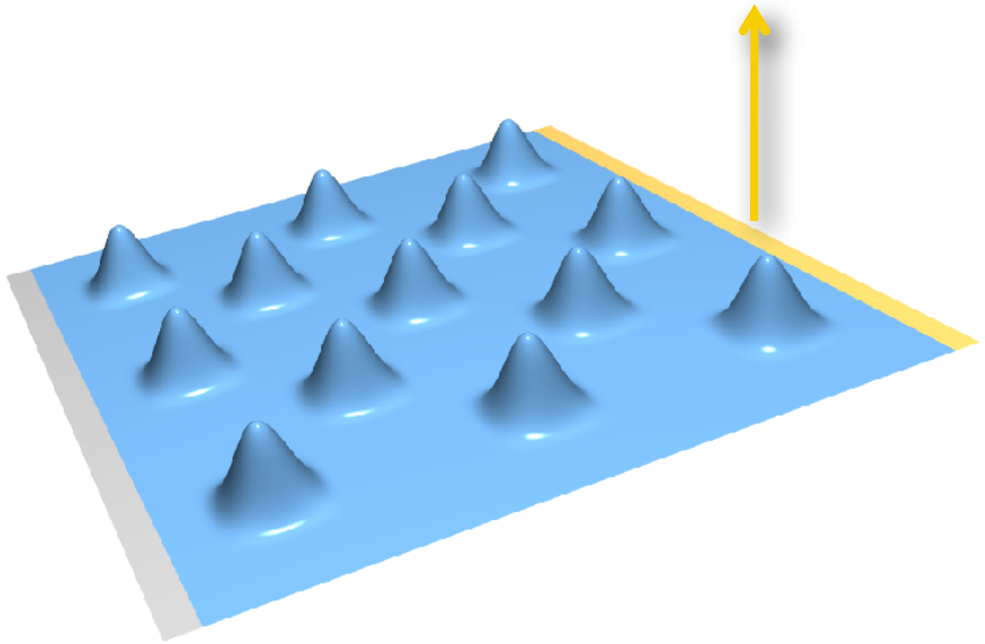
# Gradient-based Deformation

$$E(f) = \int \int_{\Omega} \|\nabla f(u,v) - \mathbf{g}(u,v)\|^2 du dv$$

$$\Delta f = \text{div } \mathbf{g}$$

# Local Rotations – Single Resolution Solutions

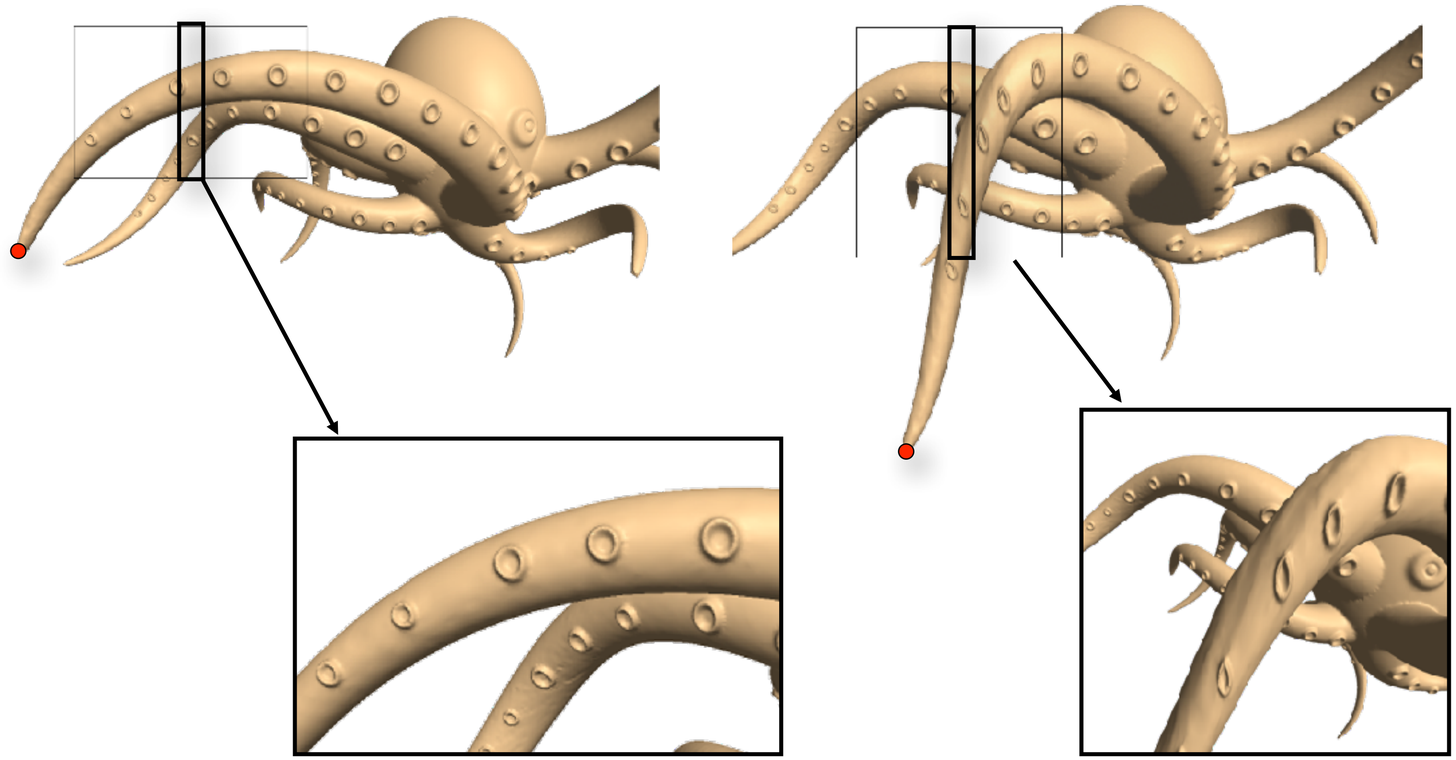
- Come up with a rotation field on the surface based on the modeling constraints
- Rotate the differential coordinates; solve



# Estimation of Rotations

**Lipman et al. 2004**

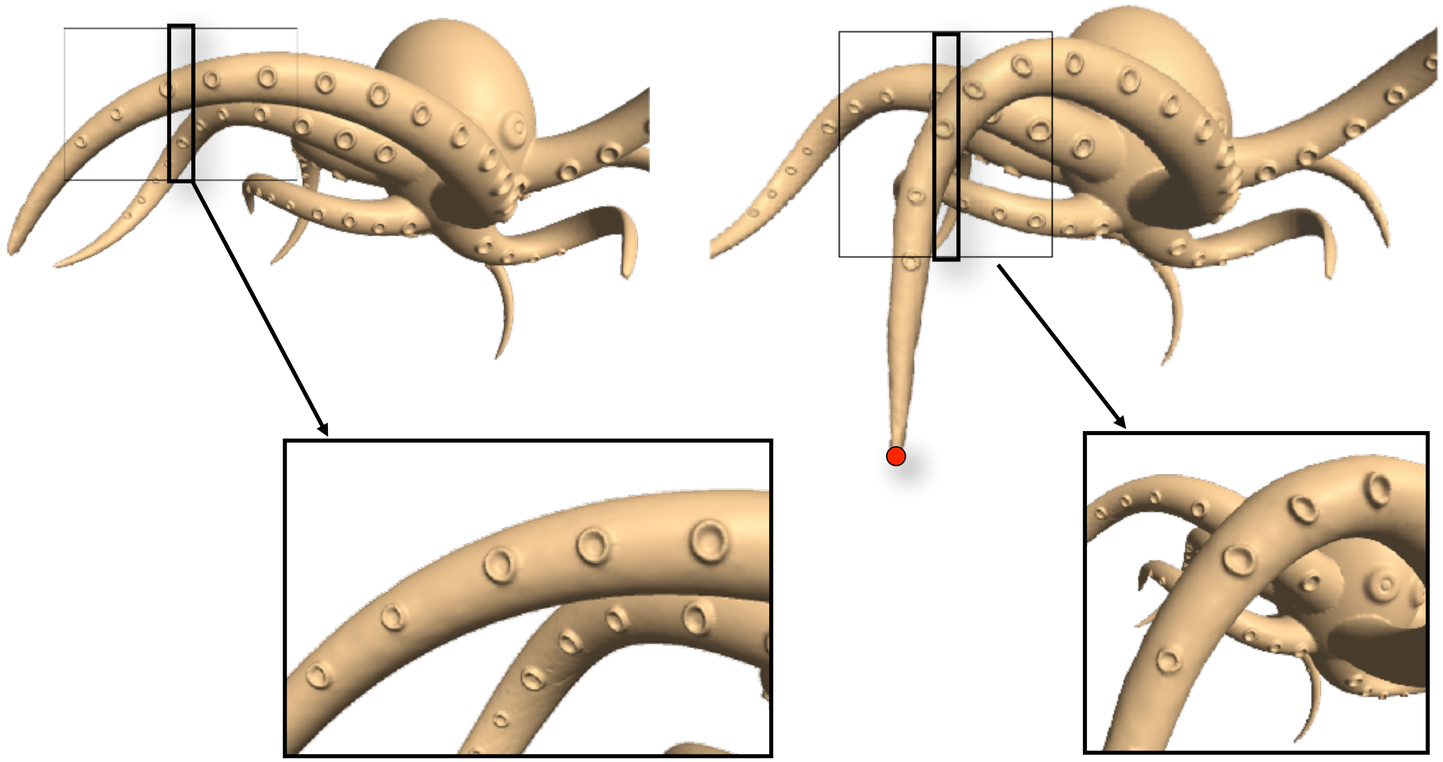
- Edit the surface using the original Laplacians δ (naïve Laplacian editing)
- Compute smoothed local frames, estimate rotation



# Estimation of Rotations

**Lipman et al. 2004**

- Then solve the optimization again with the rotated δ's!



# Estimation of Rotations

**Lipman et al. 2004**

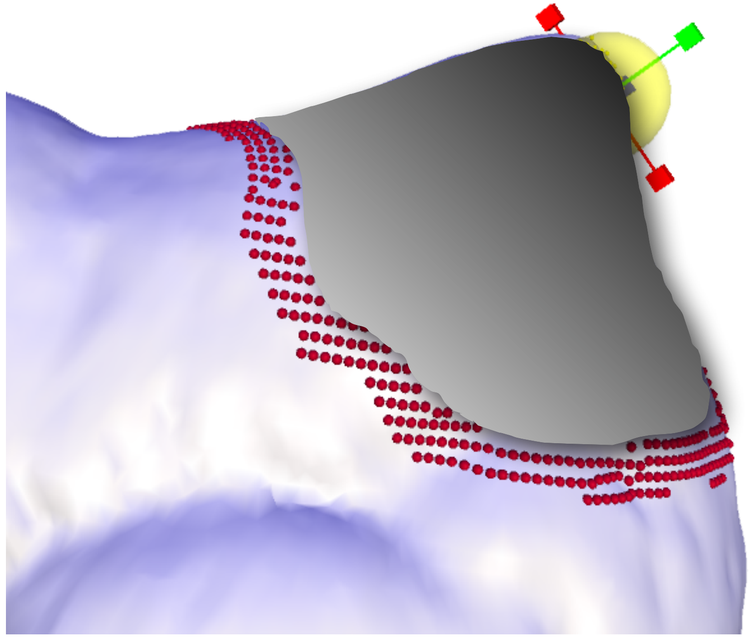
- Advantages:
  - Sparse linear solve
  - Less or no self-intersections thanks to global optimization (no more conflicting local displacements)

- Disadvantages:
  - Heuristic estimation of the rotations
  - Speed depends on the support of the smooth local frame estimation operator; for highly detailed surfaces it must be large
  - Unclear how much we need to smooth (what is the scale?)

# Rotation Propagation

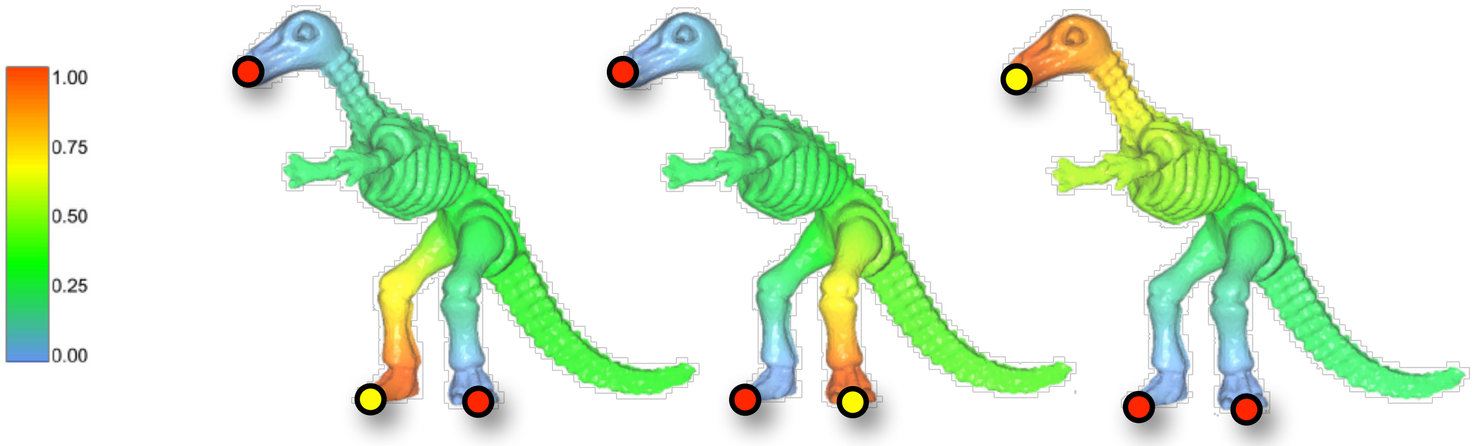
**[Yu et al. SIGGRAPH 2004][Zayer et al. EG 2005][Lipman et al. SIGGRAPH 2005]**

- Assume more user input: the user also specifies handle rotation, not just translation
- The rotation is diffused to the rest of the ROI



# Rotation Propagation

- Geodesic distance [Yu et al. 2004]
- Harmonic field [Zayer et al. 2005]
- Optimization [Lipman et al. 2005, 2006]



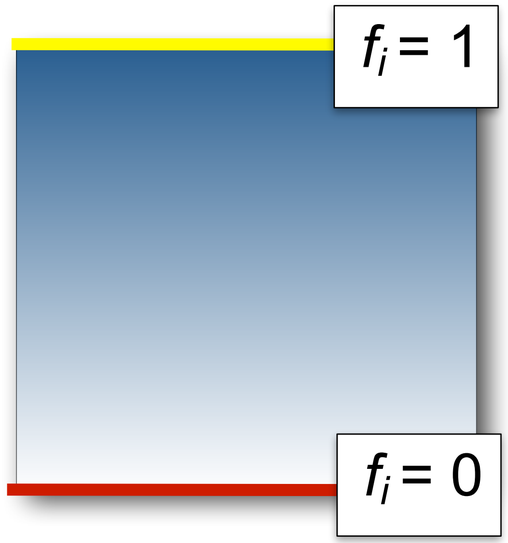

Harmonic field

# Harmonic Fields on Meshes

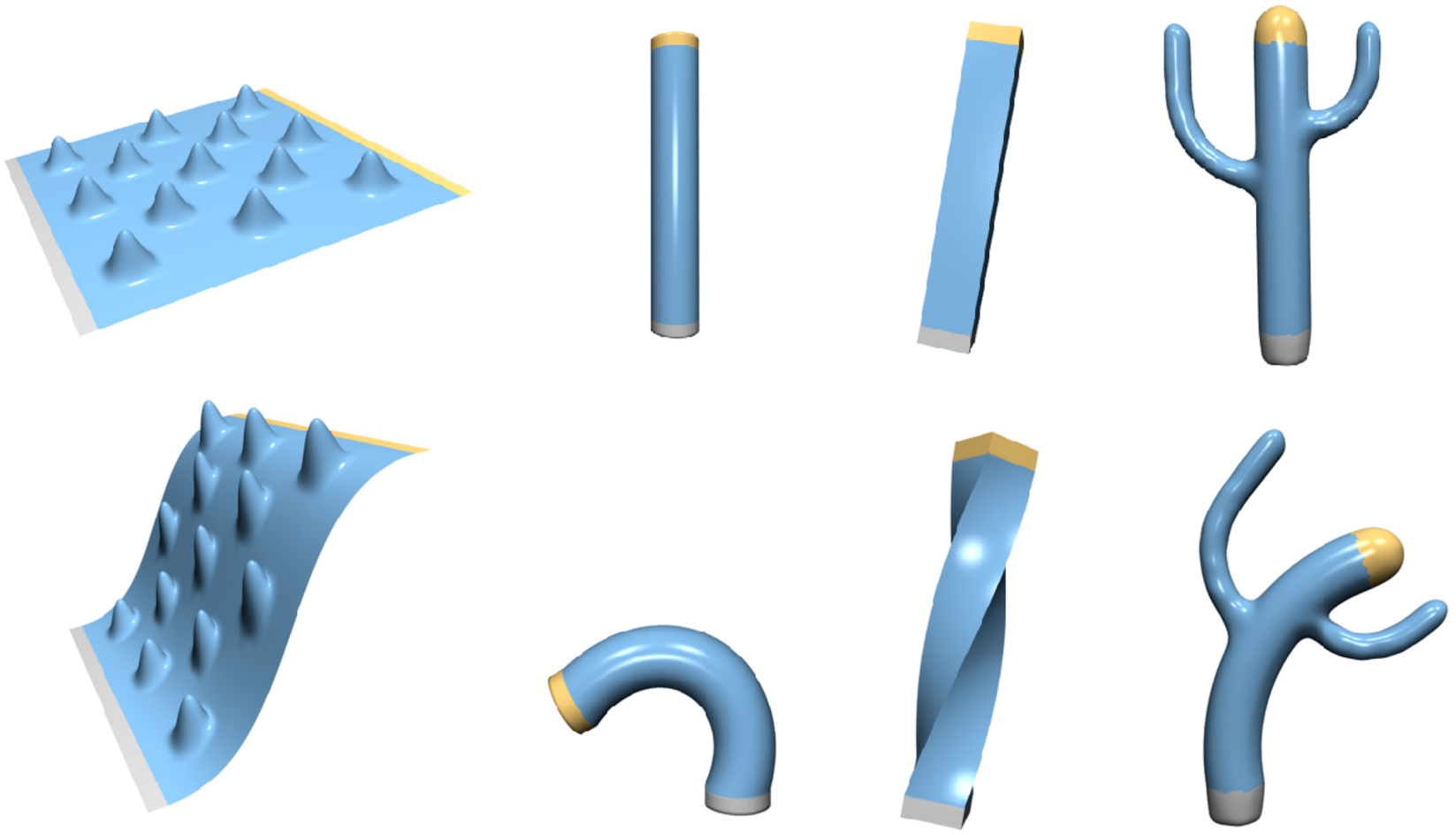
- Scalar function, attains 1 on the active handle, 0 on the static handles
- Smooth in-between, no local extrema
- Solve: $\Delta \mathbf{f} = 0$ + boundary conditions

  with constraints $f_i = 1$ on active handle,
  $f_i = 0$ on static handle

Example: in this simple case,
the harmonic field is a just a linear ramp



# Rotation Propagation w/Harmonic Fields



Why does this happen?

# Rotation Propagation w/Harmonic Fields

- If rotations are provided and consistent with the desired transformation, this works well
- However, the method is translation-insensitive
  (doesn't generate rotations when there are none provided)

# List of Literature


- Kobbelt et al.: **Interactive Multi-Resolution Modeling on Arbitrary Meshes,** ACM SIGGRAPH 1998
- Botsch and Kobbelt: **An Intuitive Framework for Real-Time Freeform Modeling,** ACM SIGGRAPH 2004
- Lipman et al.: **Differential Coordinates for Interactive Mesh Editing,**
  SMI 2004
- Sorkine et al.: **Laplacian Surface Editing**,
  SGP 2004
- Yu et al.: **Mesh Editing with Poisson-Based Gradient Field Manipulation,**
  ACM SIGGRAPH 2004
- Lipman et al.: **Linear Rotation-Invariant Coordinates for Meshes,**
  ACM SIGGRAPH 2005
- Igarashi et al.: **As-Rigid-As-Possible Shape Manipulation,**
  ACM SIGGRAPH 2005
- Kilian, Mitra, Pottmann: **Geometric Modeling in Shape Space**,
  ACM SIGGRAPH 2007