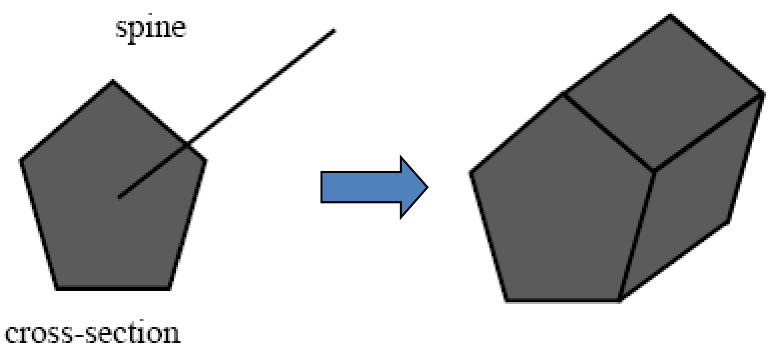
Graphics&Geometric Computing Lab @USTC

Mesh Editing

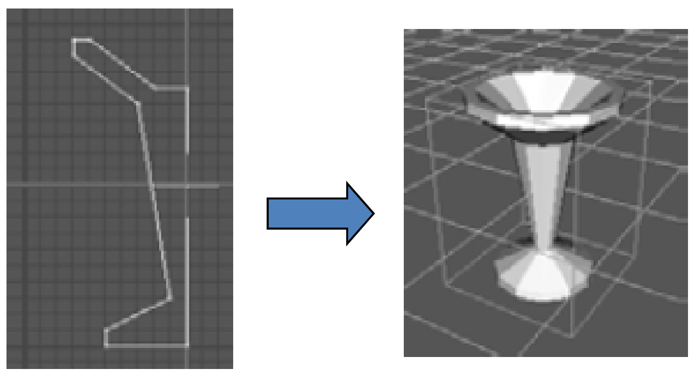
Ligang Liu
Graphics&Geometric Computing Lab
USTC
http://staff.ustc.edu.cn/~lgliu

Building Shapes – 1

- From zero
  - Create a shape by extrusion or revolution



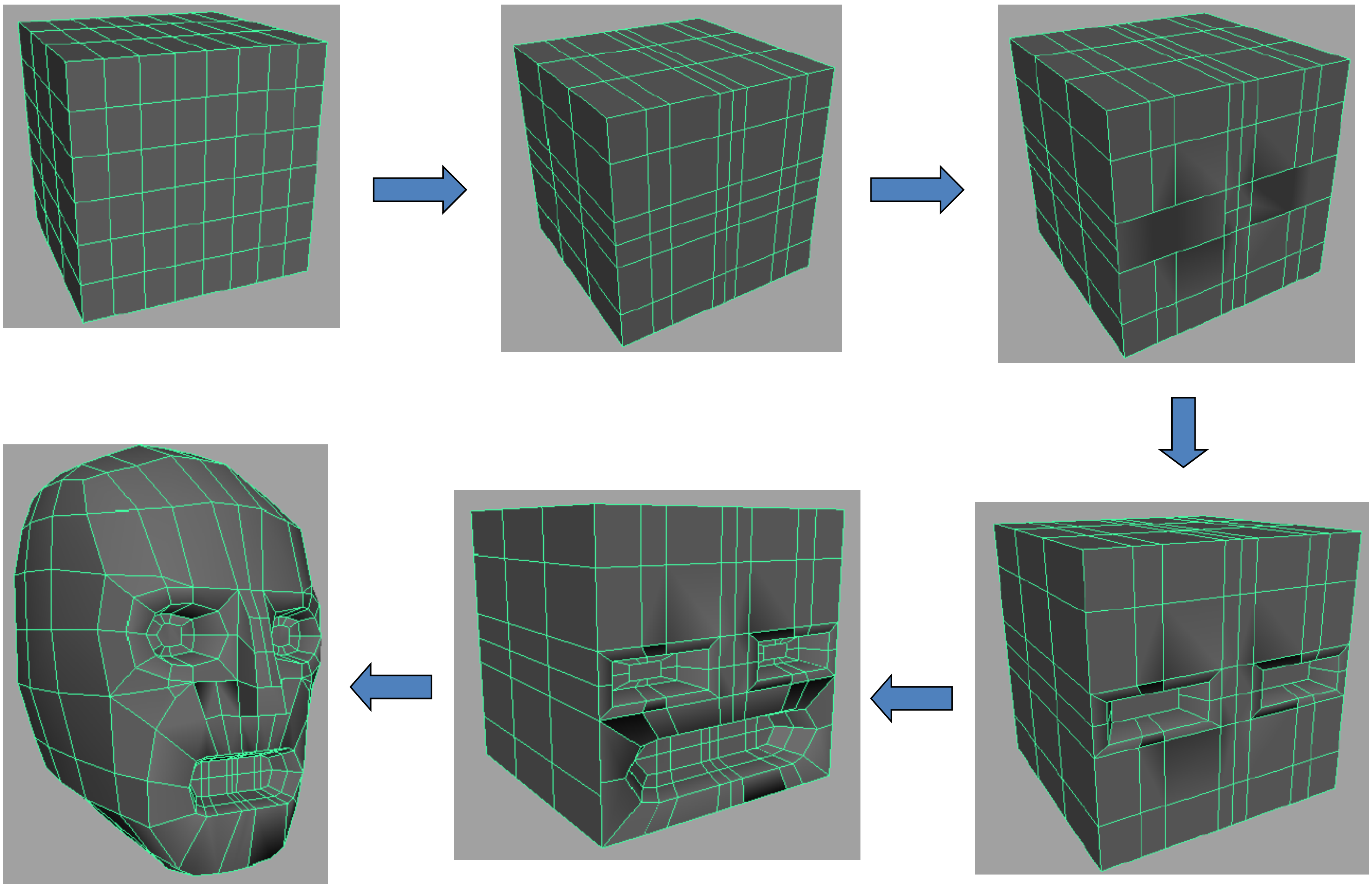
Extrusion (sweeping)



Revolution

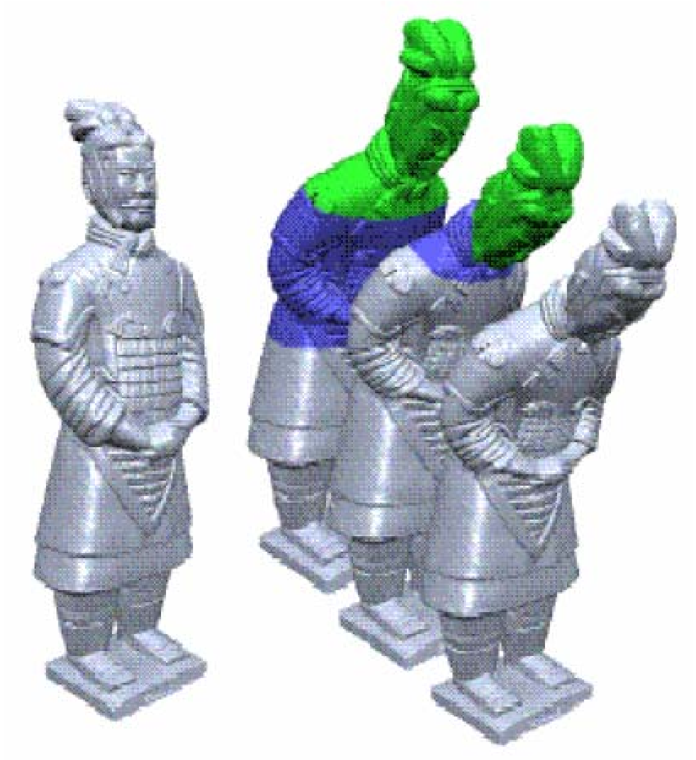
Building Shapes – 2

- Selecting a base shape
  - Create a shape by extrusion or revolution
- Editing mesh
  - Selecting and editing vertices
  - Displacing areas of the mesh



Mesh Deformation

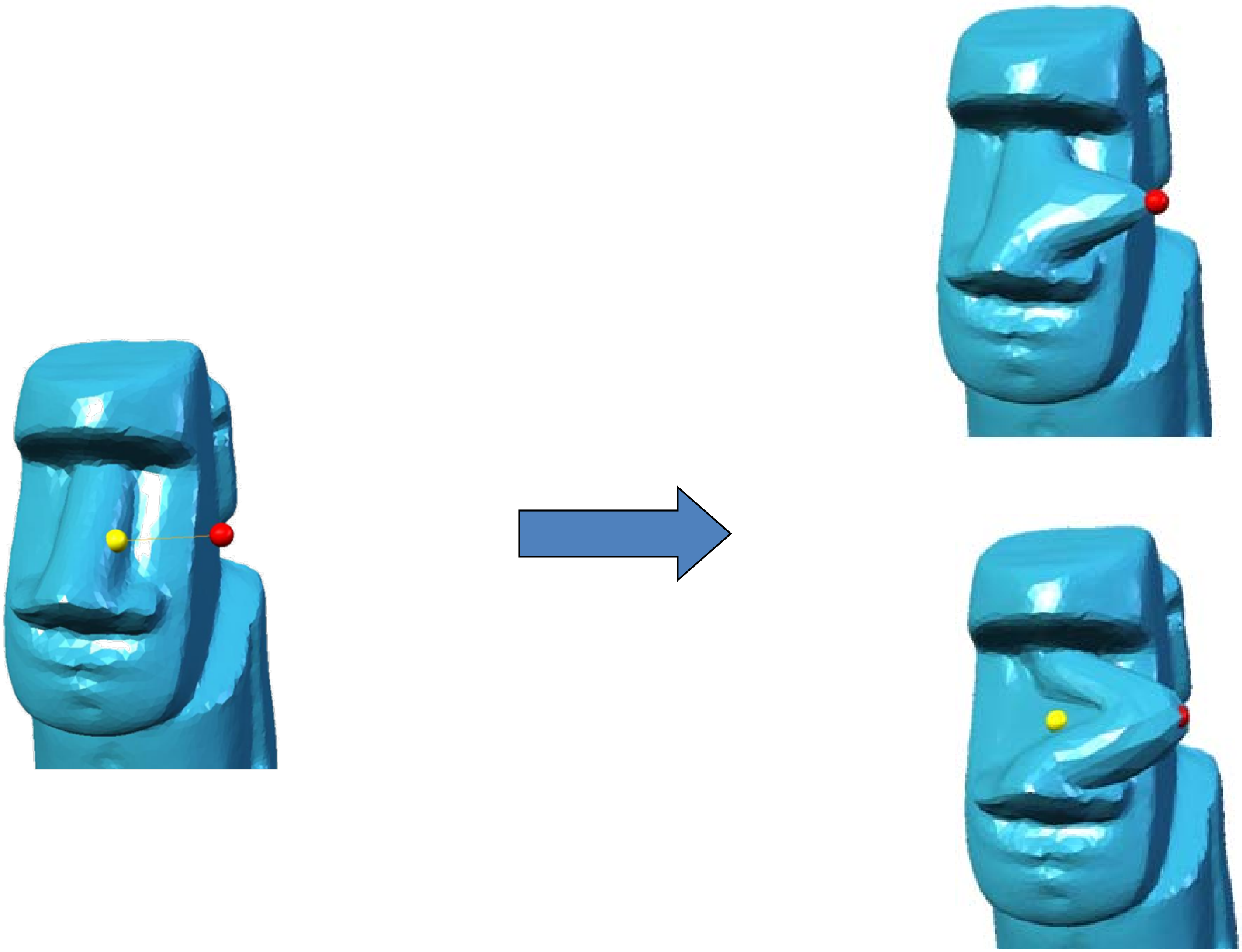
- Mesh Editing
  - Animation
  - Modeling
    - Modeling new models via editing given models



Methods

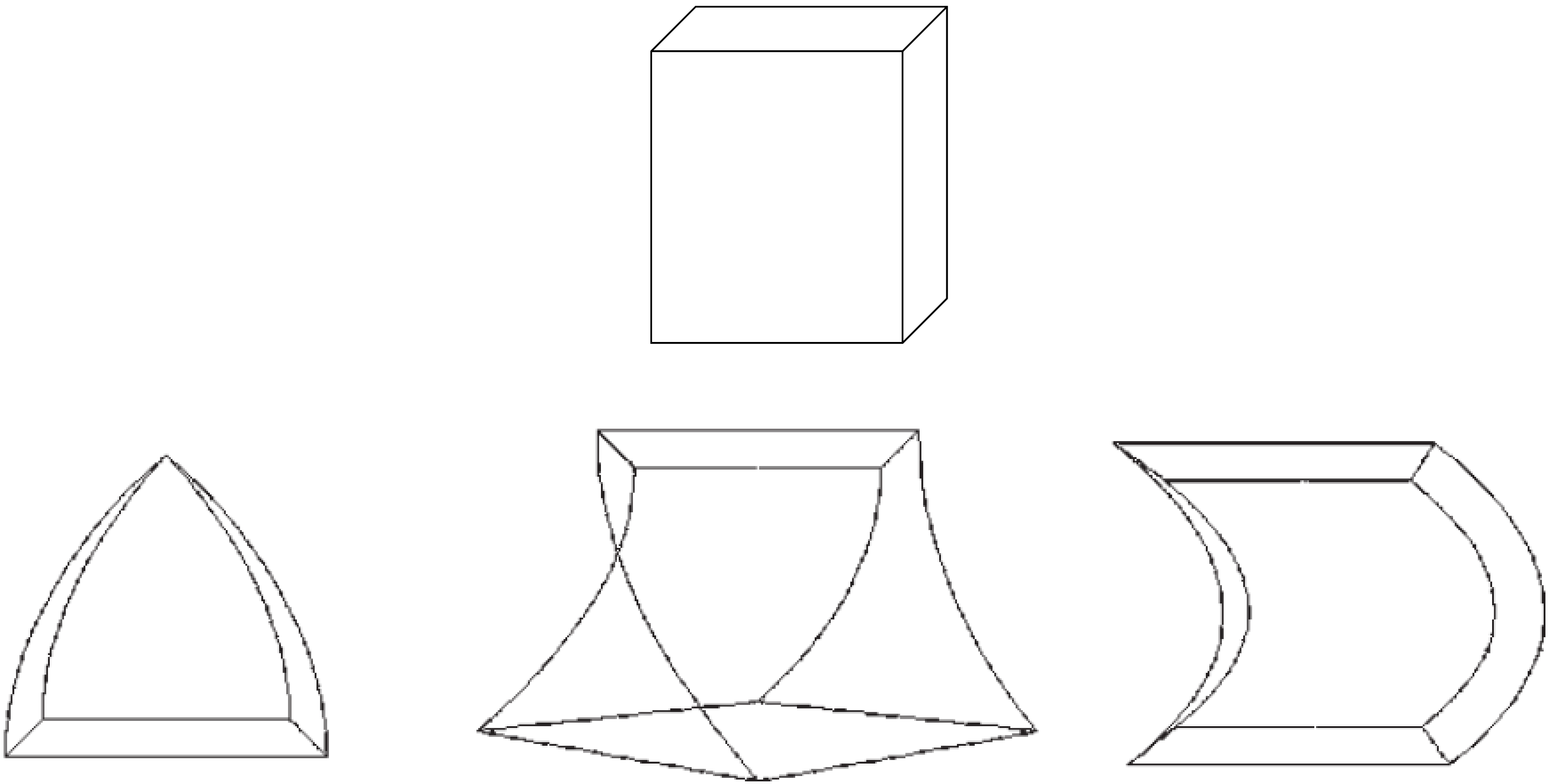
- Deform the vertex coordinates directly
  - Drag vertices
- Deform the control points
  - Bézier, NURBS
- Deform the embedded space
  - FFD, Axial deformation

Vertex Dragging



Global Deformation

[Barr 84]



taper twist bend

Free-form Deformation (FFD)

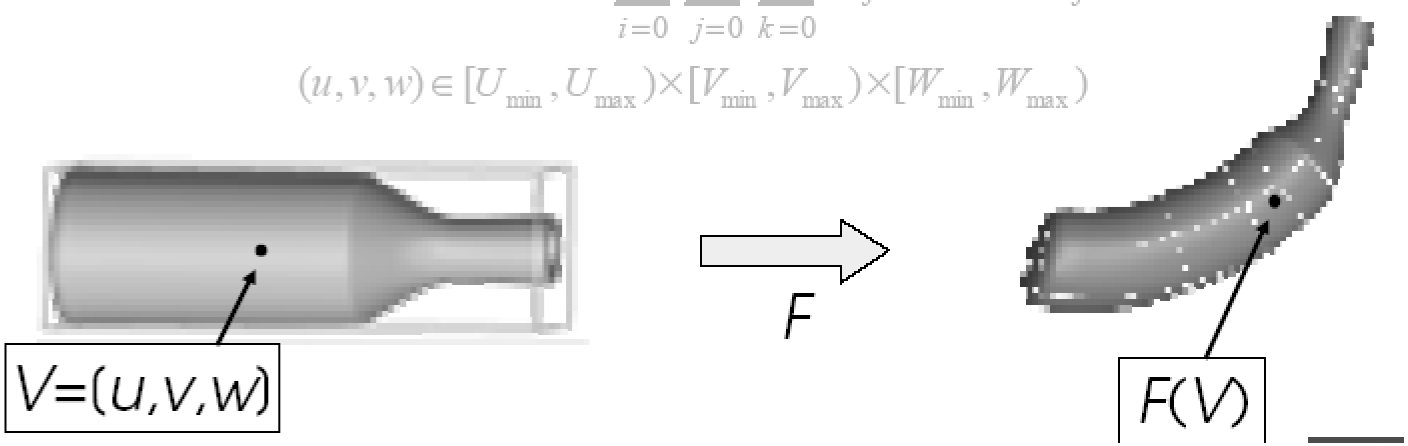
Free-form Deformation (FFD)

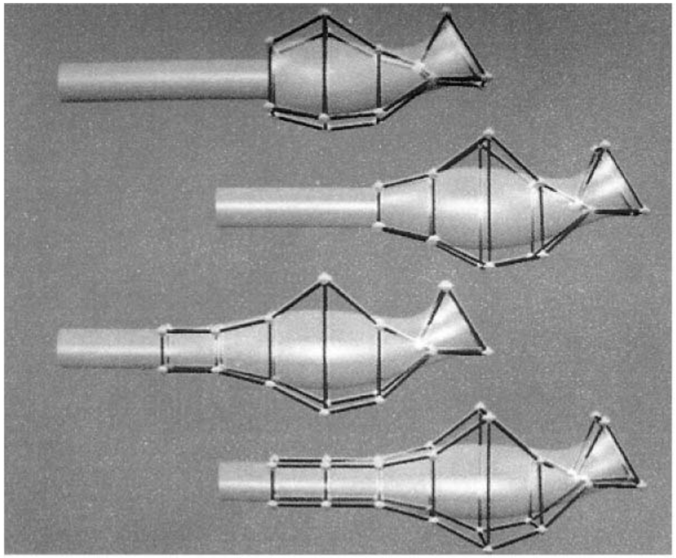
[Sederberg et al. 86]

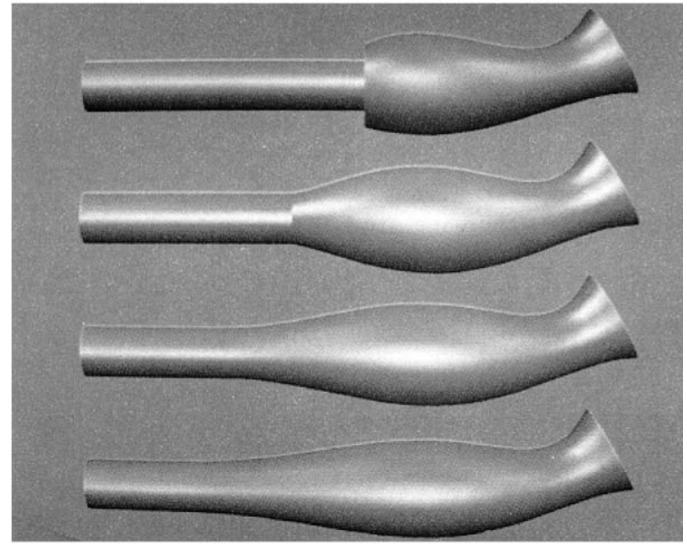
- Embed the object into a domain that is more easily parametrized than the object.
- Advantages:
  - You can deform arbitrary objects
  - Independent of object representation

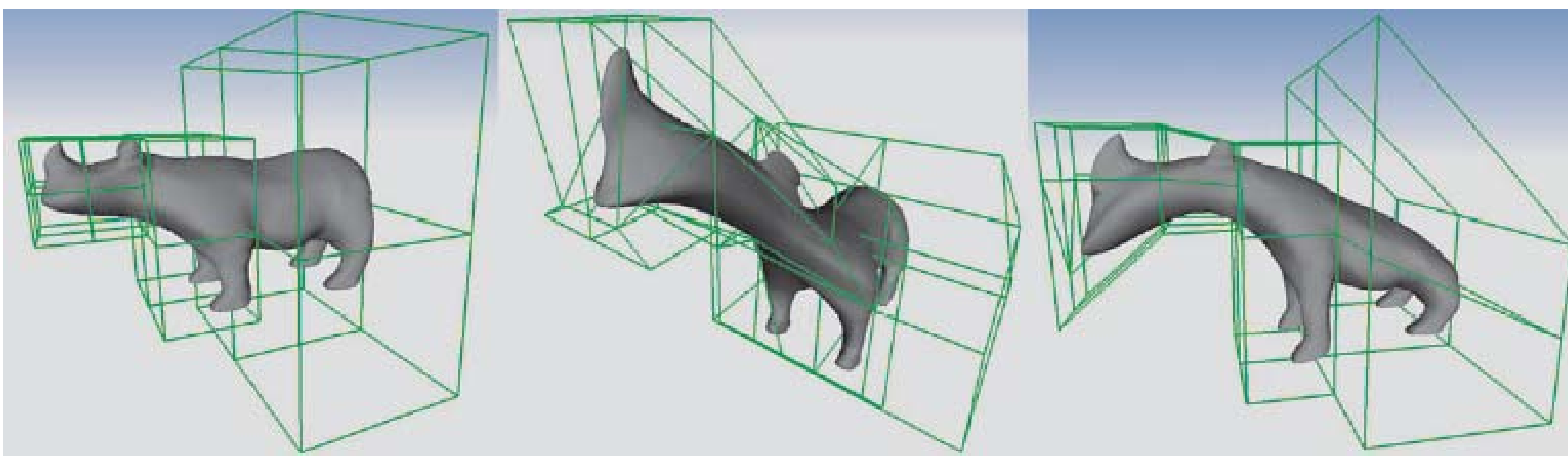
$$X_{new} = F(u,v,w) = \sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n} P_{ijk} B_i(u) B_j(v) B_k(w)$$

$$(u,v,w) \in [U_{\min}, U_{\max}) \times [V_{\min}, V_{\max}) \times [W_{\min}, W_{\max})$$



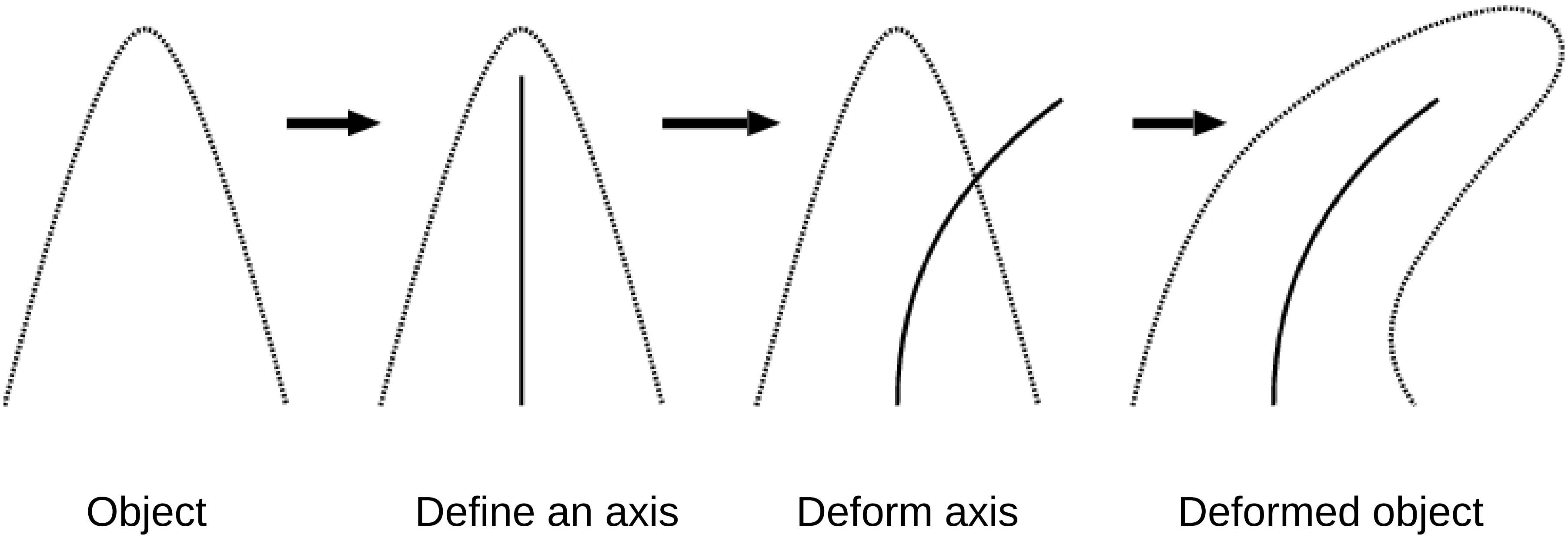




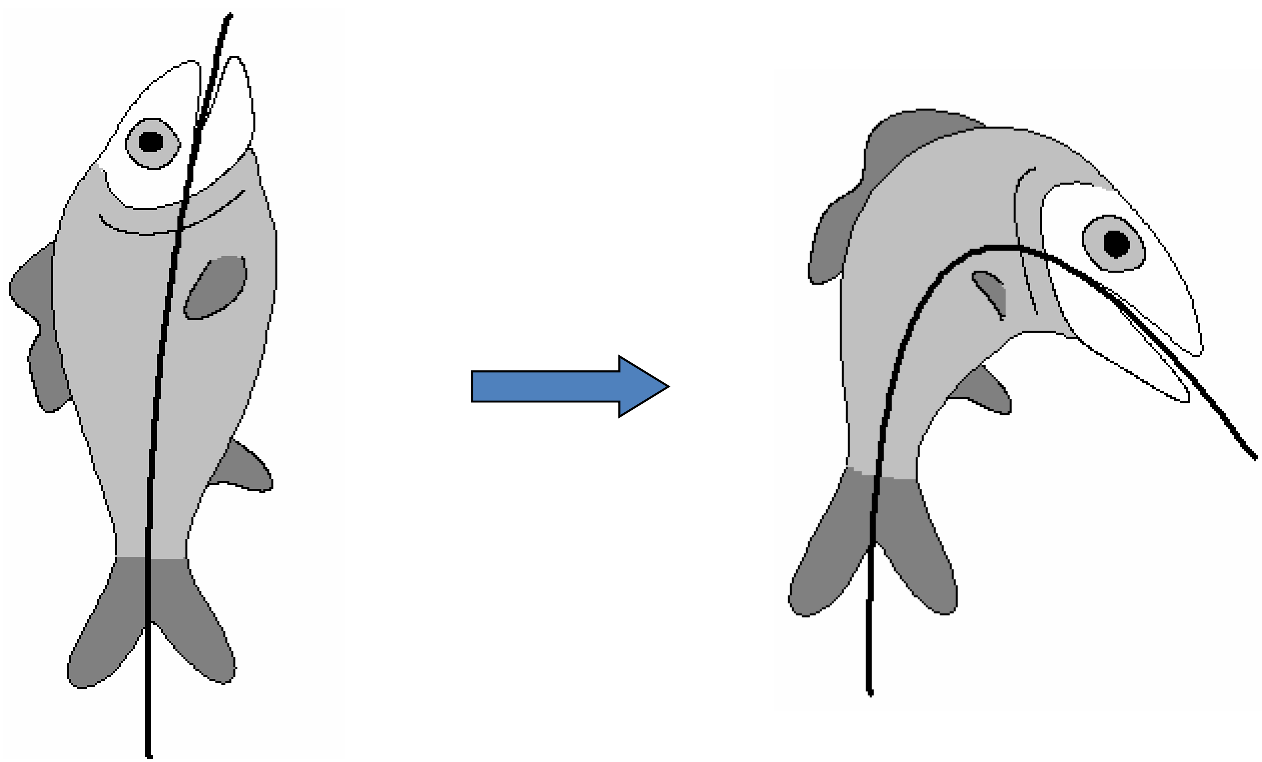


Axial Deformation (AxDf)

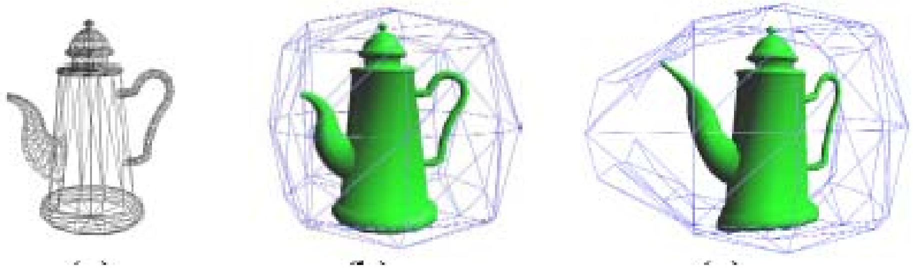
[1994]

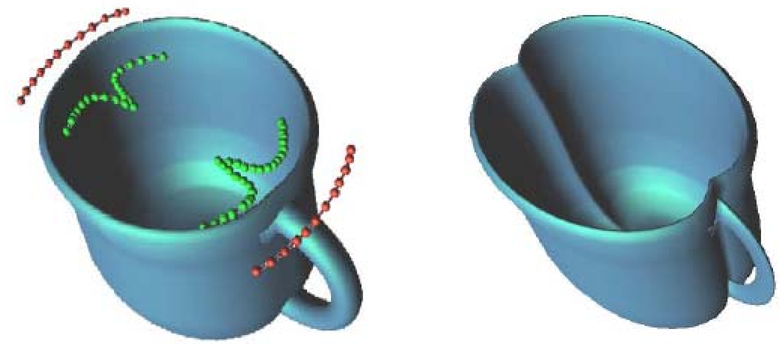


AxDf



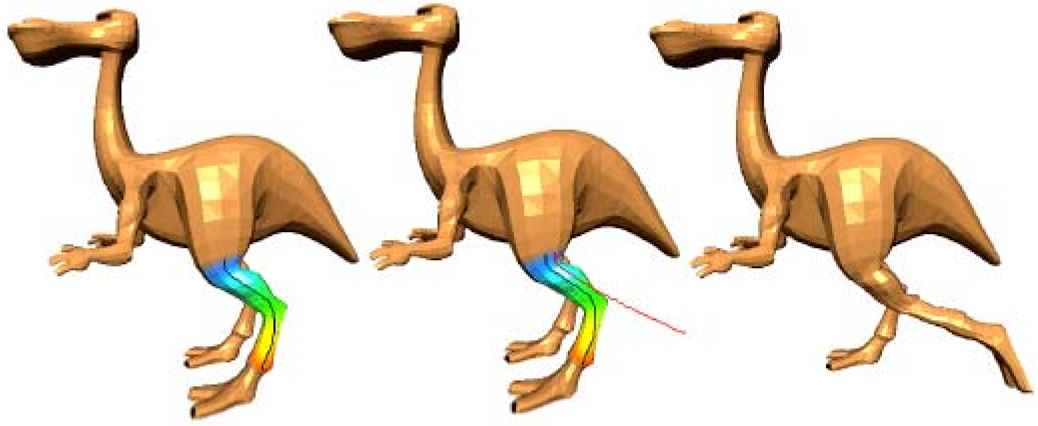
FFD via Sketching

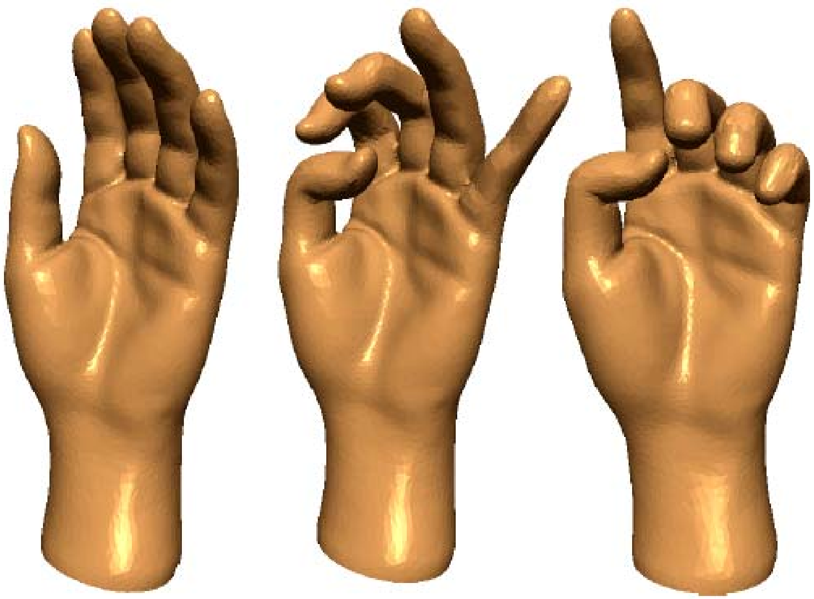


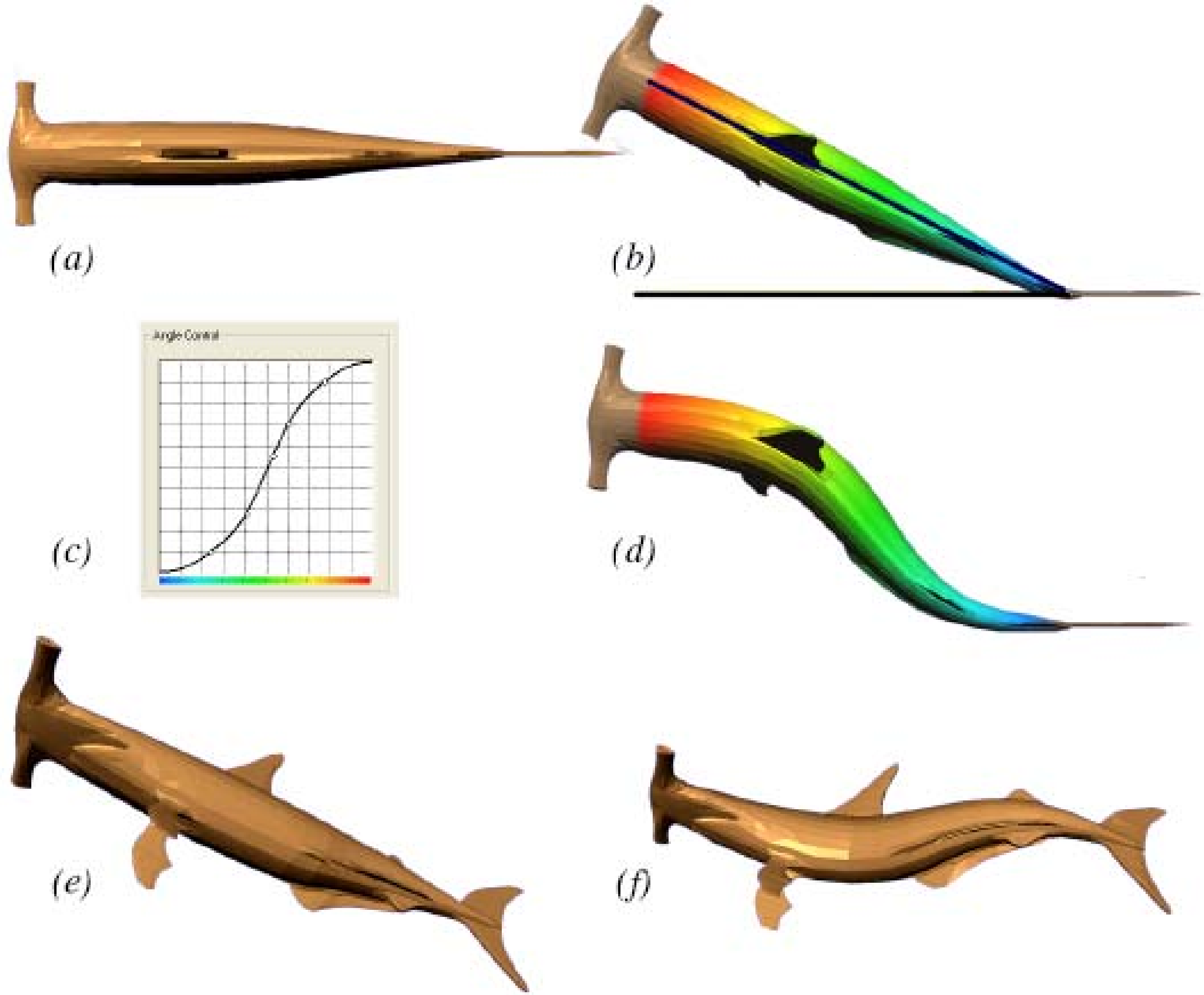


Sketching Deformations

[2005]

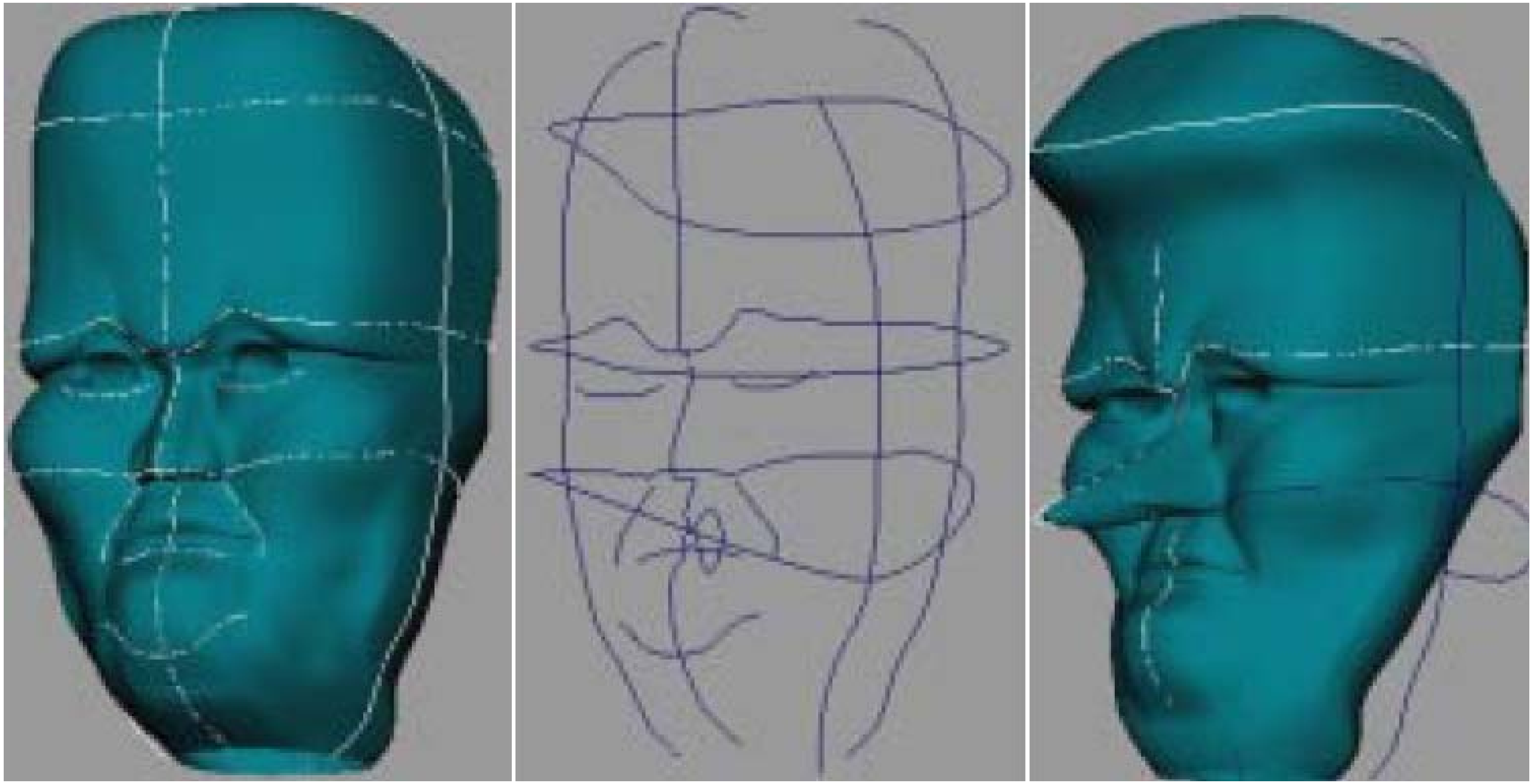





(a)
(b)
Angle Control
(c)
(d)
(e)
(f)

Wires

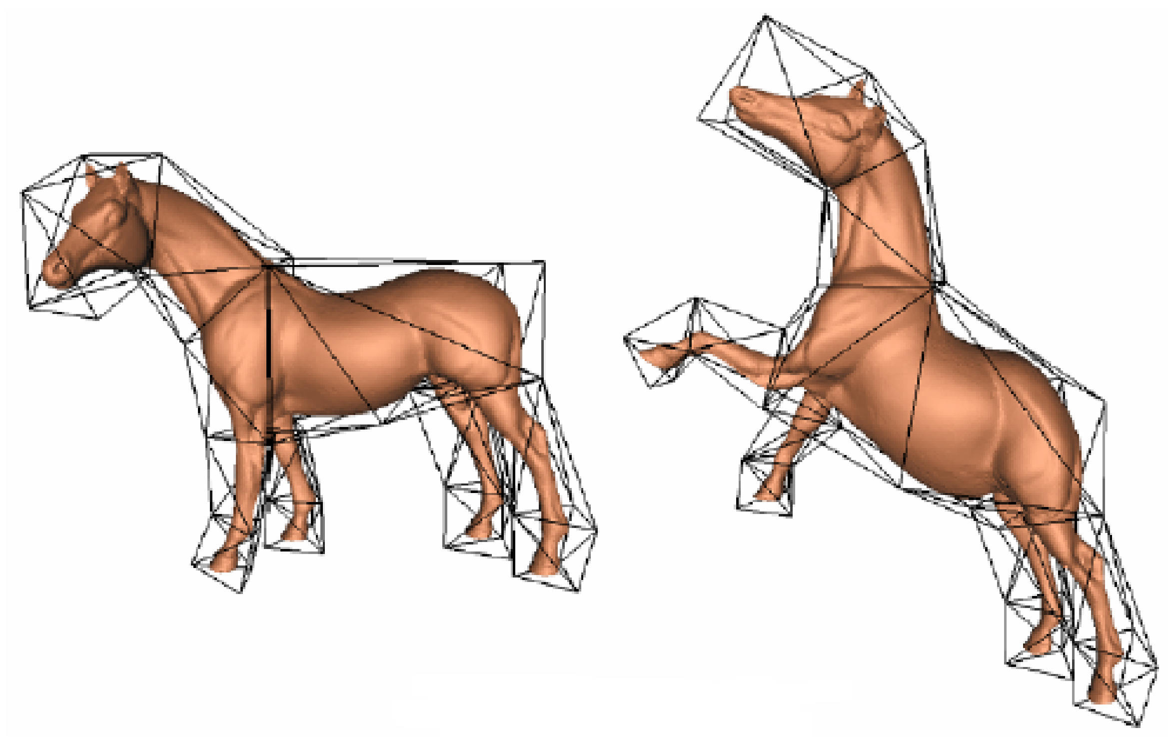
[1999]



Mean Value Coordinates

[2005]

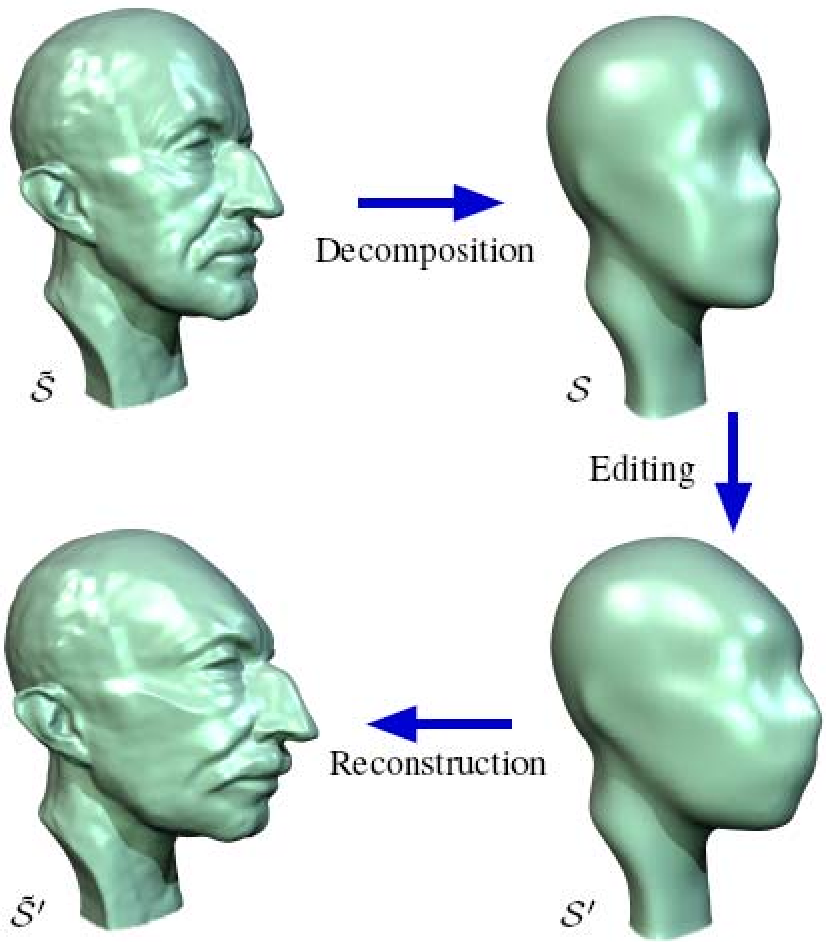
- Extended barycentric coordinates

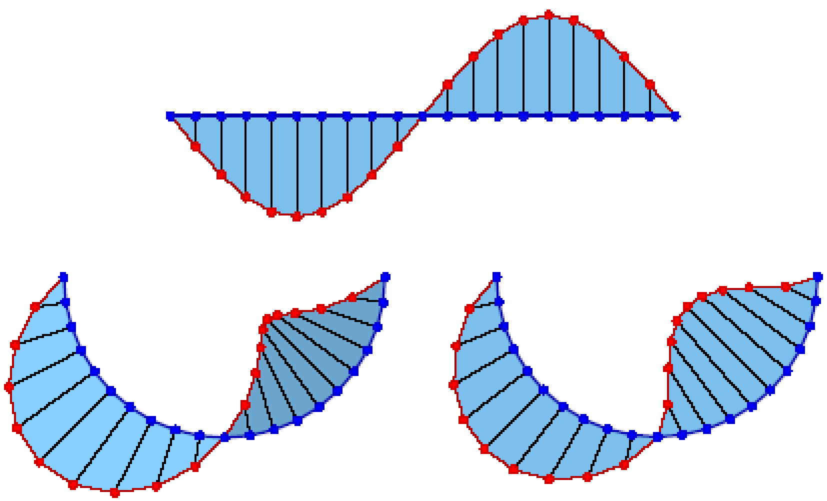


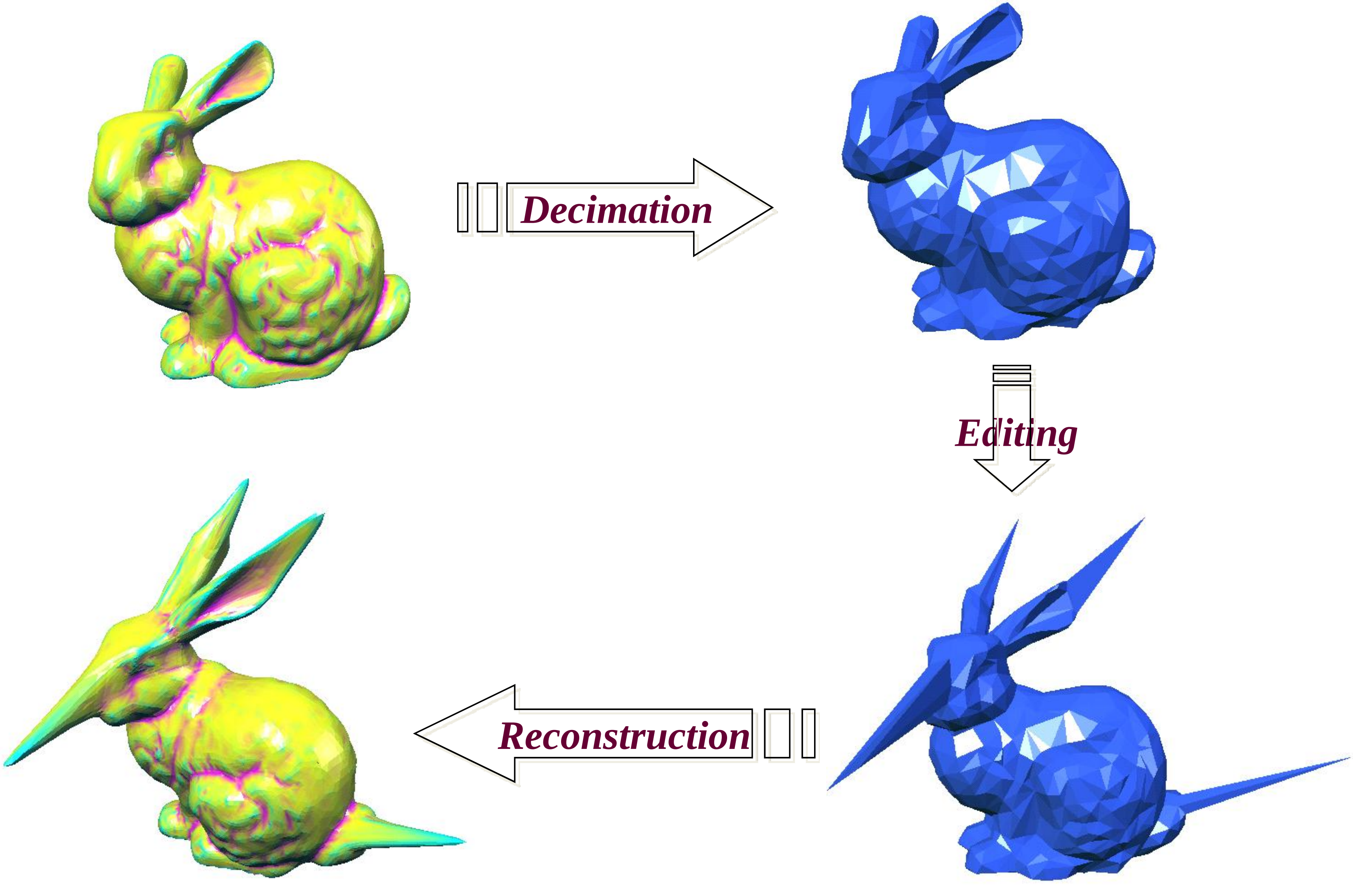
Multiresolution Editing

Multiresolution Editing

[2003]






Decimation
Editing
Reconstruction

Discrete Differential Coordinates

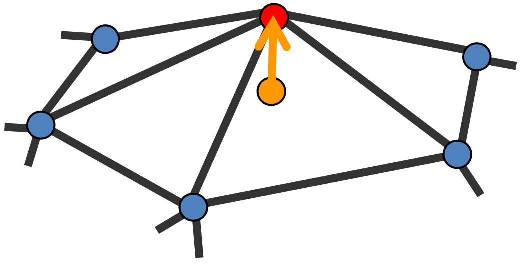
Detail Preserving Representation

Differential Coordinates

- Represent local detail at each surface point
  - better describe the shape
- Linear transition from global to differential
- Useful for operations on surfaces where surface details are important

What's are Details?

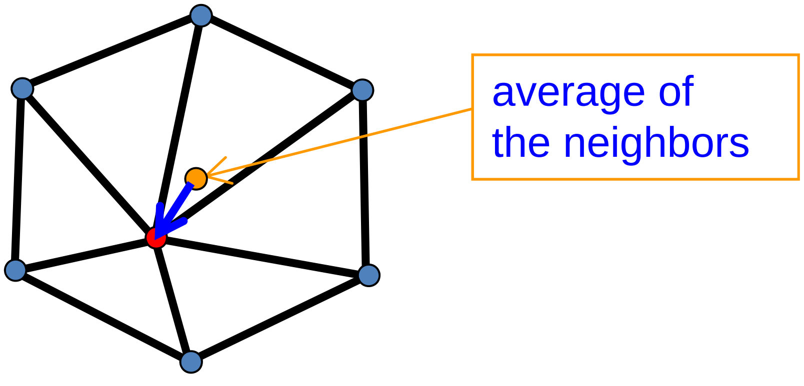
- Detail = surface – smooth (surface)
- Smoothing = averaging



Differential Coordinates

- Differential coordinates are defined by the discrete Laplacian operator:

$$\delta_i = v_i - \sum_{j \in N(i)} w_j v_j$$



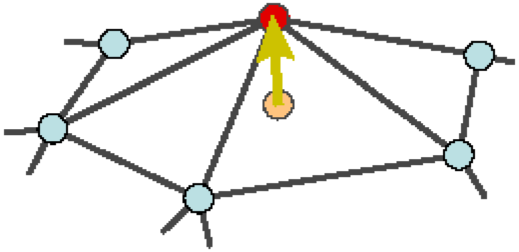
What's the Difference?

- Absolute Coordinate

$$v_i = (x_i, y_i, z_i)$$

- Relative Coordinate

$$v_i = \sum_{j \in N(i)} w_j v_j + \delta_i$$



Weighting Schemes

- Uniform weight (geometry oblivious)

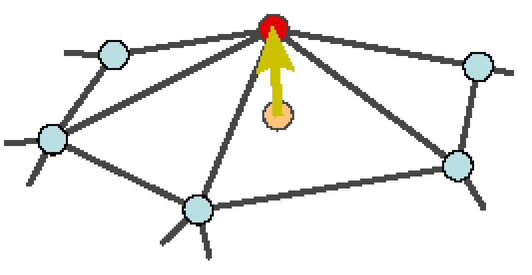
$$w_j = 1$$

- Cotangent weight (geometry aware)

$$w_j = (\cot\alpha + \cot\beta)$$

- Normalization

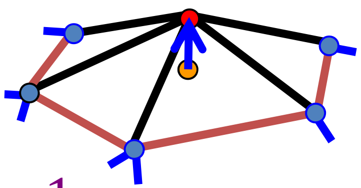
$$w_j = \frac{w_j}{\sum_j w_j}$$

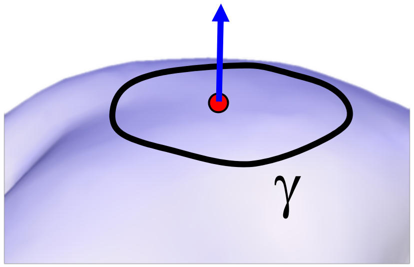


$$\delta_i = \frac{1}{d_i}\sum_{j \in N(i)} (\mathbf{v}_i - \mathbf{v}_j)$$

Geometric Meaning

- DCs represent the **local** detail / local shape description
  - The direction approximates the normal
  - The size approximates the mean curvature





$$\boldsymbol{\delta}_{\mathbf{i}} = \frac{1}{d_i} \sum_{\mathbf{v} \in N(i)} (\mathbf{v}_{\mathbf{i}} - \mathbf{v})$$

$$\frac{1}{len(\gamma)} \int_{\mathbf{v} \in \gamma} (\mathbf{v}_{\mathbf{i}} - \mathbf{v})\, ds$$

$$\lim_{len(\gamma) \to 0} \frac{1}{len(\gamma)} \int_{\mathbf{v} \in \gamma} (\mathbf{v}_{\mathbf{i}} - \mathbf{v})\, ds = H(\mathbf{v}_{\mathbf{i}})\, \mathbf{n}_{\mathbf{i}}$$

Laplacian Matrix

- The transition between the δ and *xyz* is linear:

$$\begin{pmatrix} \; \\ \mathrm{L} \\ \; \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} \delta_1^{(x)} \\ \delta_2^{(x)} \\ \vdots \\ \vdots \\ \delta_n^{(x)} \end{pmatrix}$$

$$A_{ij} = \begin{cases} 1 & i \in N(j) \\ 0 & otherwise \end{cases}$$

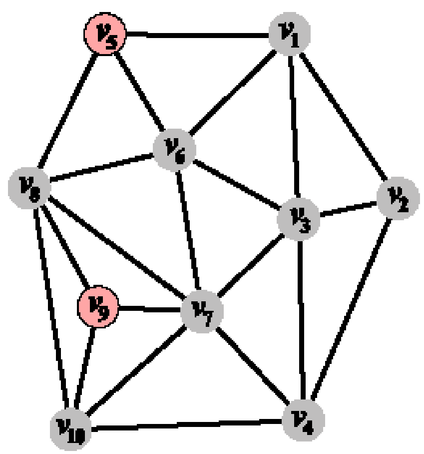
$$D_{ij} = \begin{cases} d_i & i = j \\ 0 & otherwise \end{cases}$$

$$\mathrm{L} = \mathrm{I} - \mathrm{D}^{-1}\mathrm{A}$$

Reconstruction

- From relative coordinates to absolute coordinates.
- Solving a sparse linear system

$$Lv = \delta$$



The mesh

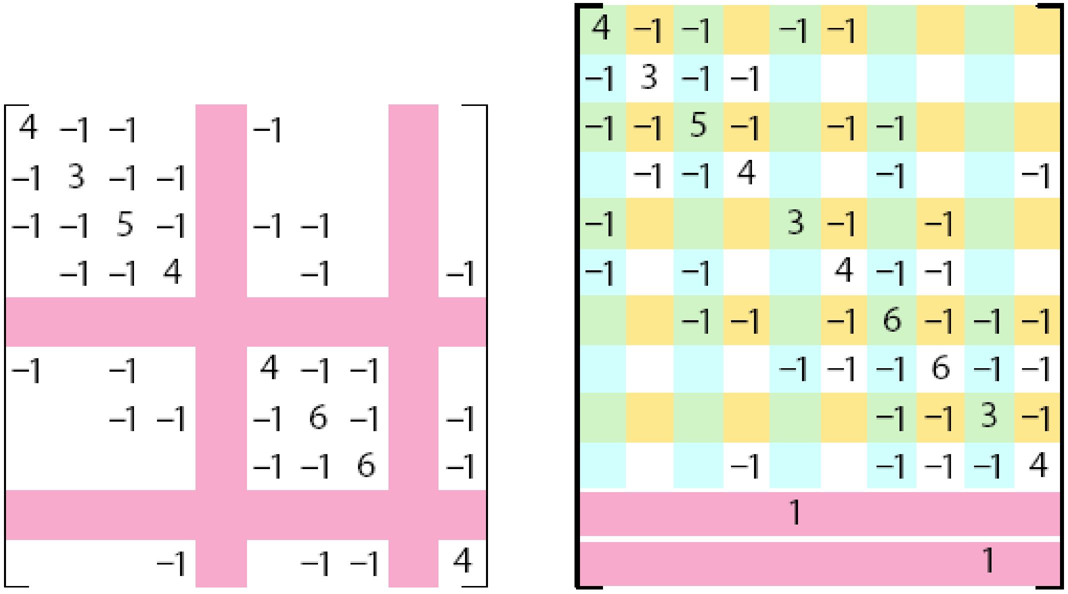
$$\begin{bmatrix} 4 & -1 & -1 & & -1 & -1 & & & & \\ -1 & 3 & -1 & -1 & & & & & & \\ -1 & -1 & 5 & -1 & & -1 & -1 & & & \\ & -1 & -1 & 4 & & & -1 & & & -1 \\ -1 & & & & 3 & -1 & & -1 & & \\ -1 & & -1 & & & 4 & -1 & -1 & & \\ & & -1 & -1 & & -1 & 6 & -1 & -1 & -1 \\ & & & & -1 & -1 & -1 & 6 & -1 & -1 \\ & & & & & & -1 & -1 & 3 & -1 \\ & & & -1 & & & -1 & -1 & -1 & 4 \end{bmatrix}$$

The symmetric Laplacian L_s

Reconstruction

- Soft constraints

$$L^T L v = L^T \delta$$



Invertible Laplacian

2-anchor $\tilde{L}$

Variational Viewpoint

- Laplacian Approximation

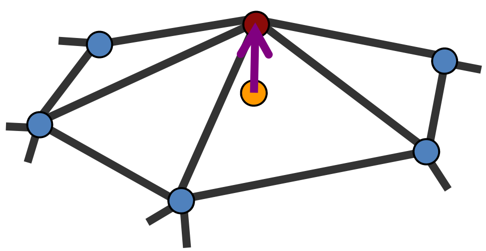
$$\tilde{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \left(\|L\mathbf{x} - \delta^{(x)}\|^2 + \sum_{j \in C} \omega^2 \|x_j - c_j\|^2 \right).$$

- Gradient Approximation

$$\min_{\phi} \int \int_{\Omega} \|\nabla \phi - \mathbf{w}\|^2 dA,$$

Laplacian matrix

- The transition between the δ and *xyz* is linear:



$$\boldsymbol{\delta}_i = \sum_{j \in N(i)} w_{ij} \left(\mathbf{v}_i - \mathbf{v}_j \right)$$

$$\mathbf{L} \, \mathbf{v_x} = \delta_{\mathbf{x}}$$

$$\mathbf{L} \, \mathbf{v_y} = \delta_{\mathbf{y}}$$

$$\mathbf{L} \, \mathbf{v_z} = \delta_{\mathbf{z}}$$

Basic properties

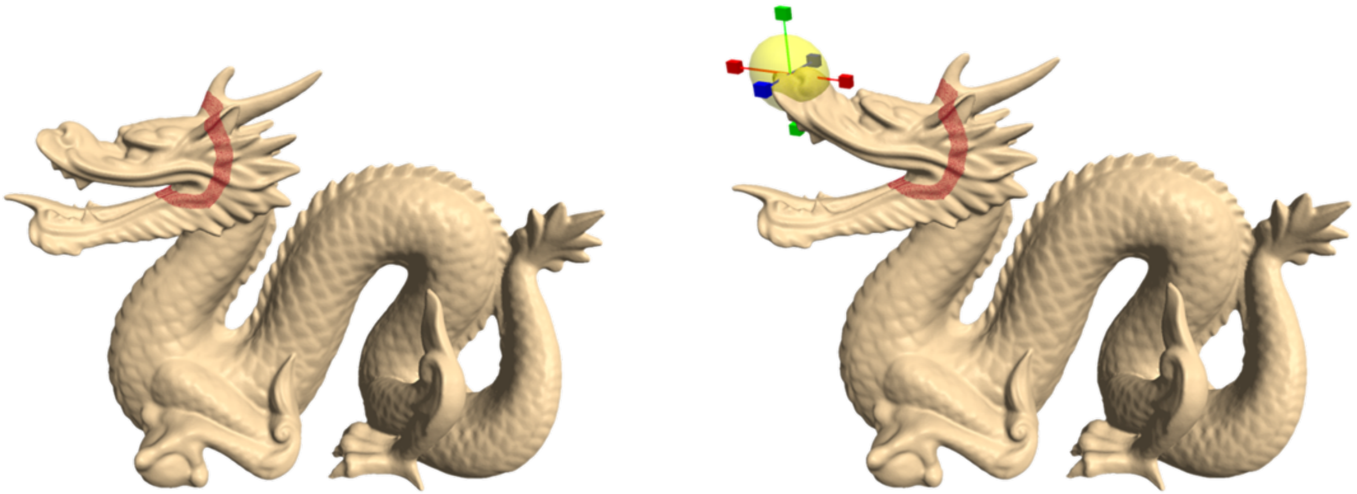
- Rank(L) = n-c   (n-1 for connected meshes)
- We can reconstruct the xyz geometry from delta up to translation

$$L\mathbf{x} = \boldsymbol{\delta}$$

Laplacian Mesh Editing

Laplacian Editing

- Local detail representation – enables detail preservation through various modeling tasks
- Representation with sparse matrices
- Efficient linear surface reconstruction



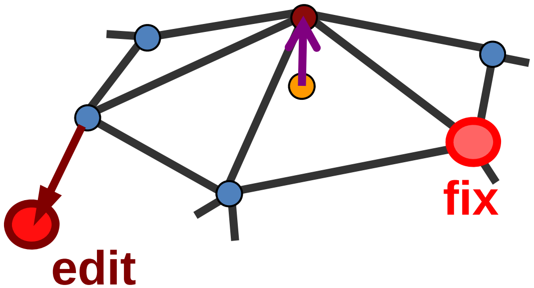
Editing framework

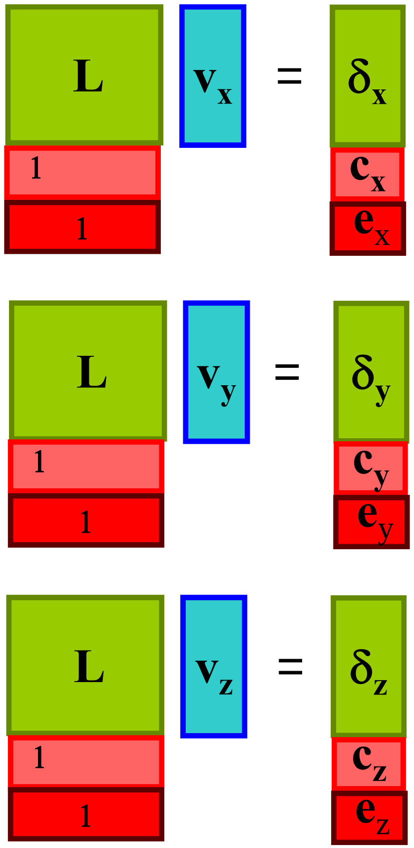
- The spatial constraints will serve as modeling constraints
- Reconstruct the surface every time the modeling constraints are changed

Detail constraints: $L\mathbf{x} = \boldsymbol{\delta}$

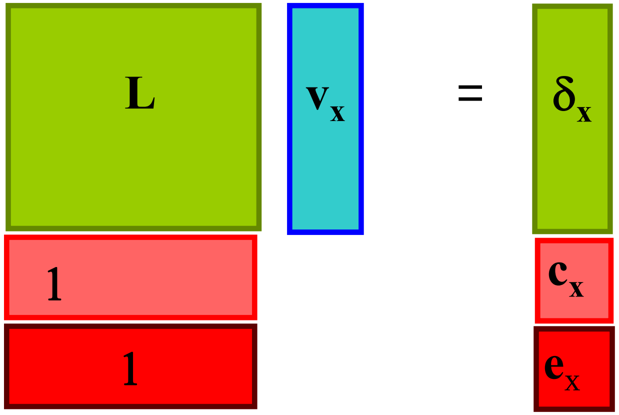
Modeling constraints: $x_j = c_j, \quad j \in \{j_1, j_2, \ldots j_k\}$

Reconstruction



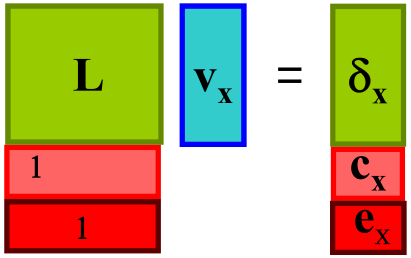


Reconstruction



$$\tilde{\mathbf{x}} = \arg\min_{\mathbf{x}} \left(\left\| L\mathbf{x} - \boldsymbol{\delta}_x \right\|^2 + \sum_{s=1}^{k} \left| x_k - c_k \right|^2 \right)$$

Reconstruction



$$A \quad x \quad = \quad b$$

Normal Equations:

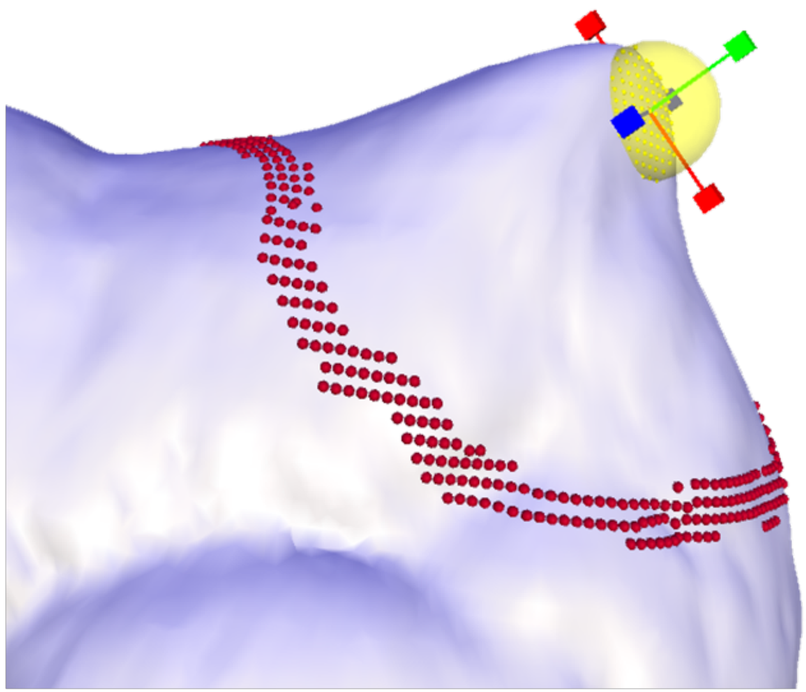
$$A^T A \; x = A^T b$$

$$x = (A^T A)^{-1} \; A^T b$$

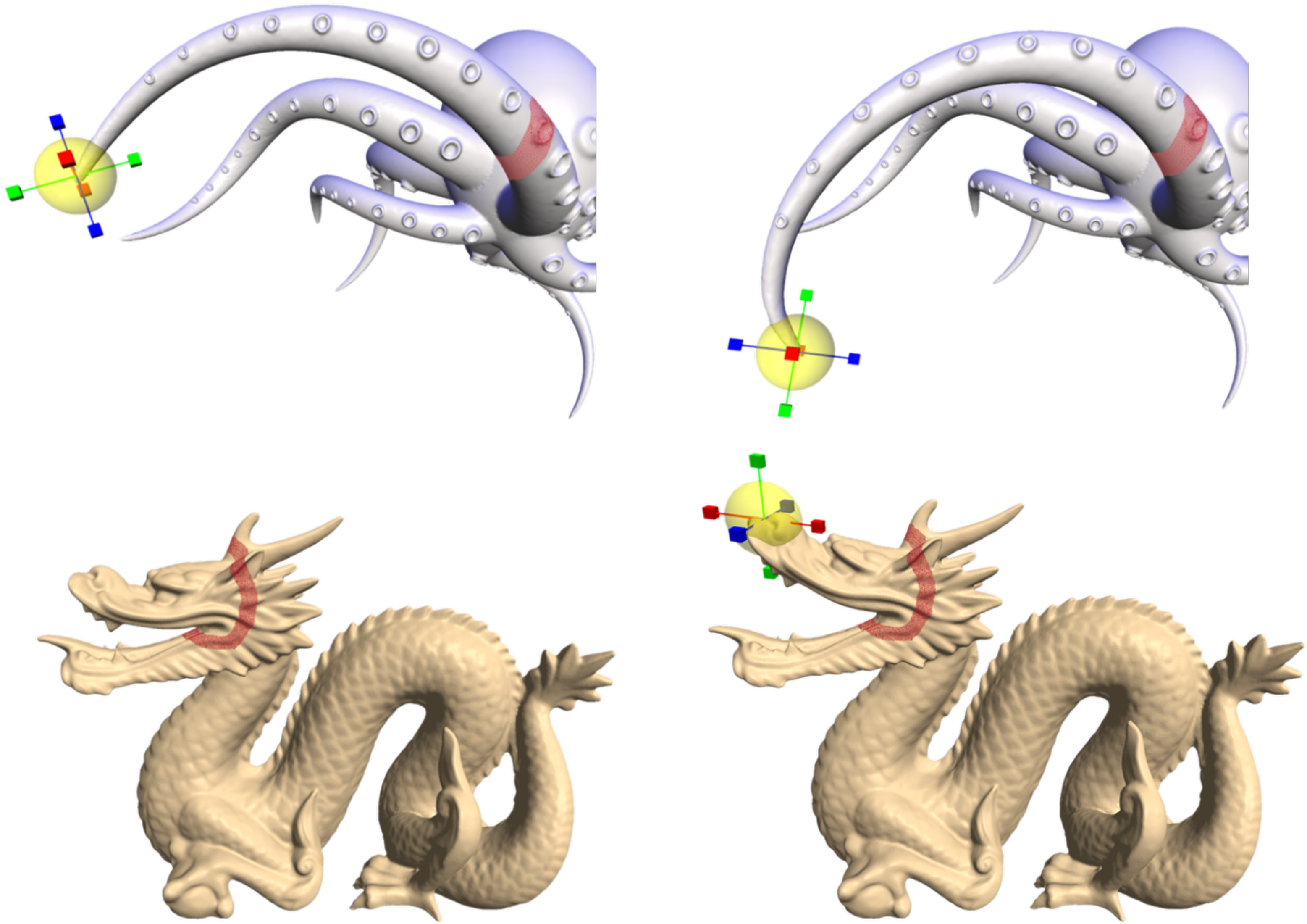
compute once

User Interfaces

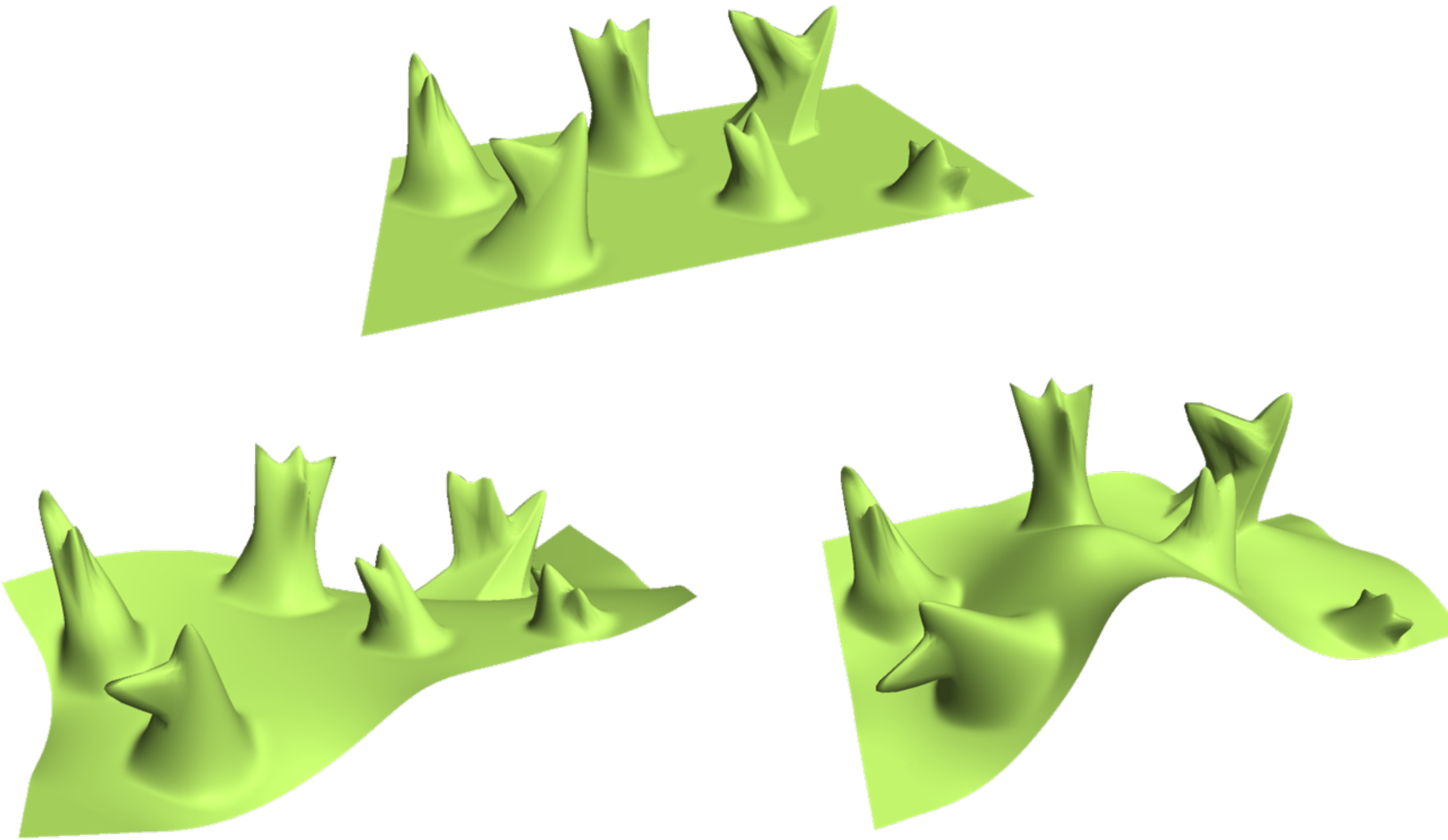
- ROI is bounded by a belt (static anchors)
- Manipulation through handle(s)



Results

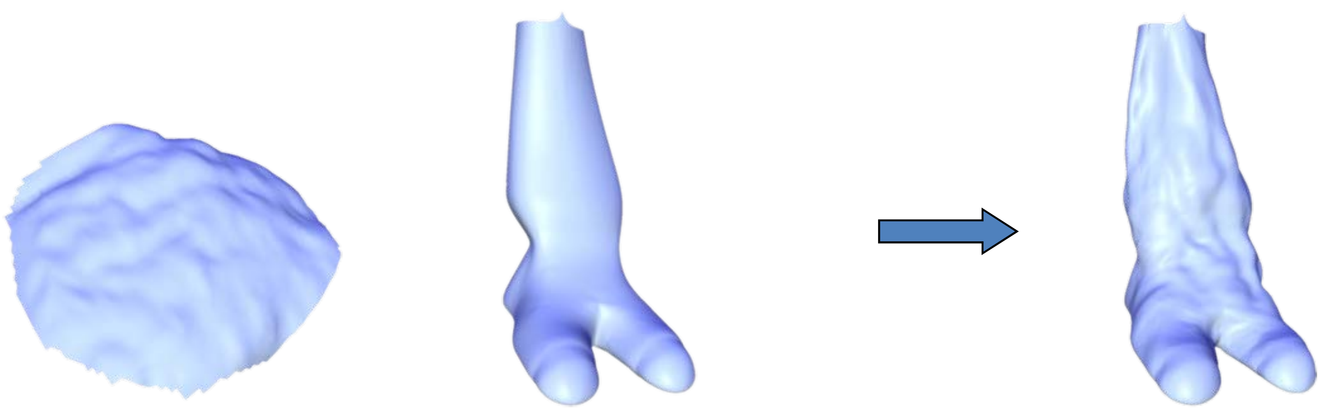


Results



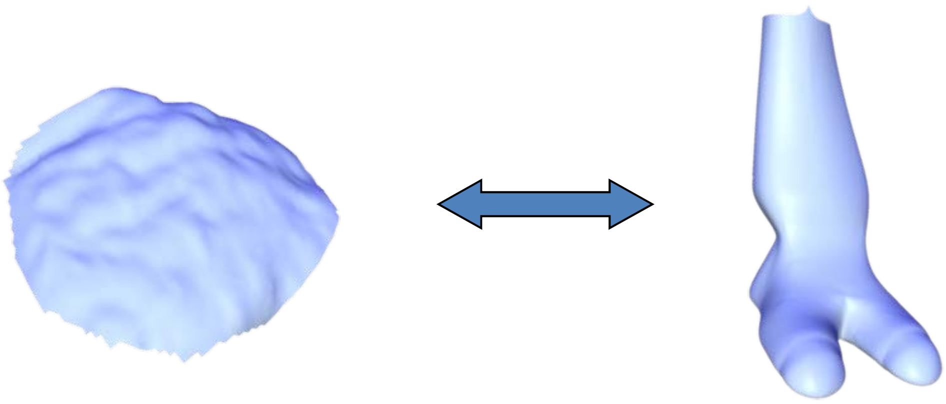
Detail transfer and mixing

- “Peel“ the coating of one surface and transfer to another



Detail transfer and mixing

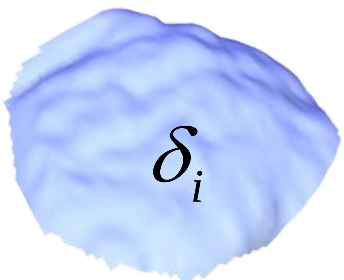
- Correspondence:

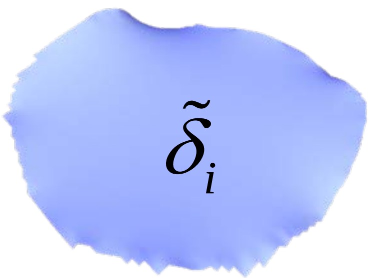


  - Parameterization onto a common domain and elastic warp to align the features, if needed

Detail transfer and mixing

- Detail peeling:



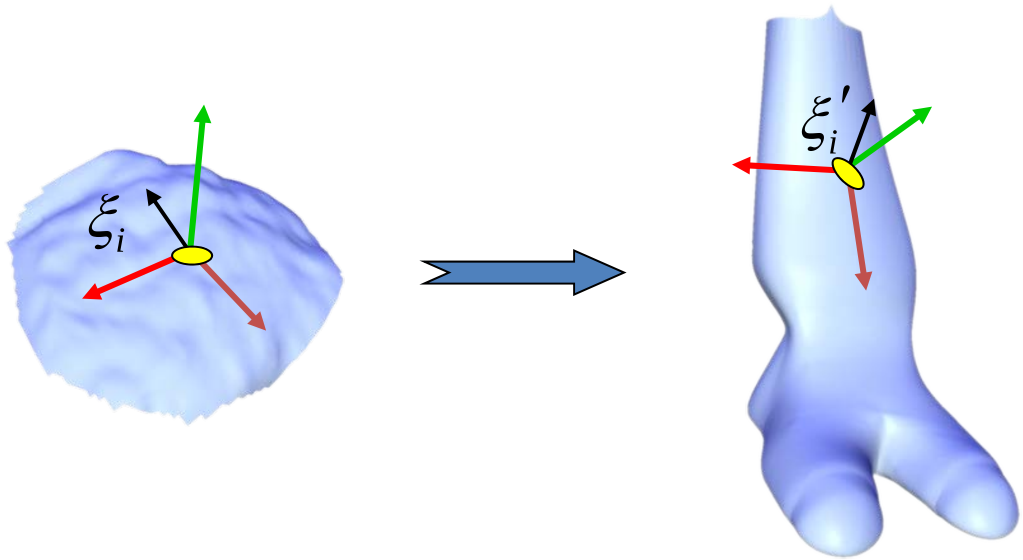


Smoothing by [Desbrun et al.99]

$$\xi_i = \delta_i - \tilde{\delta}_i$$

Detail transfer and mixing

- Changing local frames:

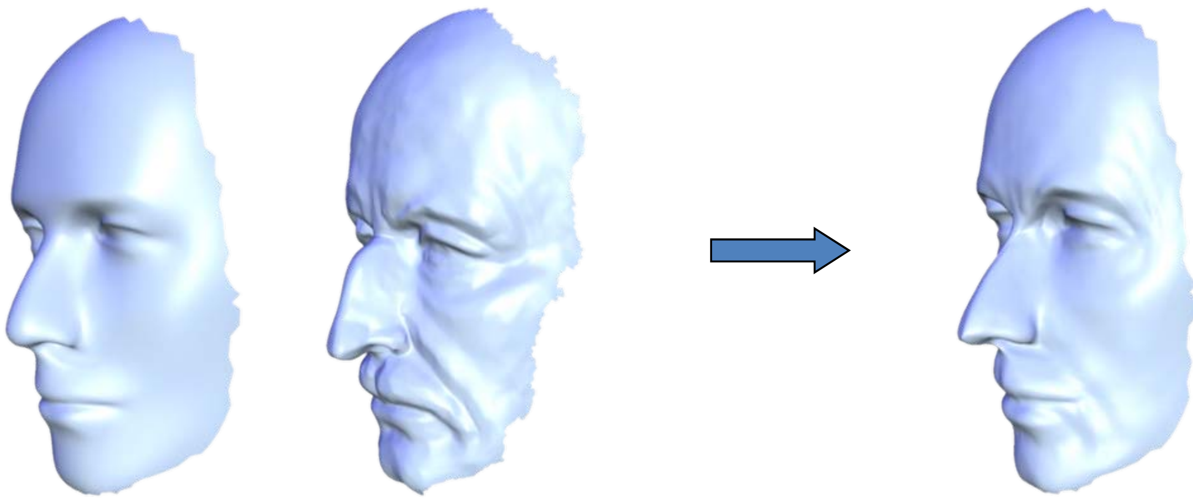


Detail transfer and mixing

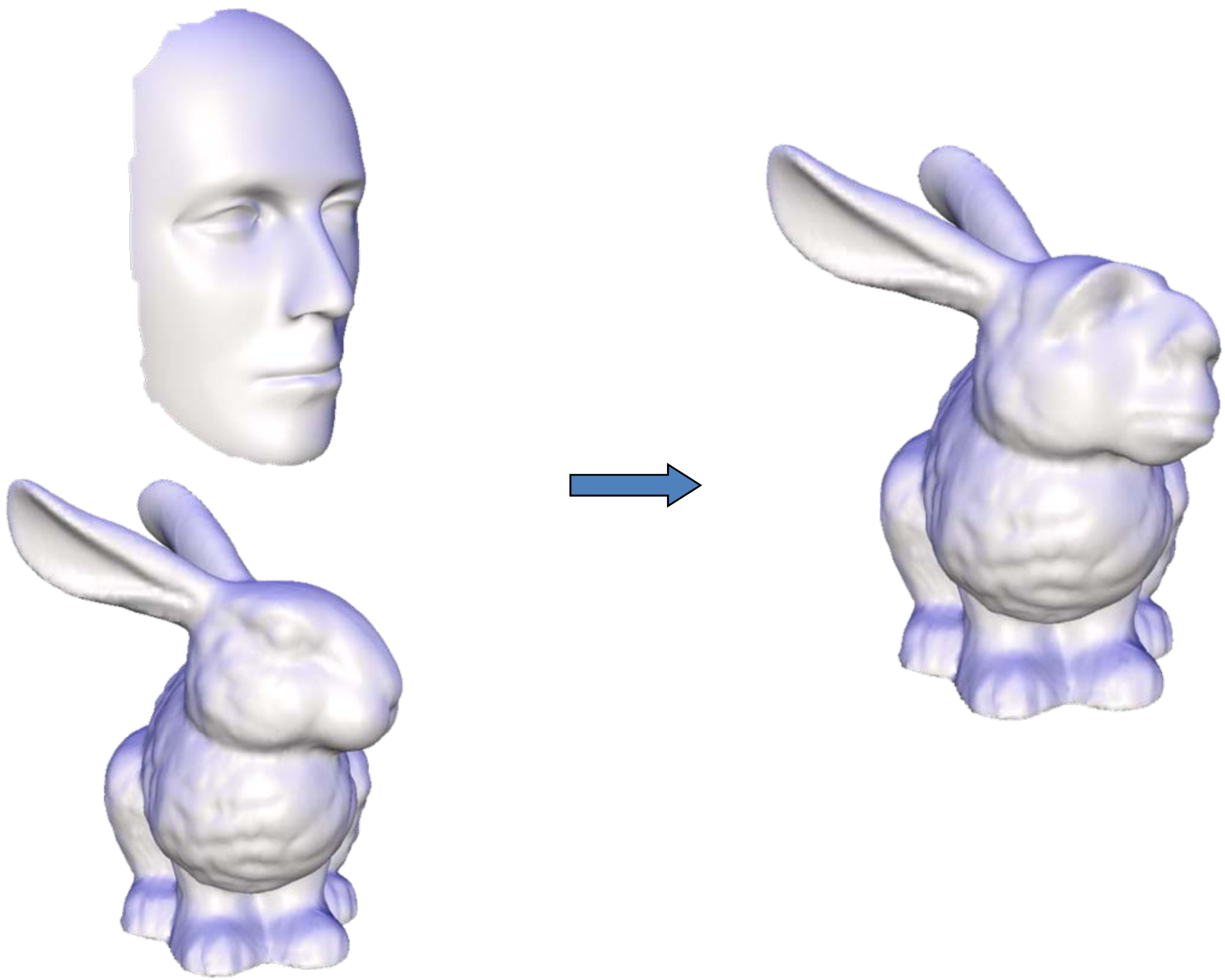
- Reconstruction of target surface from: δ_{target}

$$\delta_{\text{target}} = \delta_i' + \xi_i'$$

Examples

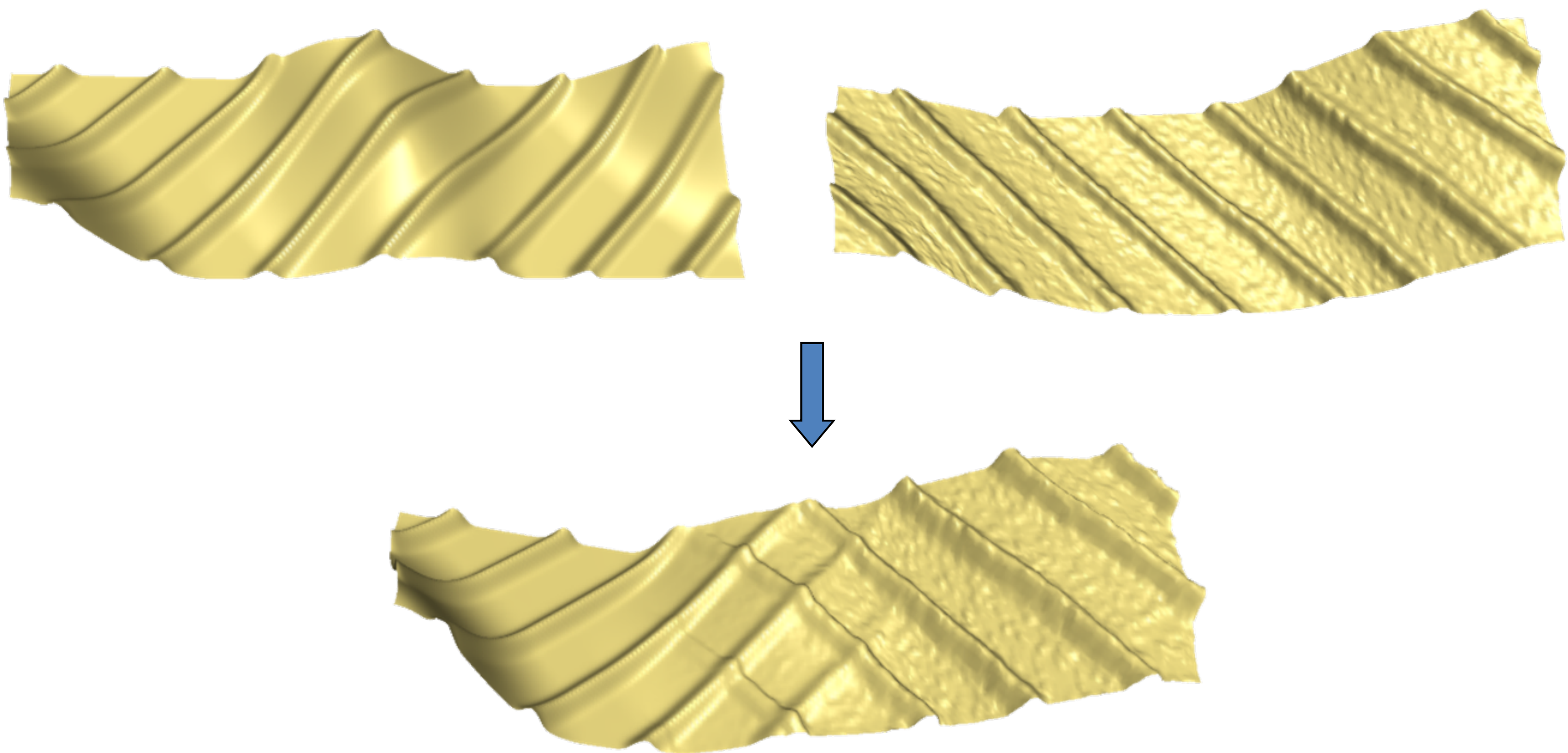


Examples



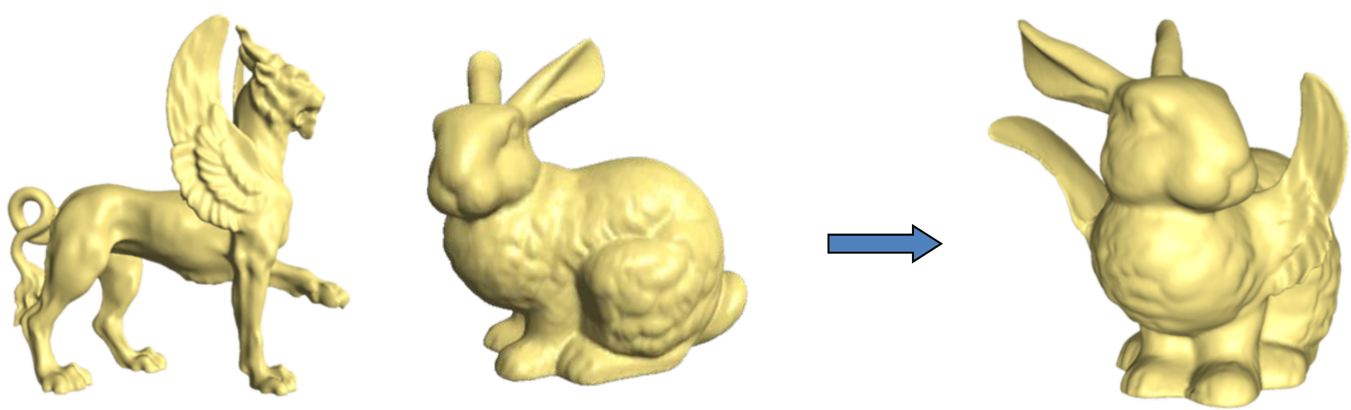
Mixing Laplacians

- Taking weighted average of δ_i and δ'_i



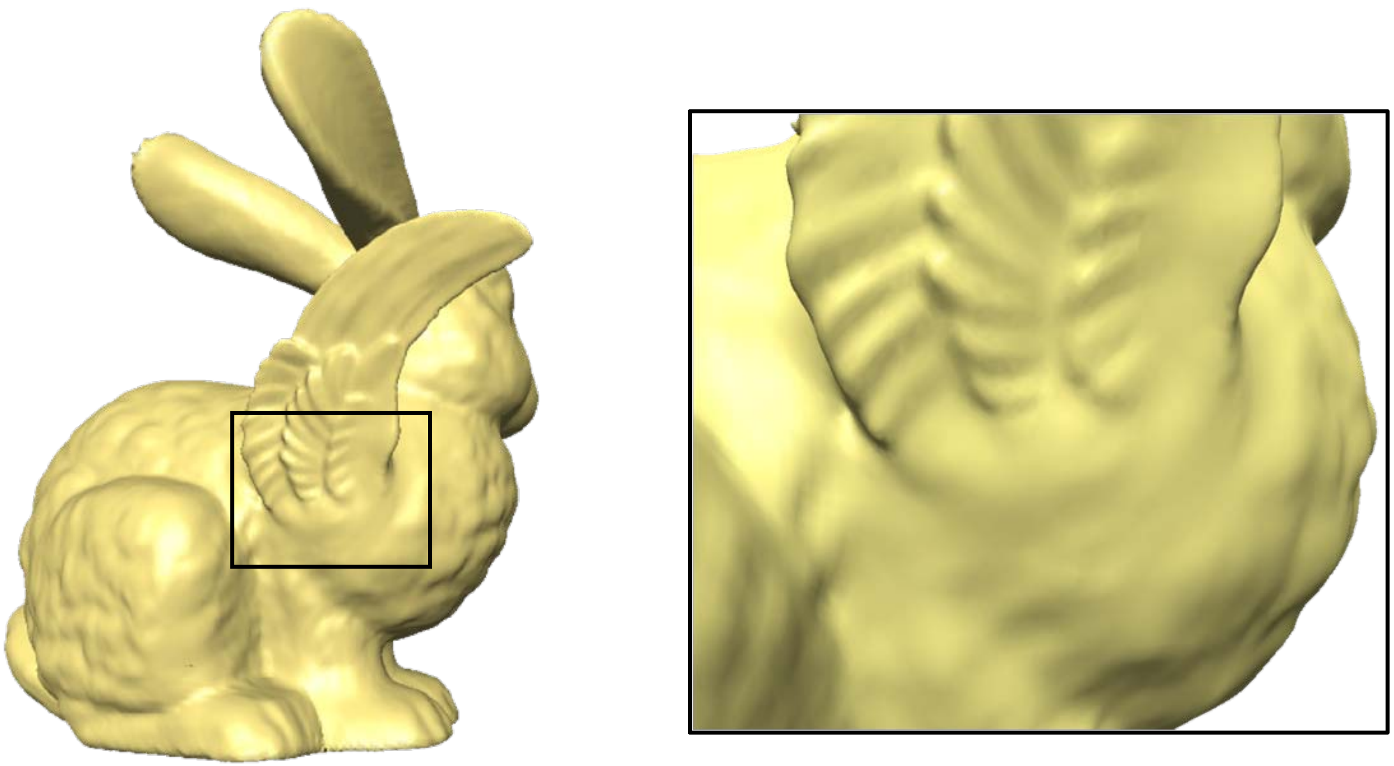
Mesh transplanting

- The user defines
  - Part to transplant
  - Where to transplant
  - Spatial orientation and scale
- Topological stitching
- Geometrical stitching via Laplacian mixing



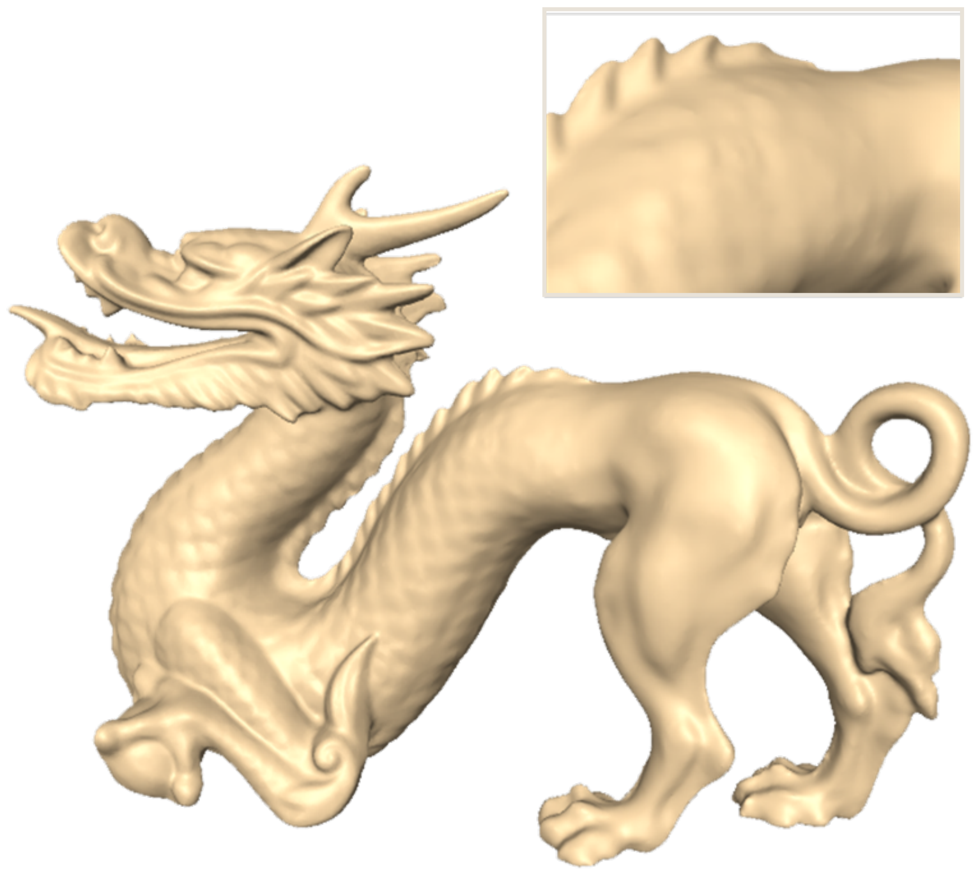
Mesh transplanting

- Details gradually change in the transition area



Mesh transplanting

- Details gradually change in the transition area



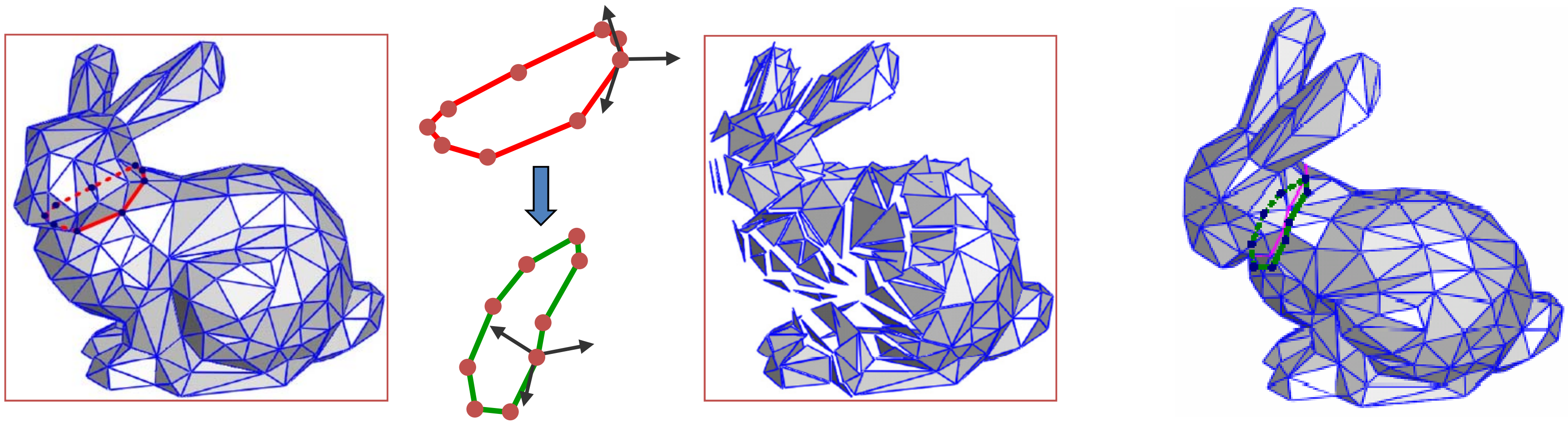
Invariance – solutions

- Explicit transformation of the differential coordinates prior to surface reconstruction
  - Lipman, Sorkine, Cohen-Or, Levin, RÖssl and Seidel [SMI 04], "Differential Coordinates for Interactive Mesh Editing",
    - Estimation of rotations from naive reconstruction
  - Yu, Zhou, Xu, Shi, Bao, Guo and Shum [SIGGRAPH 04], "Mesh Editing With Poisson-Based Gradient Field Manipulation",
    - Propagation of handle transformation to the rest of the ROI using geodesic distances
  - Zayer, RÖssl, Karni and Seidel [EG 05], "Harmonic Guidance for Surface Deformation",
    - Propagation of handle transformation to the rest of the ROI using harmonic functions

Poisson Mesh Editing

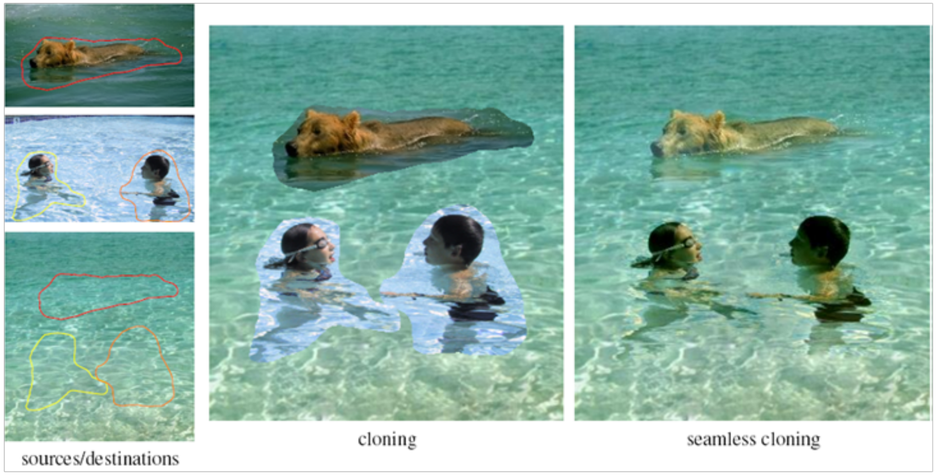
“Mesh Editing With Poisson-Based Gradient Field Manipulation”, Yu et al. 04

- The representation: the gradients of the functions X, Y, Z on each triangle of the mesh
- Deformation: propagate the transformation of the handle onto the ROI using geodesic distances



Poisson editing – images

- Inspiration: Poisson Image Editing [Pérez et al. 03]



- Reconstruct a function from its gradients via the Poisson equation:

$$\arg\min_{f} \int_{\Omega} \|\nabla f - \mathbf{w}\|^2, \quad \text{s.t.} \quad f\big|_{\partial\Omega} = f^*\big|_{\partial\Omega}$$

$$\Delta f = \operatorname{div}\mathbf{w} \quad \text{with} \quad f\big|_{\partial\Omega} = f^*\big|_{\partial\Omega}$$

Silhouette Sketch-based Editing

[Siggraph 05]

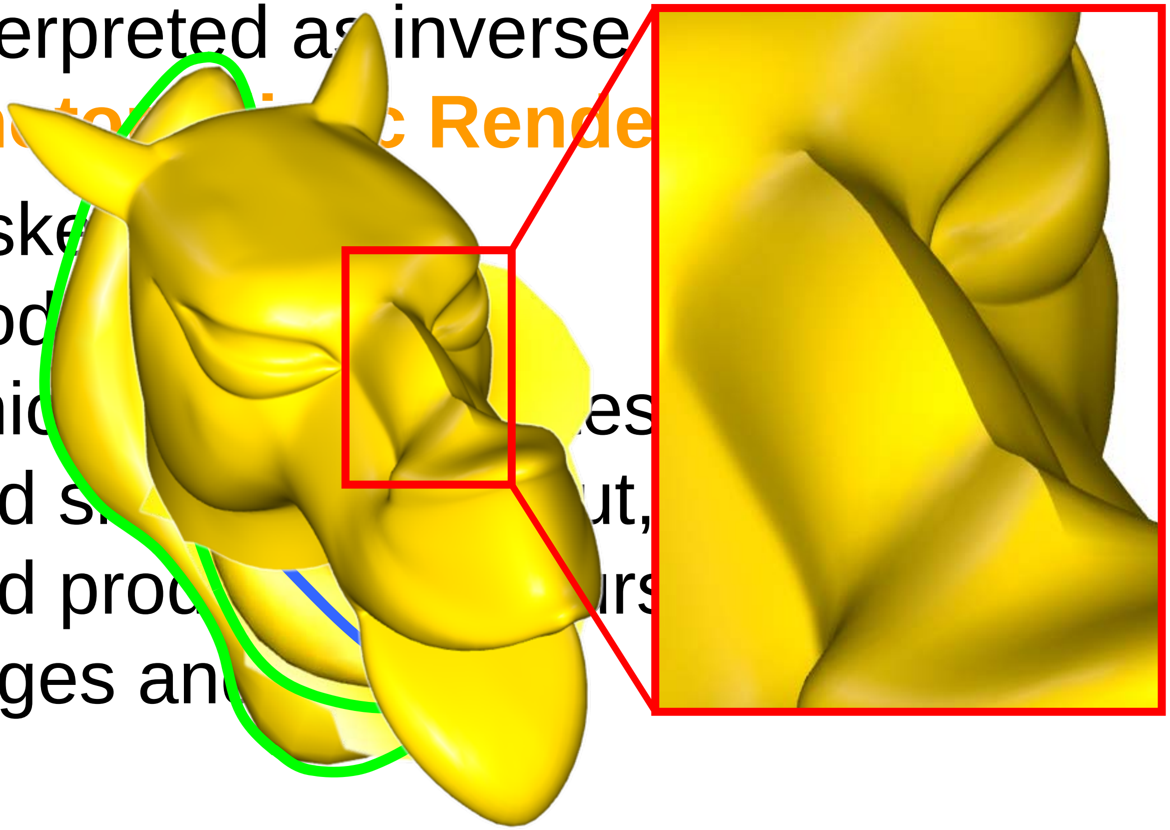
Ideas and Contributions

- Silhouette sketching

- Feature sketching

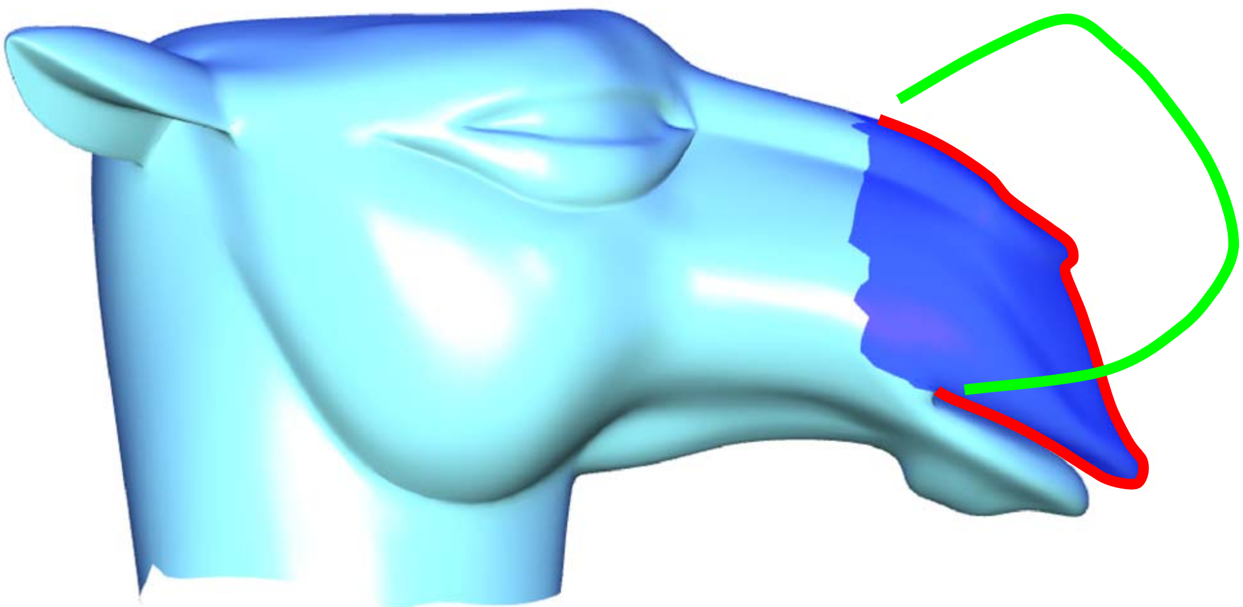
- Sketching a shape can be interpreted as inverse
Pho ic Rende

- A ske
mod
whic es
and s ut,
and proc urs
ridges an



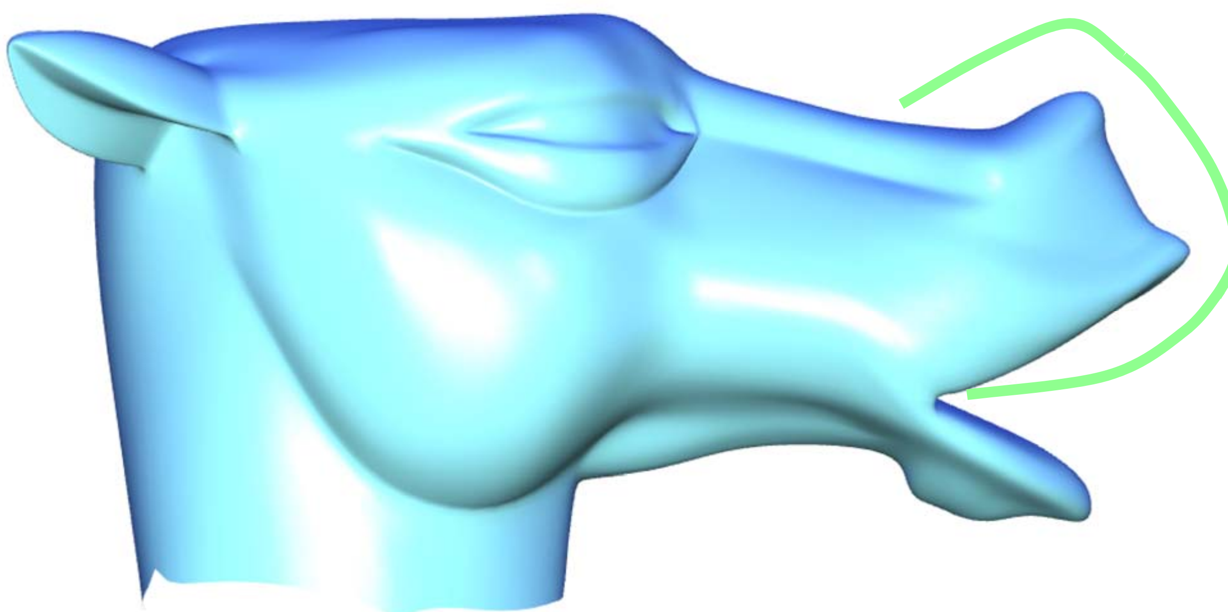
Silhouette Sketching

- Approximate sketching
  - Balance weighting between detail and positional constraints



Silhouette Sketching

- Approximate sketching
  - Balance weighting between detail and positional constraints

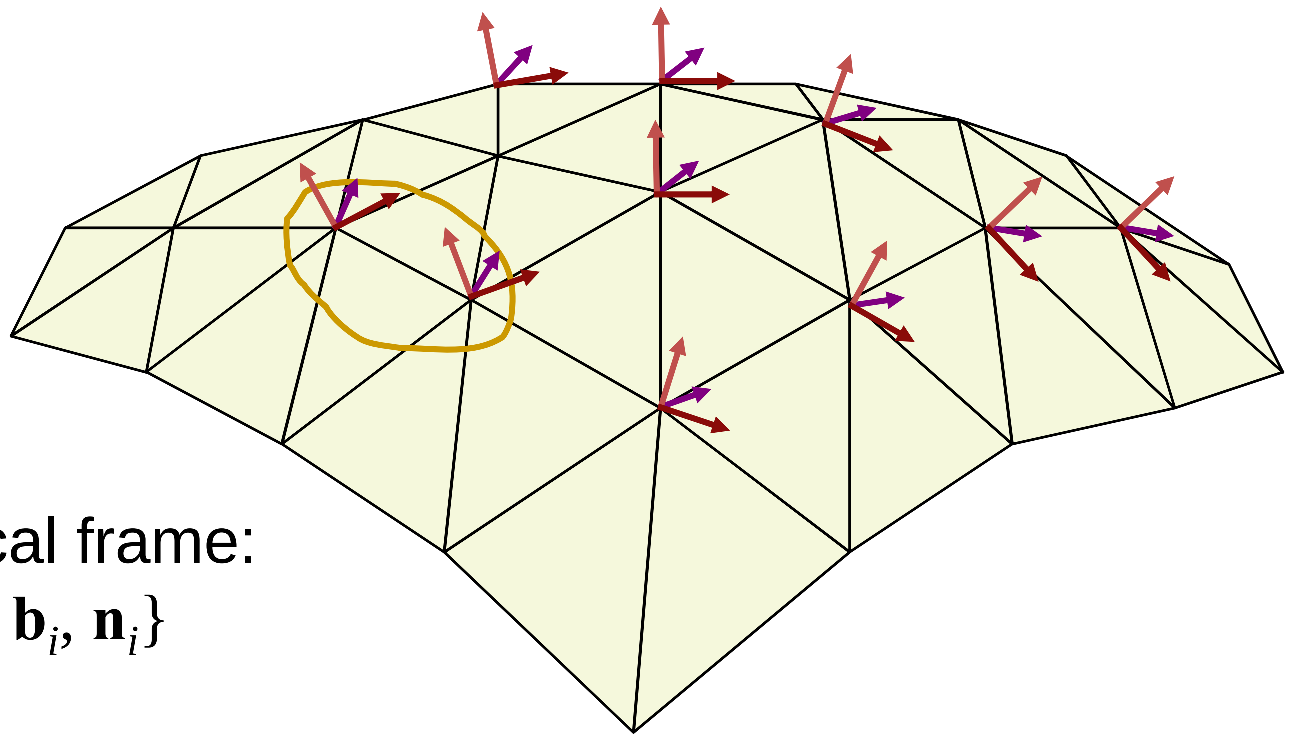


Linear Rotation-invariant Coordinates

[Siggraph 05]

Frame-based deformations

- Keep a local frame at each vertex
- Prescribe changes to some selected frames



Local frame:
$\{\mathbf{a}_i, \mathbf{b}_i, \mathbf{n}_i\}$

Frame-based deformations

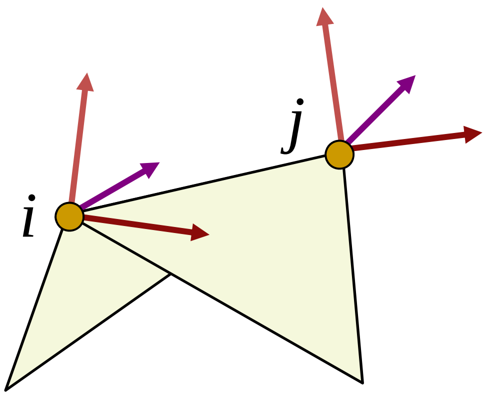
– Encode the differences between adjacent frames

– Solve for the new frames in least-squares sense

$$\mathbf{a}_i - \mathbf{a}_j = \alpha_1 \mathbf{a}_i + \alpha_2 \mathbf{b}_i + \alpha_3 \mathbf{n}_i$$
$$\mathbf{b}_i - \mathbf{b}_j = \beta_1 \mathbf{a}_i + \beta_2 \mathbf{b}_i + \beta_3 \mathbf{n}_i$$
$$\mathbf{n}_i - \mathbf{n}_j = \gamma_1 \mathbf{a}_i + \gamma_2 \mathbf{b}_i + \gamma_3 \mathbf{n}_i$$

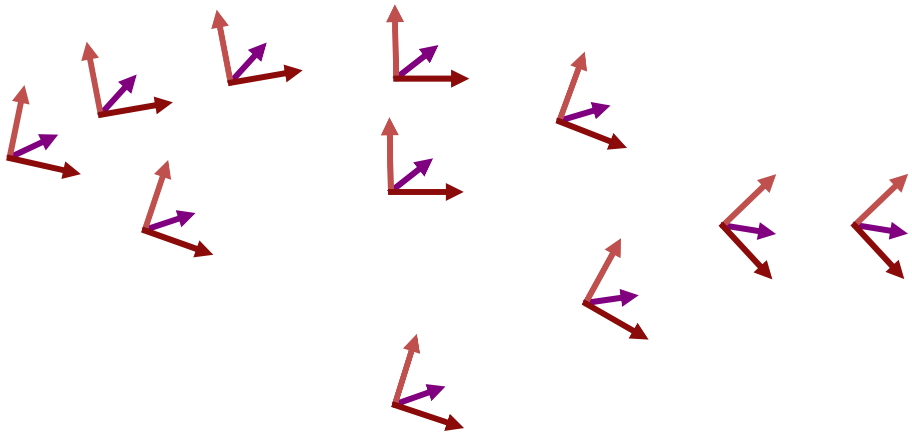
… …

constraints



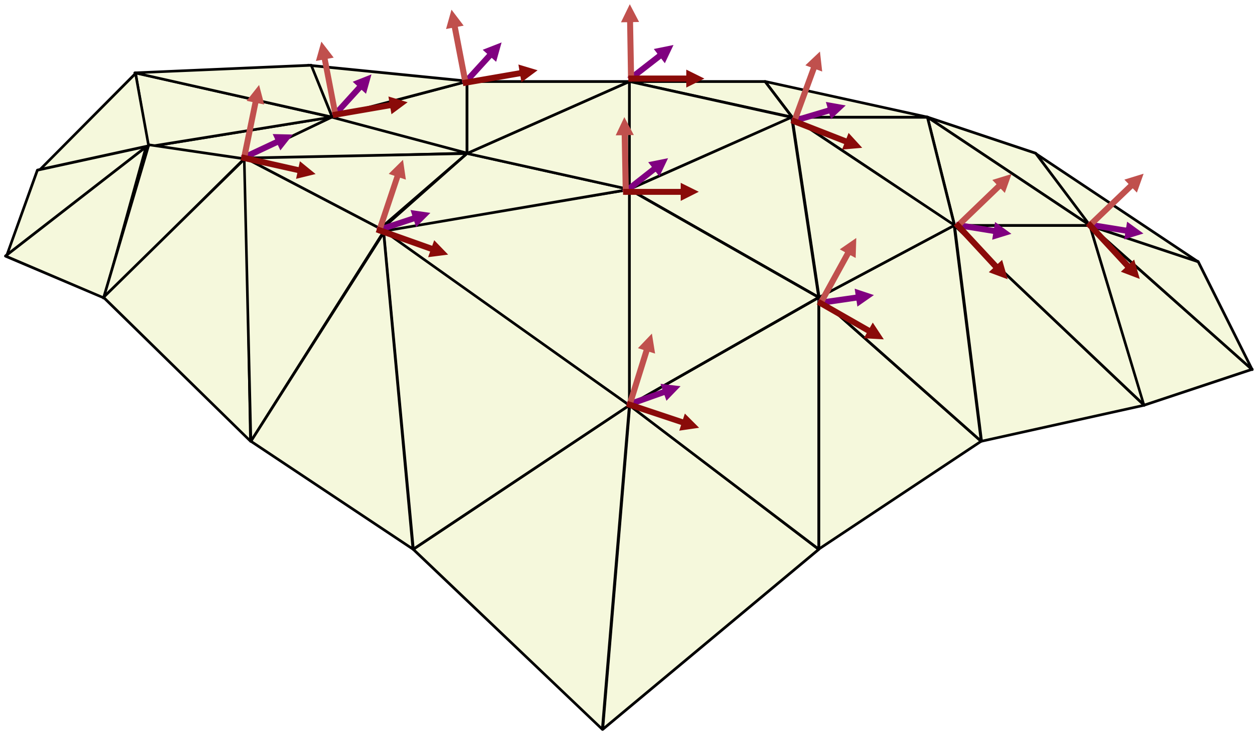
Frame-based deformations

- Reconstruction:
  - After having the frames, solve for positions

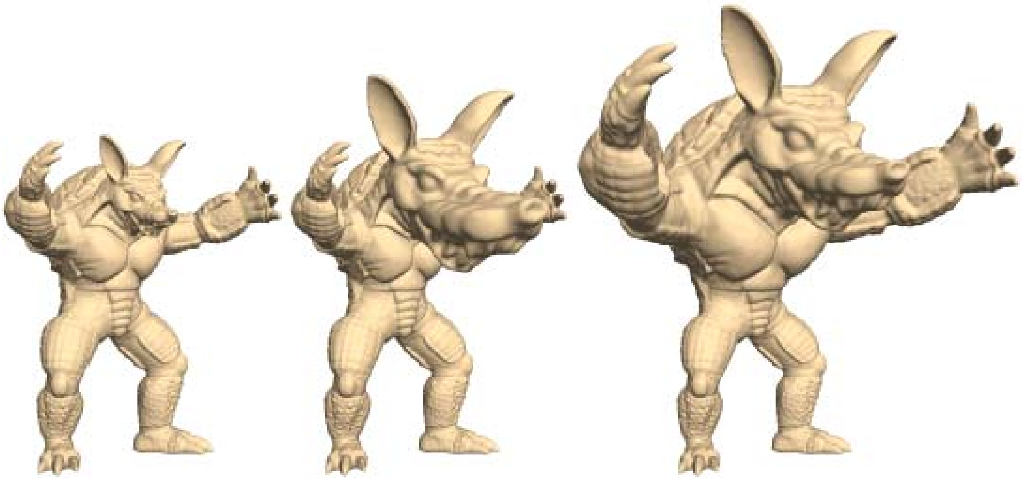


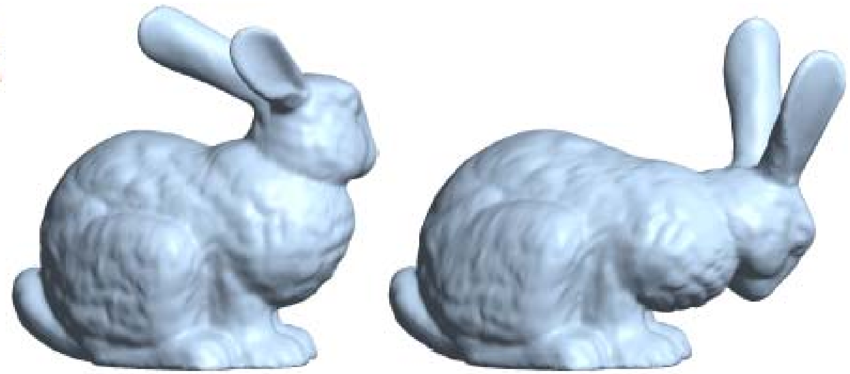
Frame-based deformations

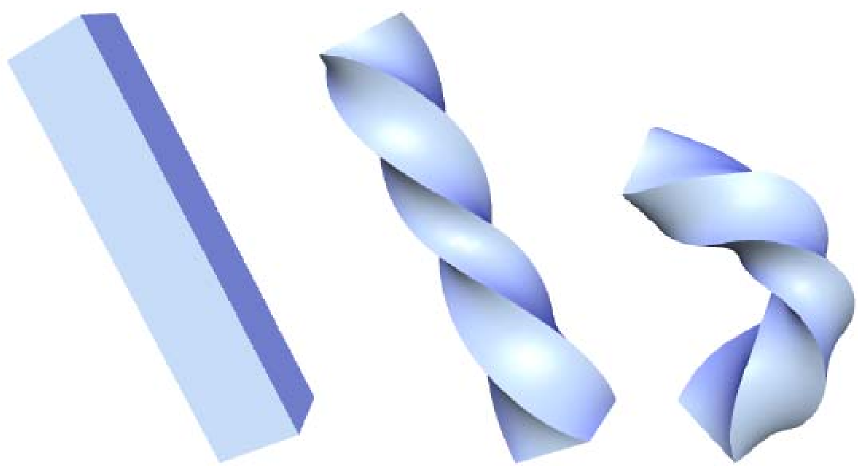
- Reconstruction:
  - After having the frames, solve for positions



Results







A Fast Multigrid Algorithm for Mesh Deformation

Siggraph 2006

Basic Model

- Two-pass pipeline
  - Local Frame Update

    R. Zayer, C. Rossl, Z. Karni and H. P. Seidel. *Harmonic Guidance for Surface Deformation*. EG2005.
  - Vertex Position Update

    Y. Lipman, O. Sorkine, D. Levin and D. Cohen-Or. *Linear rotation-invariant coordinates for meshes*. Siggraph2005.
- Multigrid Computation Method

Computation

- First Pass

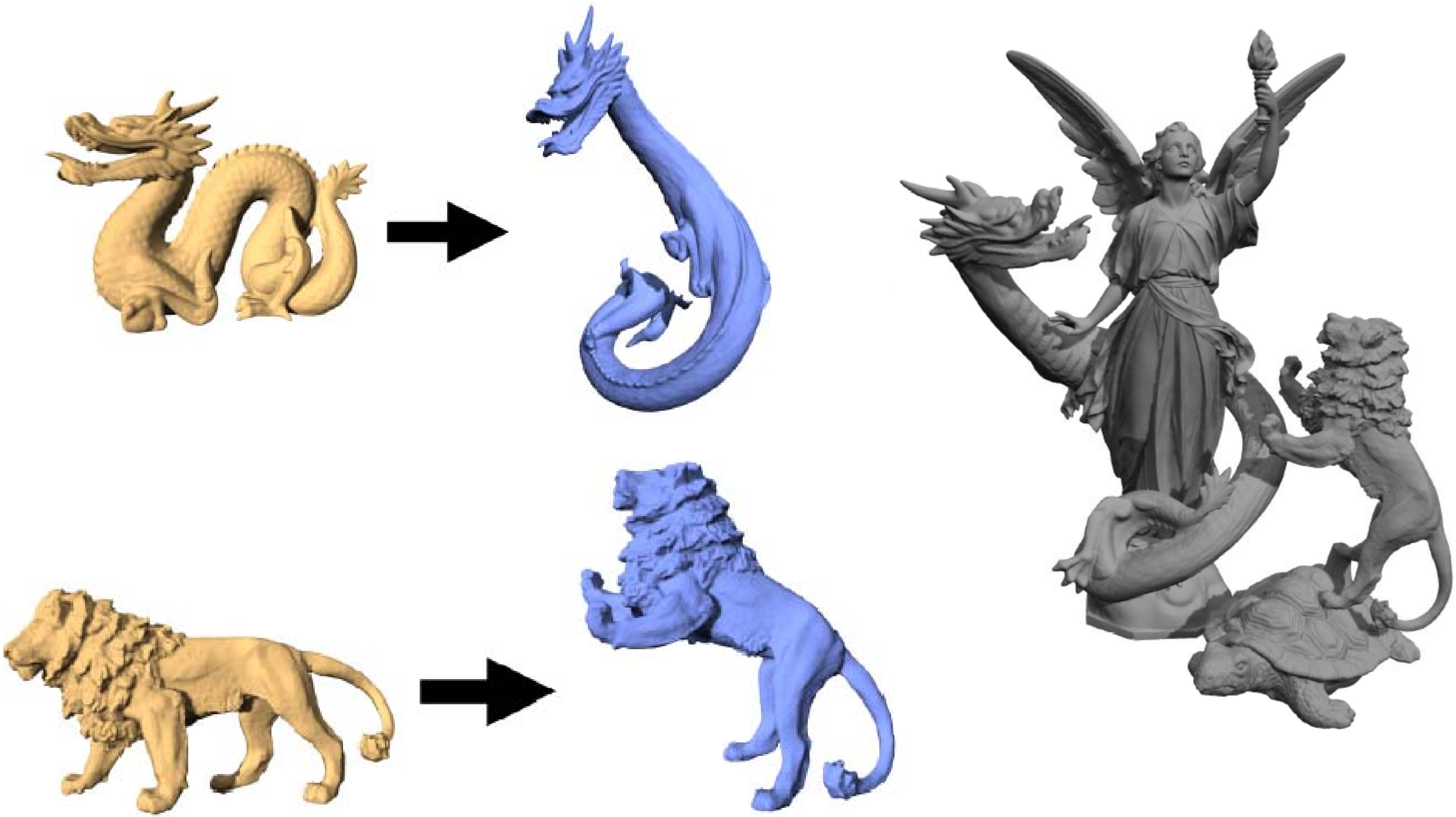
$$\nabla^2 h = 0 \Rightarrow Lh = 0$$

- Second Pass

$$\sum_{j \in N(i)} w_{ji}(\mathbf{x}_j - \mathbf{x}_i) = -\sum_{j \in N(i)} w_{ji}\mathbf{d}_{ji}$$

Multigrid Method

Examples



Performance

- Deformation for large meshes

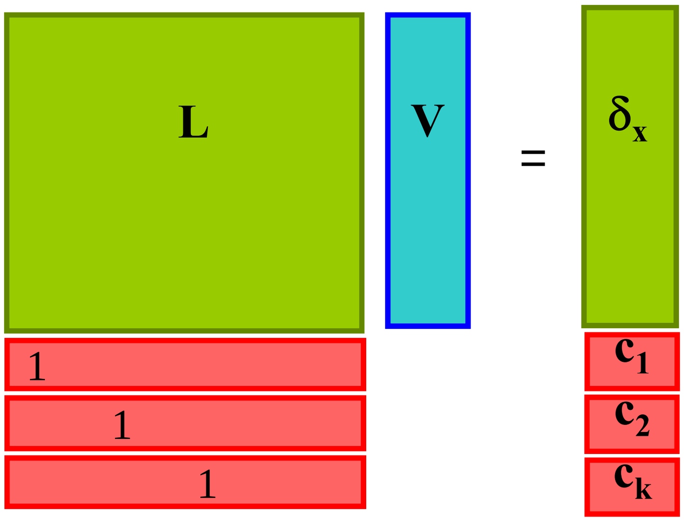
		SPRING	DINO	CAMEL	FELINE	FEMALE	LUCY	DRAGON
#Free Vertices		24,188	43,494	99,588	181,292	415,619	822,204	3,447,861
UMFPACK	Factor	1.63 sec	2.72 sec	20.59 sec	37.29 sec	113.11 sec	n/a	n/a
	Substitute	0.16 sec	0.26 sec	1.04 sec	1.95 sec	5.00 sec	n/a	n/a
	Memory	52 MB	70 MB	398 MB	710 MB	1,838 MB	>2 GB	>2 GB
CHOLMOD	Factor	0.43 sec	0.83 sec	5.48 sec	12.20 sec	31.9 sec	69.32 sec	n/a
	Substitute	0.03 sec	0.05 sec	0.15 sec	0.30 sec	0.78 sec	1.36 sec	n/a
	Memory	26 MB	35 MB	139 MB	292 MB	695 MB	1,311 MB	>2 GB
TAUCS	Factor	0.60 sec	1.04 sec	4.70 sec	10.46 sec	25.90 sec	57.65 sec	n/a
	Substitute	0.09 sec	0.16 sec	0.57 sec	1.197 sec	2.63 sec	5.35 sec	n/a
	Memory	25 MB	41 MB	139 MB	277 MB	643 MB	1,190 MB	>2 GB
Trilinos ML	Setup	0.15 sec	0.34 sec	0.57 sec	1.06 sec	2.63 sec	4.87 sec	12.60 sec
	Solve	0.57 sec	2.19 sec	5.37 sec	9.15 sec	24.00 sec	47.22 sec	148.80 sec
	Memory	15 MB	21 MB	52 MB	87 MB	200 MB	388 MB	1,080 MB
Our Multigrid	Setup	0.06 sec	0.16 sec	0.13 sec	0.24 sec	0.58 sec	0.94 sec	2.64 sec
	Solve	0.19 sec	0.39 sec	0.89 sec	1.99 sec	4.19 sec	8.47 sec	39.70 sec
#Levels /	#V-cycles	3 / 3	4 / 4	4 / 4	5 / 6	5 / 6	6 / 7	9 / 8
	Memory	10 MB	16 MB	31 MB	56 MB	119 MB	232 MB	740 MB

Shape From Connectivity

Shape from connectivity

"Least-squares Meshes", Sorkine and Cohen-Or 04

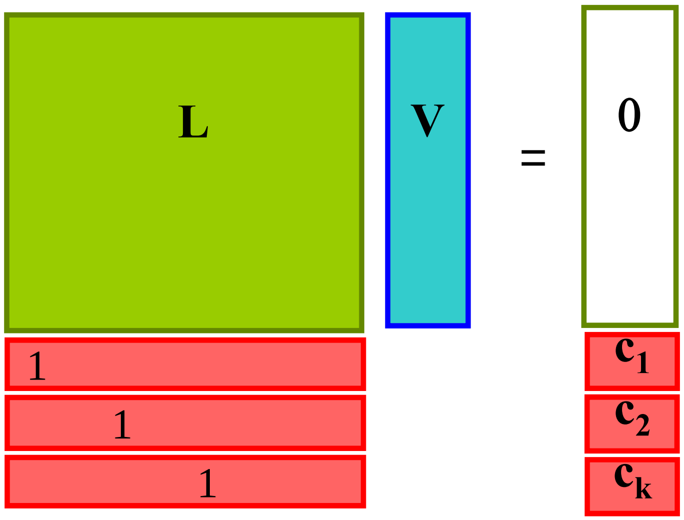
- What if we reduce delta information to zero?
- Can we still reconstruct some geometry?



Shape from connectivity

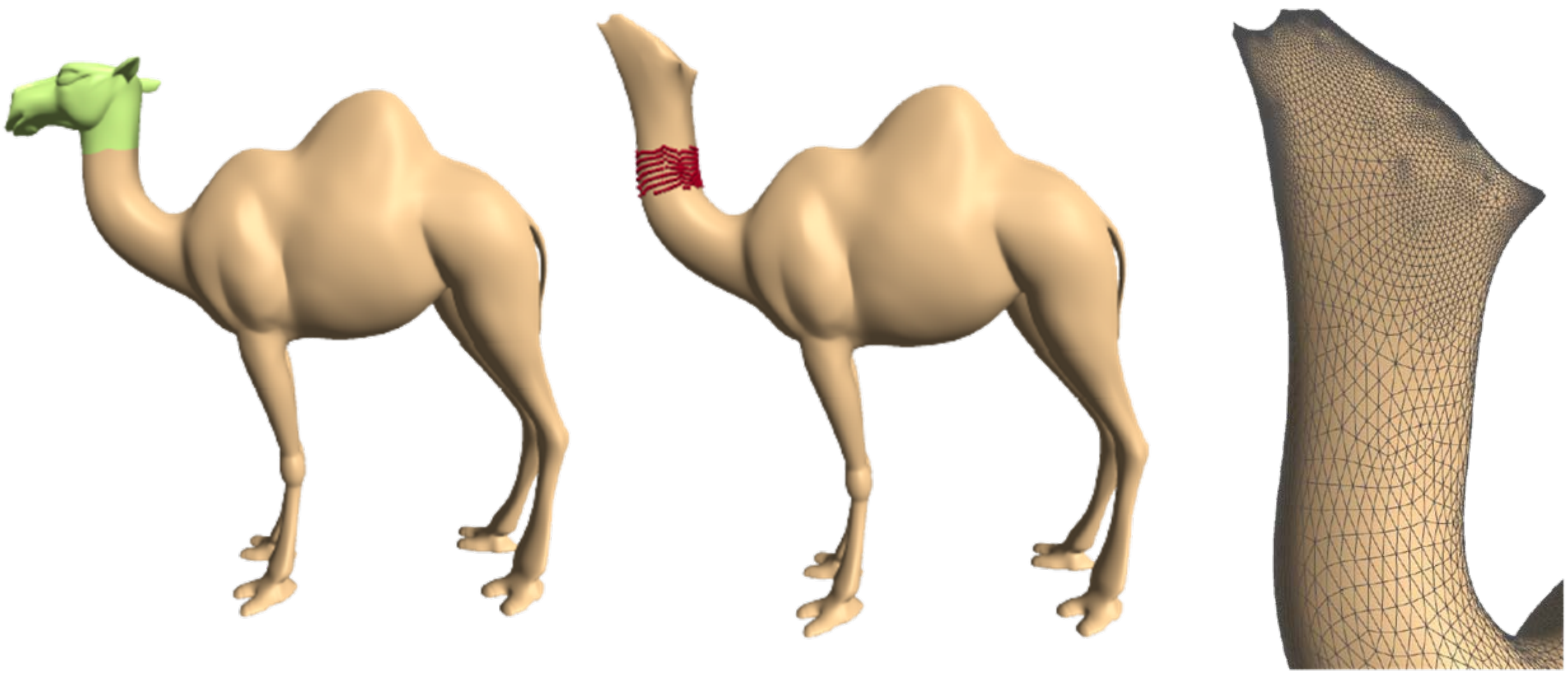
"Least-squares Meshes", Sorkine and Cohen-Or 04

- What if we reduce delta information to zero?
- Can we still reconstruct some geometry?



Geometry hidden in connectivity

"Least-squares Meshes", Sorkine and Cohen-Or 04

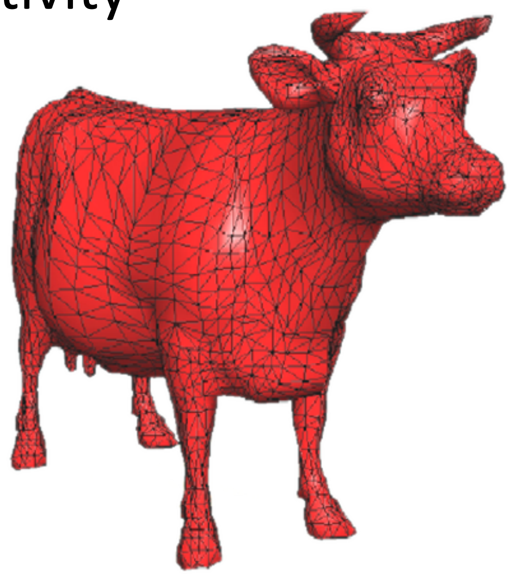


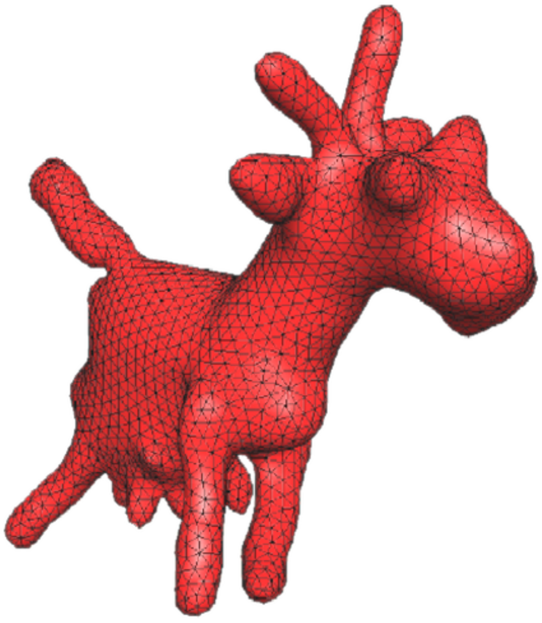
There is geometry in connectivity

Connectivity Shapes

"Connectivity Shapes", Gumhold and Gotsman 01

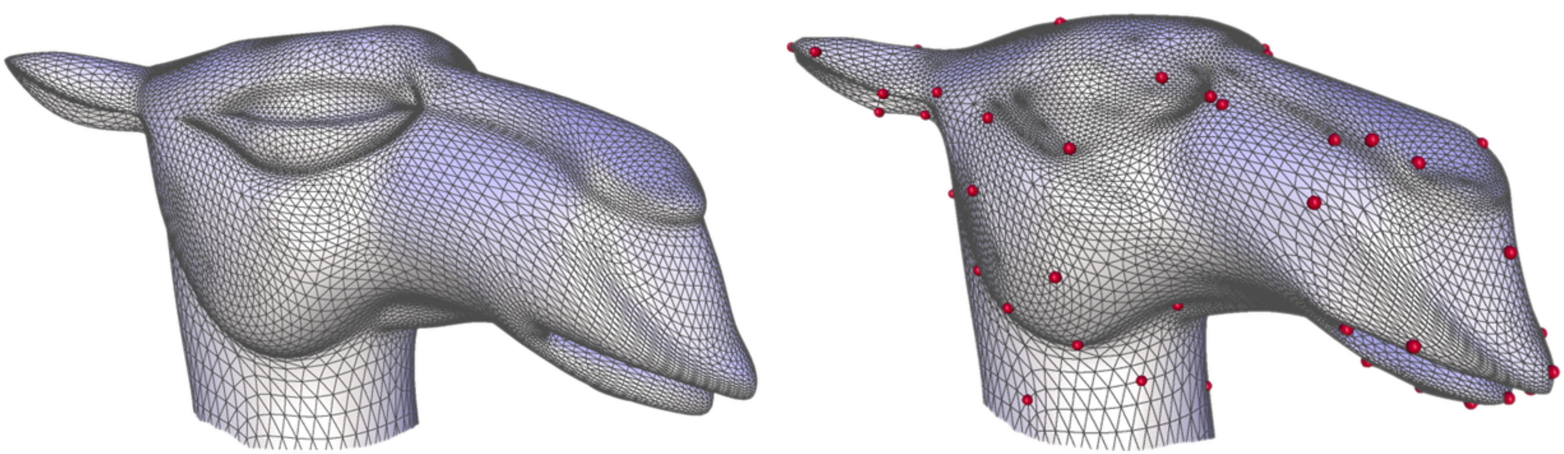
- Connectivity has geometric information in it
- Isenburg et al. showed how to get a shape from connectivity by assuming uniform edge length and smoothness
  - Non-linear optimization process to get shape from connectivity





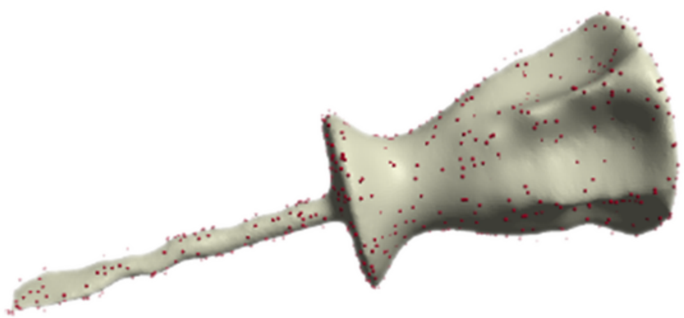
Least-squares Meshes

- Enrich the connectivity by sparse set of control points with geometry
- Solve a linear least-squares problem to reconstruct the geometry of all vertices (the approximated shape)

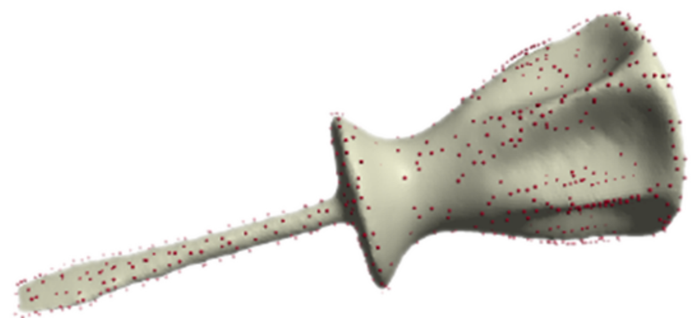


Selecting the control points

- Random selection
  - Faster, but less effective approximation
- Greedy approach
  - Place one-by-one at vertices with highest reconstruction error
  - Fast update procedure for the system inverse matrix



Random selection



Greedy approach

1000 control points

Results

Original camel
39074 vertices

100 control points

600 control points

1200 control points

3600 control points

Results

Original feline
49864 vertices

100 control points

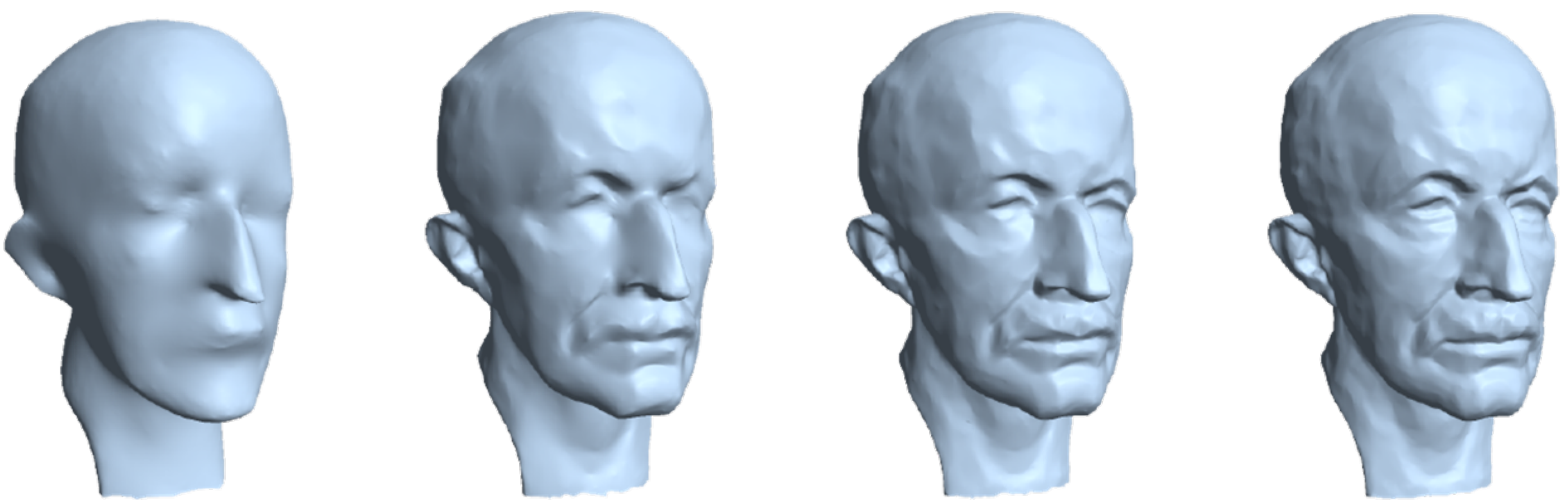
500 control points

4000 control points

9000 control points

Applications

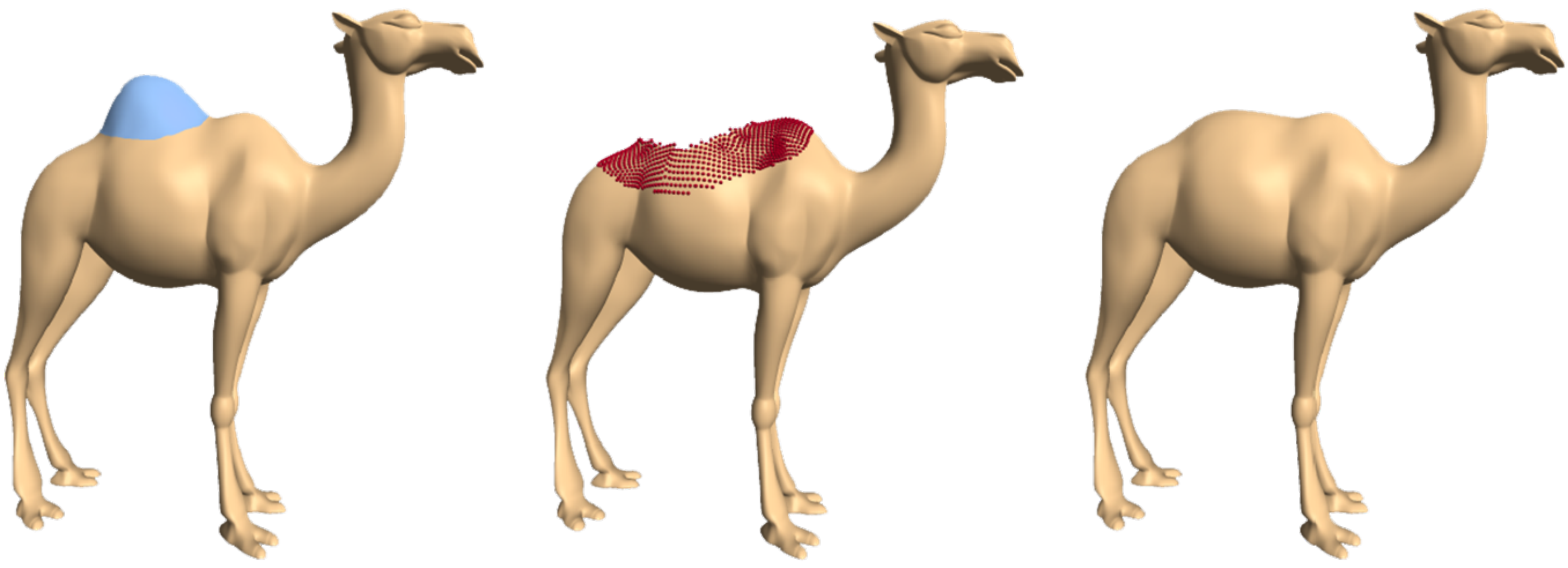
- Progressive geometry compression and streaming



100 control points | 1000 control points | 3000 control points | 10000 control points

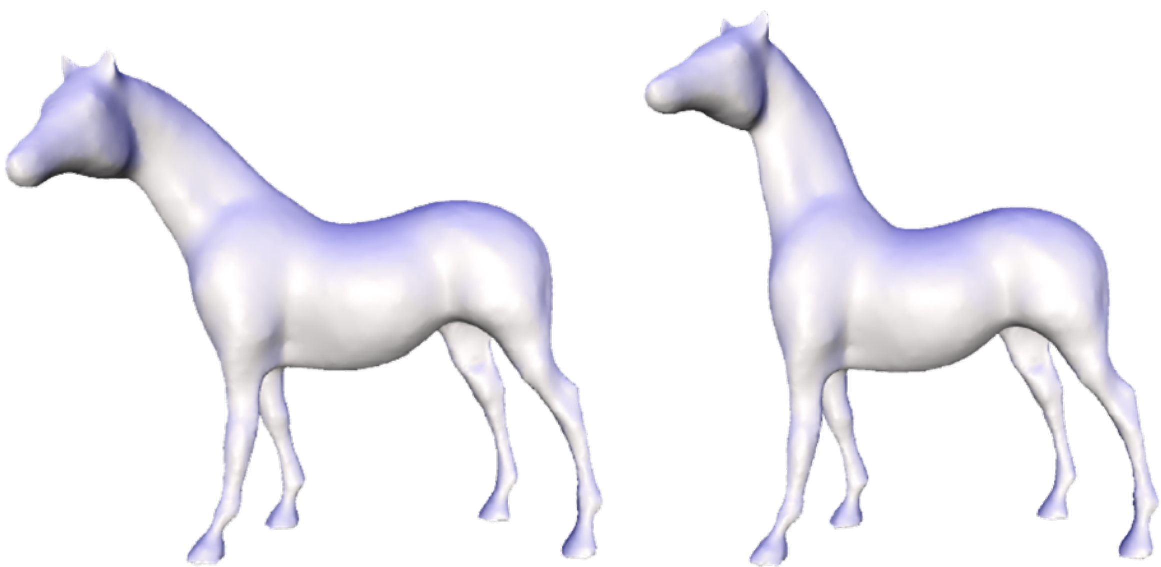
Applications

- Progressive geometry compression and streaming
- Hole filling



Applications

- Progressive geometry compression and streaming
- Hole filling
- Mesh editing



Differential Processing

- Local detail representation
- Representation with sparse matrices
- Efficient linear surface reconstruction

See:

[EG05 – Laplacian Mesh Processing]

Conclusions

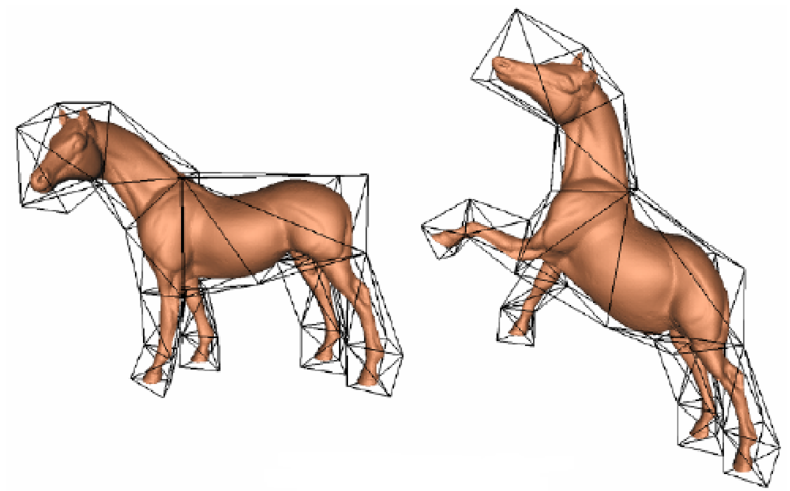
- Differential coordinates represent local details
- Good for applications that wish to preserve local details
  - shape approximation
  - shape editing
- Reconstruction by linear least-squares
  - smoothly distributes the error across the domain
  - reasonably efficient

Mesh Editing Papers

- FFD
  - [1986, Sederberg and Parry] Free-form deformation of solid geometric models
  - [1994, Lazarus et al.] Axial deformations: An intuitive deformation technique
  - [2005, Juet al.] Mean value coordinates for closed triangular meshes
- Multiresolution editing
  - [1995, Eck et al.] Multiresolutionanalysis of arbitrary meshes
  - [1998, Kobbelt et al.] Interactive multi-resolution modeling on arbitrary meshes
  - [2003, Botsch and Kobbelt] Multiresolution surface representation based on displacement volumes
- Differential editing
  - [2004, Yu et al.] Mesh editing with Poisson-based gradient field manipulation
  - [2004, Sorkineet al.] Laplacian surface editing
  - [2005, Lipman et al.] Linear rotation-invariant coordinates for meshes

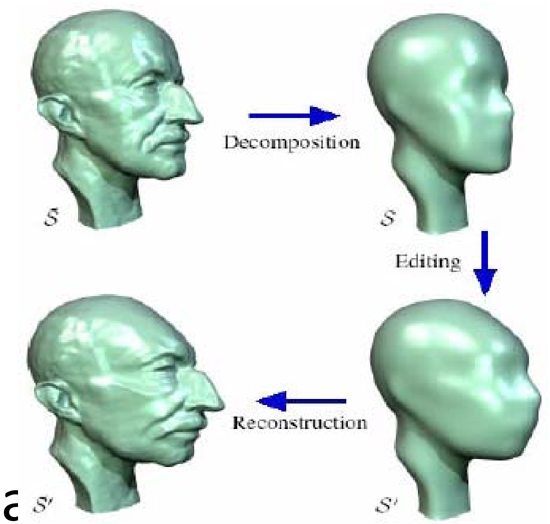
1. FFD

- Manipulating
  - Embedding object
- Pros
  - Simple and intuitive
- Cons
  - Hard to preserve details
  - Suitable for smooth objects
- Invariant variables
  - Parametric coordinates



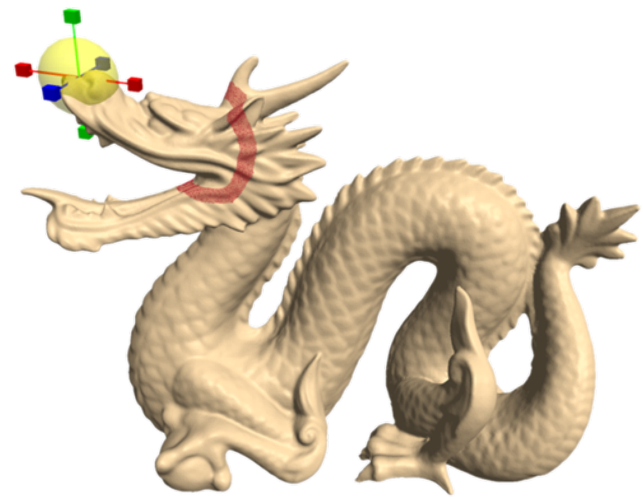
2. Multiresolution Editing

- Manipulating
  - Simplified model
- Pros
  - Preserving details, scalable
- Cons
  - Instable reconstruction for large deforma
  - Resampling problem
- Invariant variables
  - Detail information

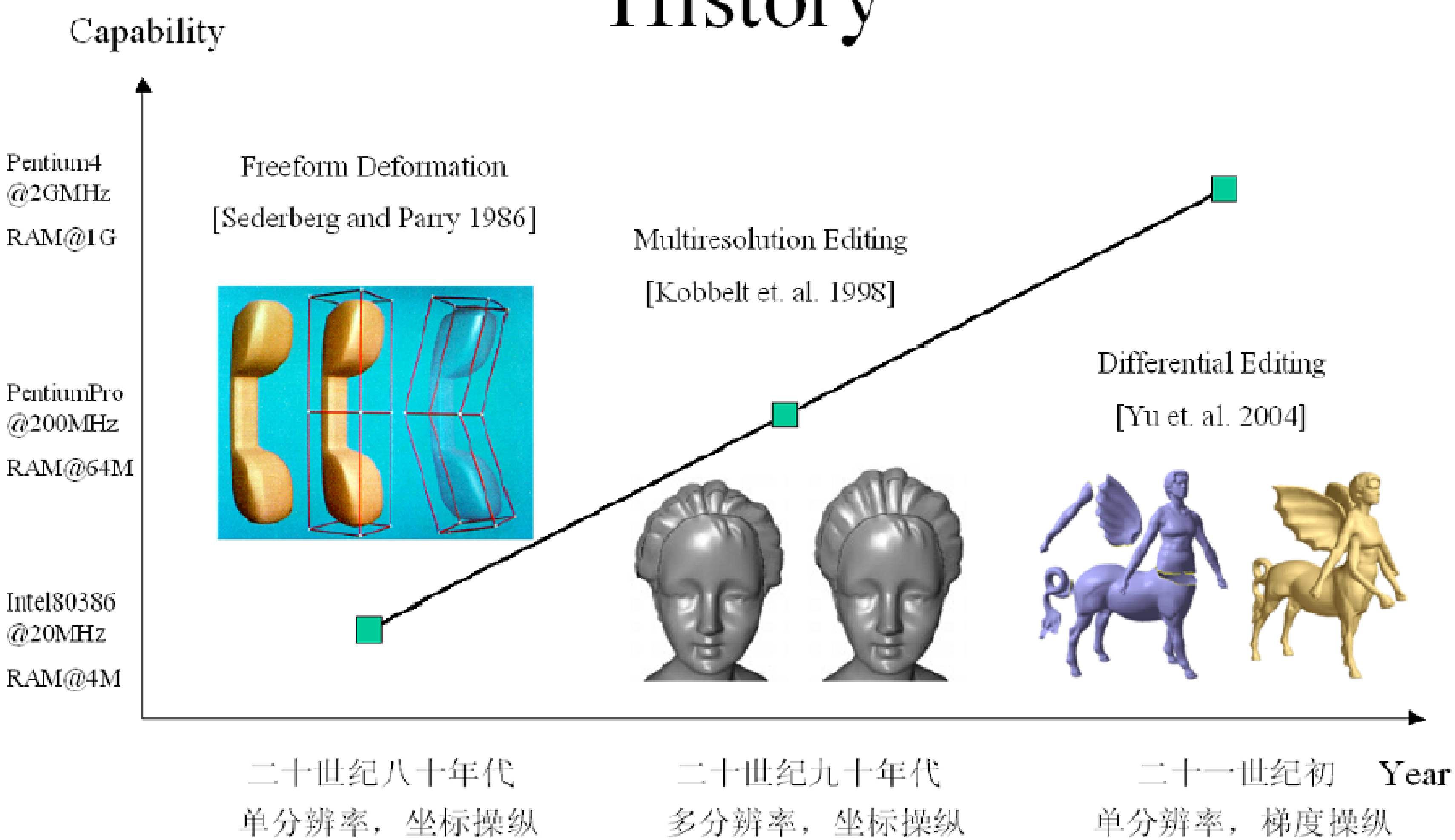


3. Differential Editing

- Manipulating
  - Handles
- Pros
  - Preserving details
  - Enhance user interaction
- Cons
  - Much computation cost
- Invariant variables
  - Differential information



History

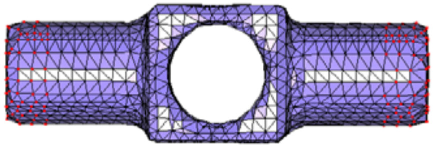


[Xu, 2006]

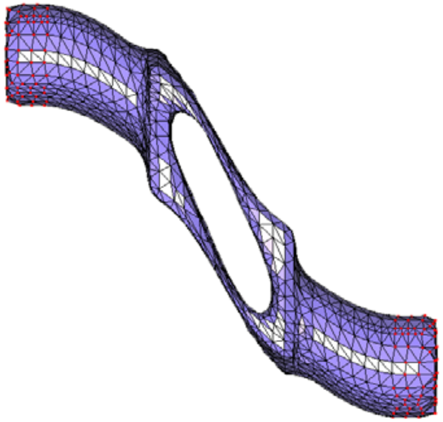
Dual Laplacian Editing

IEEE TVCG 2006

Laplacian Editing - Problems



- The LCs are encoded in global coordinate system
  - Local structures of deformed surface may be rotated or stretched
  - Minimizing changes from LCs of original mesh is not appropriate



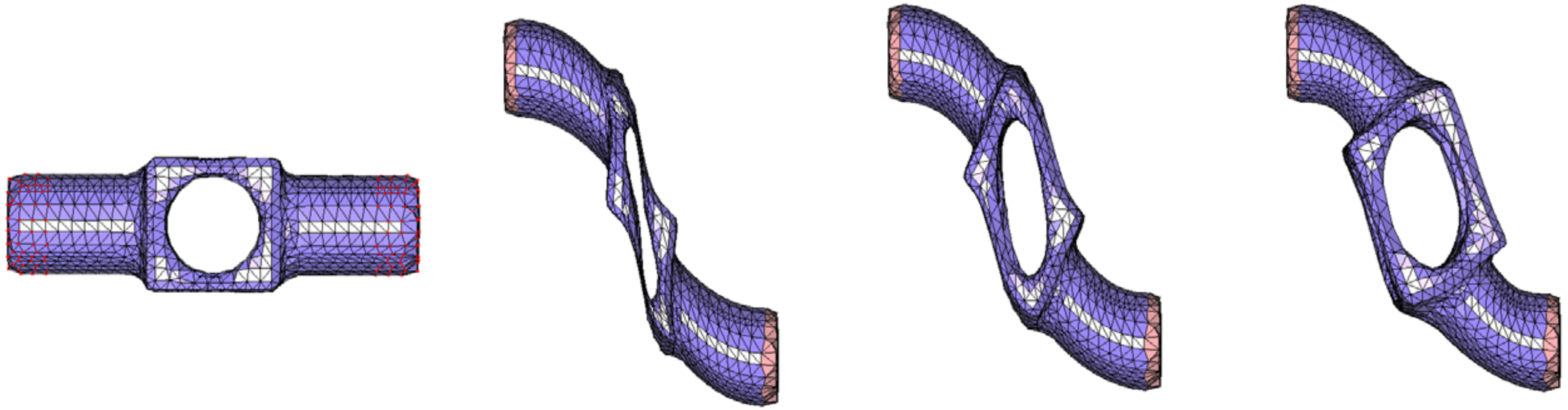
- The LCs should be properly reoriented

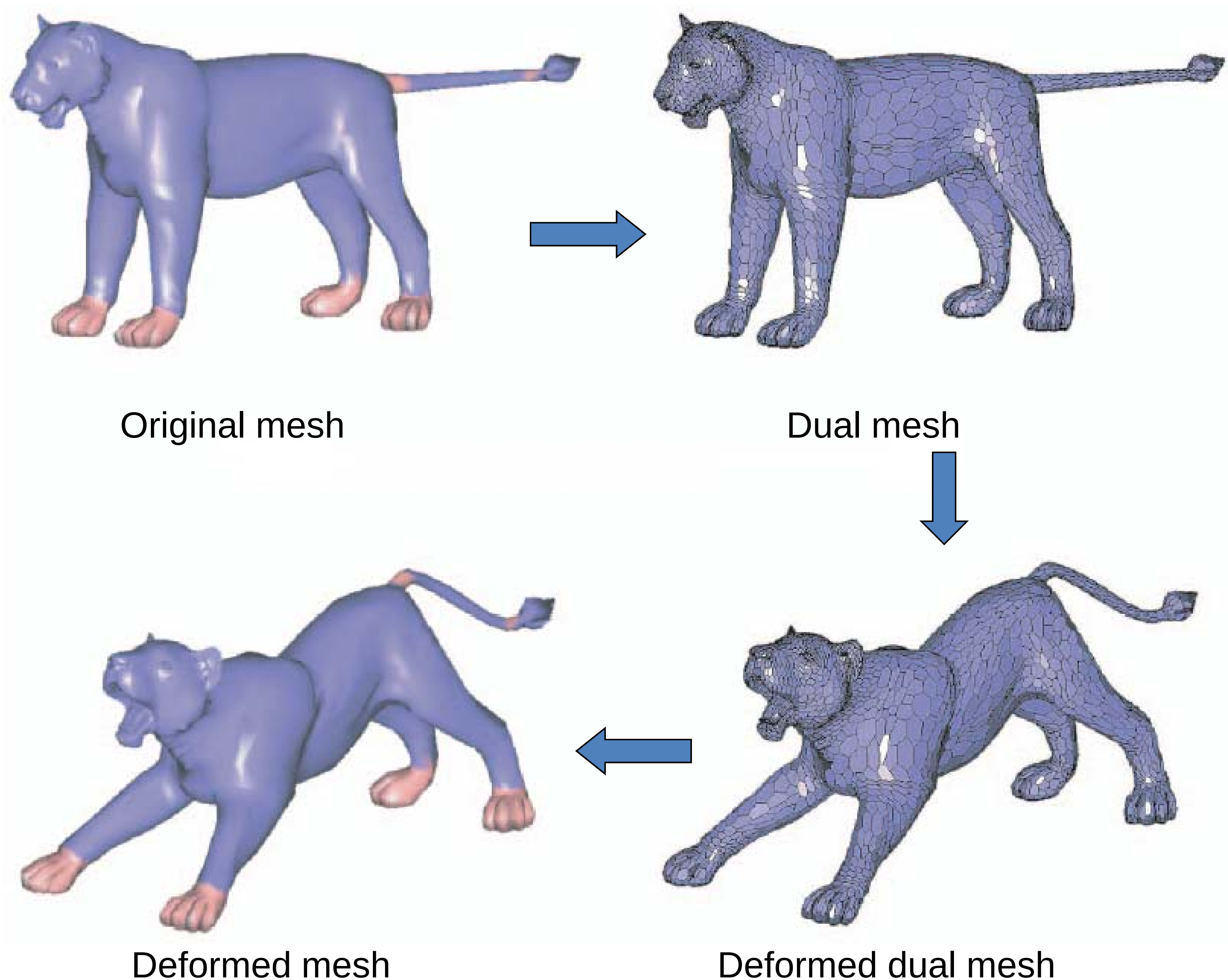
Observations

- The deformed mesh should have
  - Similar triangle shapes as the original mesh
    - Preserve parameterization information (i.e., shapes of local features)
    - Shape distortion causes undesired shearing and stretching
  - Similar local feature sizes as the original mesh
    - Preserve geometry information (i.e., sizes of local features)

Framework

- An iterative method
  - Refine vertex positions and LCs iteratively
  - Minimize parameterization error iteratively while keeping small geometry error





Original mesh
Dual mesh
Deformed mesh
Deformed dual mesh

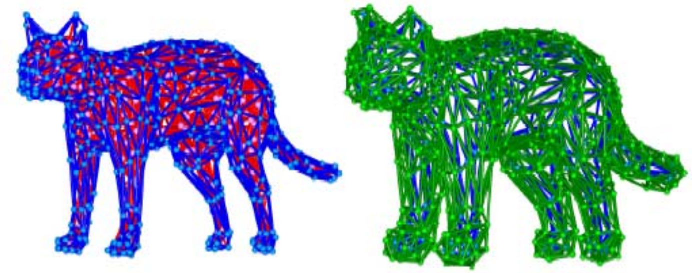
Volumetric Graph Laplacian (VGL)

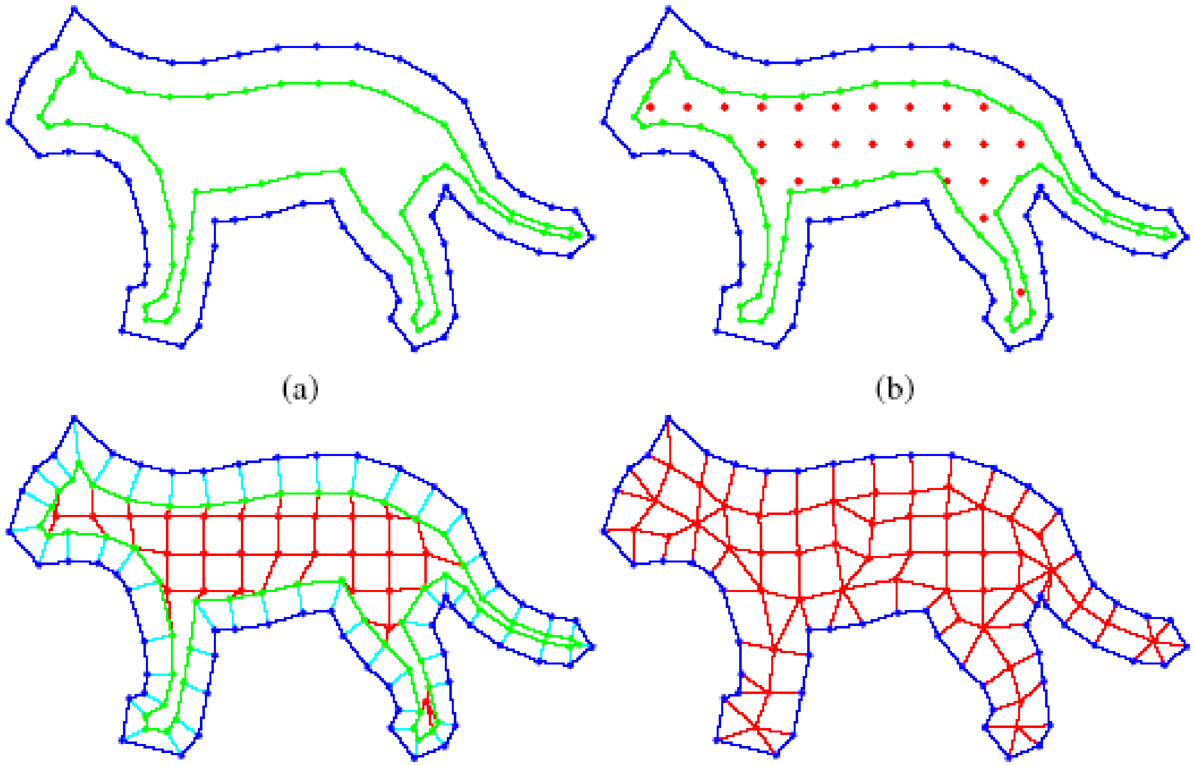
Siggraph 2005

3D Laplacian Editing

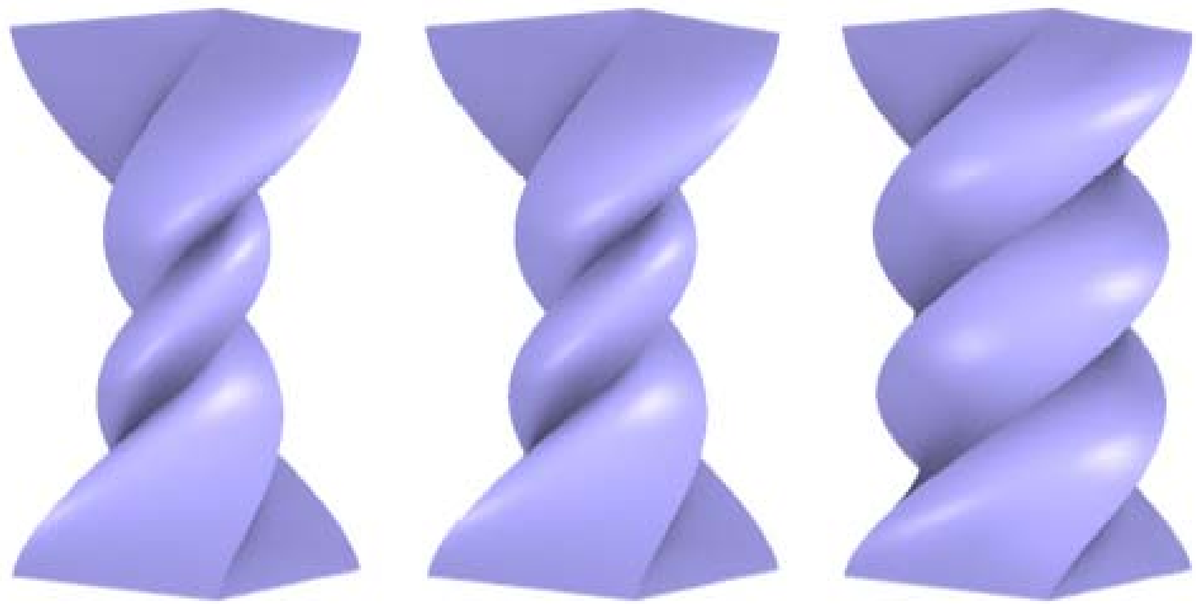
- Tetrahedral meshes
  - Hard to creat

Volumetric Gr





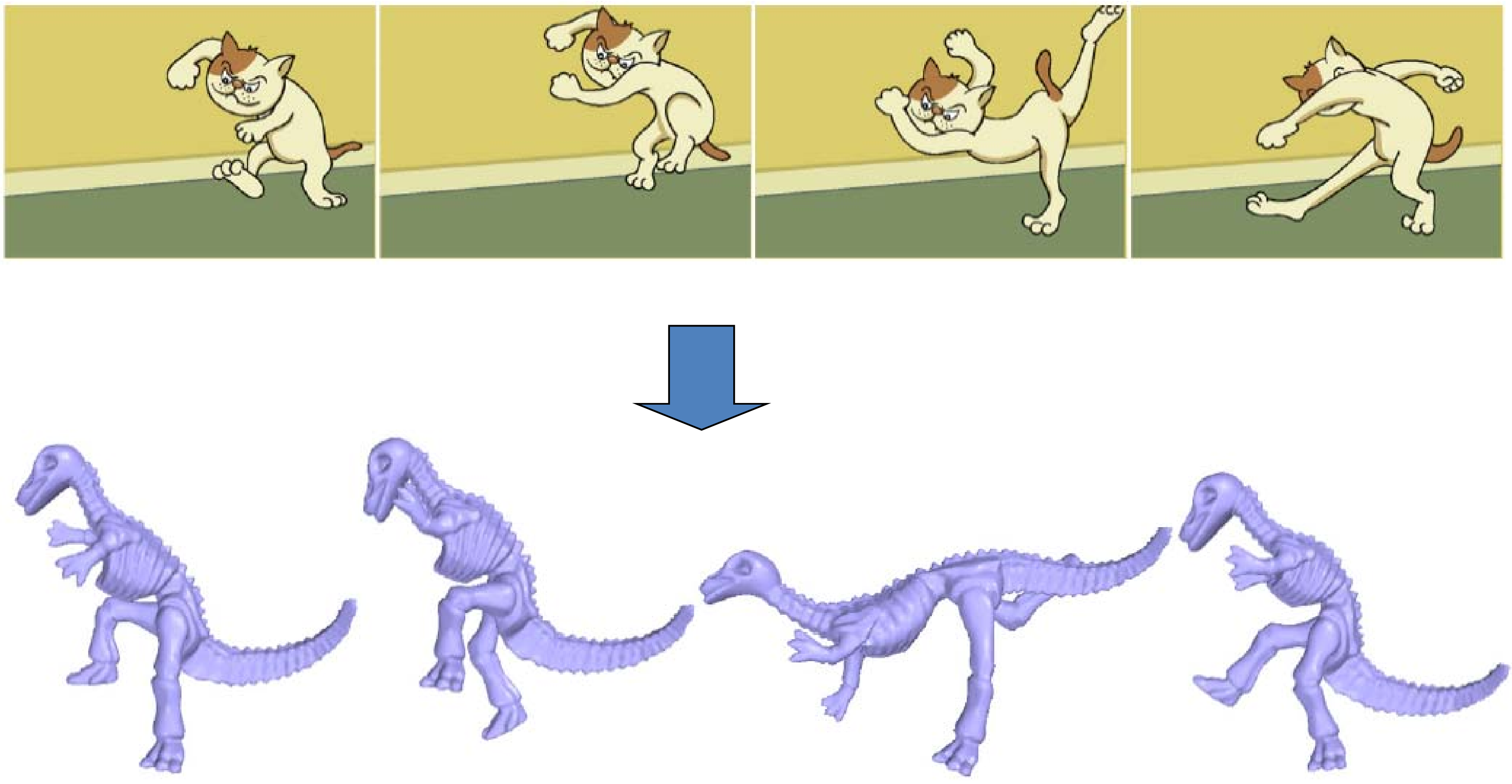
(a) (b) (c) (d)



(a) Laplacian surface (b) Poisson mesh (c) VGL

(a) Laplacian surface (b) Poisson mesh (c) VGL

Application



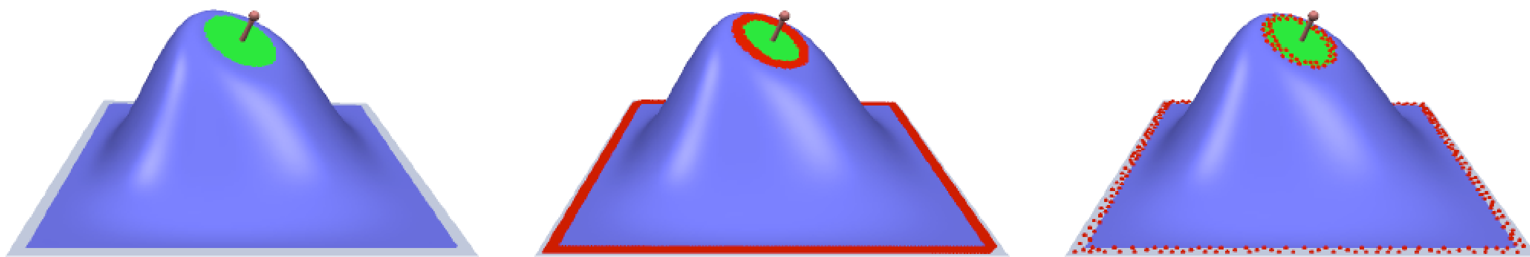
Other Approaches

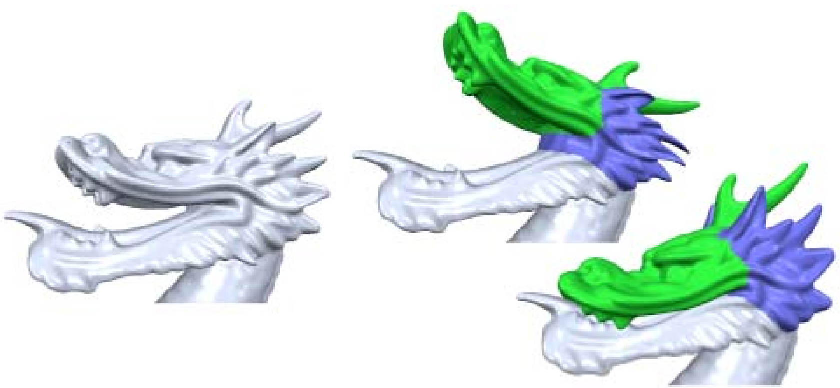
Skeleton-based Deformation

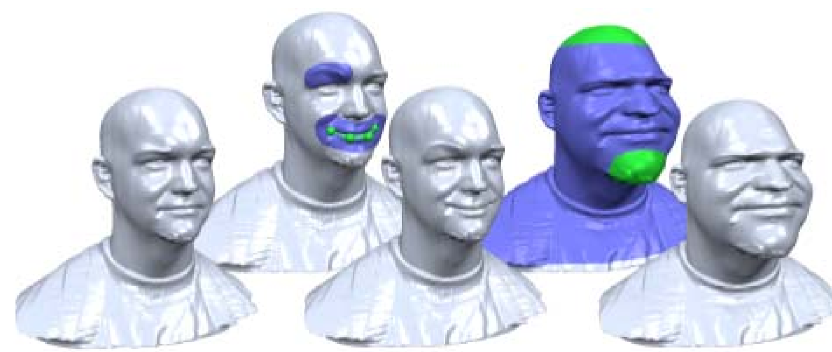
[2002]

RBF-based Editing

[2005]

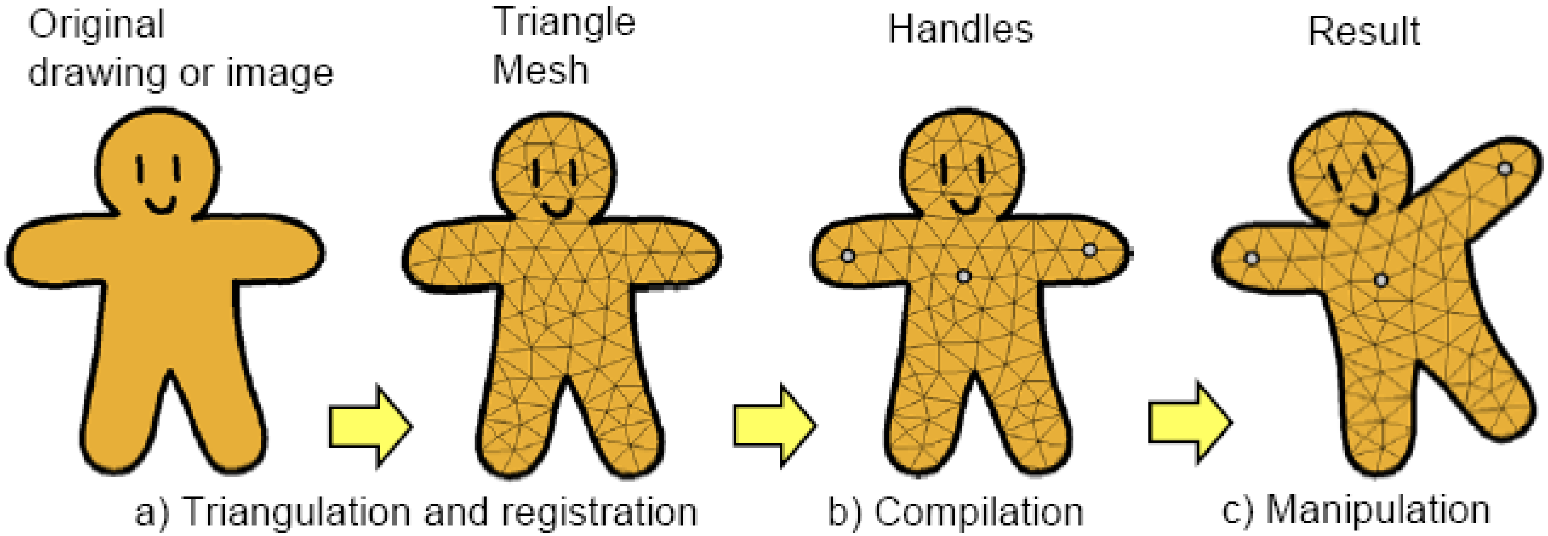


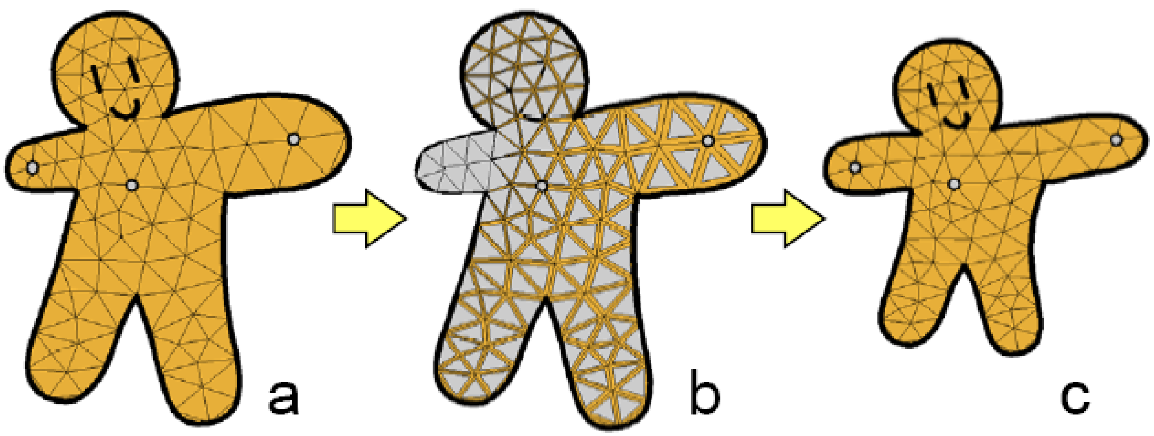




As-rigid-as Possible Deformation

Siggraph 2005


Original drawing or image
Triangle Mesh
Handles
Result
a) Triangulation and registration
b) Compilation
c) Manipulation


a
b
c

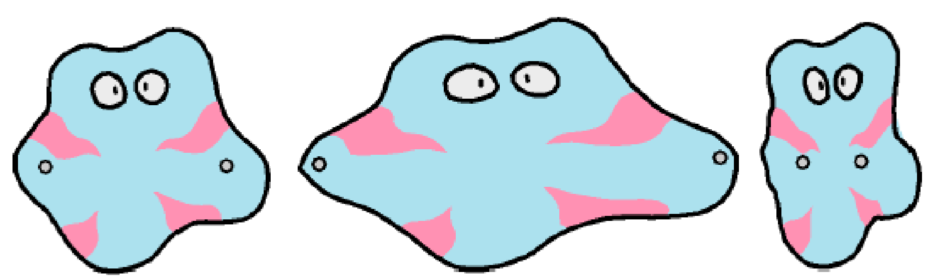
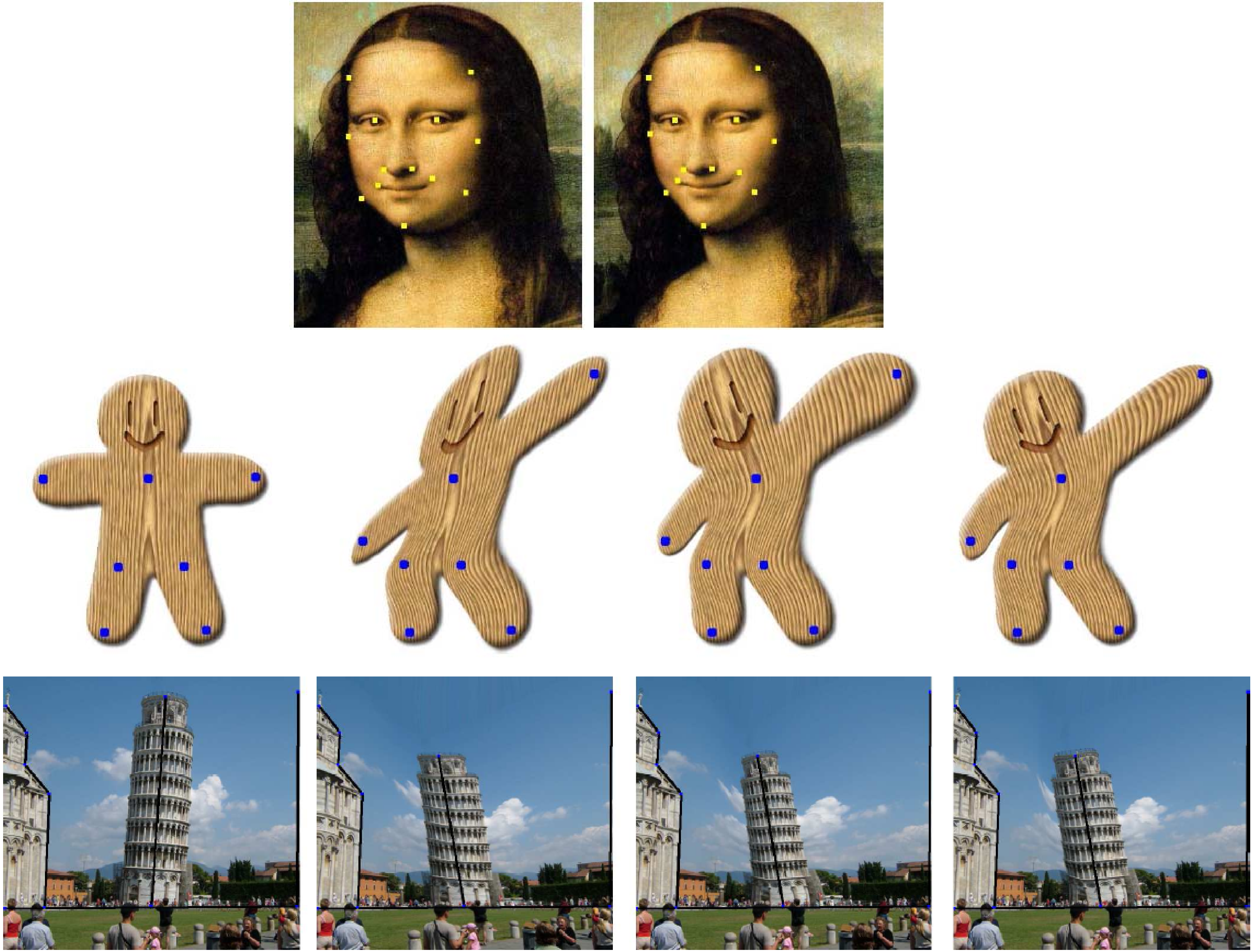


Image Deformation Using Moving Least Squares

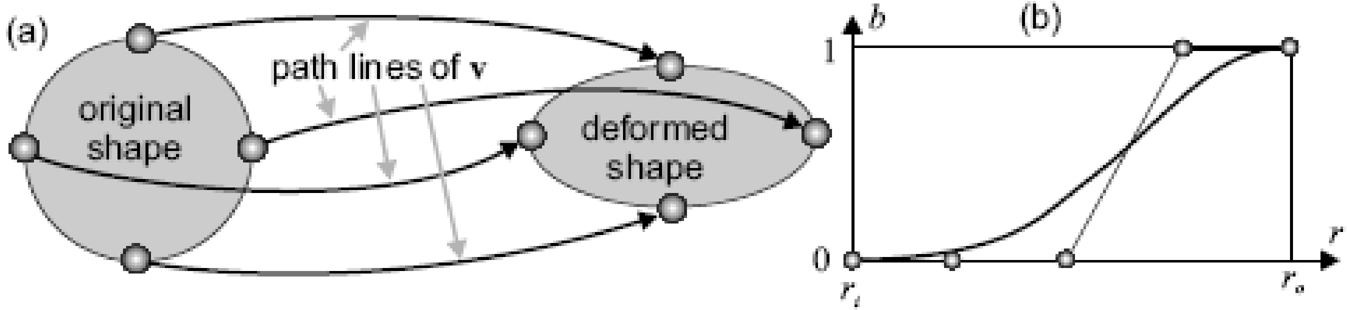
Siggraph 2006



Vector Field Based Shape Deformations

Siggraph 2006

Basic Model



Moving vertex along the deformation orbit – defined by the *path lines* of a vector field v.

Path Line of Vector Field

t_0 t

Given a time-dependent vector field **V**(**X**, t), a *Path Line* in space is **X**(t):

$$\frac{d}{dt}\mathrm{X}(t) = V\left(\mathrm{X}(t), t\right), \qquad \mathrm{X}(t_0) = \mathrm{X}_0$$

OR

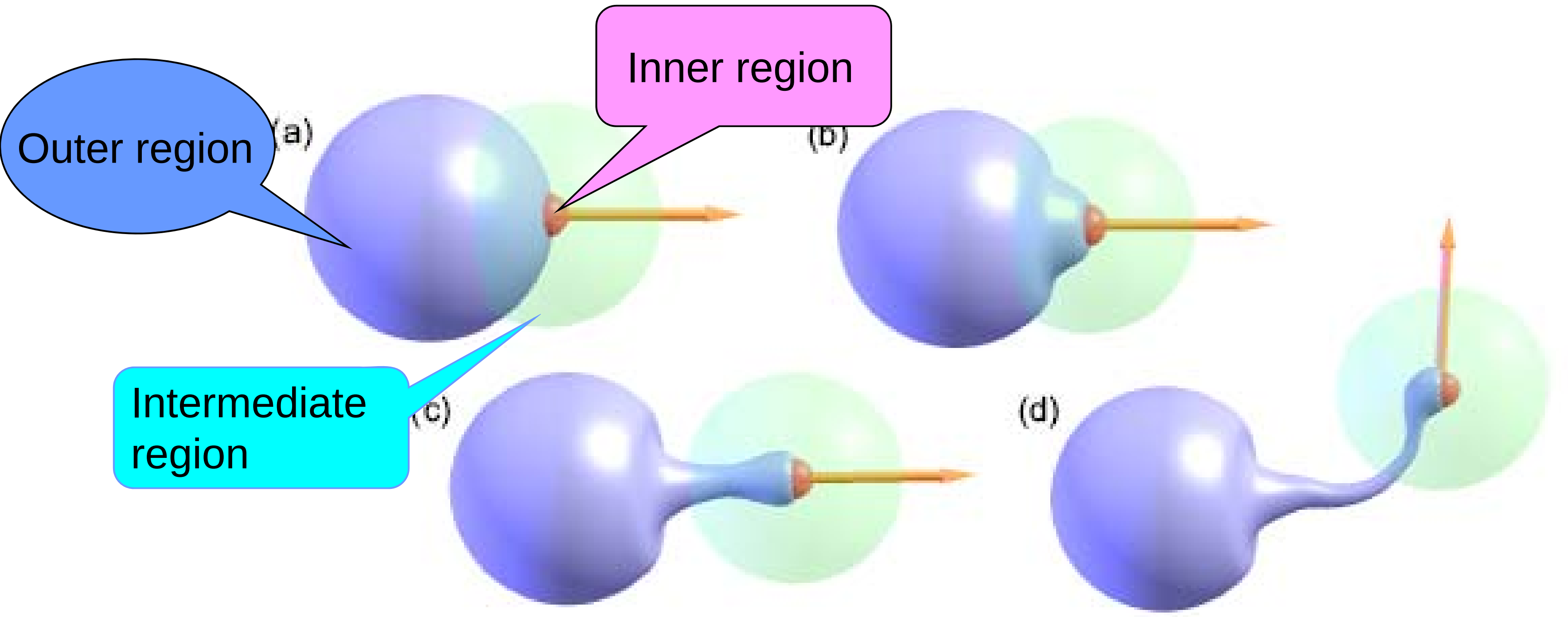
$$\mathrm{X}(t) = \mathrm{X}_0 + \int_{t_0}^{t} V(\mathrm{X}(s), s)\, ds$$

Vector Field Selection

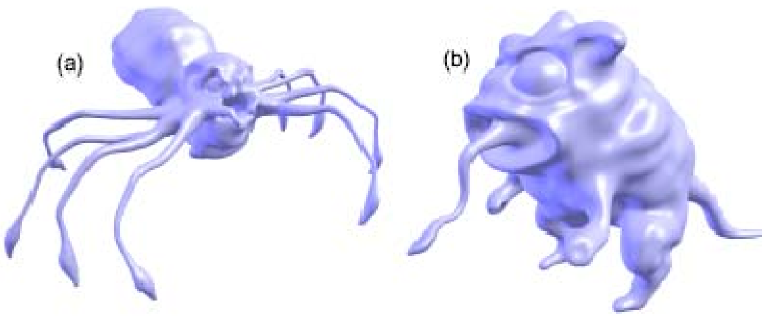
- Deformation Request:
  - No self-intersection
  - Volume-preserving
  - Details-preserving
  - Smoothness of shape in deformation
- Divergence-free Vector Field: ***v***=(V_1, V_2, V_3)

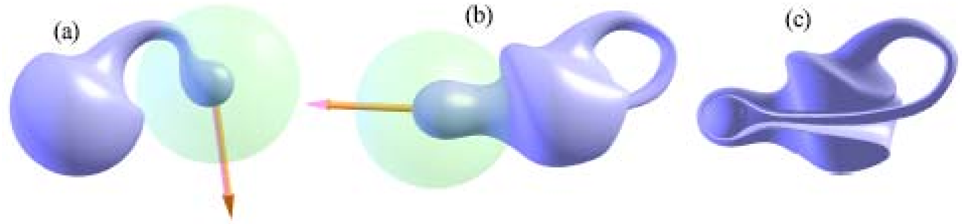
$$div\,\mathbf{V} = \frac{\partial V_1}{\partial x} + \frac{\partial V_2}{\partial y} + \frac{\partial V_3}{\partial z} = 0$$

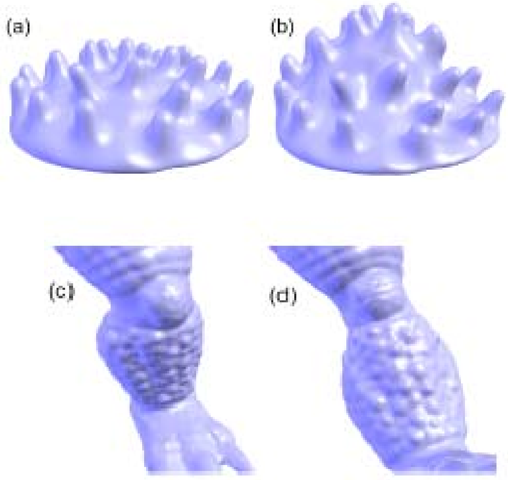
Piecewise Field for Deformation

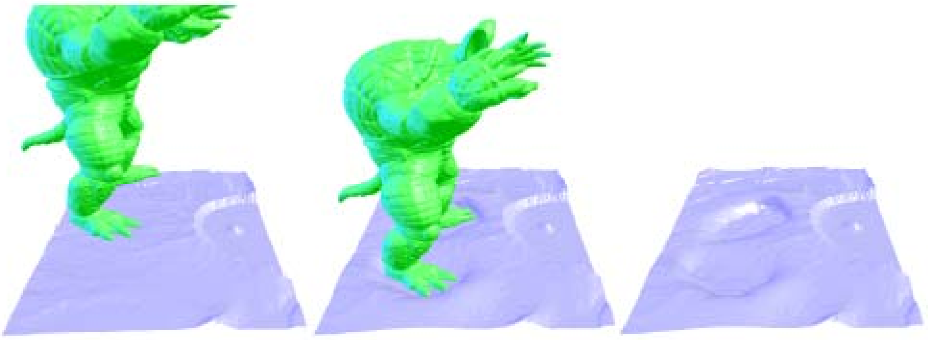


Examples



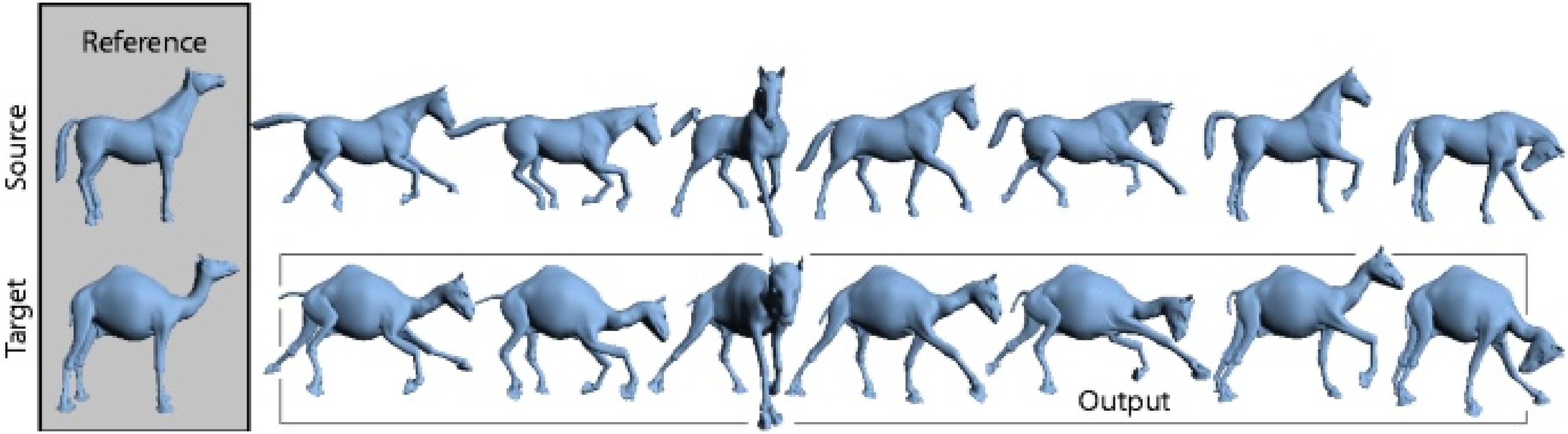






Deformation Transfer

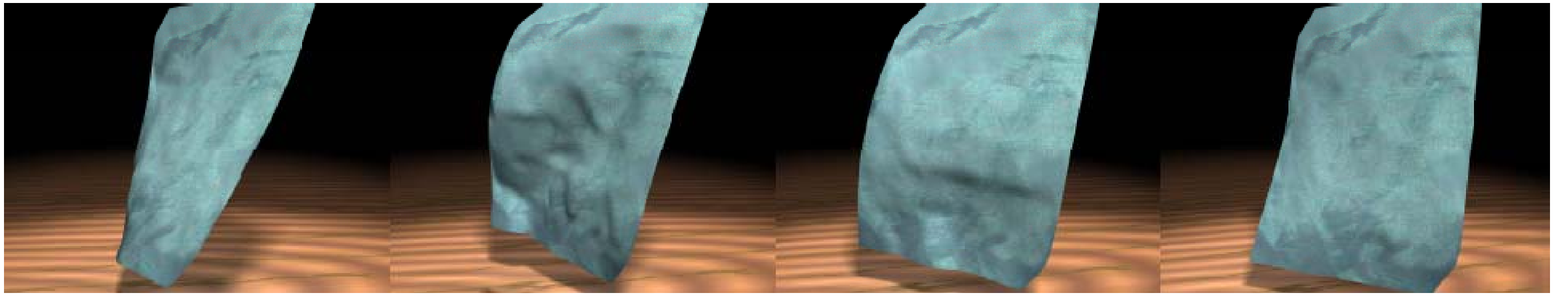
Siggraph 2004



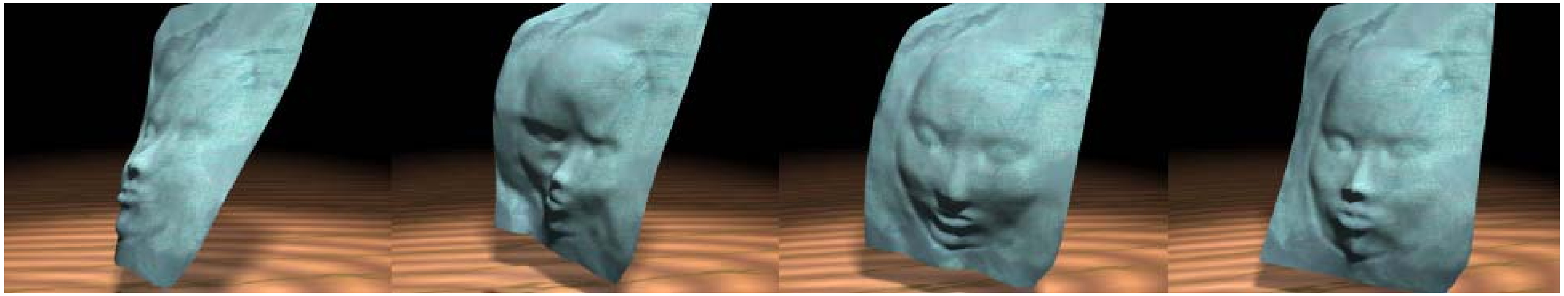
Editing Arbitrary Deforming Surface Animations

Siggraph 2006

Deforming Surface



Editing Surface



Discussions