

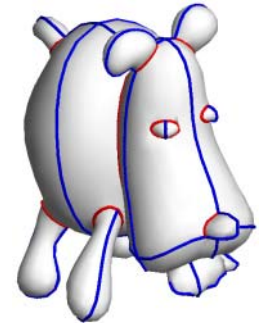
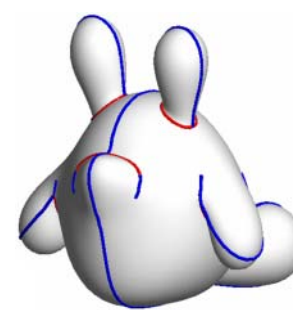
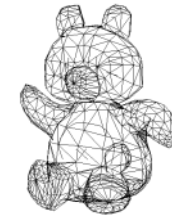
CS 523: Computer Graphics, Spring 2009

Shape Modeling

Sketch-based interfaces and
algorithms for shape creation
and editing




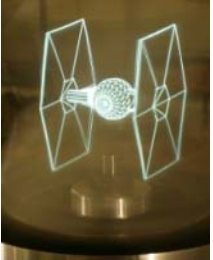
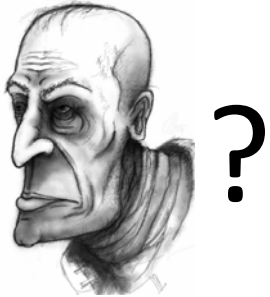

Overview

- **Teddy**
[Igarashi et al. 1999]
- **Sketch based mesh editing** [Nealen et al 2005]
- **FiberMesh**
[Nealen et al. 2007]
- **SilSketch**
[Zimmermann et al. 2008]






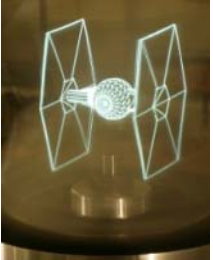
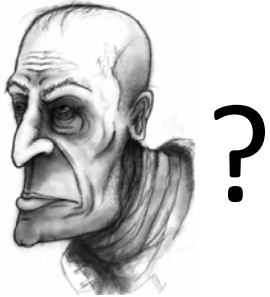

Primary Goal

- Find a possible solution to the human “video out” problem

	Machine	Human
Video in		
Video out	 	 

The Main Question

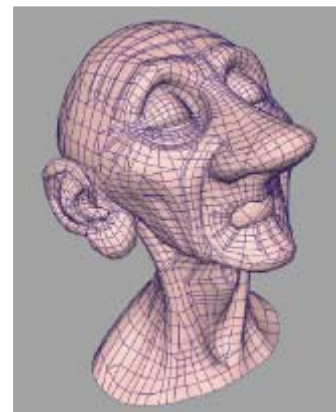
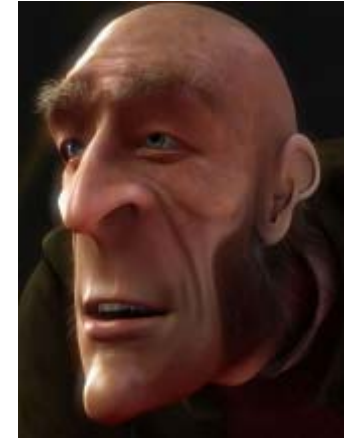
- How can we convey **a mental model of shape** to a digital computer ?

	Machine	Human
Video in		
Video out	 	 

Motivation

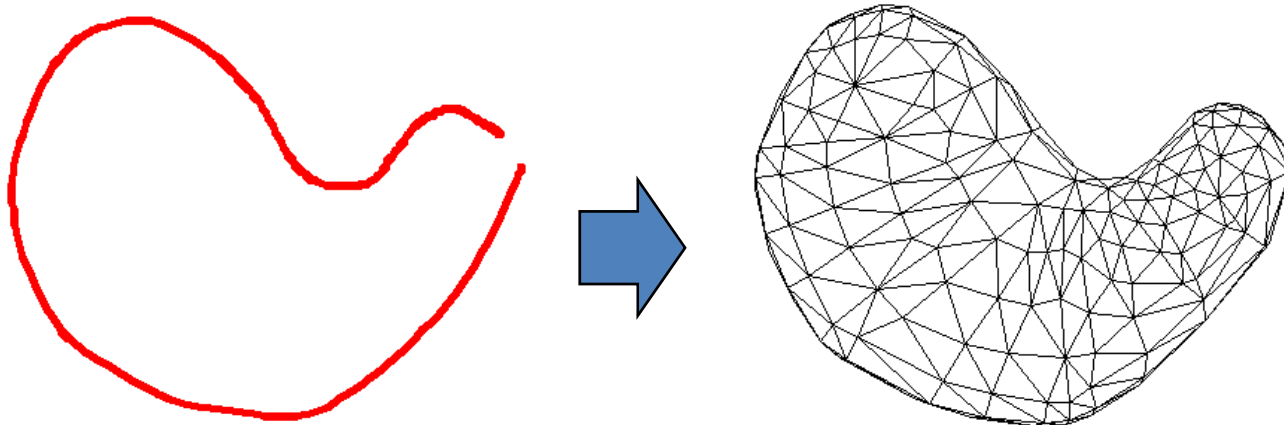
Model Creation

- Digital 3D model **creation from scratch** is hard
- Common workflow(s)
 - Create clay model + scan
 - Laser scanner
 - Position tracking
 - Create coarse 3D model + refine
 - Subdivision surfaces
 - Parametric patches
 - Detail maps



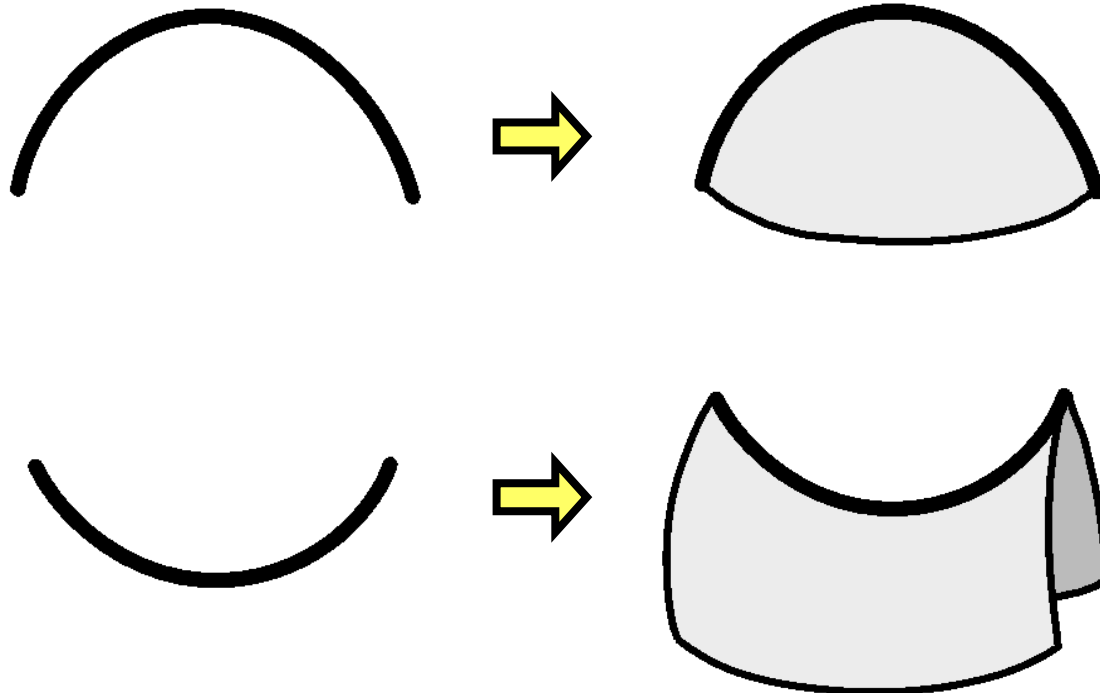
Problem

- How to construct 3D surface from 2D sketch?
 - Infer the missing depth information
 - Construct a discrete surface
 - What rule(s) to use?



Background

- Study on Human perception of 3D shape from 2D drawings [Hoffman 2004]



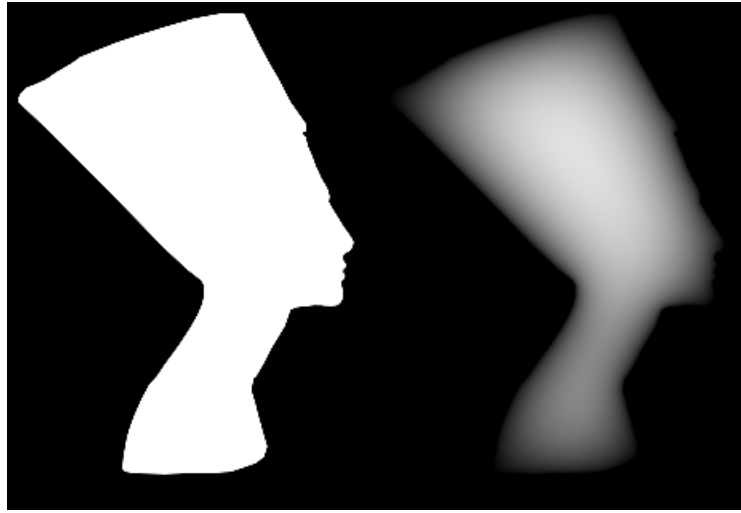
Specific Algorithms

- Level-set method [Williams 91]
- Heuristic mesh construction [Igarashi 99]
- Volumetric construction [Owada 03]
- Convolution surface [Alexe 05]
- Blob tree (Implicit surface) [Schmidt 05]
- Mass-spring system [Karpenko 06]
- Optimization [Nealen 07]

Level-set Method

[Williams 91]

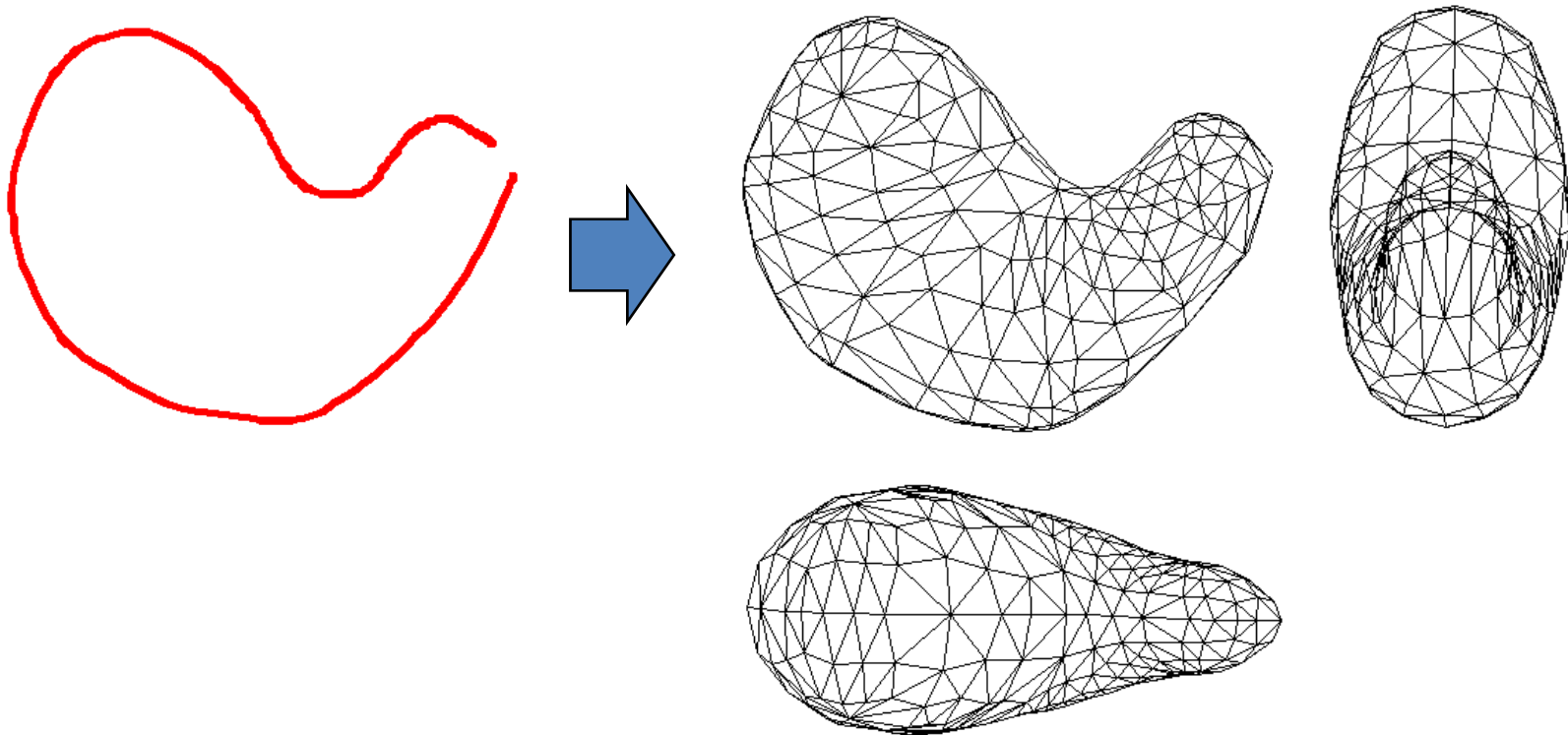
- Compute implicit representation of the contour



Teddy

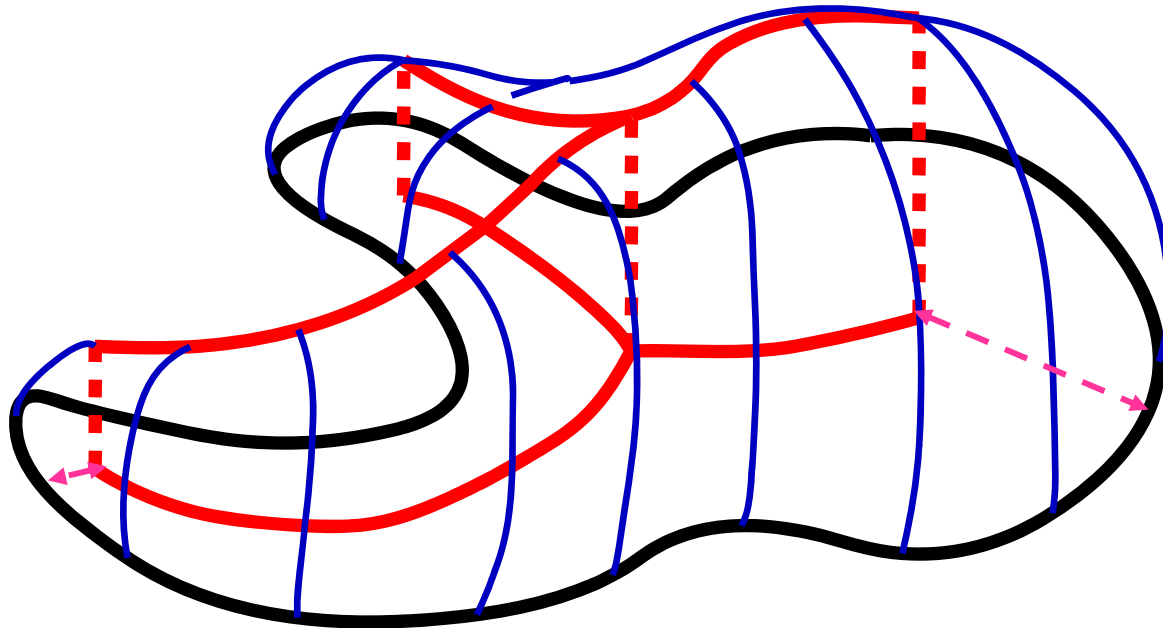
Heuristic Mesh Construction

Teddy



Heuristic Mesh Construction

Teddy



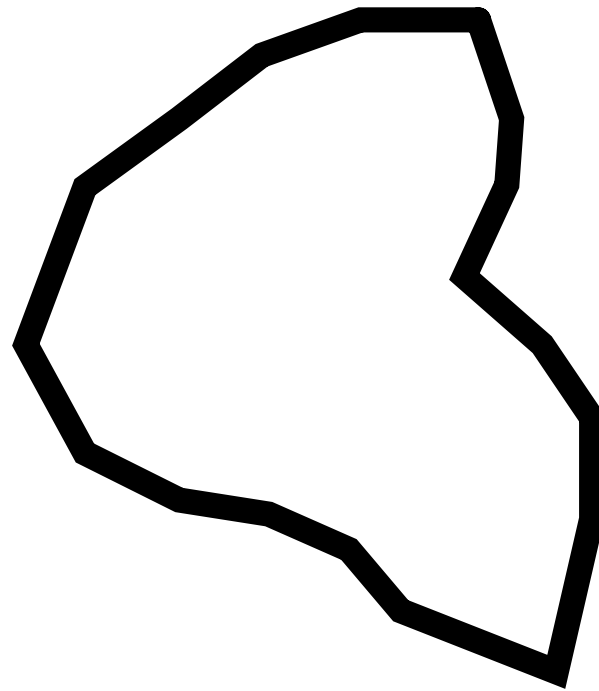
1. Find axes

2. Elevate axes

3. Wrap polygon and axes

Heuristic Mesh Construction

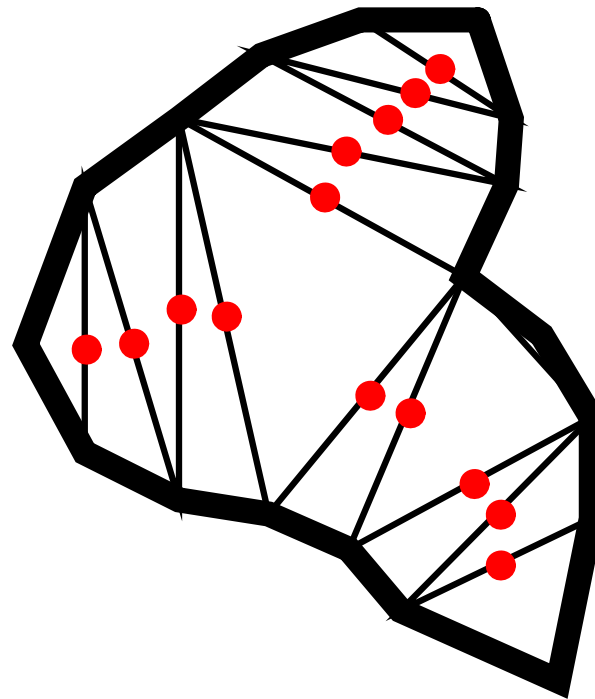
Teddy



Input 2D polygon

Heuristic Mesh Construction

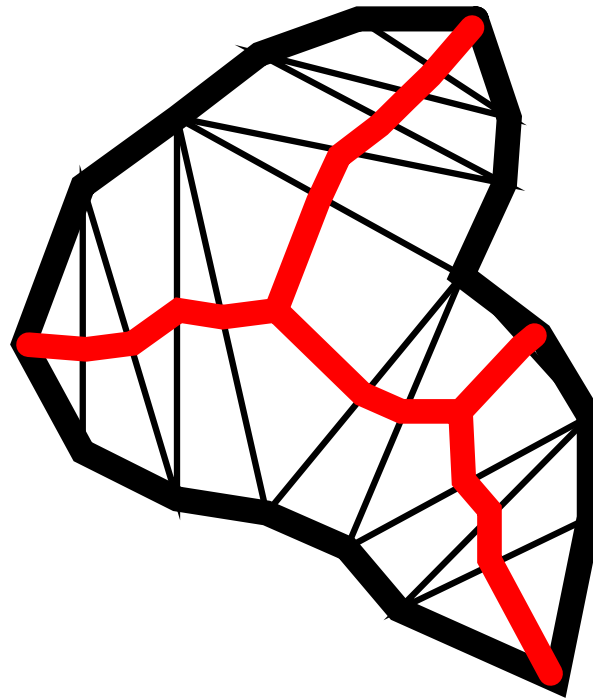
Teddy



Constrained Delaunay Triangulation

Heuristic Mesh Construction

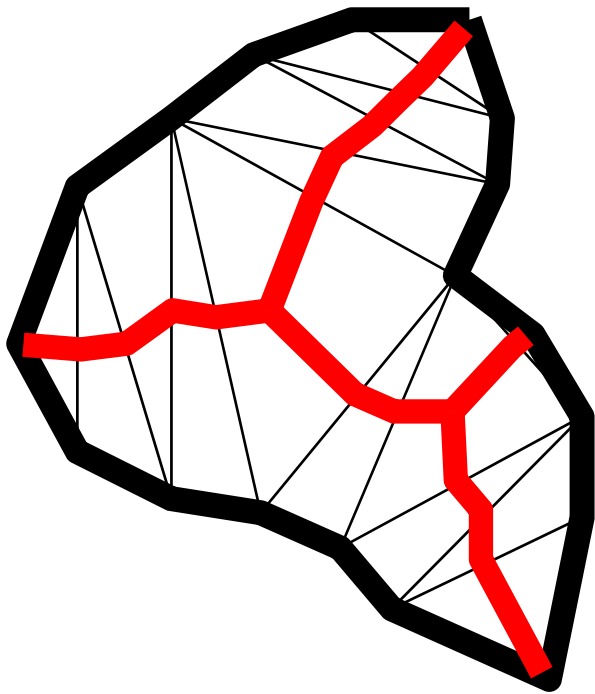
Teddy



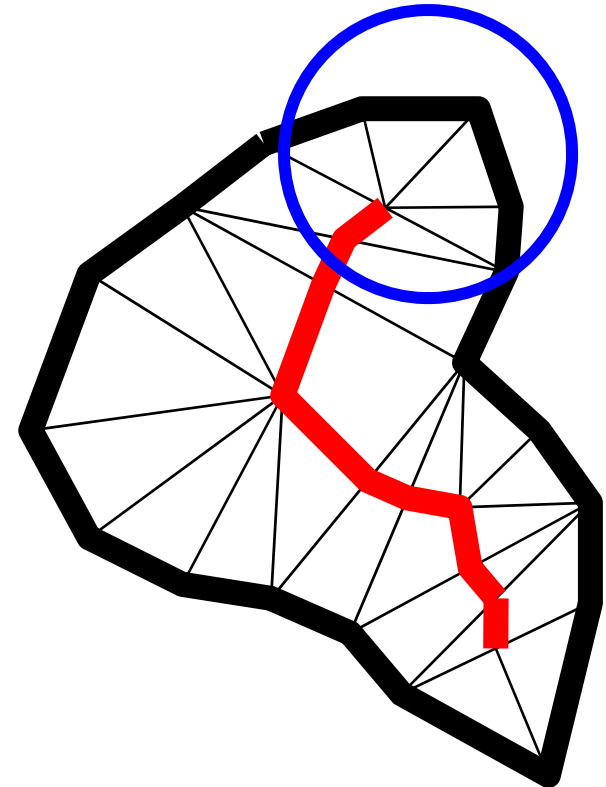
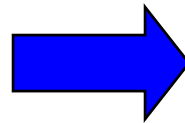
Discrete chordal axis [Prasad 97]

Heuristic Mesh Construction

Teddy



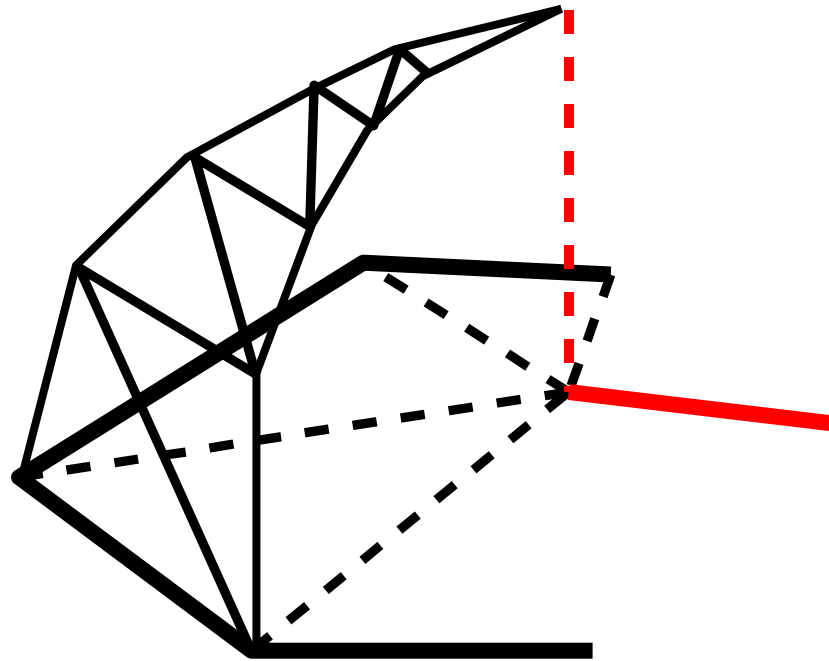
Before trimming



After trimming

Heuristic Mesh Construction

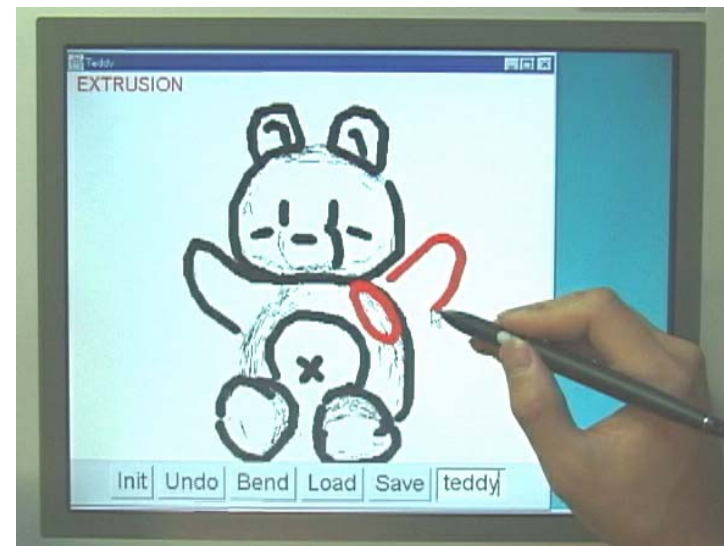
Teddy



Lift the axes, put quarter ovals on the internal edges, and generate mesh.

Teddy

- Results



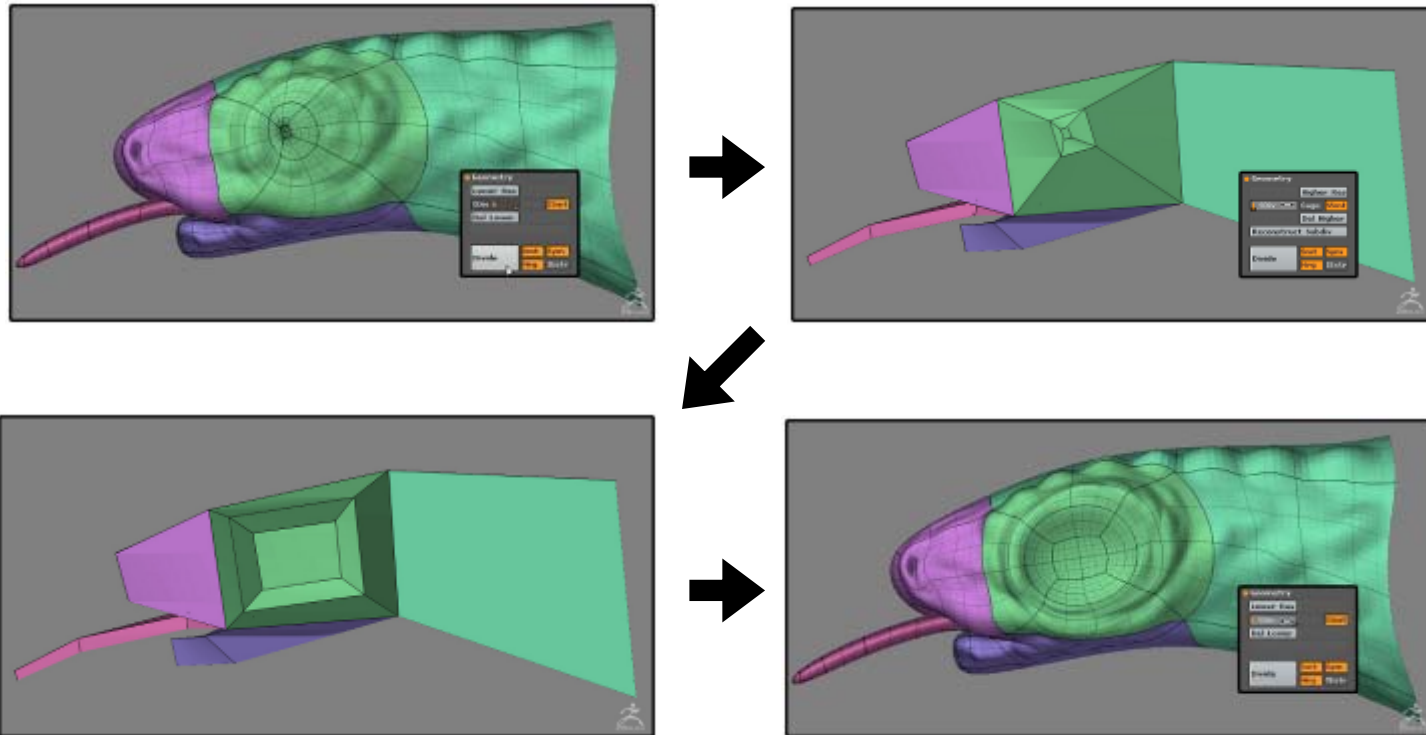
- Fast and first of its kind
- Meshes are of low quality

Sketch-based mesh editing

Motivation

Model **Modification**

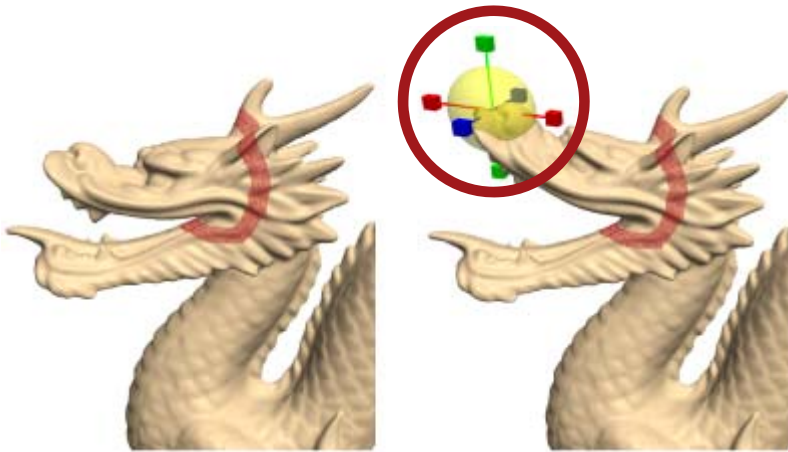
- Digital 3D model **modification** is nontrivial
- Multiresolution mesh editing /w subdivision



Motivation

Model **Modification**

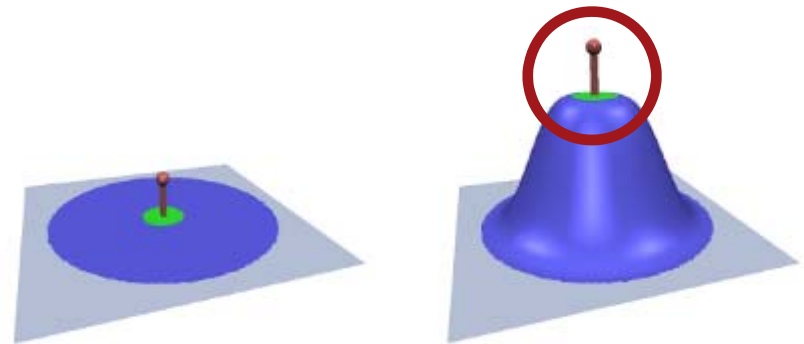
- Digital 3D model **modification** is nontrivial
- Laplacian / Poisson mesh editing /w handle



[Sorkine et al. 2004]

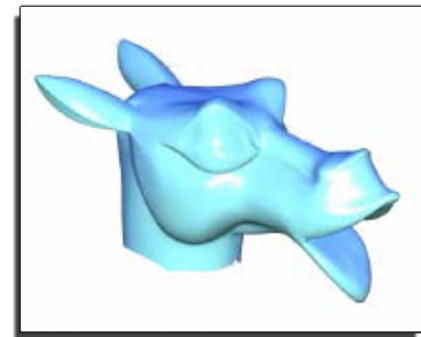
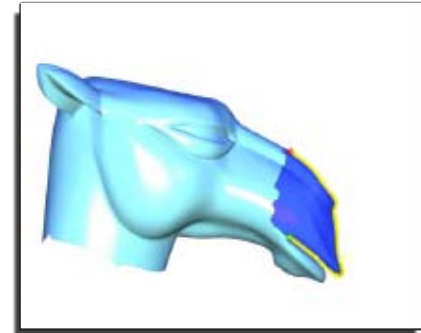
Affine Handle

[Botsch and Kobbelt 2004]



Ideas and Contributions

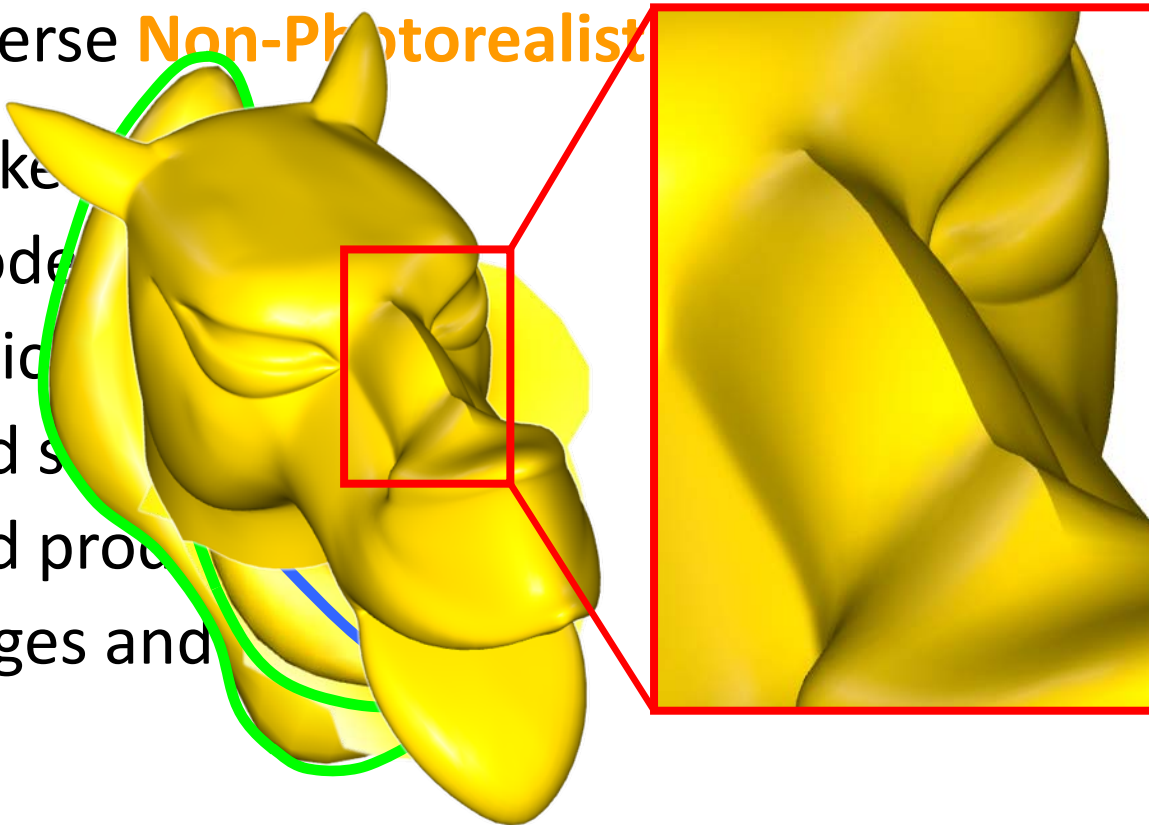
- **A sketch-based interface...**
 - Feature modification
(object-space silhouettes)
 - Feature creation
(sharp features, ridges, ravines)



- **... For detail-preserving mesh editing**
 - Adjust remaining geometry around the modified/created feature such that shape characteristics are preserved

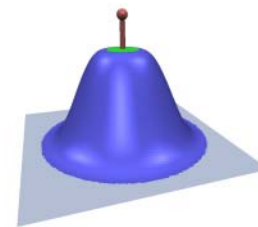
Ideas and Contributions

- Silhouette sketching
- Sketching a shape can be interpreted as inverse **Non-Photorealistic**
- A sketching model which and s and prod ridges and

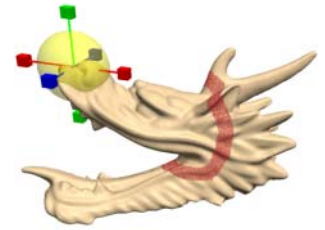


Inspiration and Motivation

- The (affine) handle metaphor
 - Used in (almost) every editing tool
 - Nice, but can be unintuitive for specific editing tasks



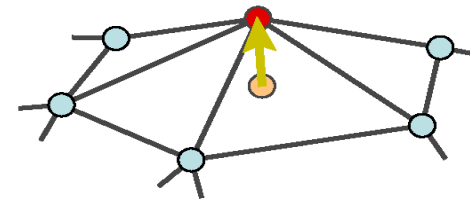
[Botsch and Kobbelt 04]



[Sorkine et al. 04]

- Laplacian Mesh Editing

- Preserve local detail **after** imposing editing constraints

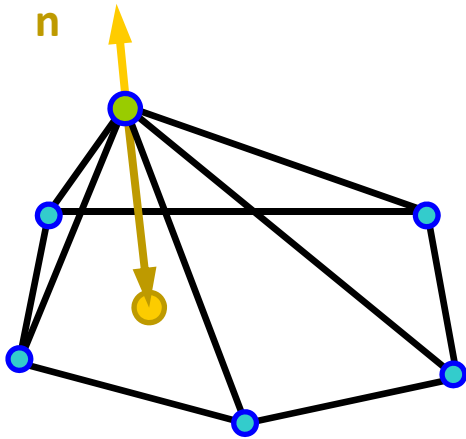


$$\delta_i = \frac{1}{d_i} \sum_{j \in N(i)} (\mathbf{v}_i - \mathbf{v}_j)$$

[Sorkine et al. 04] [Zhou et al. 05]

Mesh Modeling Framework

- Discrete Laplacians



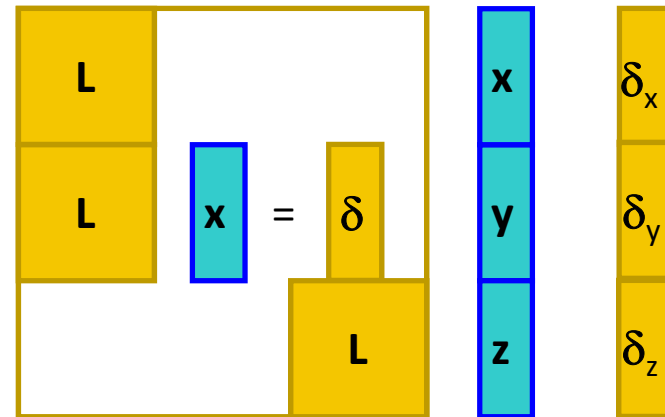
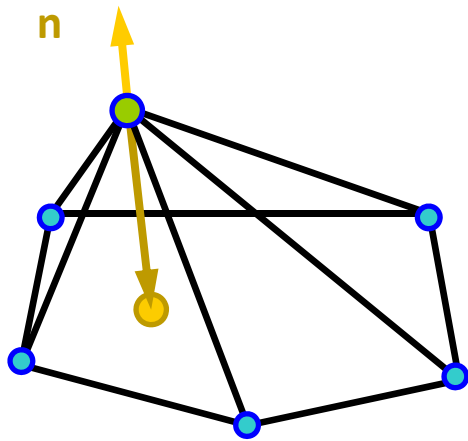
$$\mathbf{L} \mathbf{x} = \boldsymbol{\delta}$$

$$\boldsymbol{\delta}_i = \mathbf{x}_i - \frac{1}{\sum_{(i,j) \in E} w_{ij}} \sum_{(i,j) \in E} w_{ij} \mathbf{x}_j$$

$$\boldsymbol{\delta}_{\text{cotangent}} : w_{ij} = \cot \alpha_{ij} + \cot \beta_{ij}$$

Mesh Modeling Framework

- Surface reconstruction

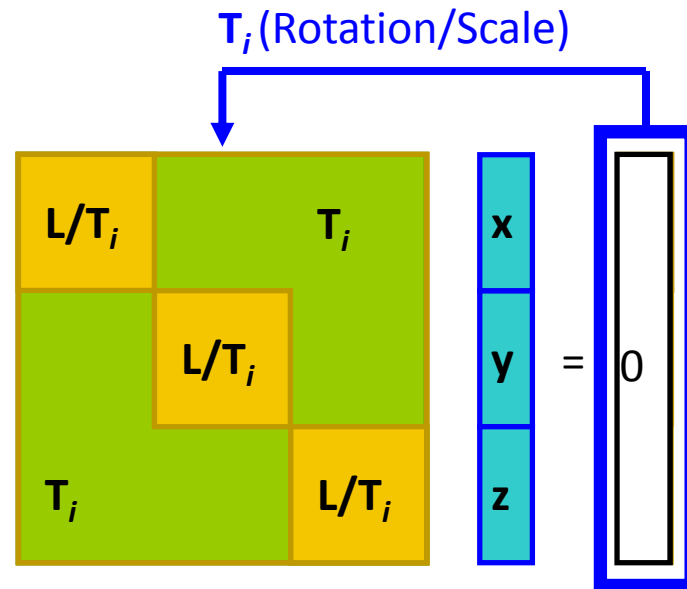
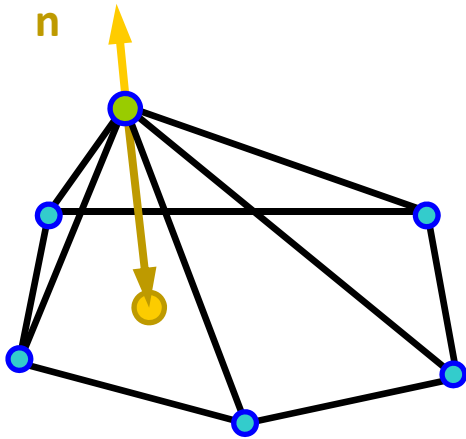


$$\delta_i = \mathbf{x}_i - \frac{1}{\sum_{(i,j) \in E} w_{ij}} \sum_{(i,j) \in E} w_{ij} \mathbf{x}_j$$

$$\delta_{\text{cotangent}} : w_{ij} = \cot \alpha_{ij} + \cot \beta_{ij}$$

Mesh Modeling Framework

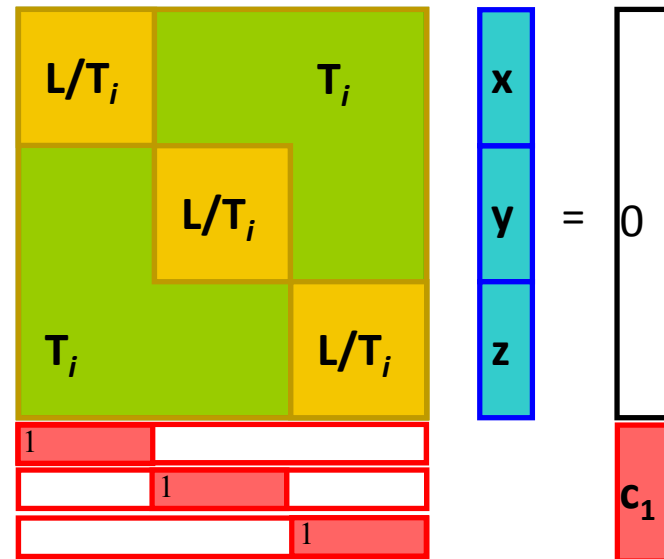
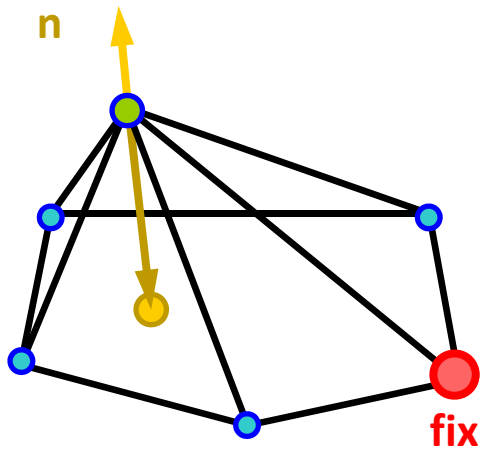
- Implicit transformations



Implicitly compute T_i
For the details see:
transformations by comparing 1-
Laplacian Mesh Editing
[Sorkine et al. 04]
deformed mesh

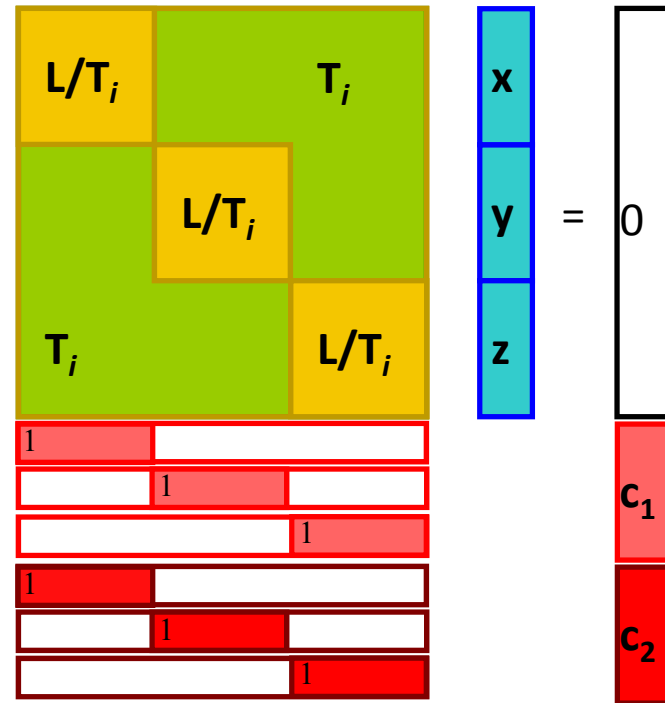
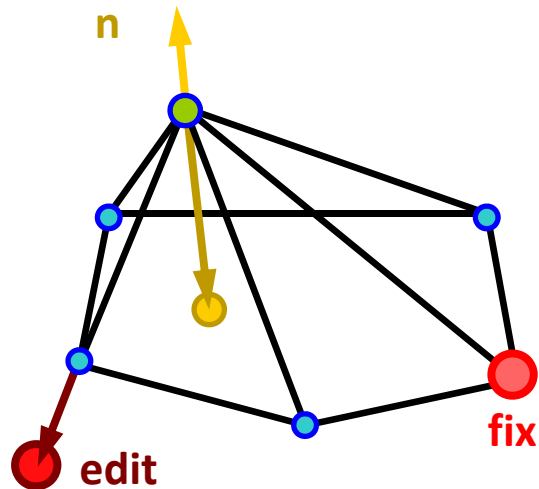
Mesh Modeling Framework

- Surface reconstruction



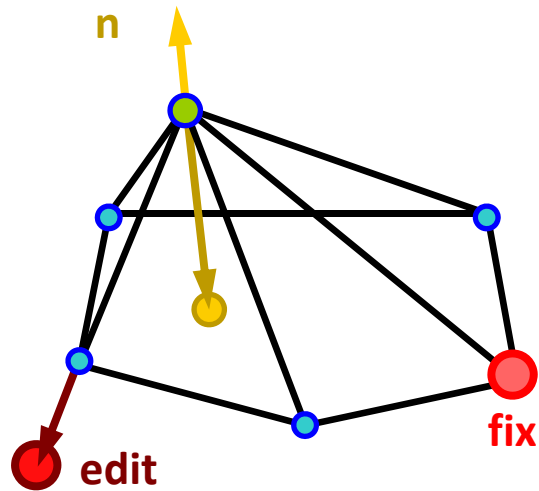
Mesh Modeling Framework

- Editing operations



Mesh Modeling Framework

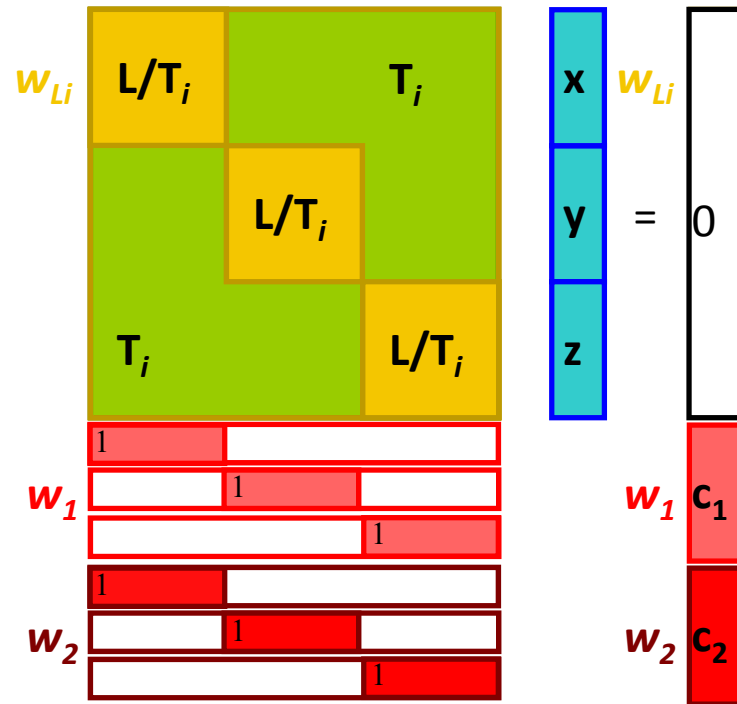
- Least-Squares solution



Normal Equations

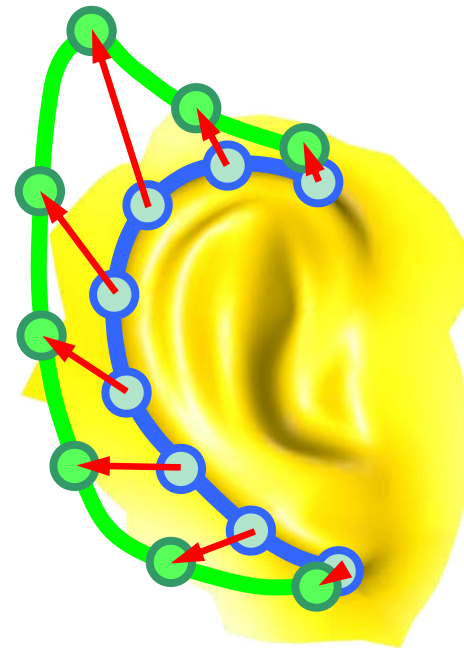
$$A^T A \mathbf{x} = A^T \mathbf{b}$$

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



Silhouette Sketching

- Using silhouettes as handles
 - Detect object space silhouette
 - Project to screen space and parametrize $[0,1]$
 - Parametrize sketch $[0,1]$
 - Find correspondences



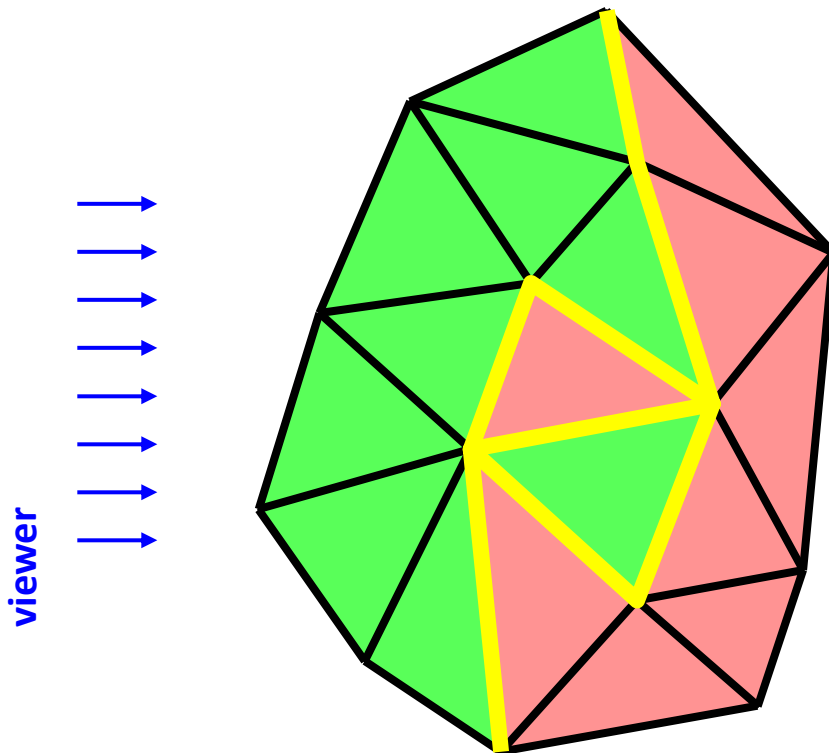
Silhouette Sketching

- Using silhouettes as handles
 - Detect object space silhouette
 - Project to screen space and parametrize $[0,1]$
 - Parametrize sketch $[0,1]$
 - Find correspondences
 - Use as positional constraints while retaining depth value



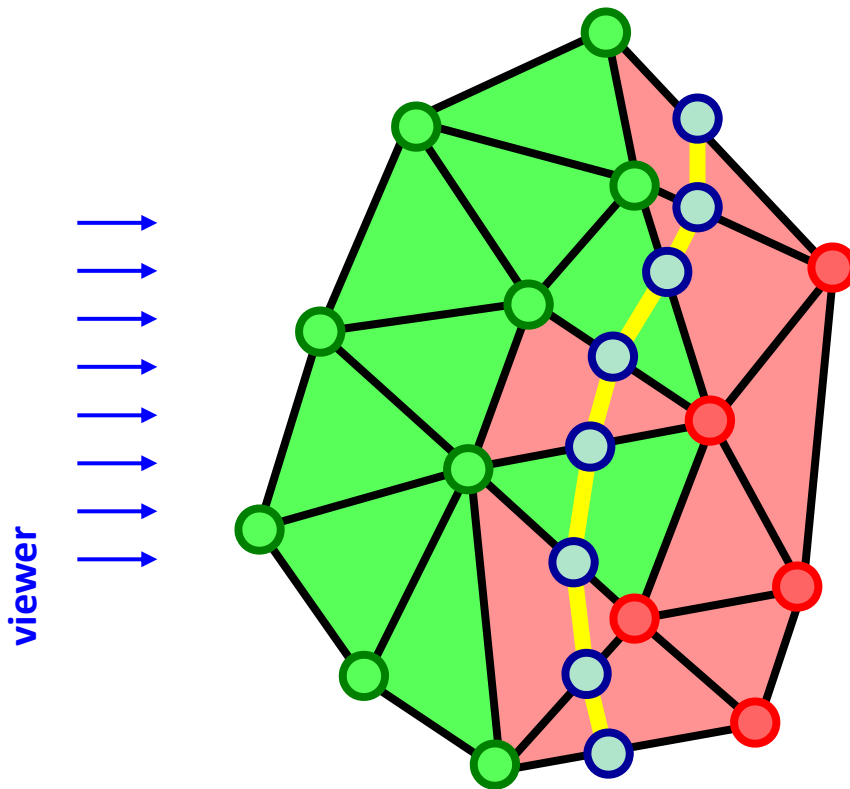
Silhouette Sketching

- What is a good silhouette?



Silhouette Sketching

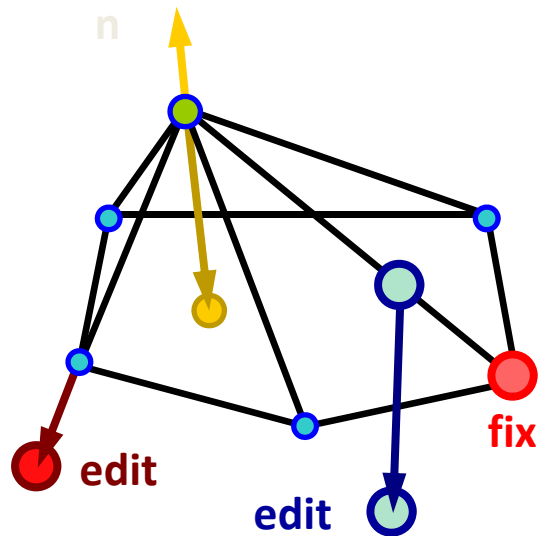
- What is a good silhouette?



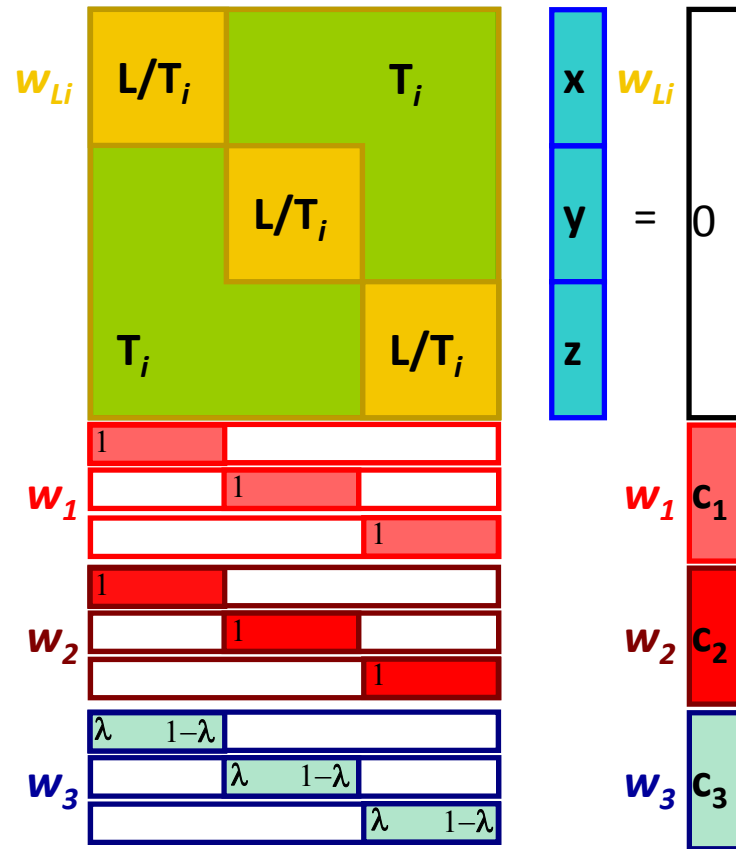
Illustrating Smooth Surfaces
[Hertzmann and Zorin 00]

Silhouette Sketching

- On edge constraints

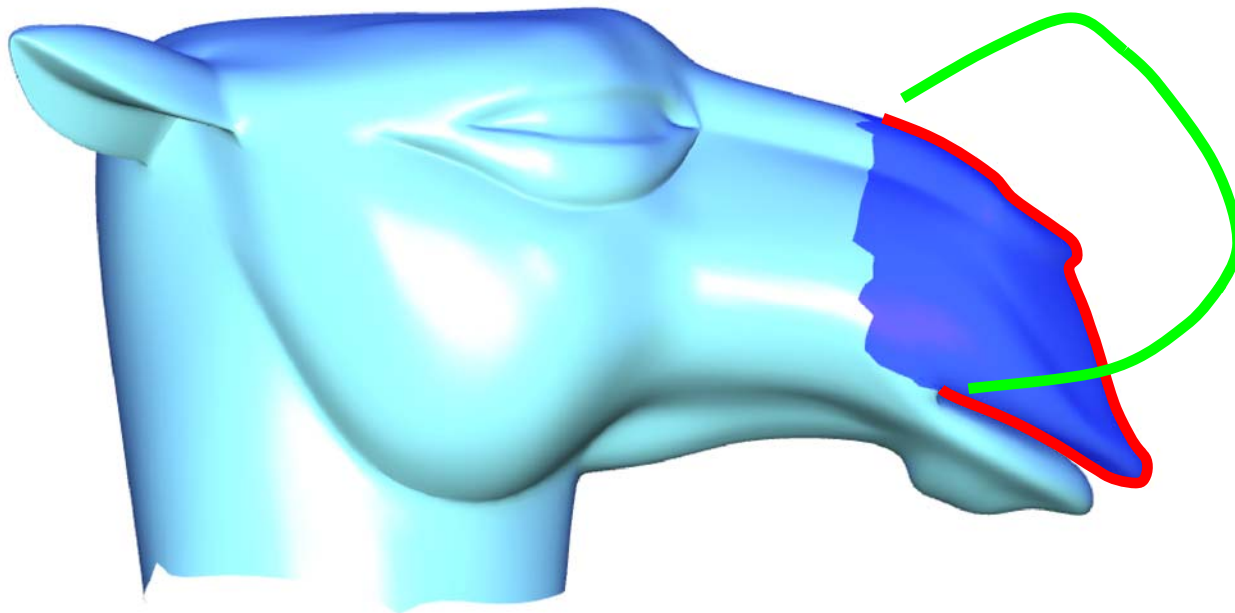


$$\lambda \mathbf{x}_i + (1-\lambda) \mathbf{x}_j$$



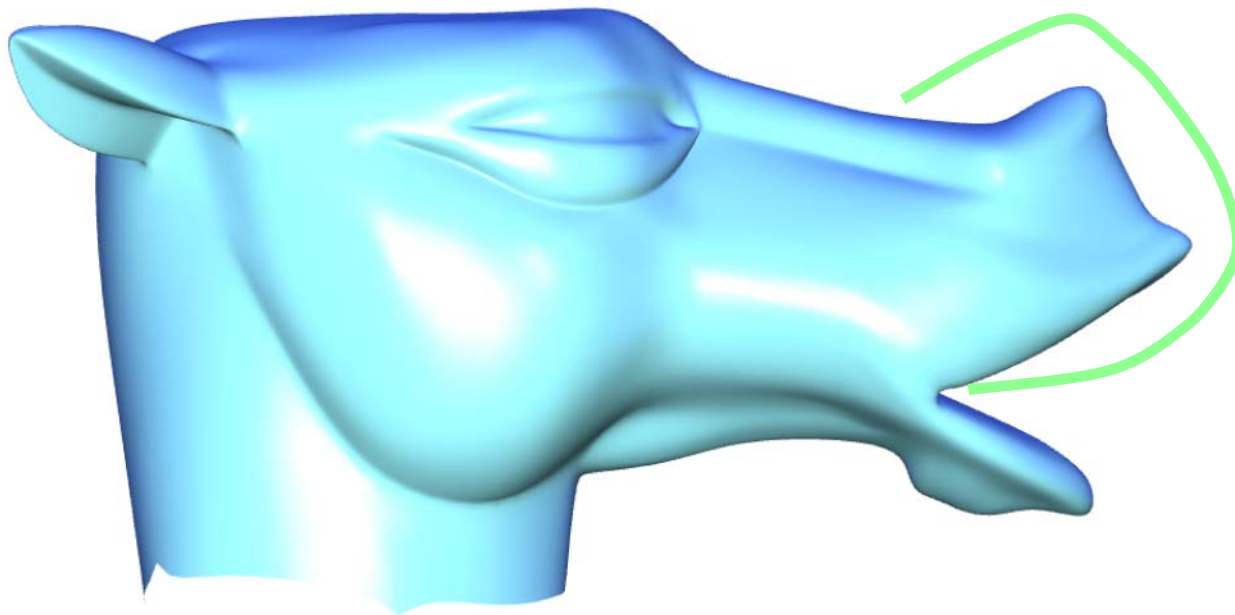
Silhouette Sketching

- Approximate sketching
 - Balance weighting between detail and positional constraints



Silhouette Sketching

- Approximate sketching
 - Balance weighting between detail and positional constraints

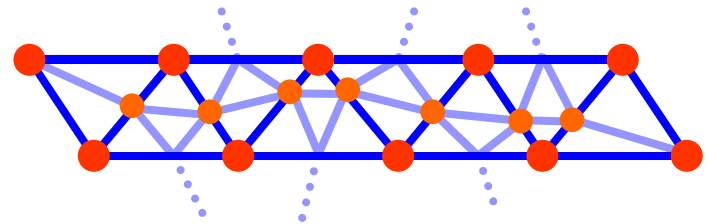
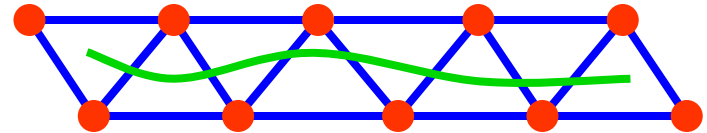
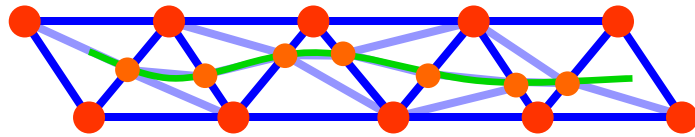


Feature Sketching

- We wish to influence (discrete) differential properties of the mesh for **arbitrary** sketches

- Possible solution

- Cut existing polygons along the sketch and add new edges

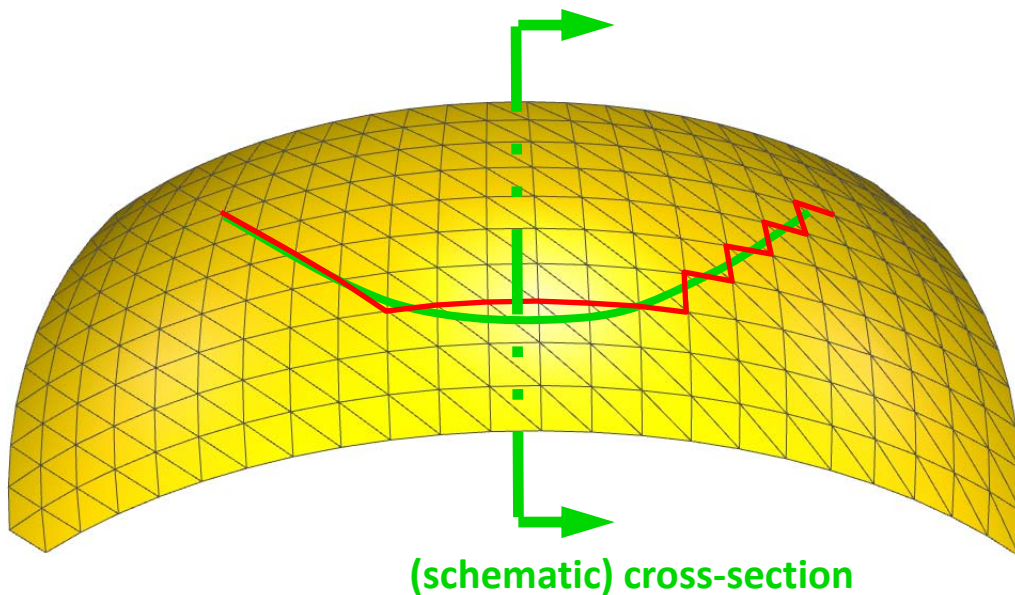


- Our solution

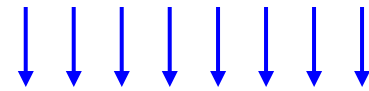
- Adjust mesh **geometry** to lie under the sketch (as seen from the camera), while preserving mesh **topology** and ensuring well shaped triangles

Geometry Adjustment

- First: **min cost edge path (close to sketch)**
 - Potentially *jaggy* appearance

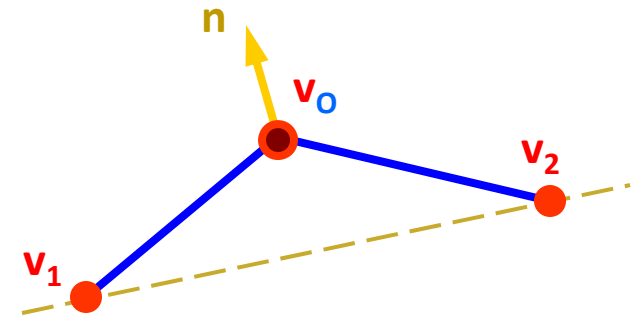


(orthographic) viewer



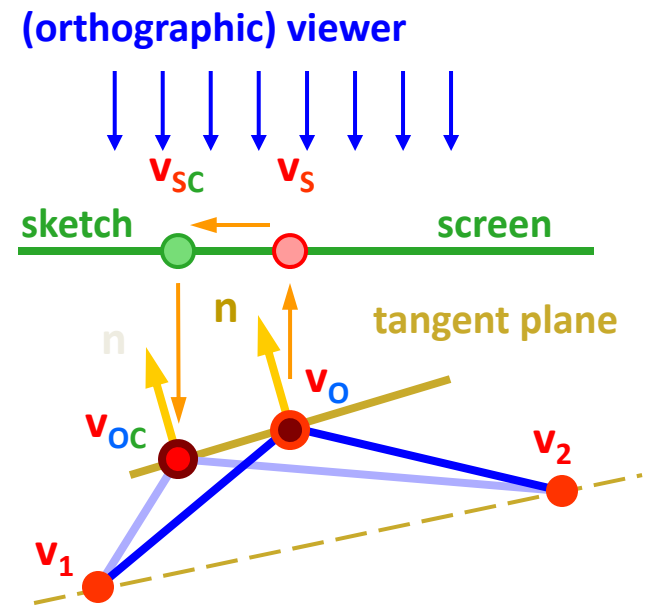
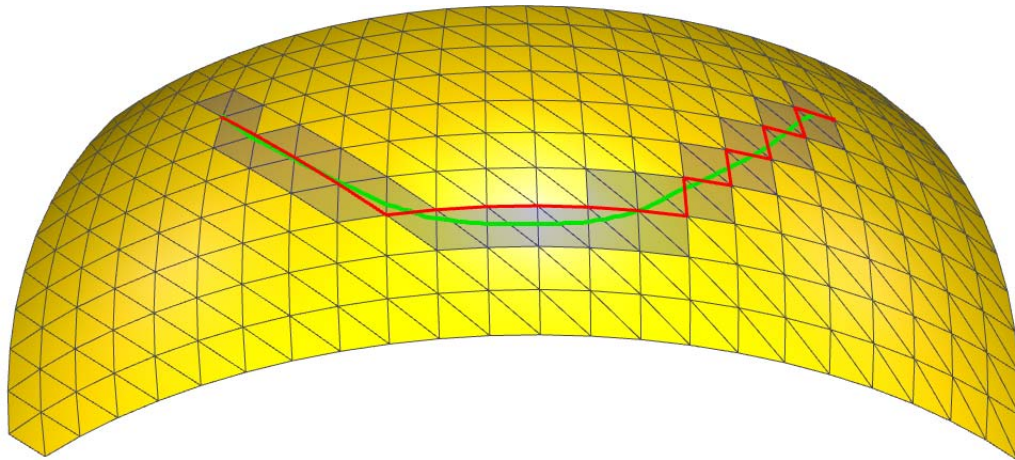
sketch

screen



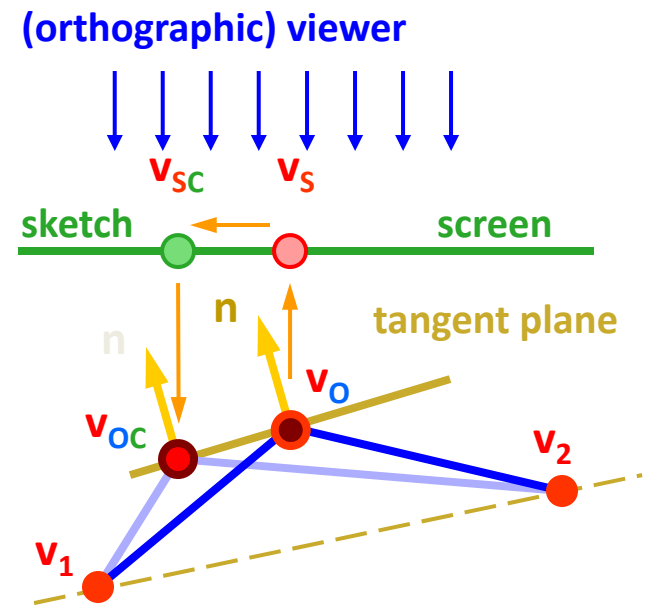
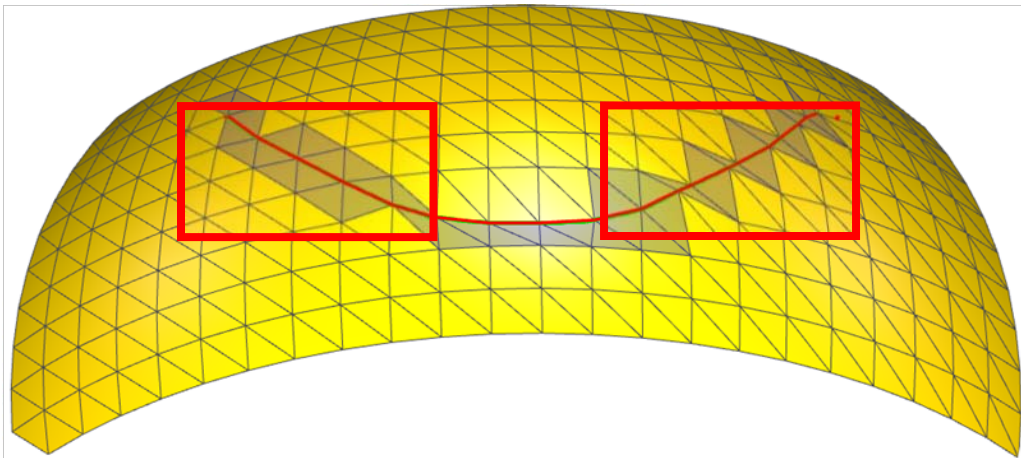
Geometry Adjustment

- Second: **projection onto sketch**



Geometry Adjustment

- Second: **projection onto sketch**
 - Approximates the sketch very well
 - Can introduce badly shaped tri's

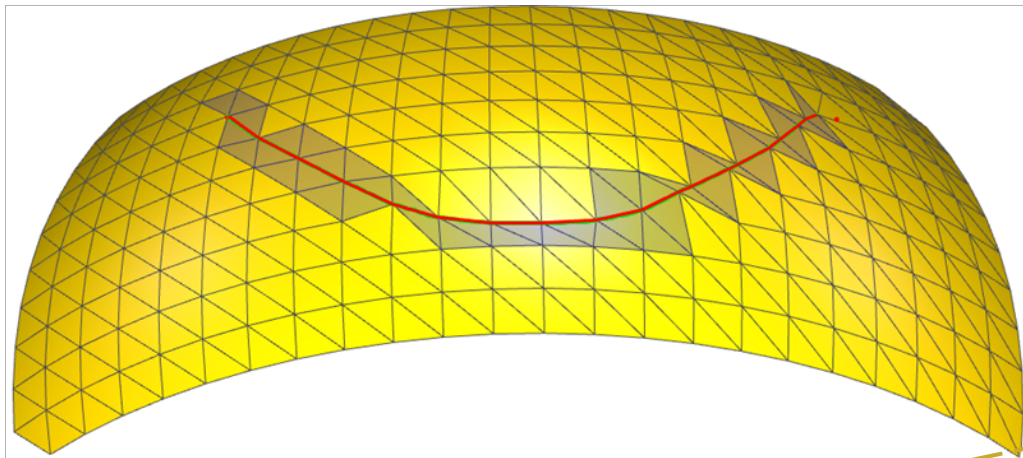


Geometry Adjustment

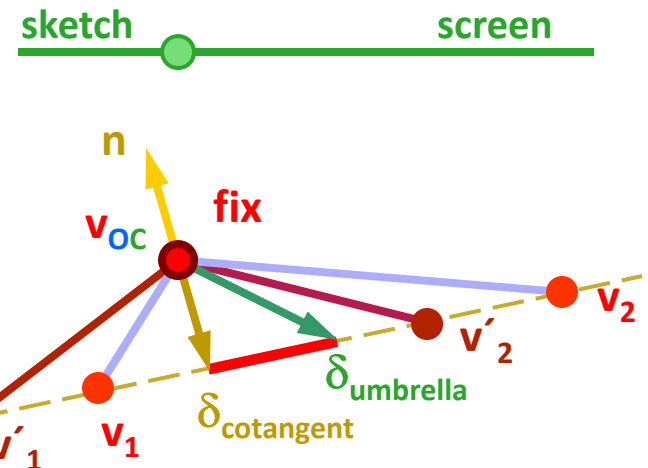
- Third: **local mesh regularization**

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

- Ask uniformly weighted Laplacian to become cotangent weighted Laplacian, while fixing path vertices



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

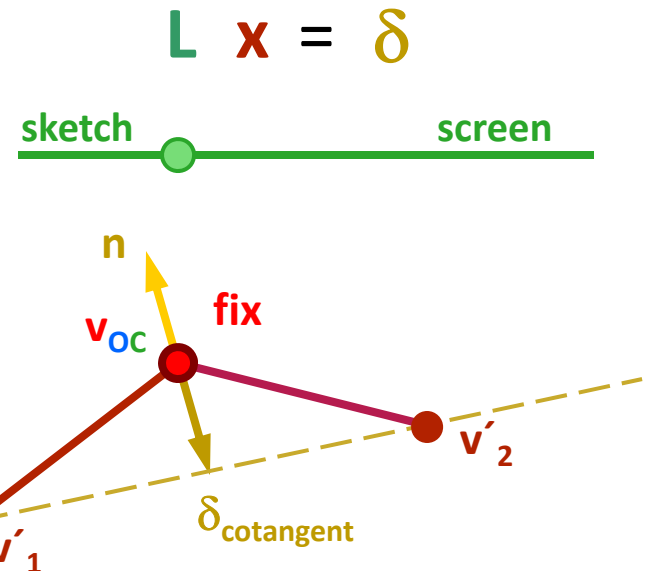
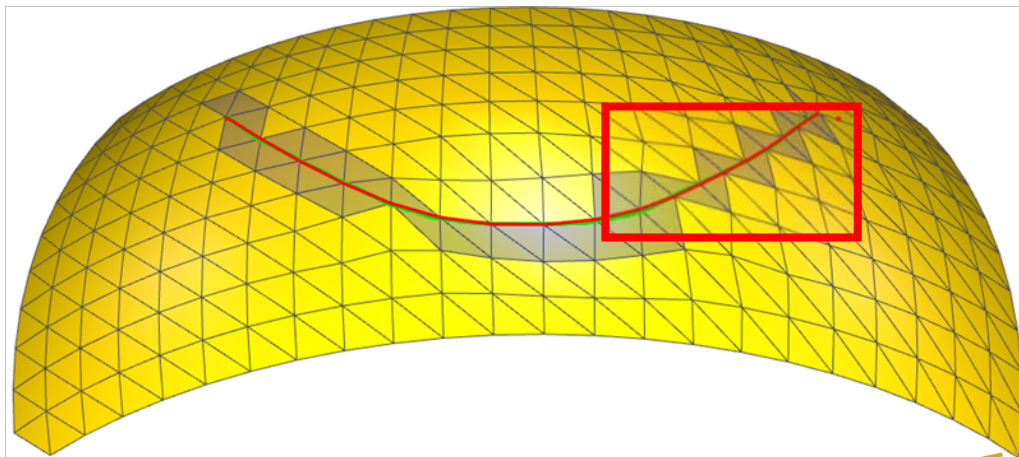


Geometry Adjustment

- Third: **local mesh regularization**

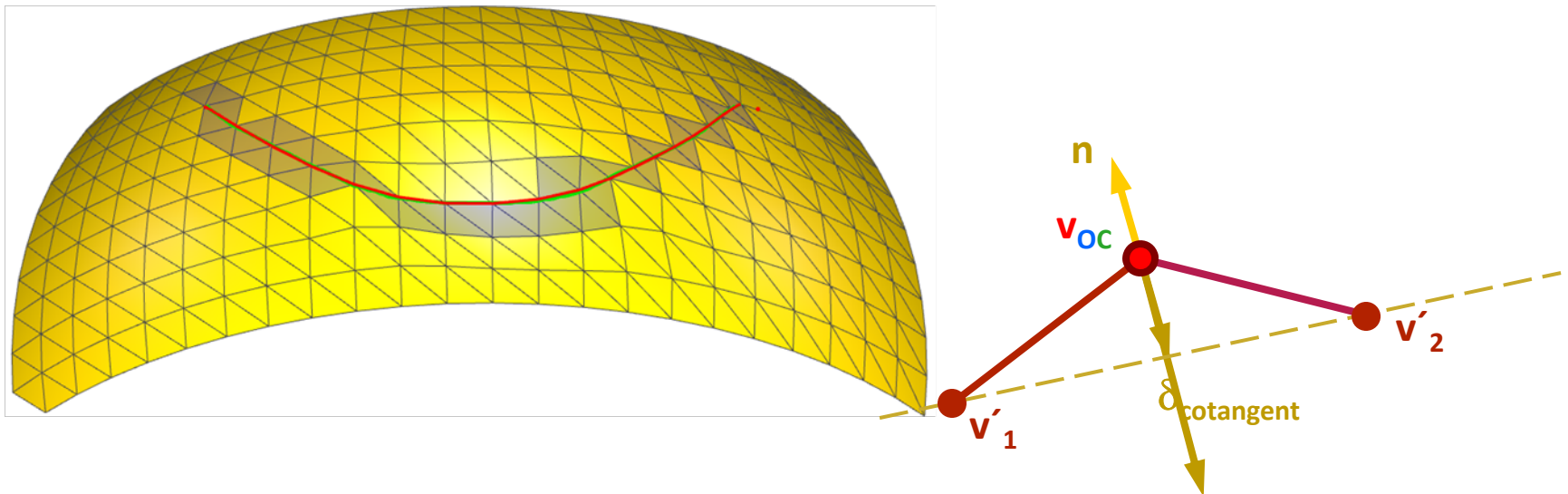
- Well shaped triangles and nice piecewise linear approximation of the users sketch

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



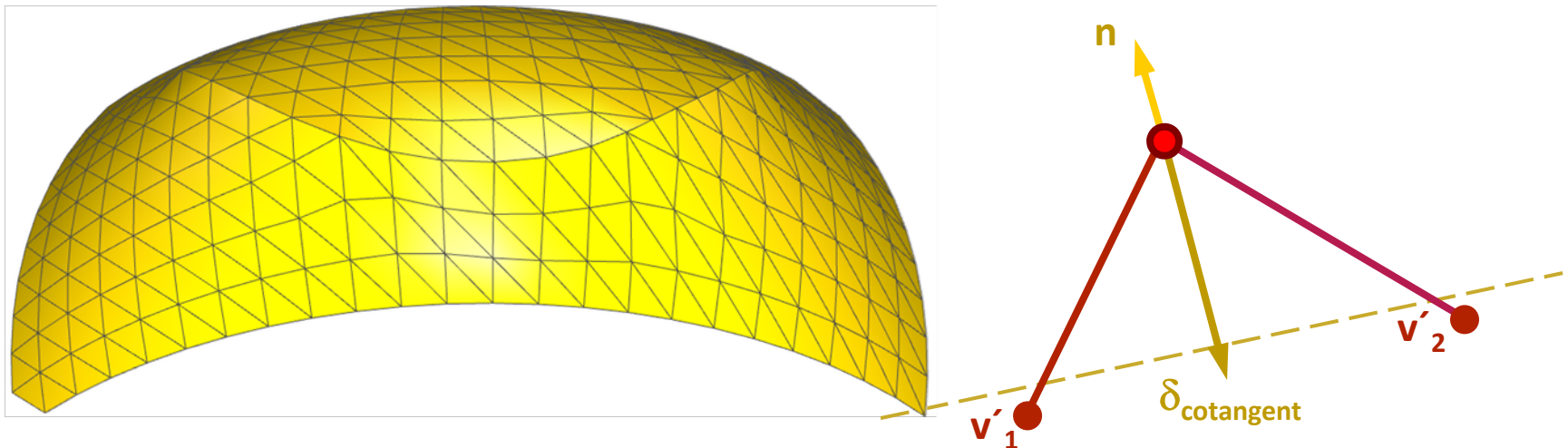
Feature Edit

- Edit: **scale (or add to) Laplacians**

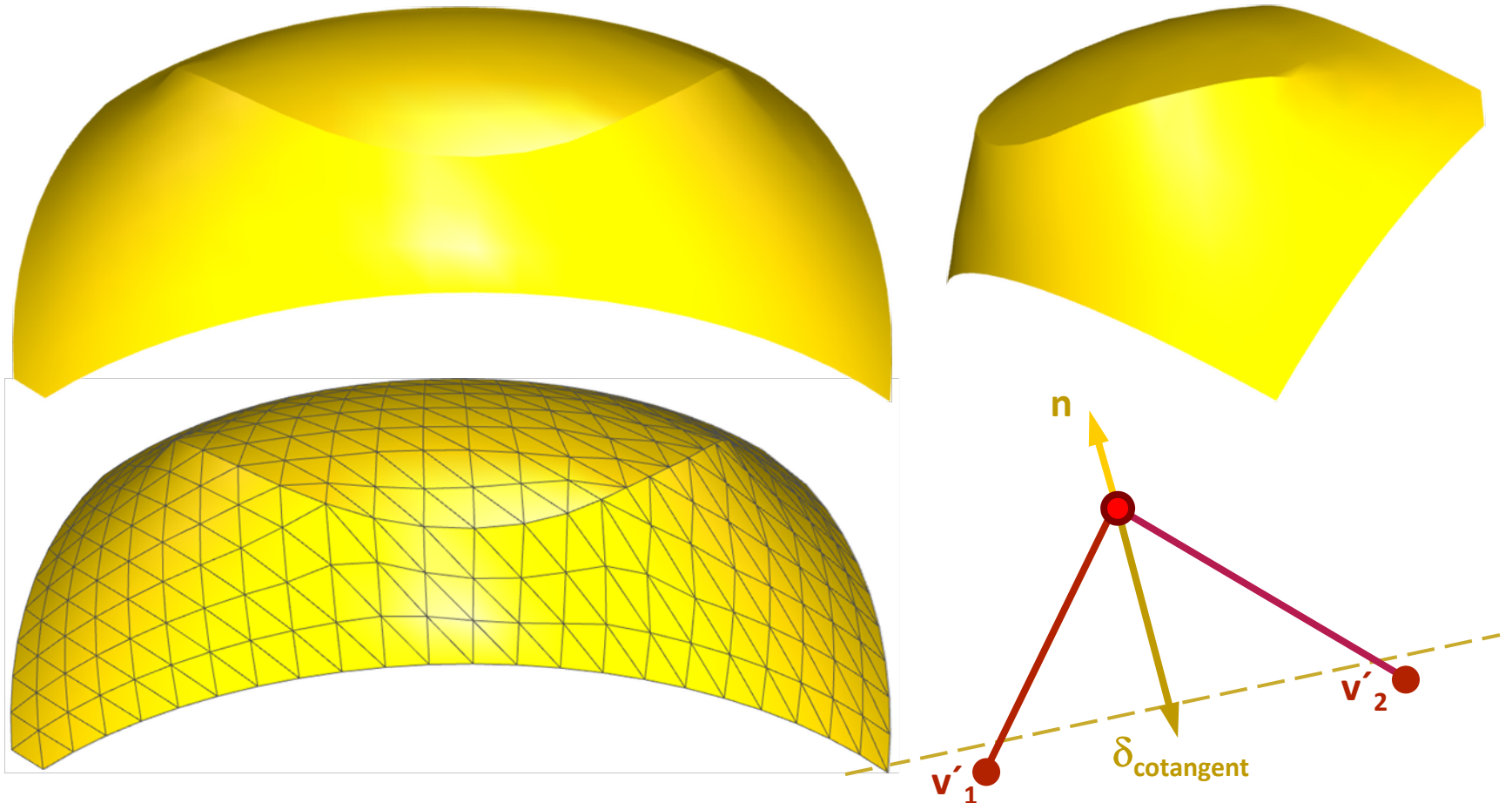


Feature Edit

- Edit: **scale (or add to) Laplacians**

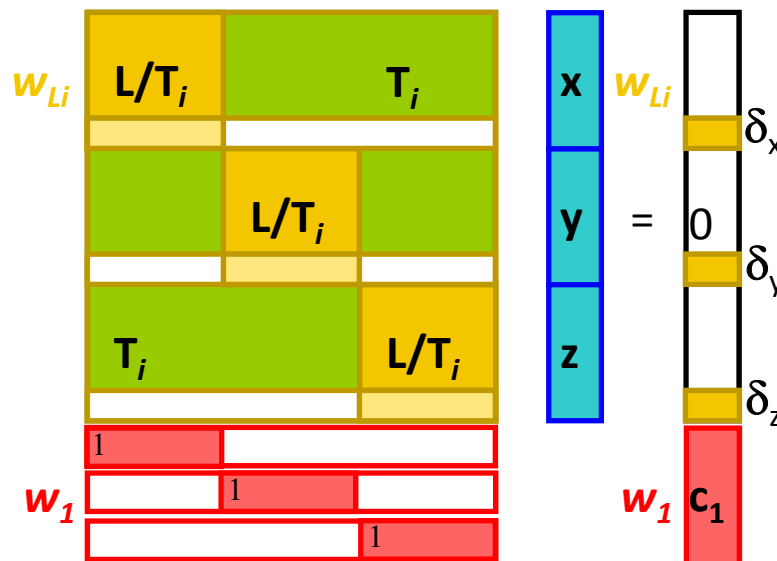
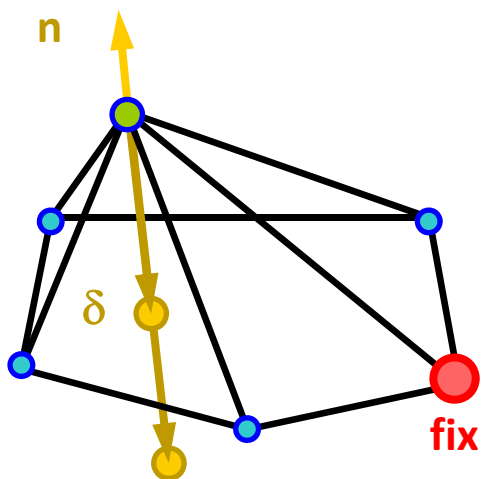


Feature Edit



Laplacian Constraints

- Scale (or add to) Laplacians

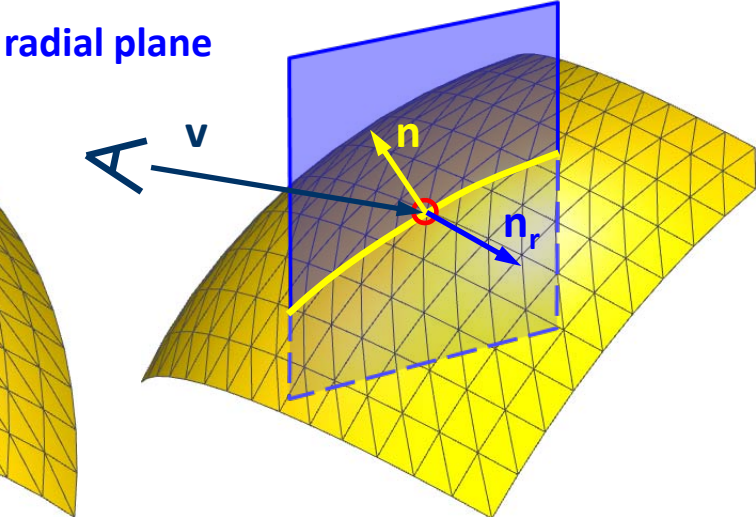
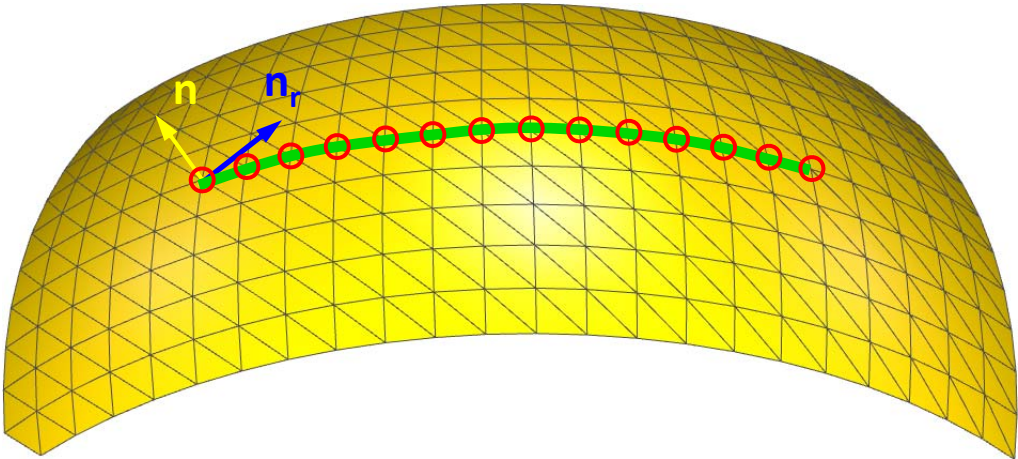


Normal Equations

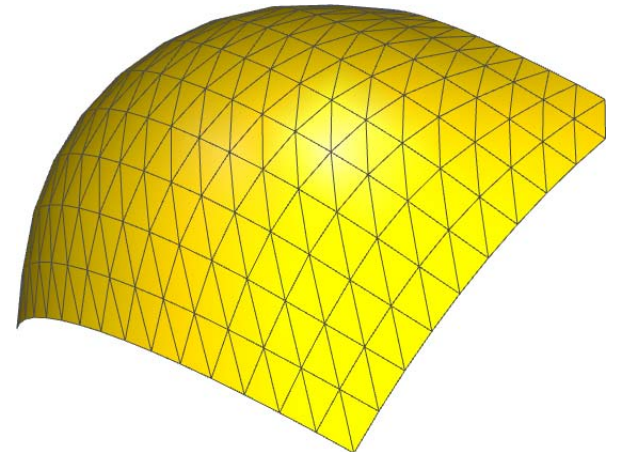
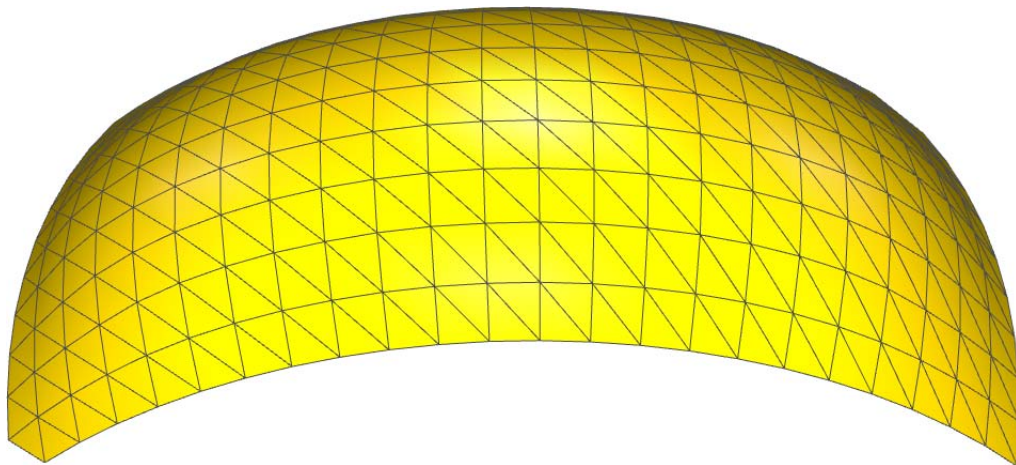
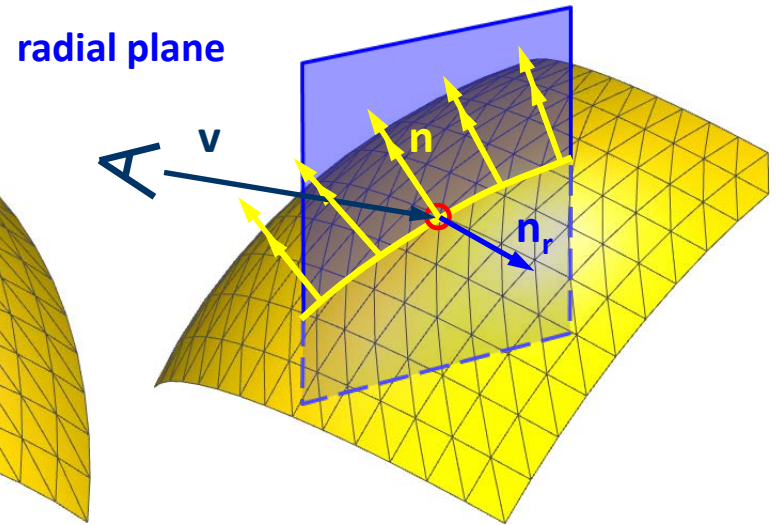
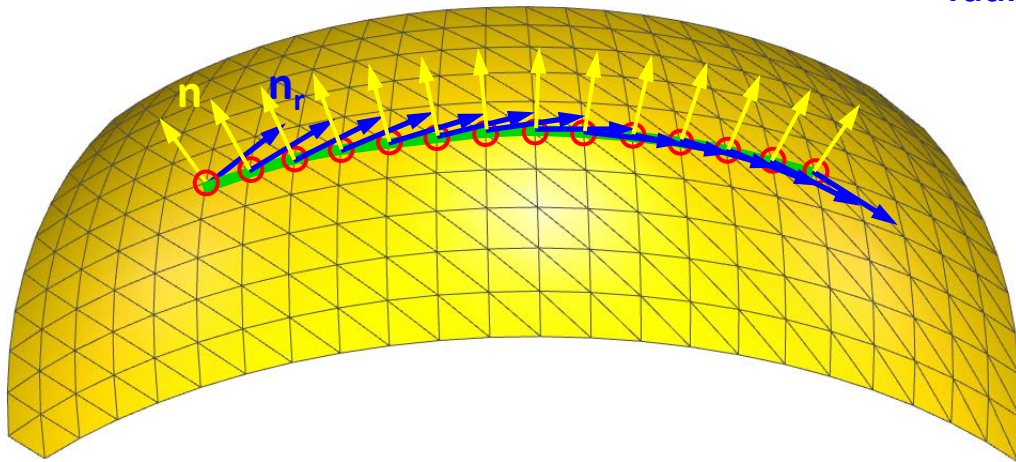
$$A^T A \mathbf{x} = A^T \mathbf{b}$$

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

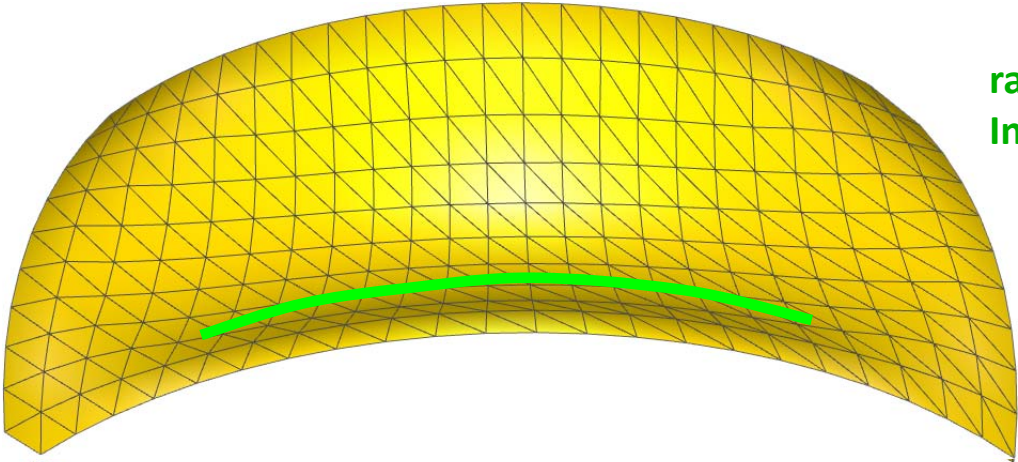
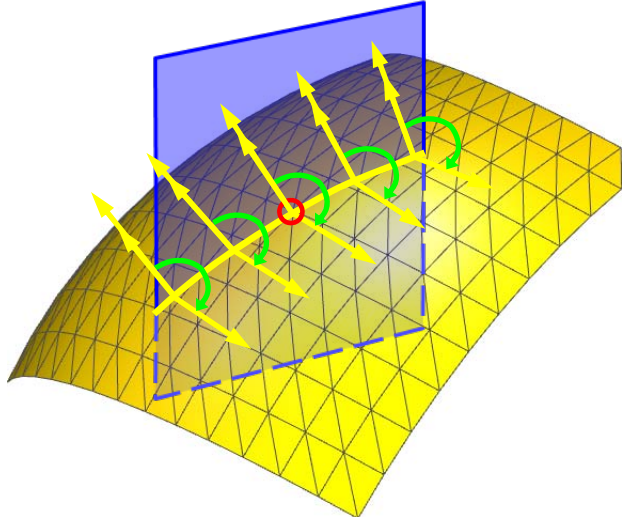
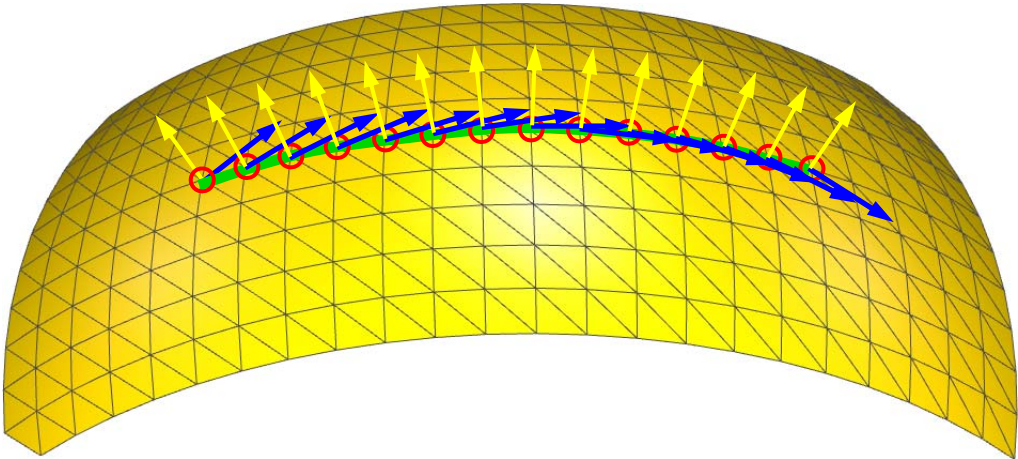
Contour Edit



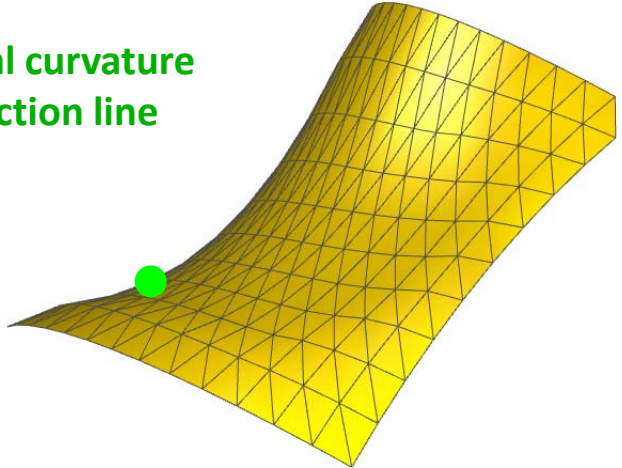
Contour Edit



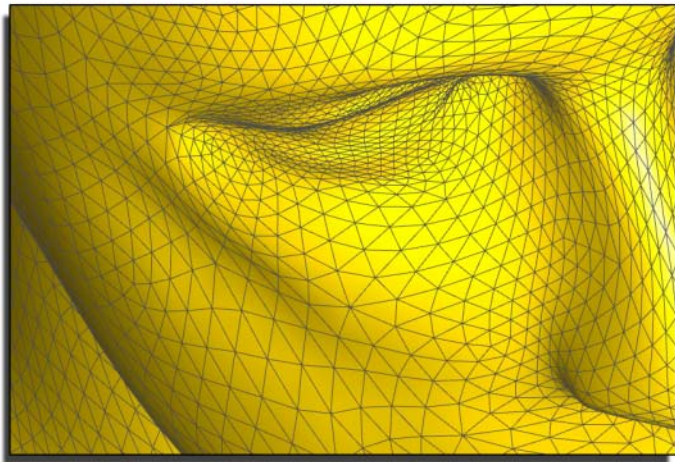
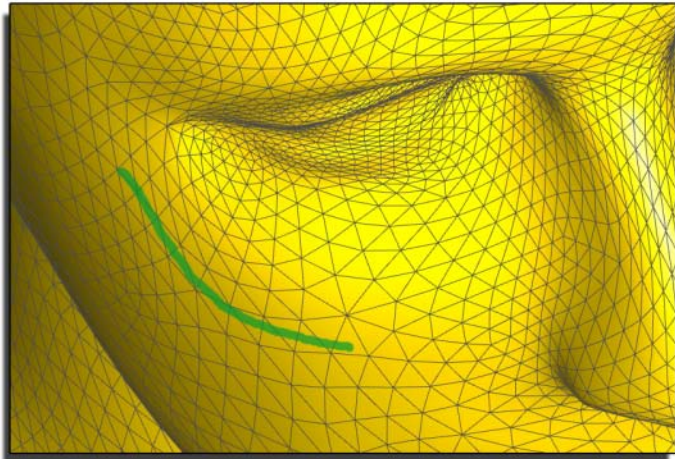
Contour Edit



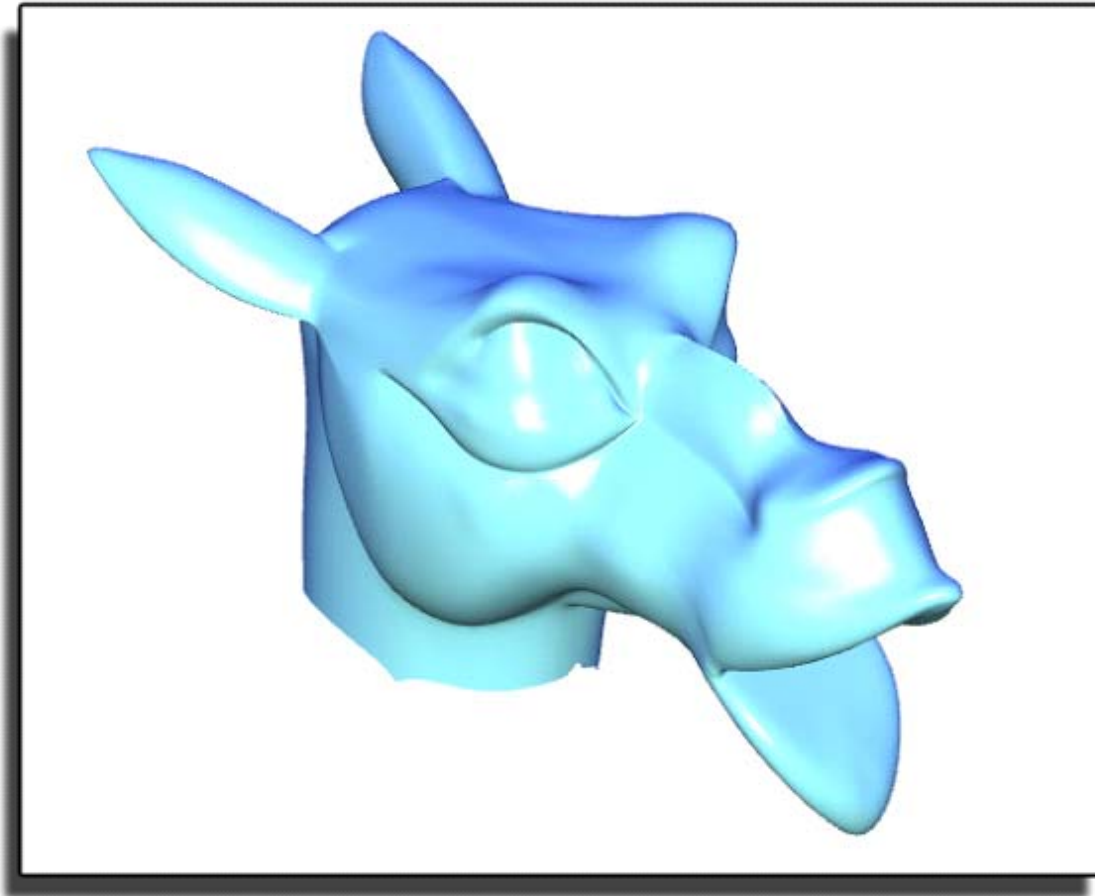
radial curvature
Inflection line



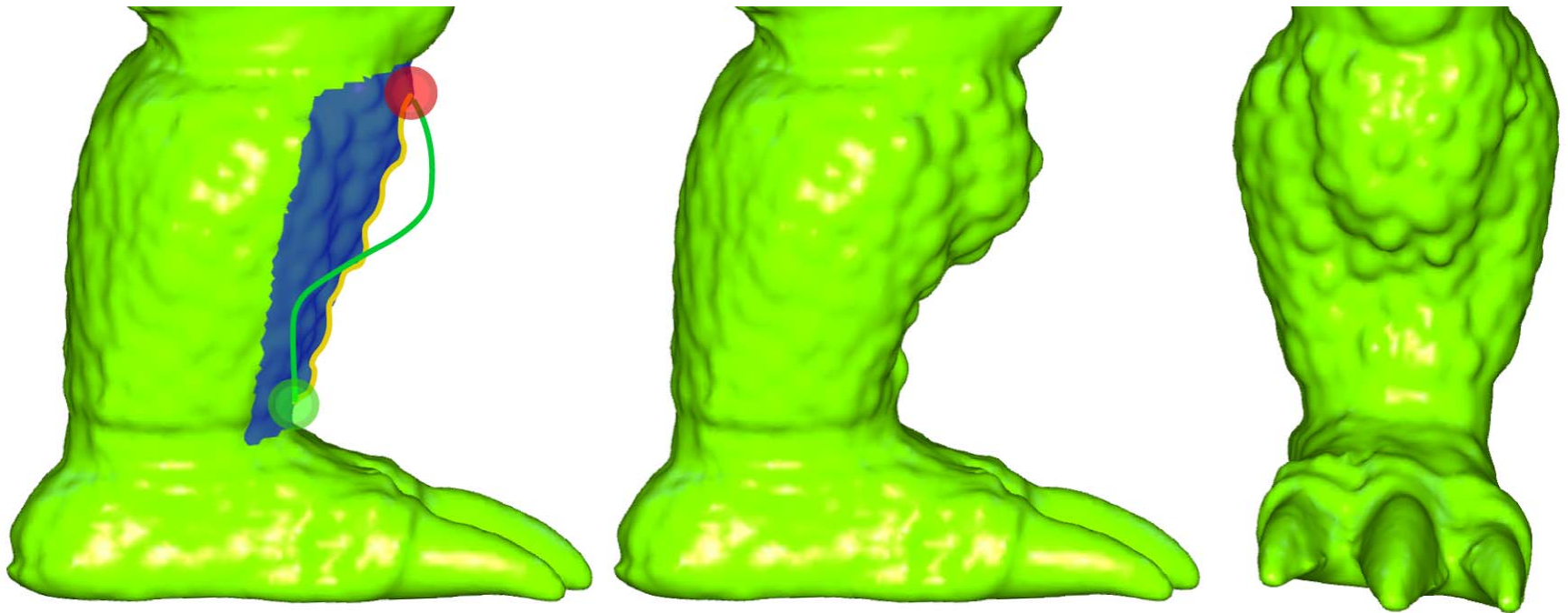
Contour Edit



Editing Session Result



Problem: noisy Surface Silhouette



Discussion...

- The good...
 - Intuitive, sketch-based User Interface for silhouette deformation and feature creation/modification
 - Fast model updates after sketch (Iterative Modeling)
 - Preserves surface detail as much as possible
- ... and the *not so good*
 - Object-Space sil's useless in the presence of heavy noise
 - Editing differential properties can take time to learn

FiberMesh

Problem Statement

- 3D modeling from scratch is **difficult**

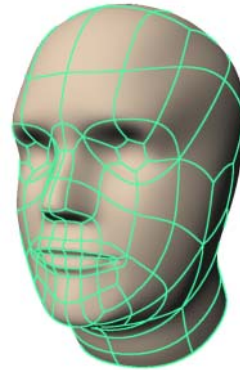
Parametric patches

**Subdivision
surfaces**

Initial patch layout
can be tedious

Sketching

Produces simple,
rough models



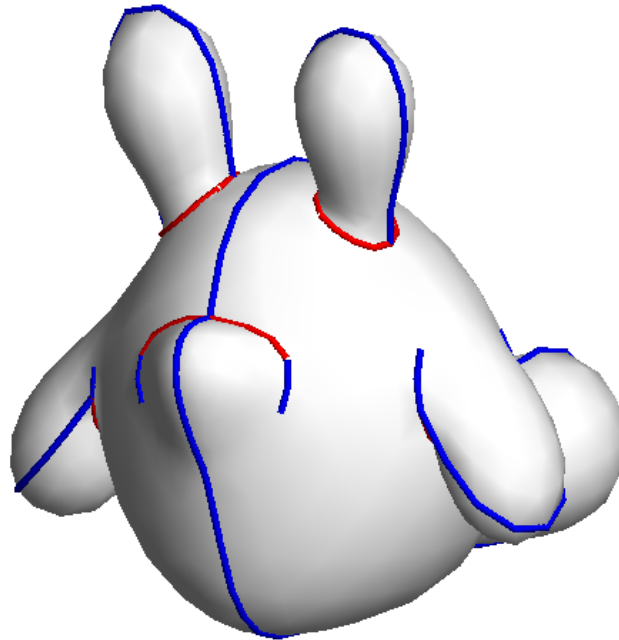
FiberMesh

Overview and Contributions

- **Curves as the user interface**
 - The user's sketches are persistent, and used as a modeling handles
 - Topologically flexible = add / remove anywhere
- **Fast surface construction by (nonlinear) functional optimization**
 - Incorporates curve constraints
 - Runs at interactive rates

FiberMesh Demo

File Config Iterations



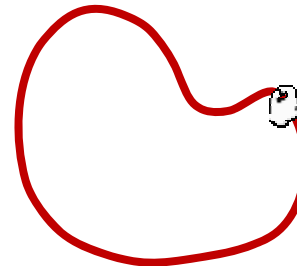
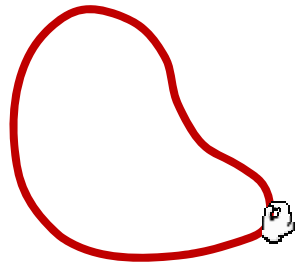
init undo style LSM MVC w recomp w precomp w unconstr remesh rotate

Algorithm Overview

- **3D curve deformation**

Please see paper for details...

Handle position → Curve geometry

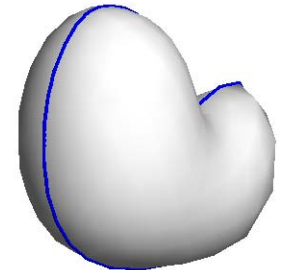
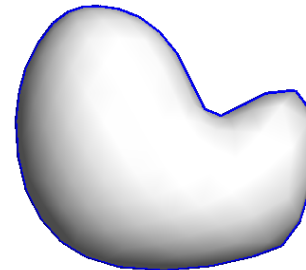


- **Surface optimization**

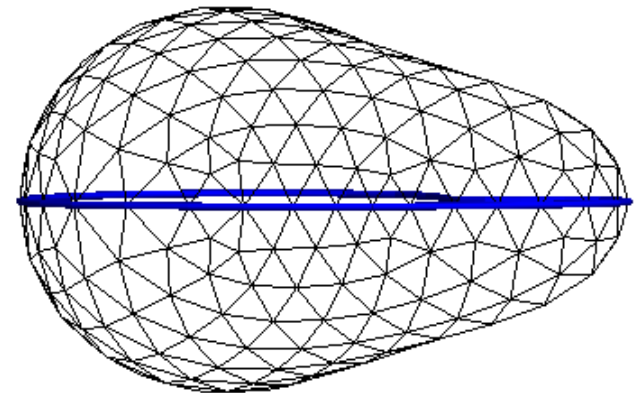
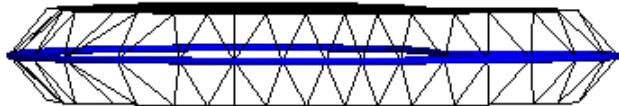
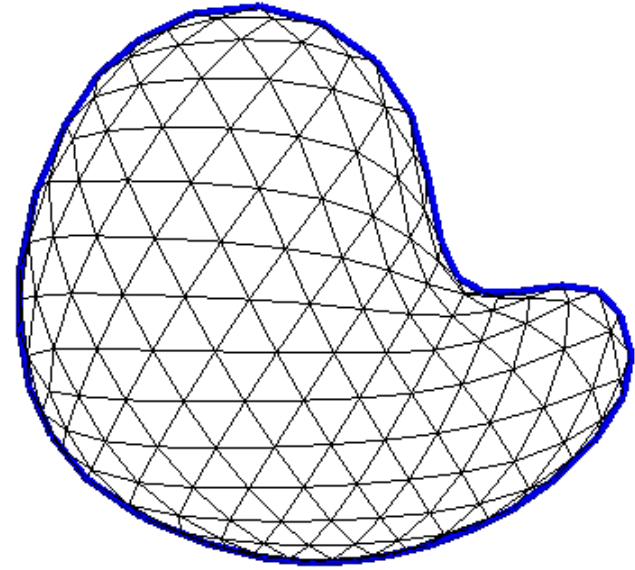
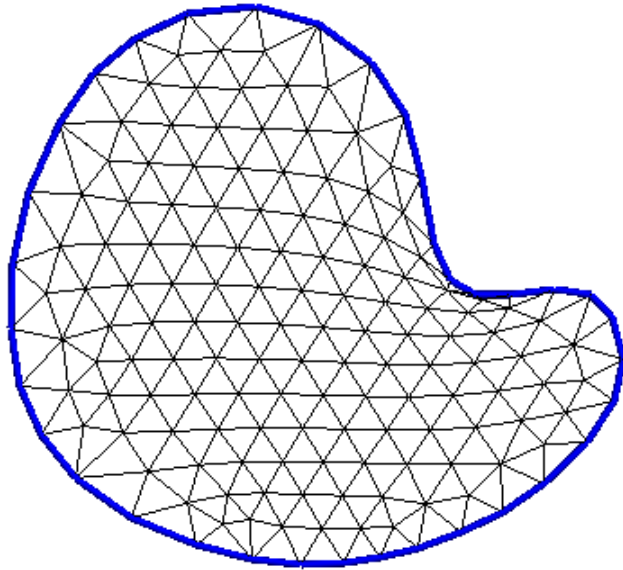
Curve geometry



Surface geometry



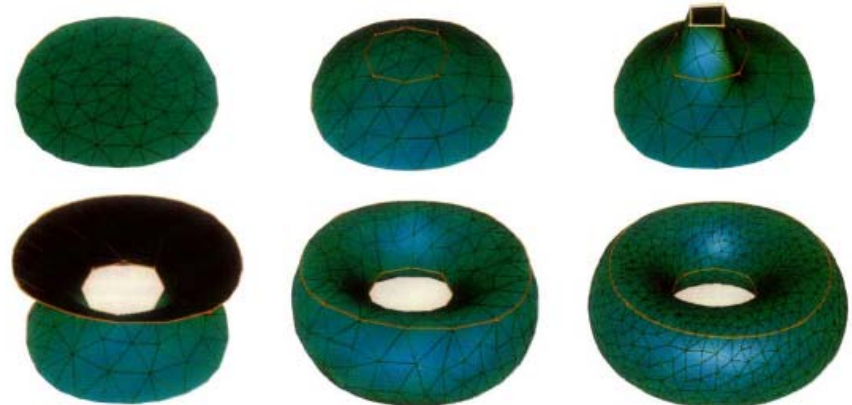
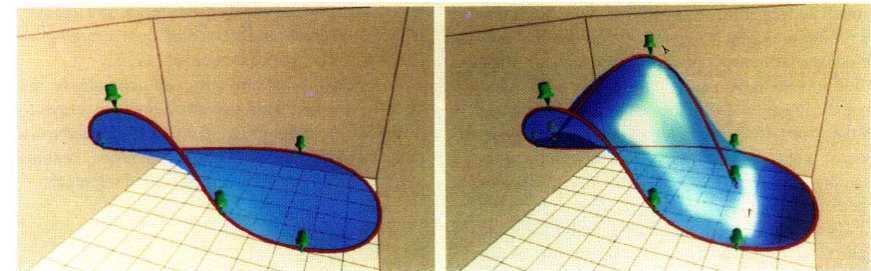
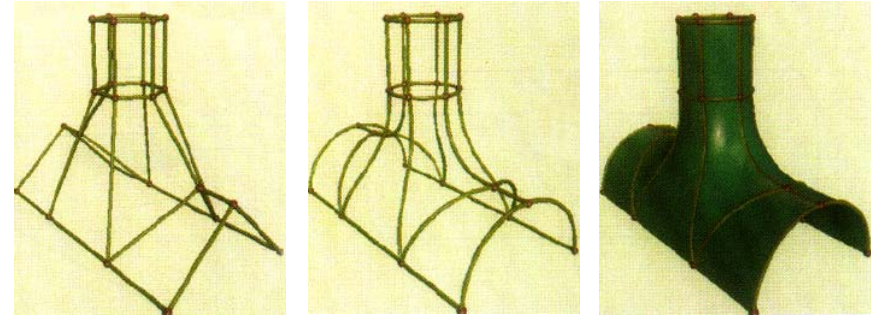
Surface Optimization ?



Inspirations

Curve and Surface Optimization

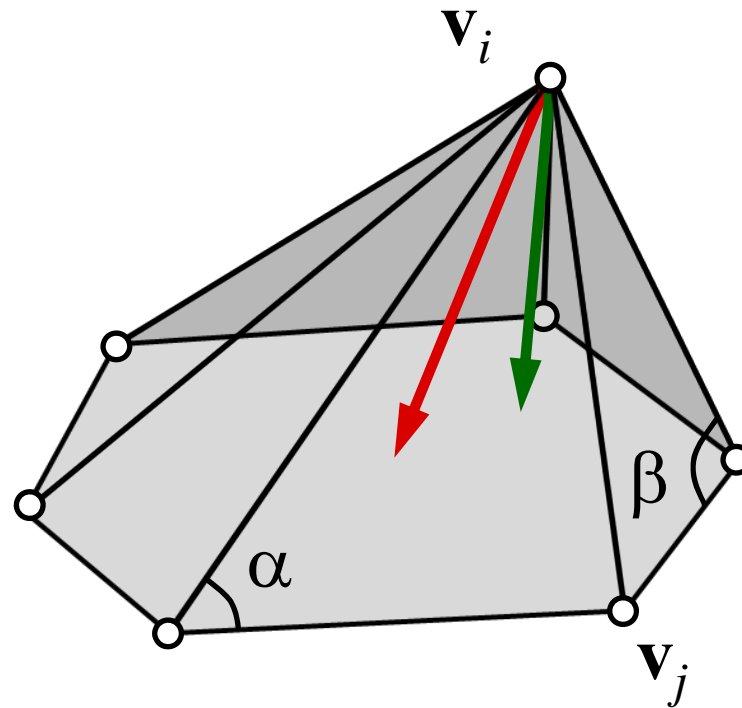
- Minimal energy networks [Moreton and Sequin 1991]
- Functional optimization [Moreton and Sequin 1992]
- Variational surface modeling [Welch and Witkin 1992]
- Shape design with triangulated surfaces [Welch and Witkin 1994]
- **What defines „smooth“ ?**



$$\mathbf{L}^k(x) = 0$$

Surface Optimization

Discrete Differential Geometry



See work from

Polthier, Desbrun, Meyer, Alliez, Grinspun, Schröder, etc.

Surface Optimization

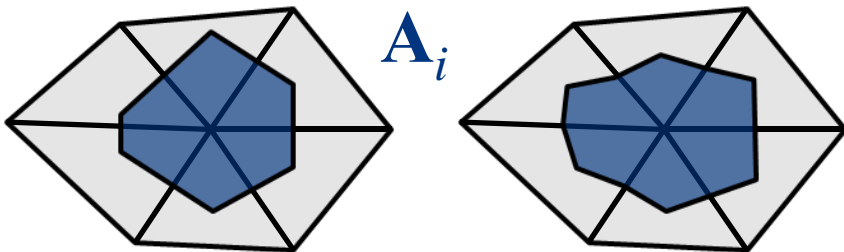
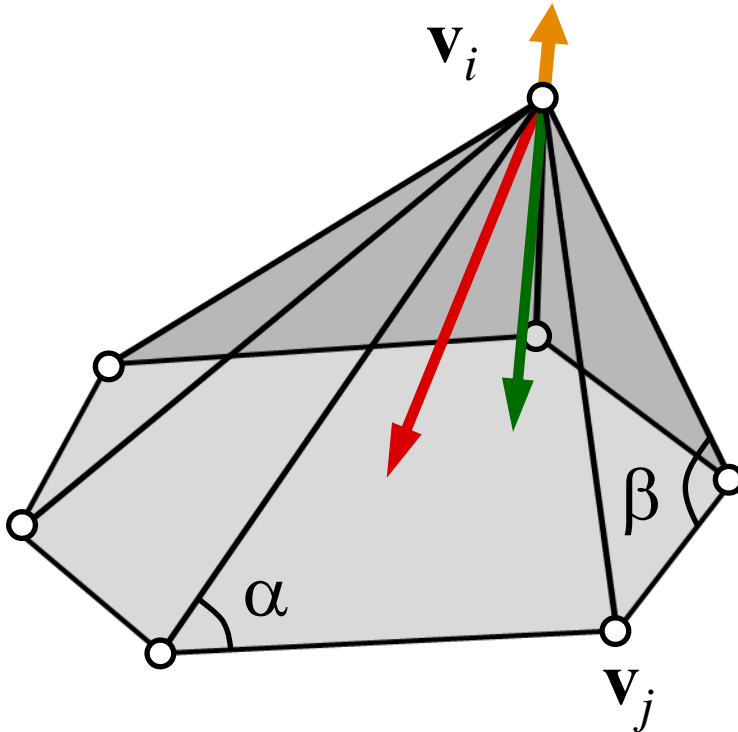
Discrete Differential Geometry

- Laplacian operators

- **Uniform Laplacian** $L_u(\mathbf{v}_i)$

- **Cotangent Laplacian** $L_c(\mathbf{v}_i)$

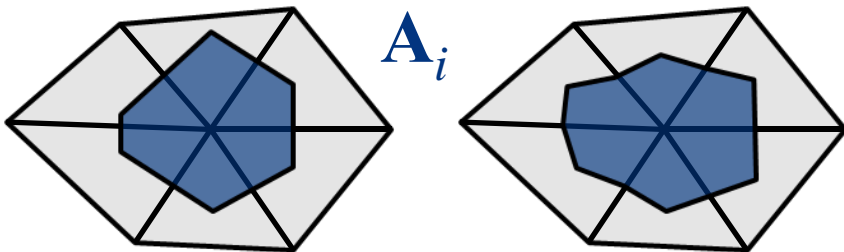
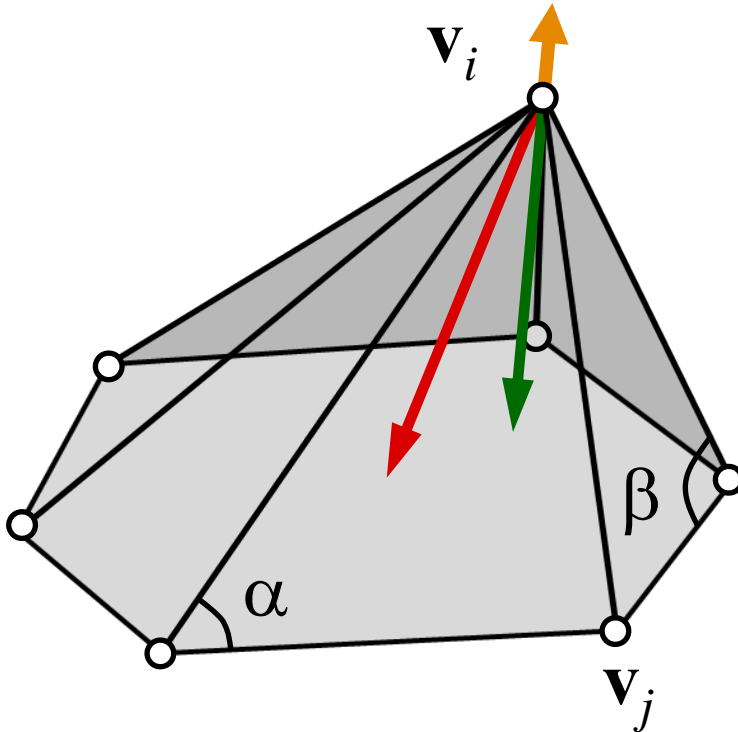
- **Mean curvature normal**



Surface Optimization

Two Important Observations !

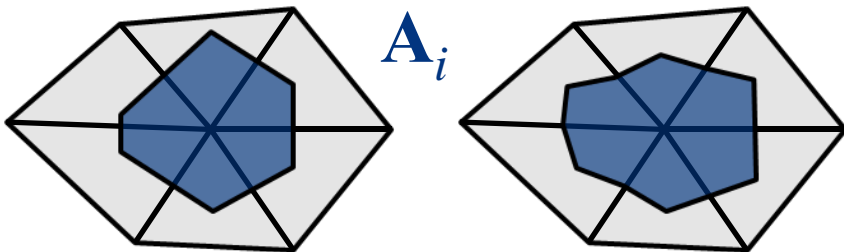
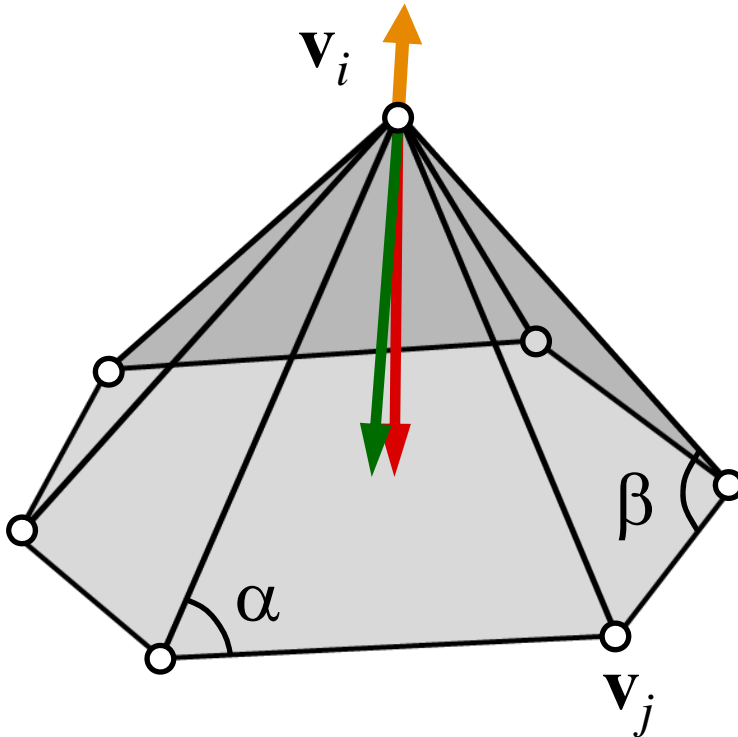
- Laplacian operators
 - **Uniform Laplacian** $L_u(\mathbf{v}_i)$
 - **Cotangent Laplacian** $L_c(\mathbf{v}_i)$
 - **Mean curvature normal**
- **Cotangent Laplacian = mean curvature normal** \times **vertex area** (A_i)
- For nearly equal edge lengths **Uniform** \approx **Cotangent**



Surface Optimization

Two Important Observations !

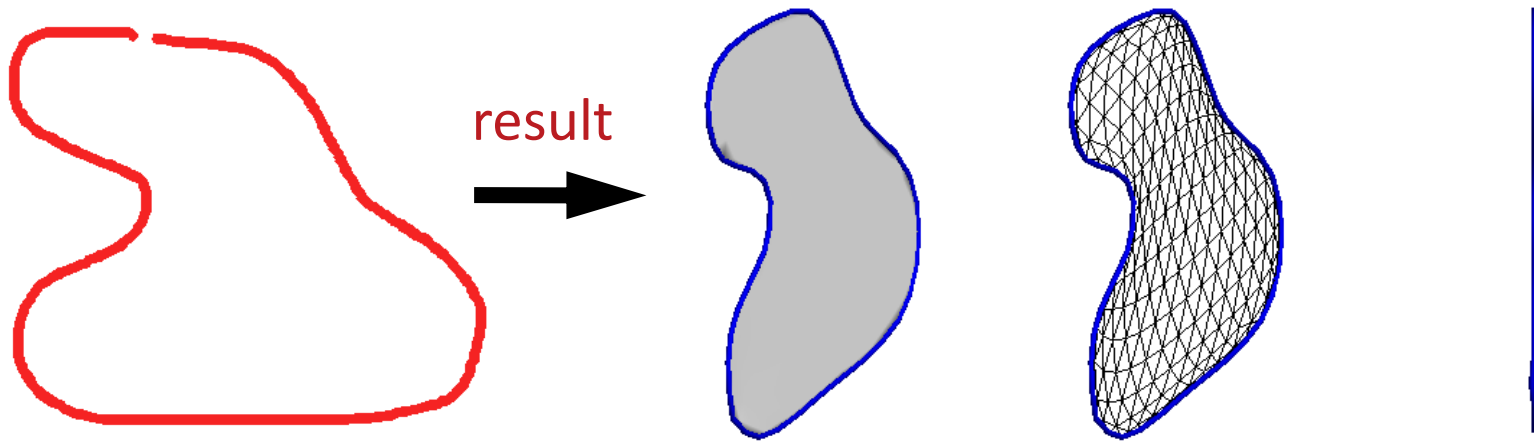
- Laplacian operators
 - **Uniform Laplacian** $L_u(\mathbf{v}_i)$
 - **Cotangent Laplacian** $L_c(\mathbf{v}_i)$
 - **Mean curvature normal**
- **Cotangent Laplacian = mean curvature normal** \times **vertex area** (A_i)
- For nearly equal edge lengths **Uniform** \approx **Cotangent**



Surface Optimization

Linear- or Nonlinear Optimization ?

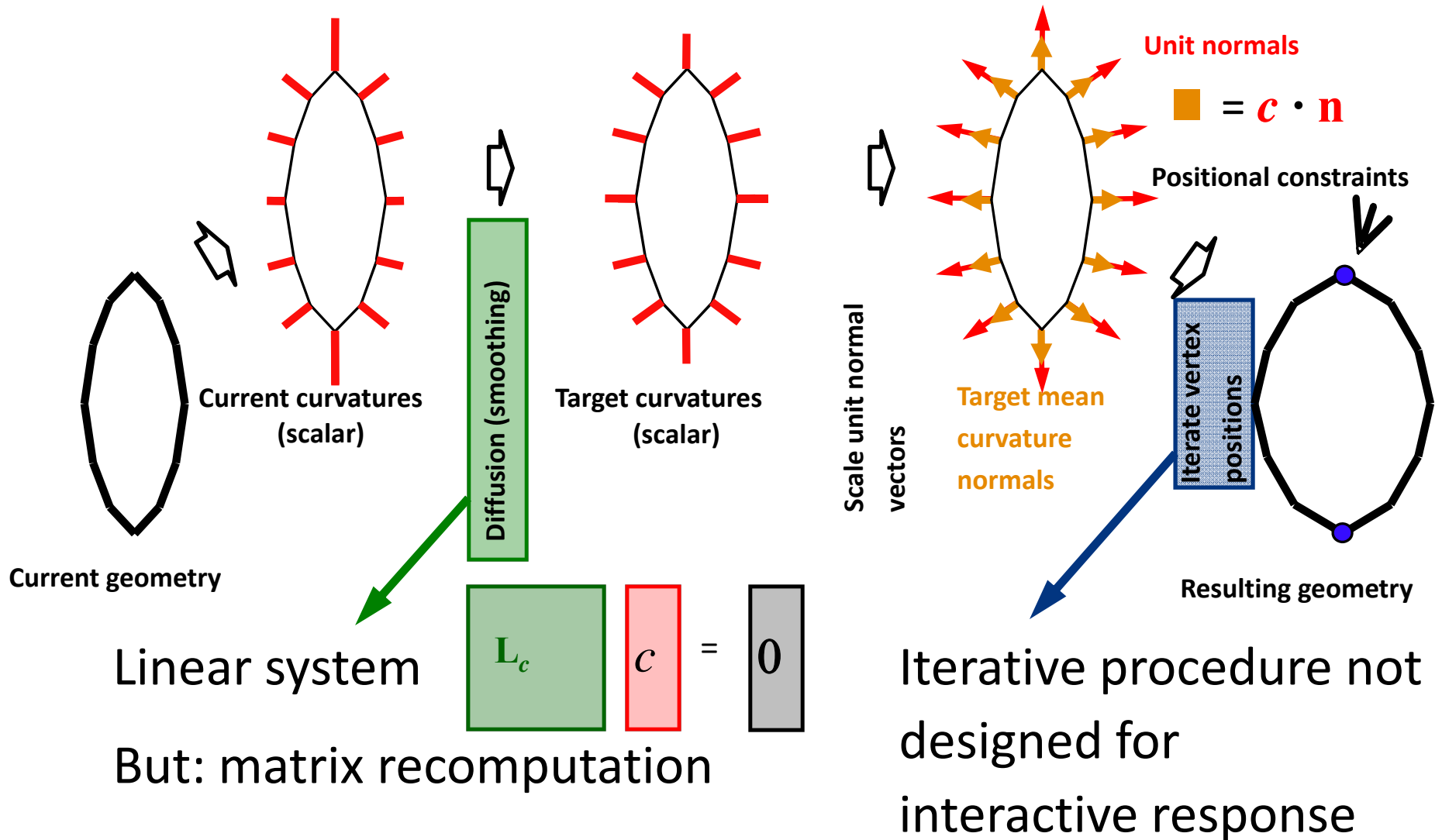
- Linear variational methods [Botsch and Sorkine 2007]



- Instead: nonlinear optimization
- Inspired by a discrete fairing algorithm
 - Geometric fairing of irregular meshes for free-form surface design [Schneider and Kobbelt 2002]

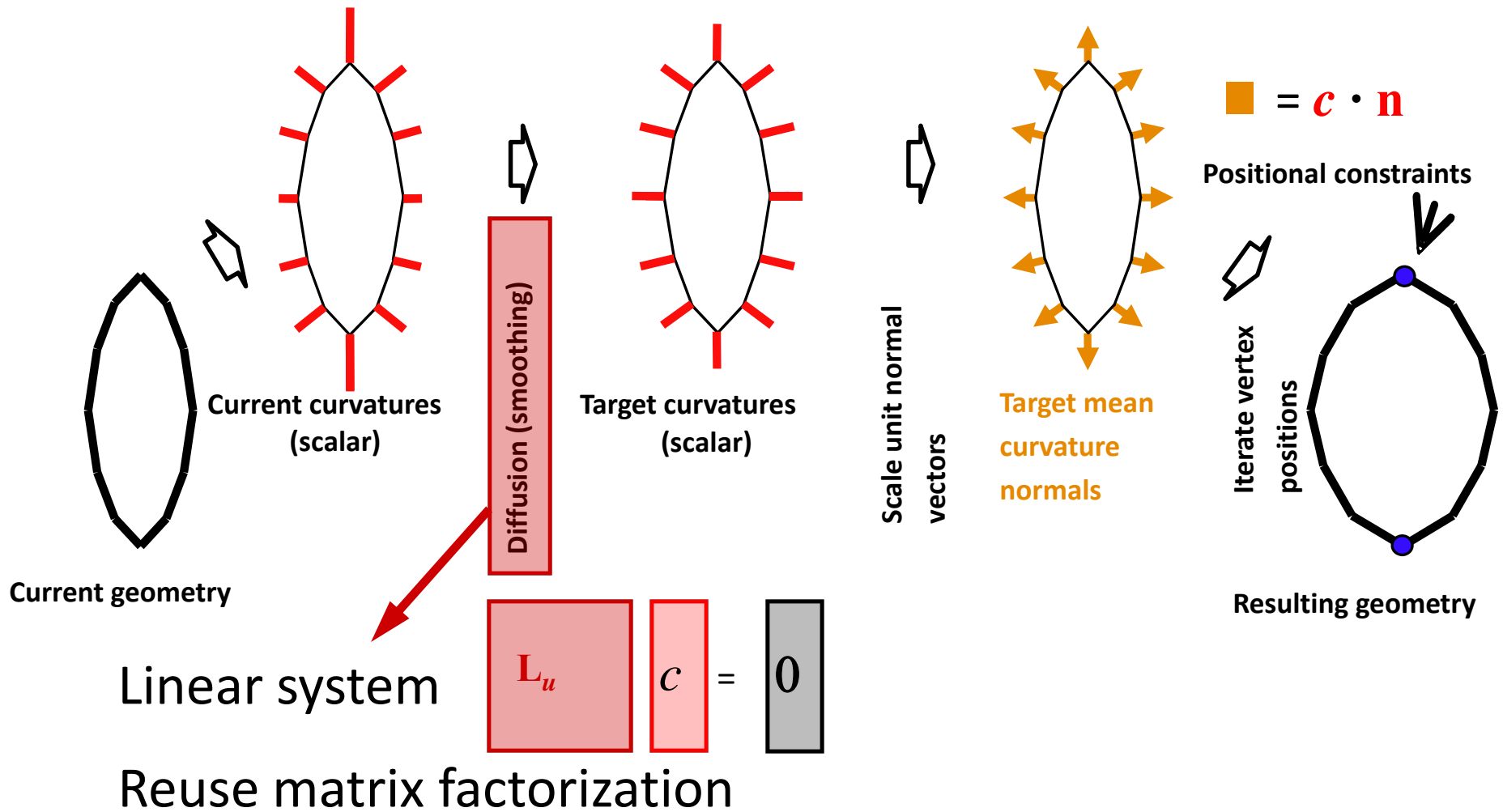
Surface Optimization

Overview of a Single Step



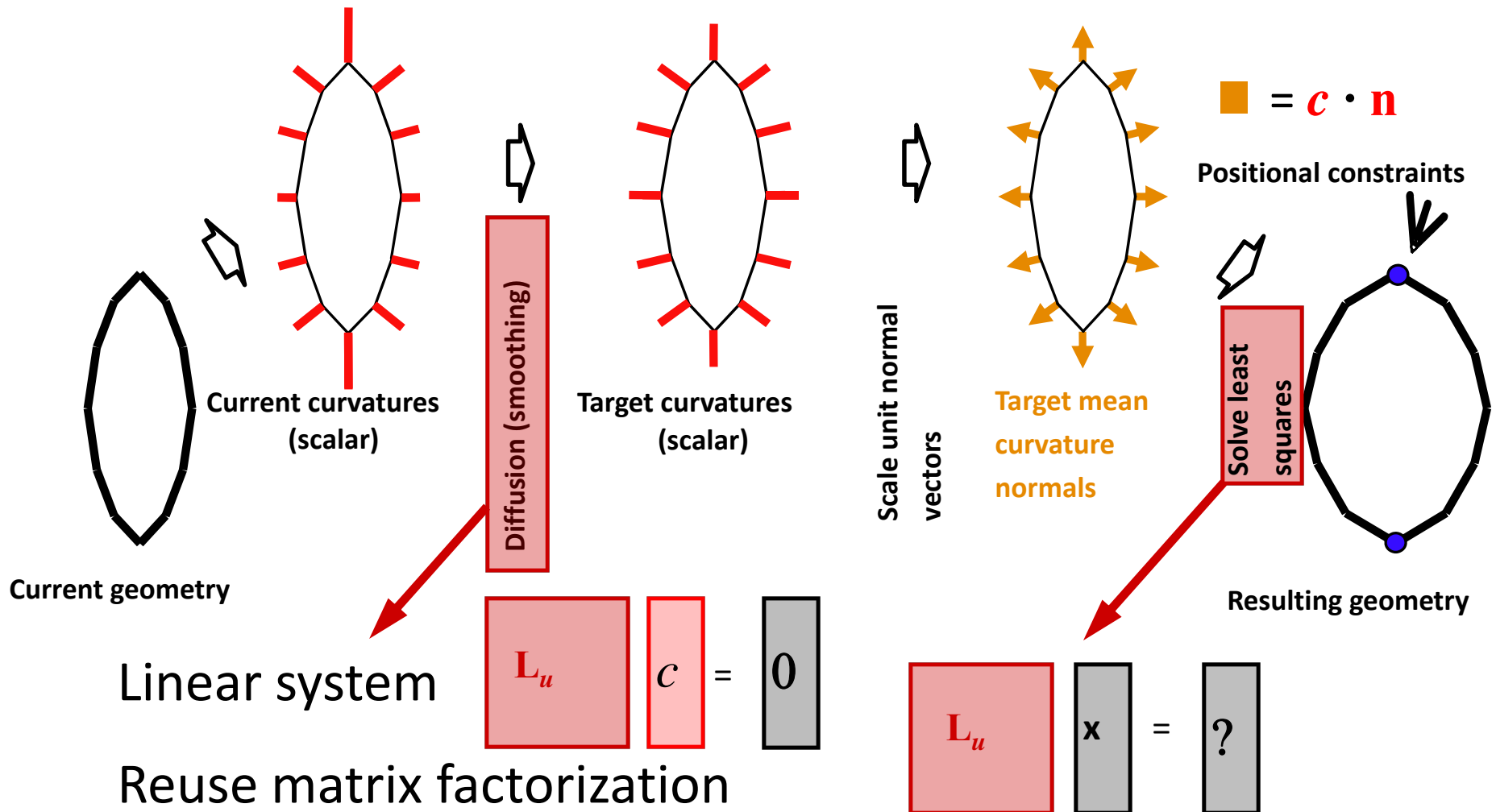
Surface Optimization

Our Optimizations and Approximations



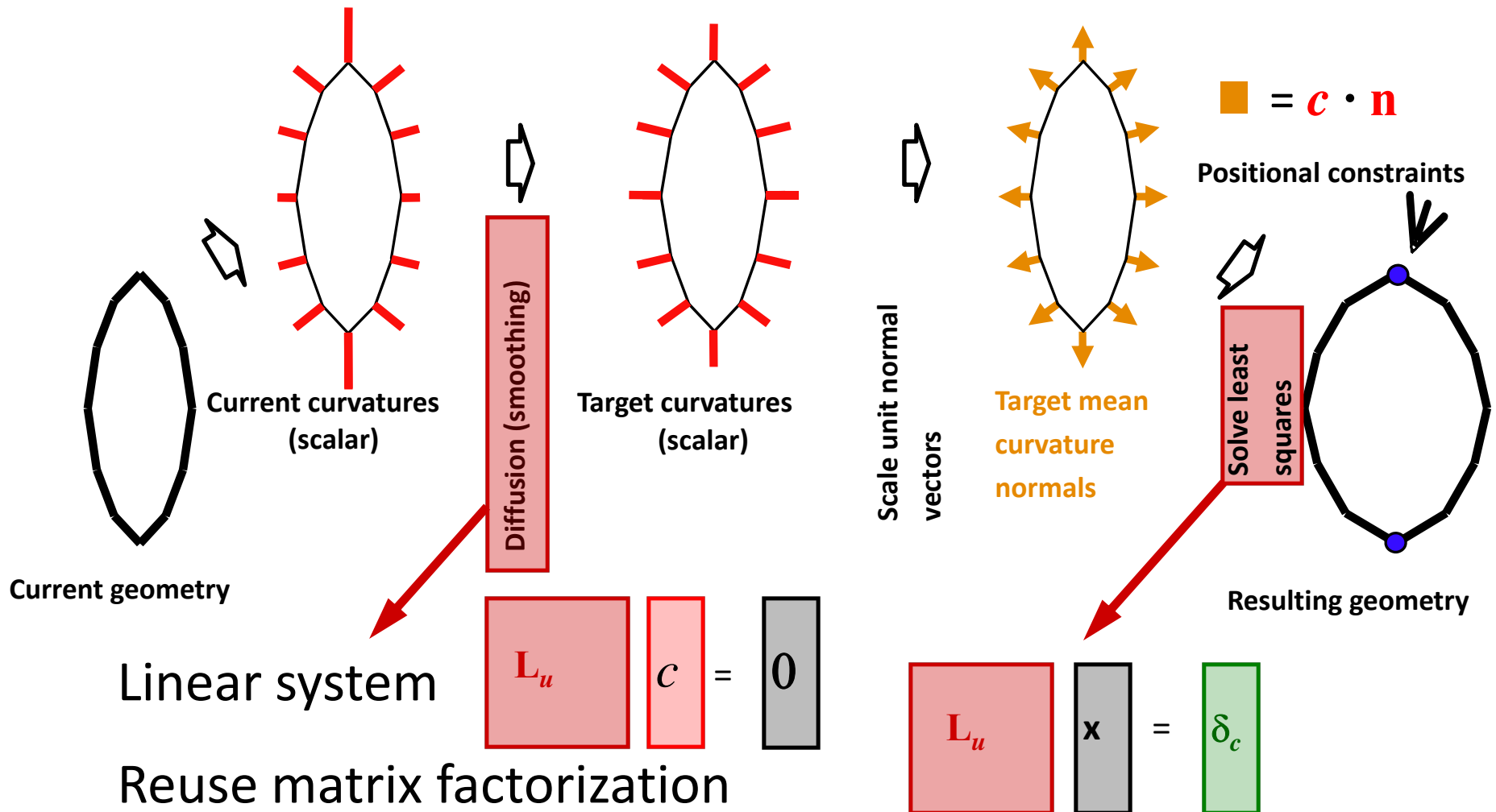
Surface Optimization

Our Optimizations and Approximations



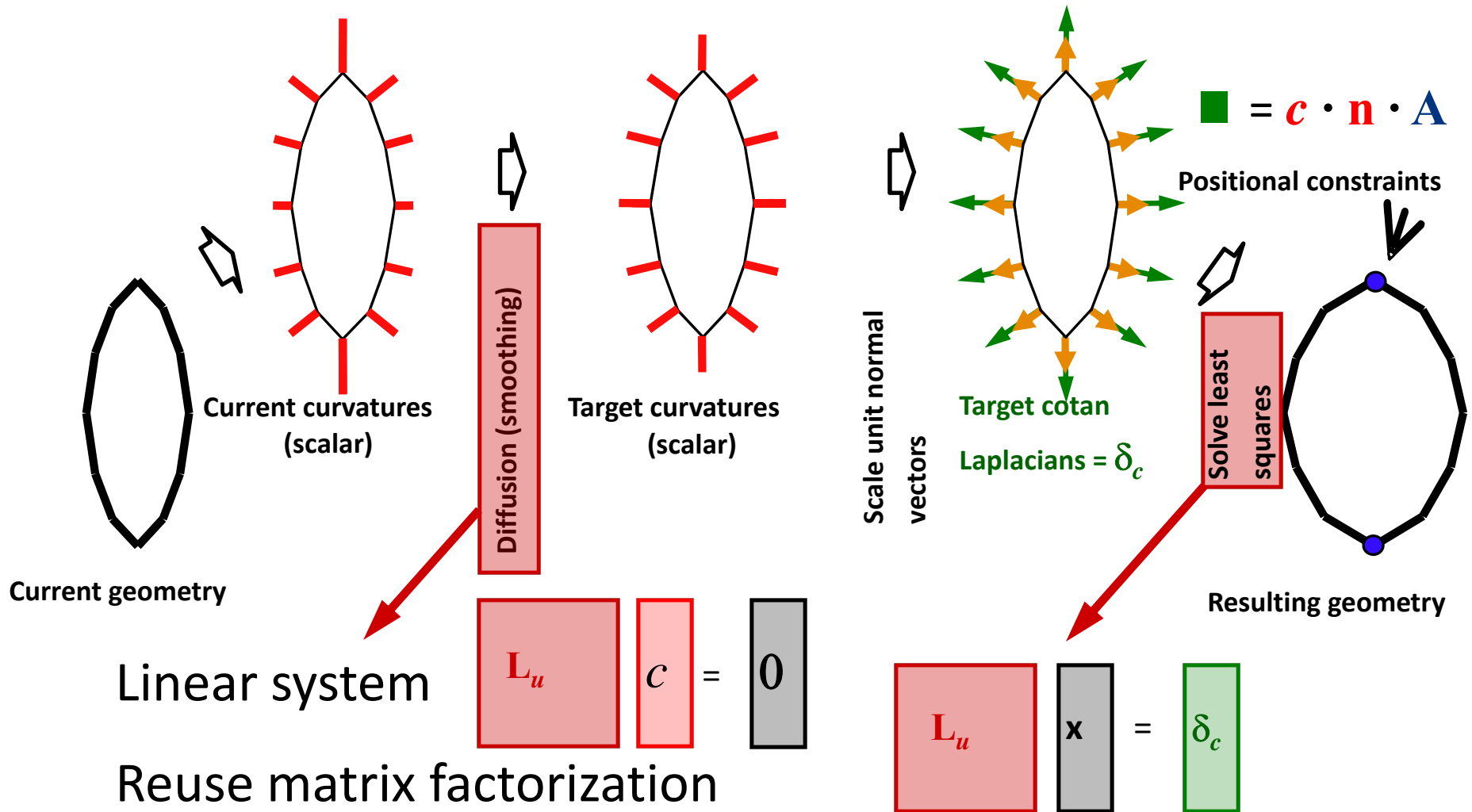
Surface Optimization

Our Optimizations and Approximations



Surface Optimization

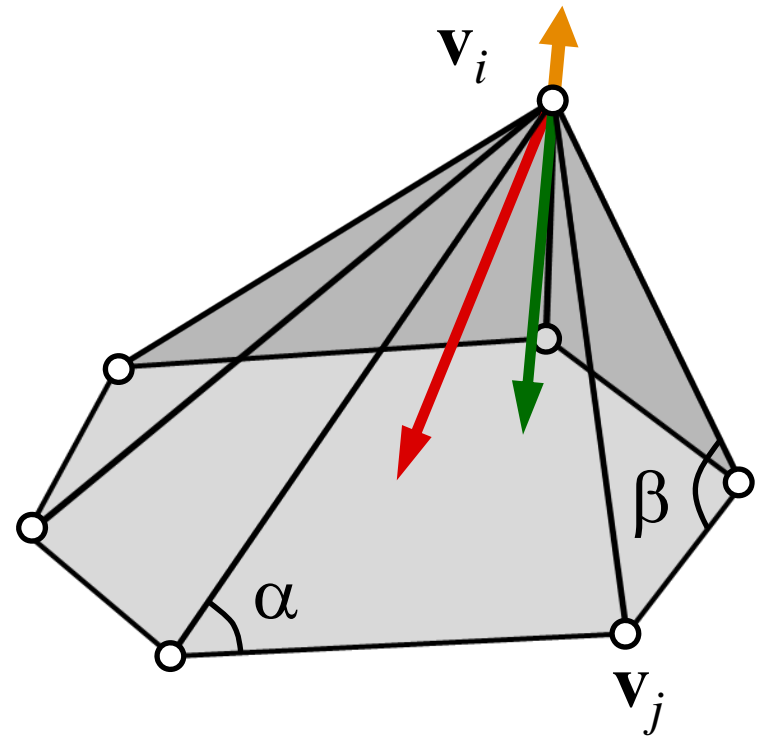
Our Optimizations and Approximations



Surface Optimization

Requirements for our Approximations

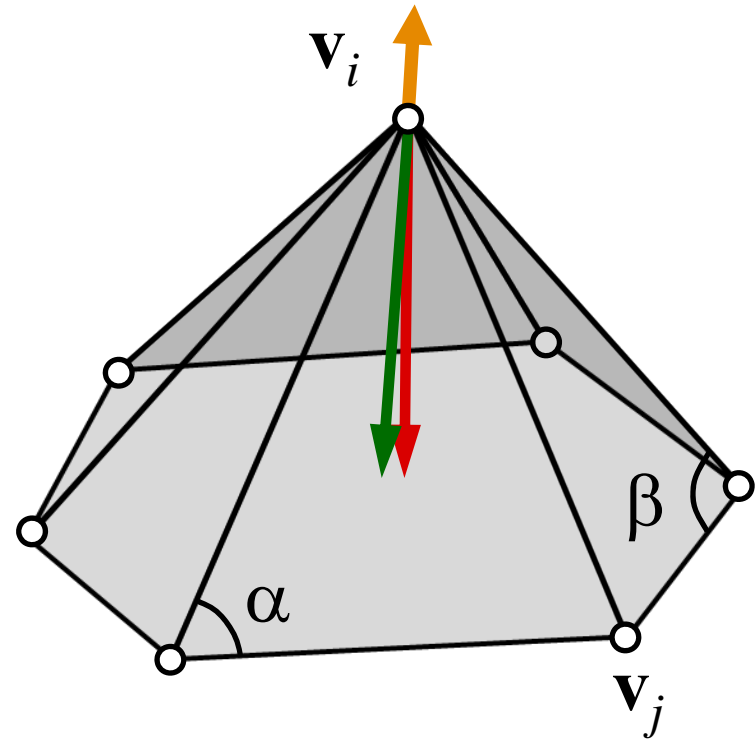
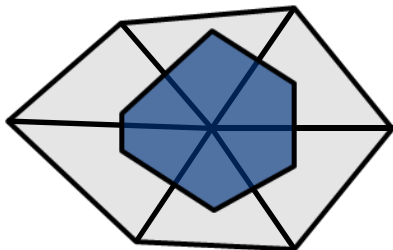
- We replace \mathbf{L}_c with \mathbf{L}_u in both steps
- This a viable approximation for:



Surface Optimization

Requirements for our Approximations

- We replace \mathbf{L}_c with \mathbf{L}_u in both steps
- This a viable approximation for:
 - (A) nearly equal edge lengths (smooth metric)
 - (B) nearly equal vertex areas



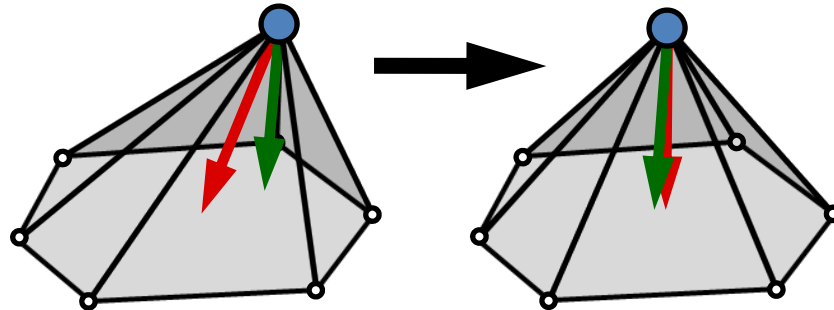
Surface Optimization

How to Enforce (A) and (B)

Setting $\mathbf{L}_u(\mathbf{x})$ equal to δ_c improves inner fairness

$$\mathbf{L}_u \mathbf{x} = \delta_c$$

[Nealen et al. 2006]

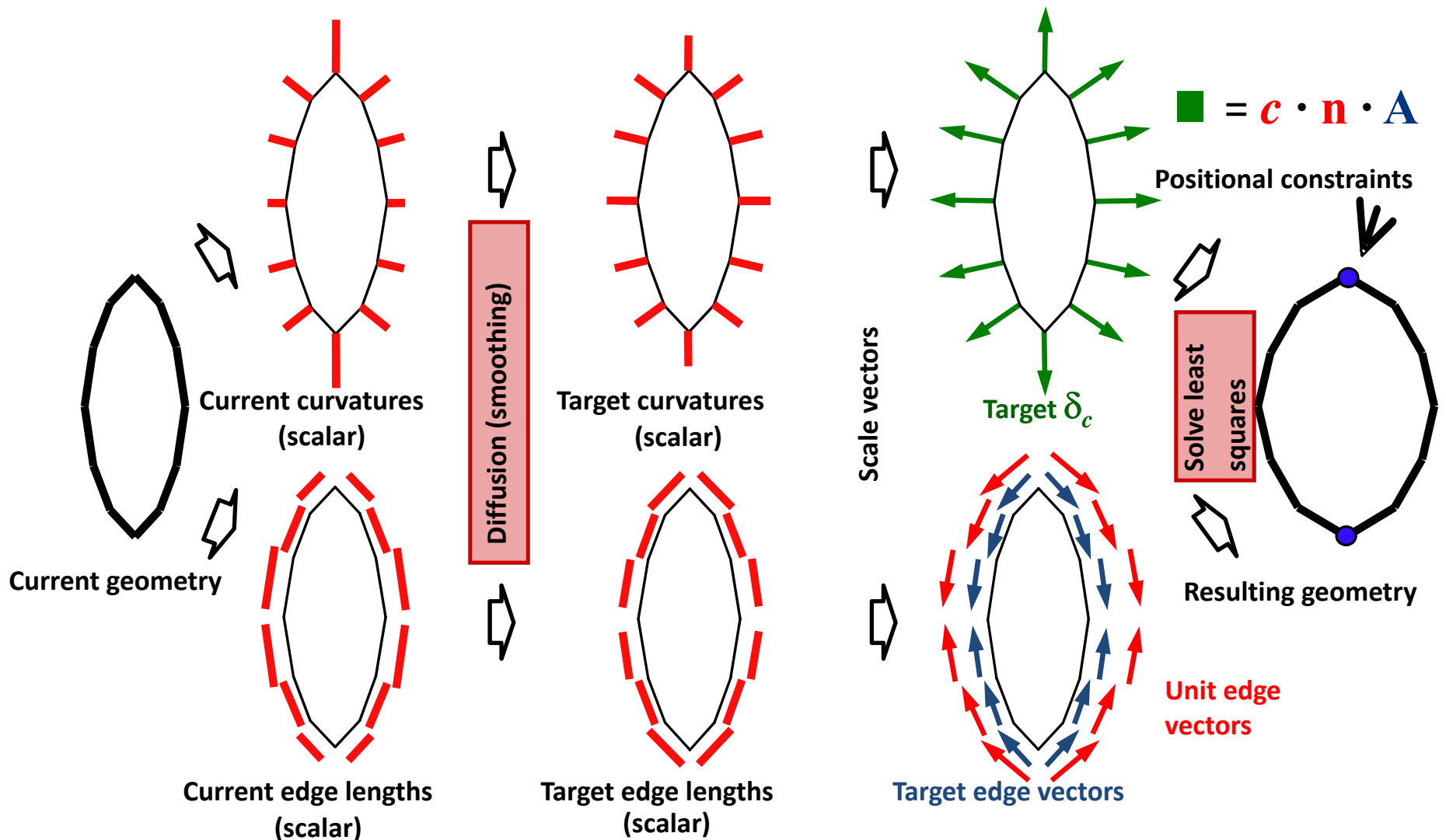


Smooth the edge lengths

Constrain edge vectors

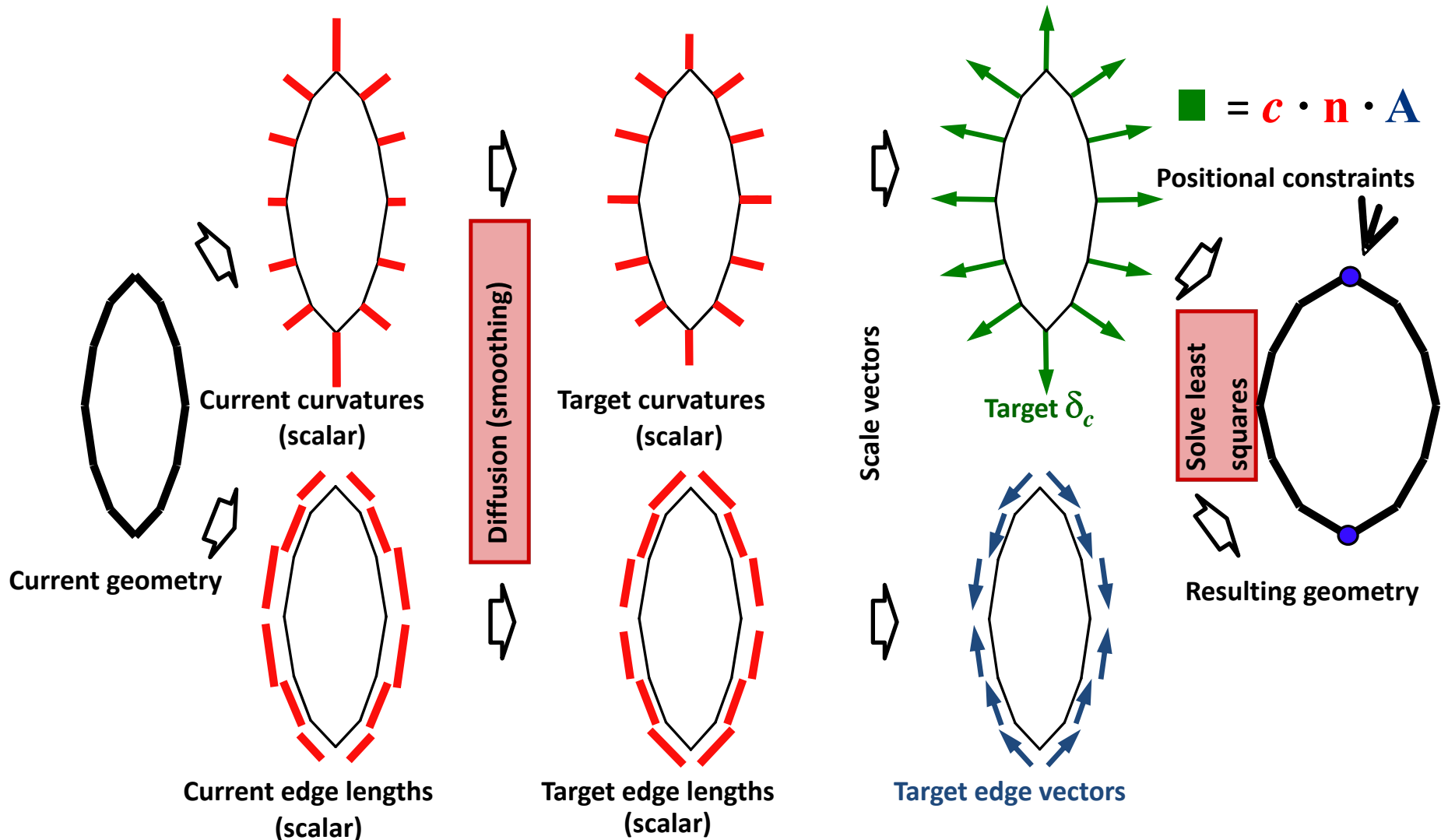
Surface Optimization

Adding Edge Vector Constraints to LS

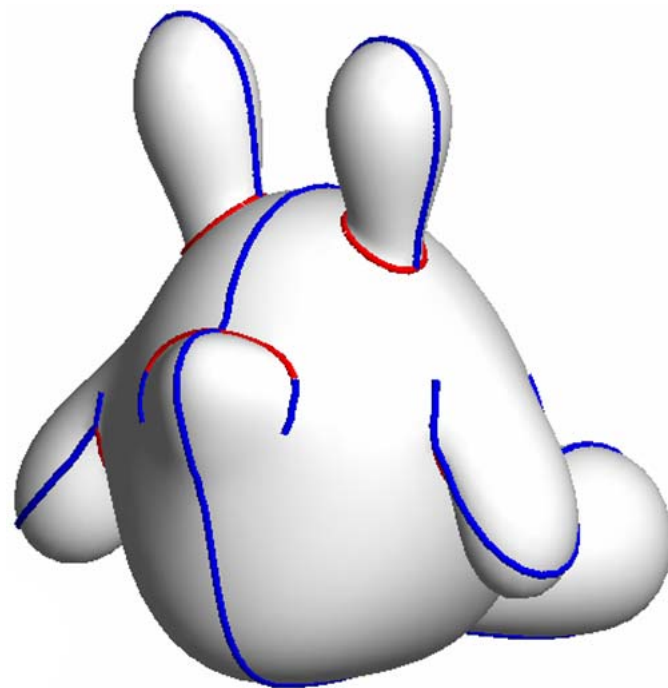
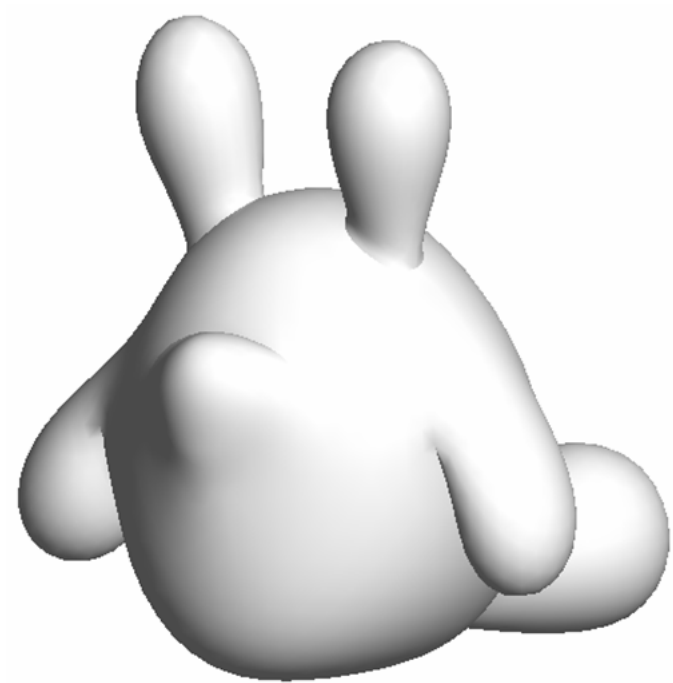


Surface Optimization

A Single Step of the Optimization

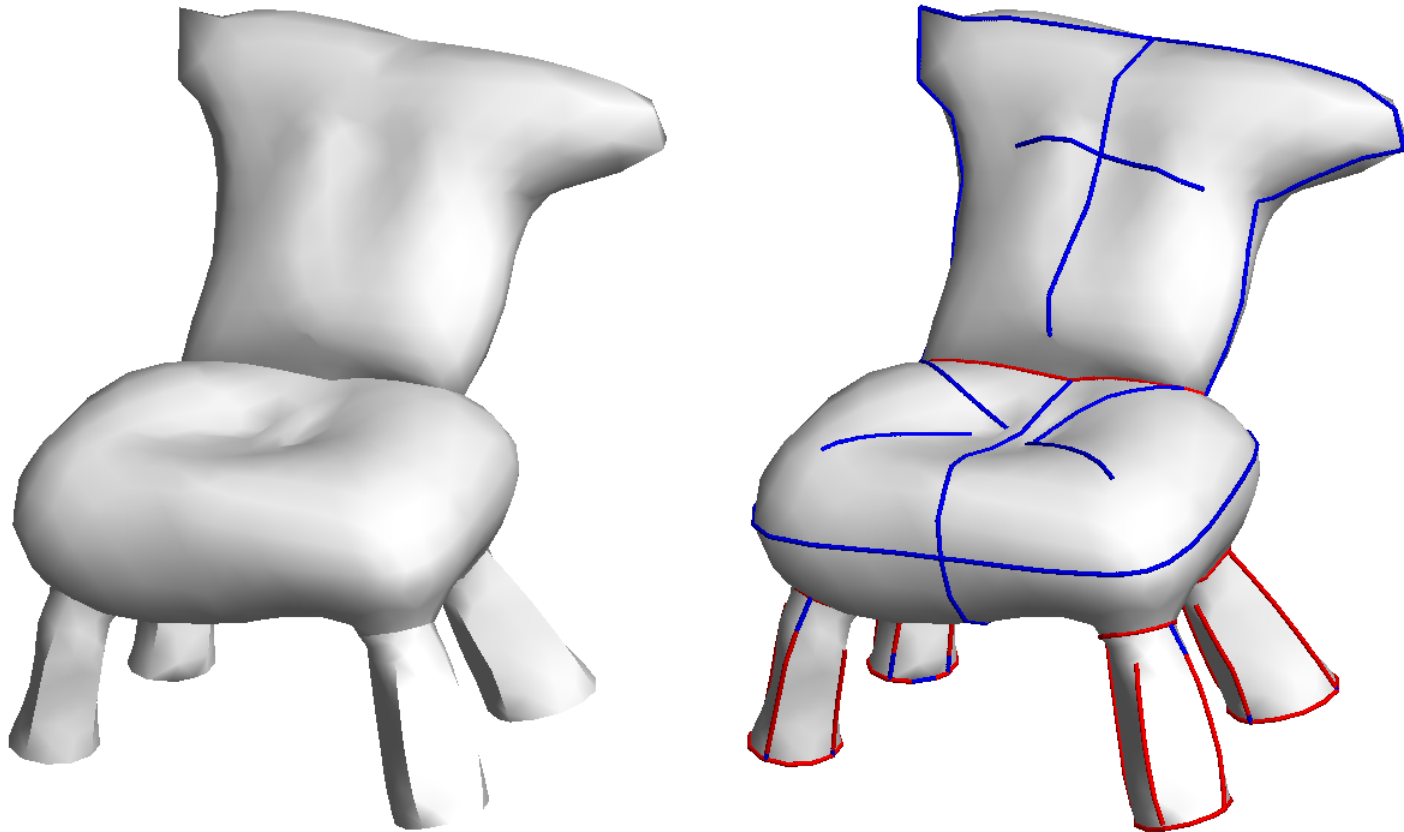


Results



Results

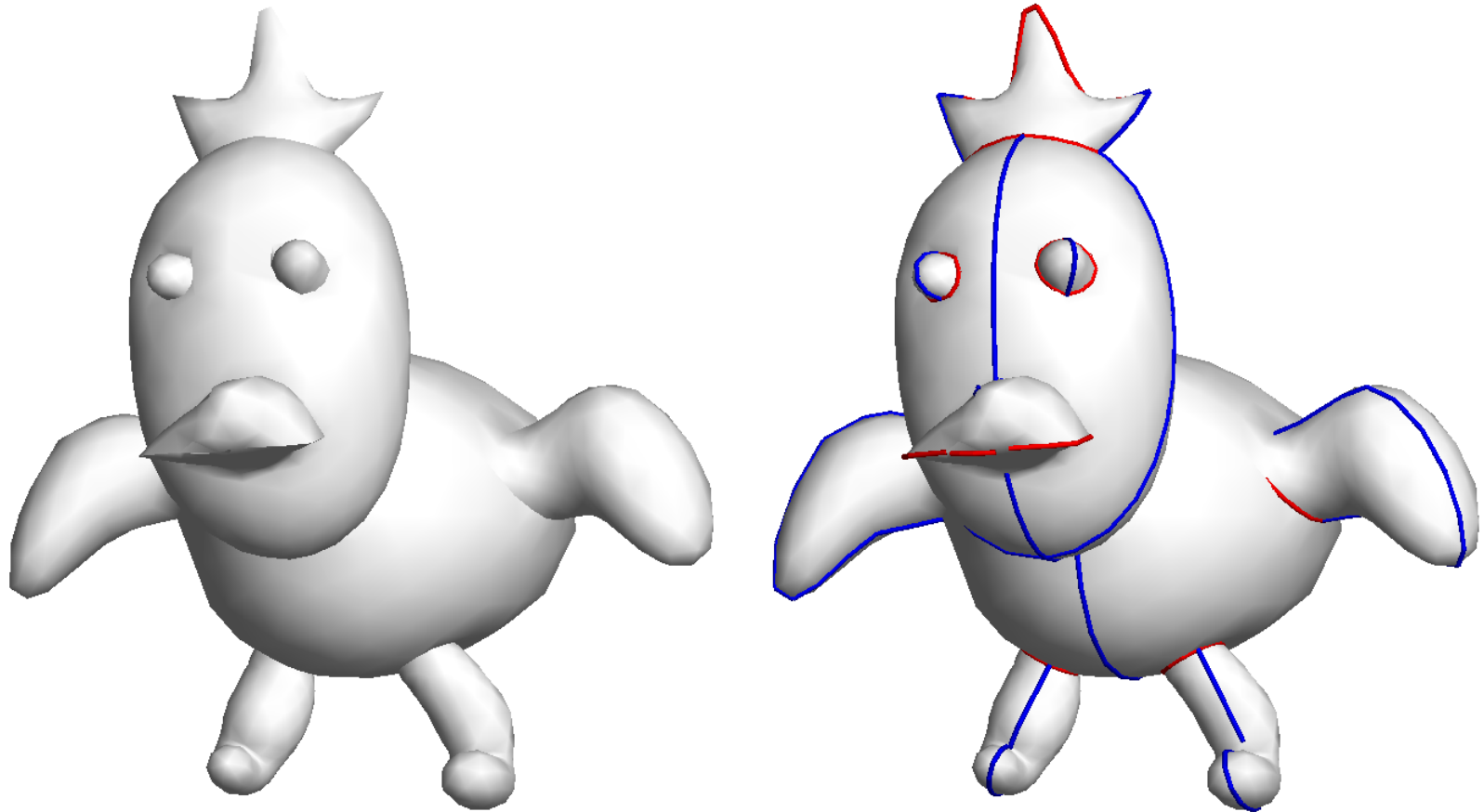
Novice User



15 min instructions + 20 minutes tryout

Results

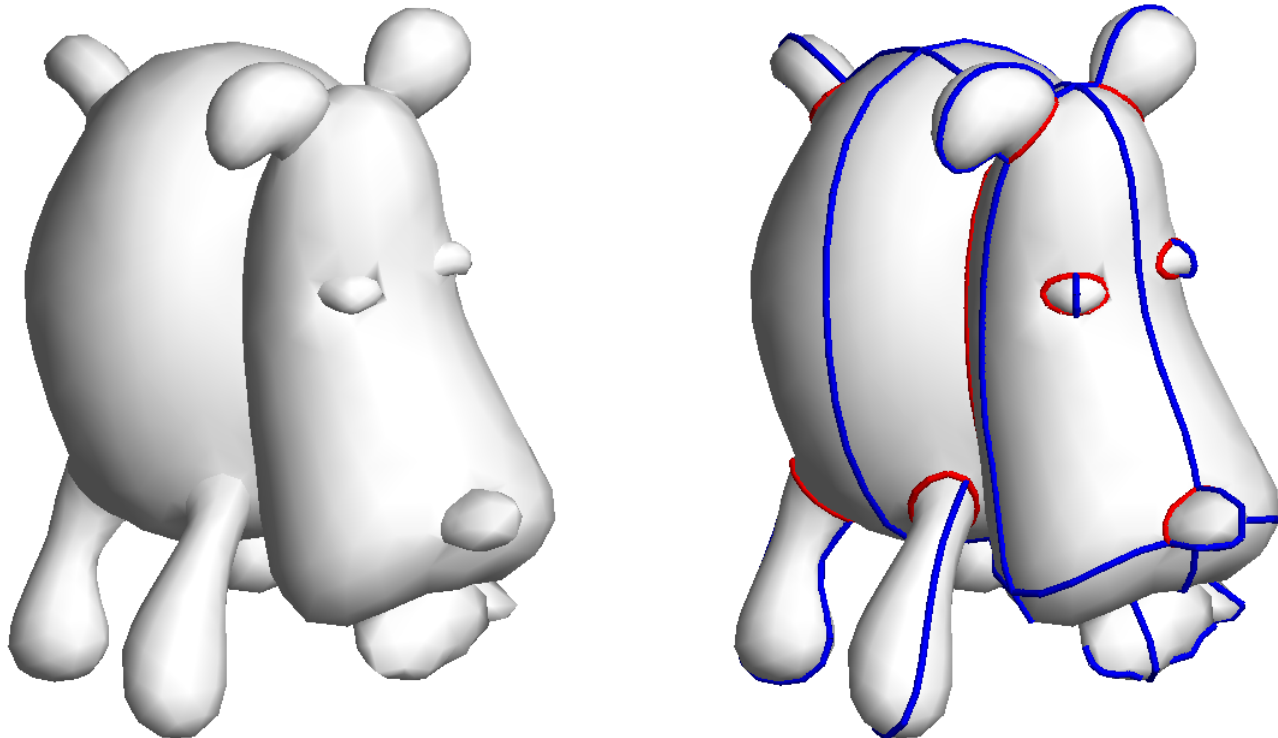
2D Artist



15 min intructions + 10 minutes tryout

Results

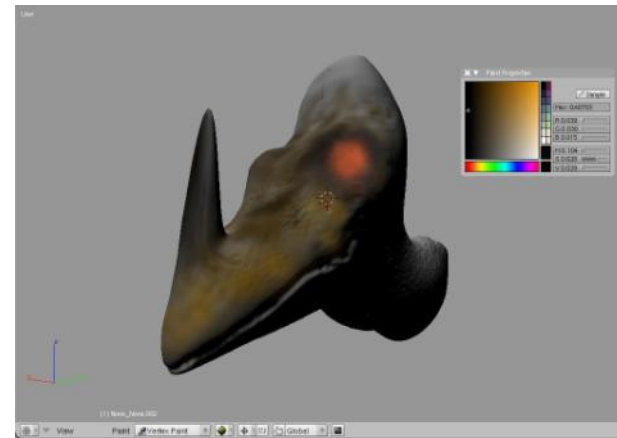
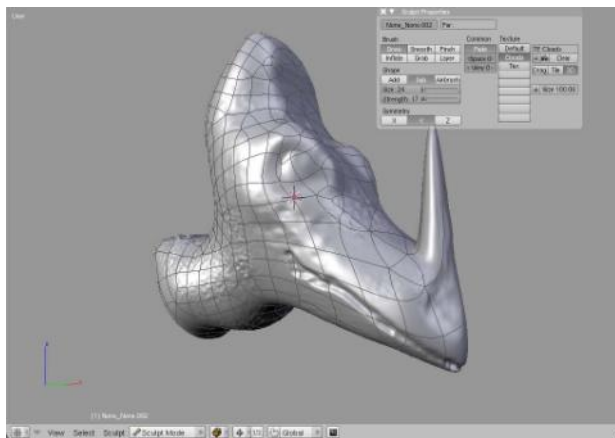
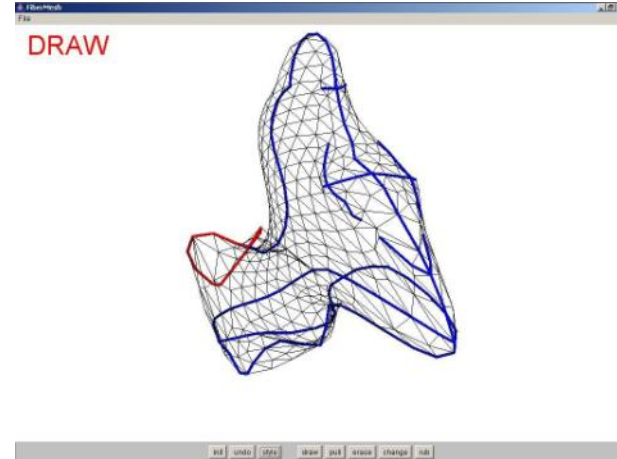
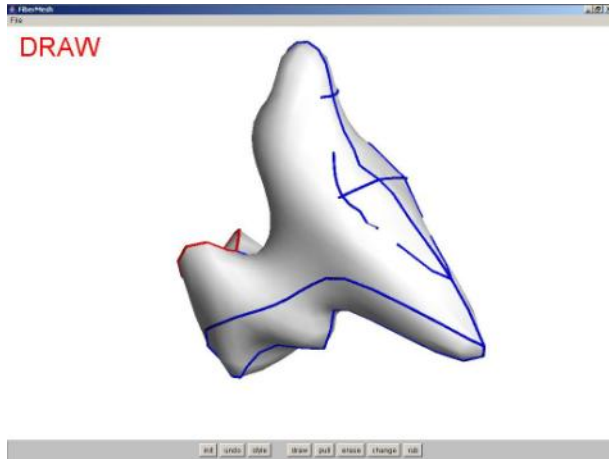
2D Artist



... + 20 minutes tryout

Results

Base Model Creation



Discussion

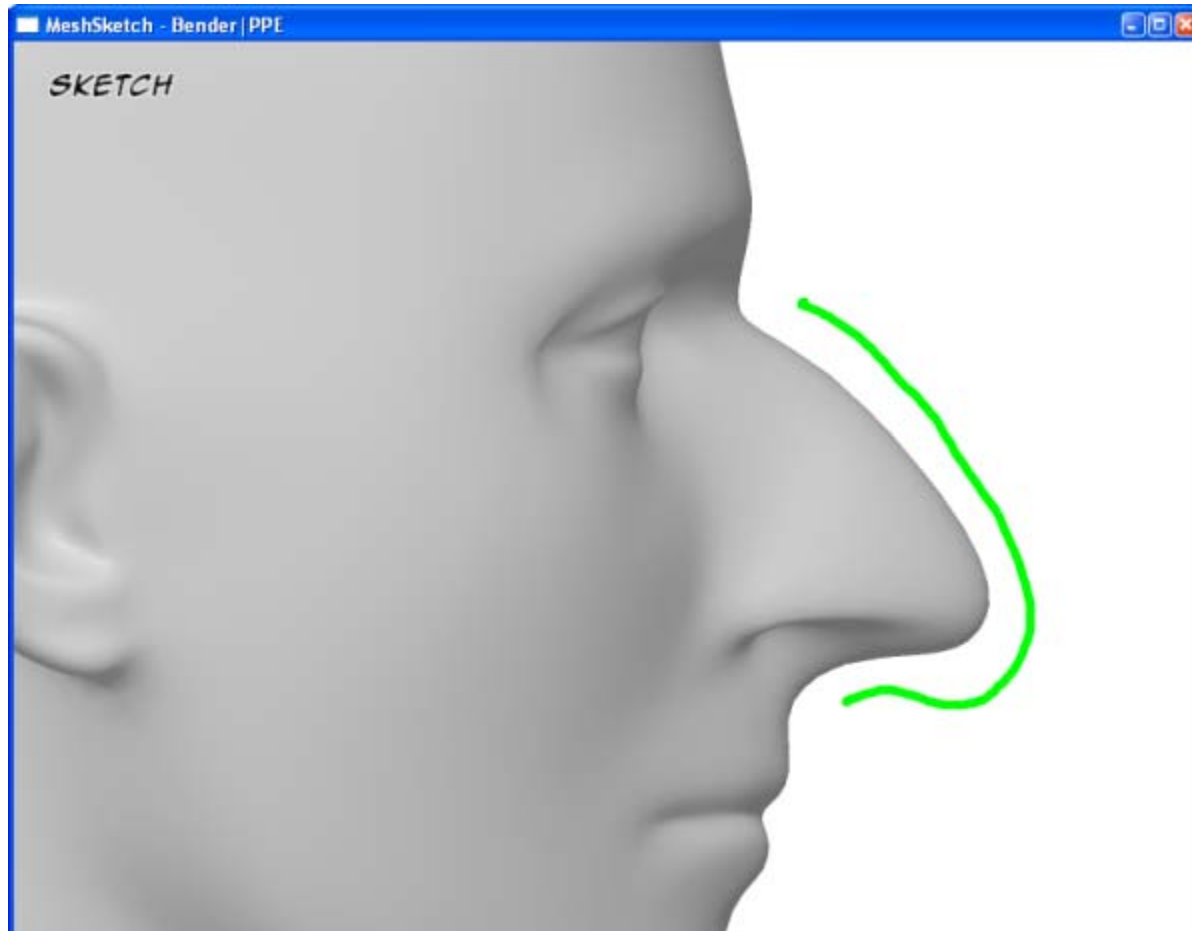
To be Improved...

- Takes a bit to learn the interface operations and their combinations
- Larger meshes would be nice
 - Perhaps „freeze“ part of the mesh
In general: entire mesh is optimized
 - Multigrid / Multiresolution acceleration techniques
- Add more intuitive modeling operations: symmetry, vertex snapping, merging, etc.

SilSketch

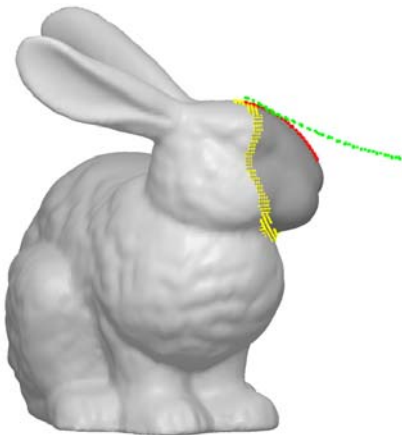
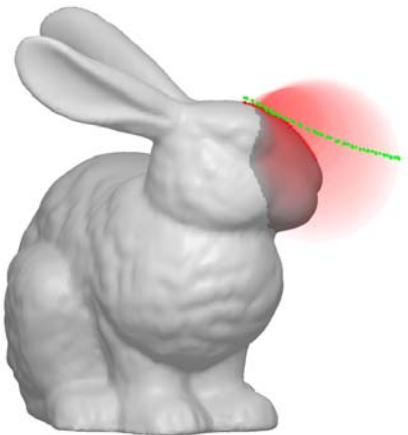
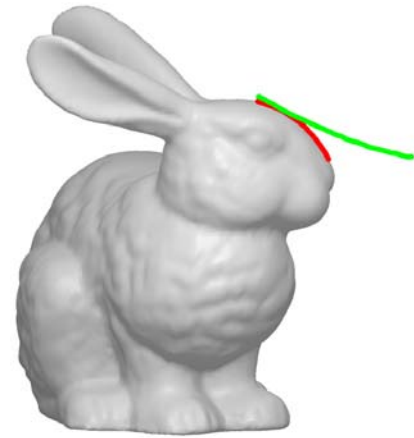
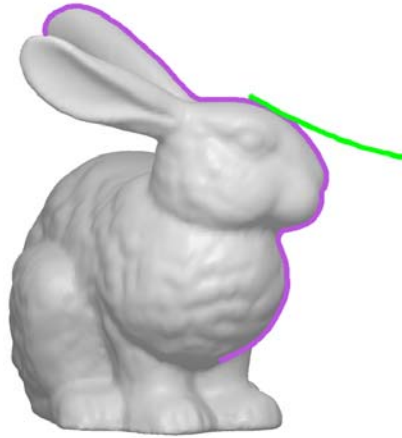
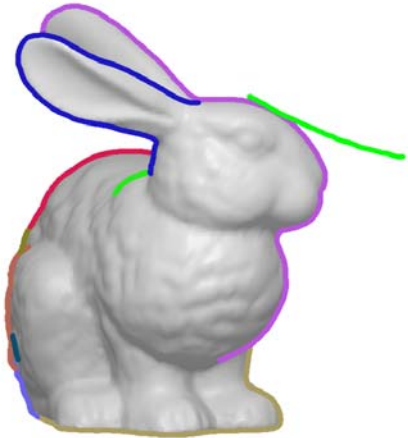
SilSketch

Demo



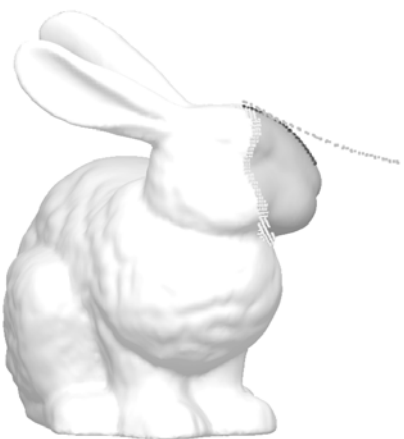
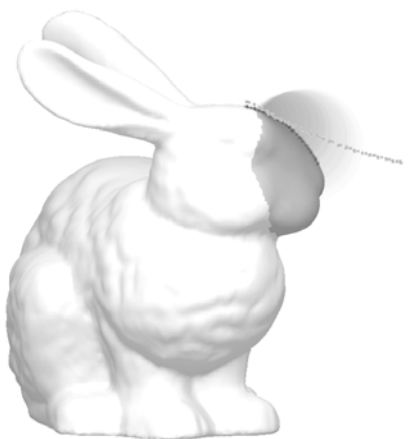
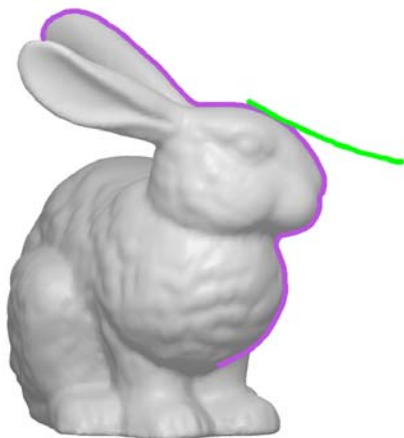
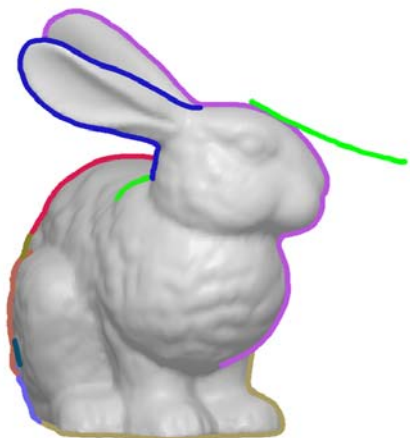
SiSketch

Algorithm Overview



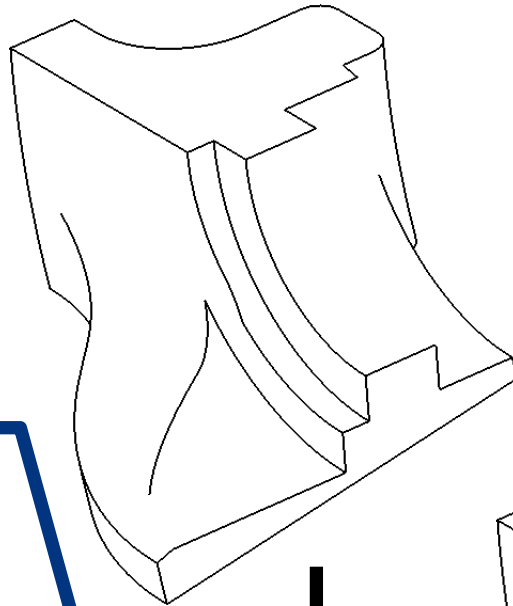
Sil Detection and Segmentation

Image Space

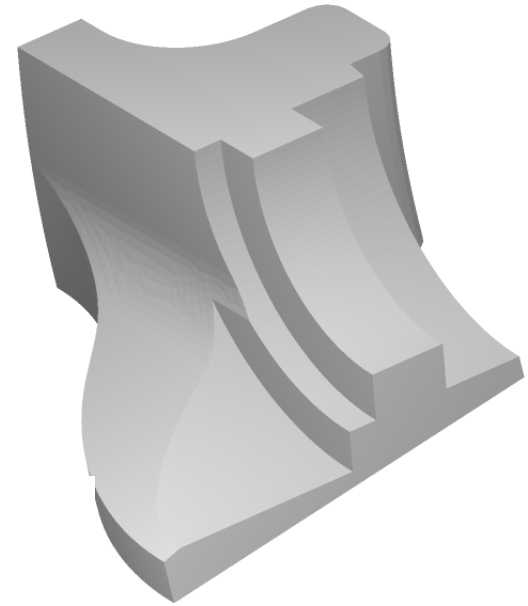


Which Lines to Match Against ?

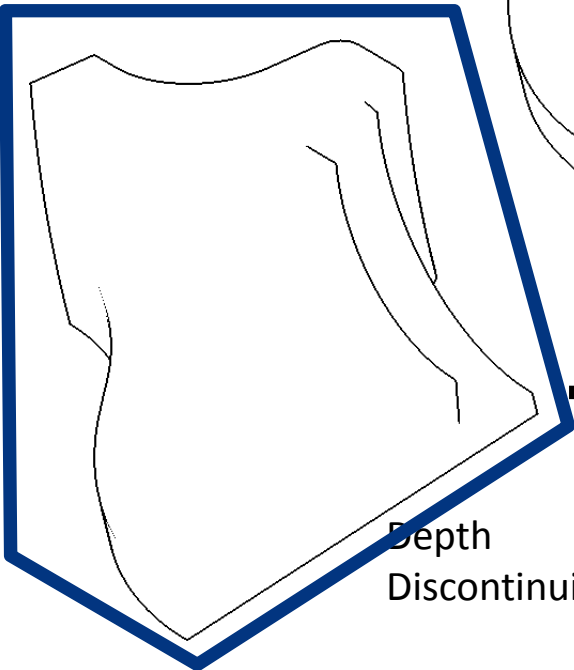
Normal
Discontinuities



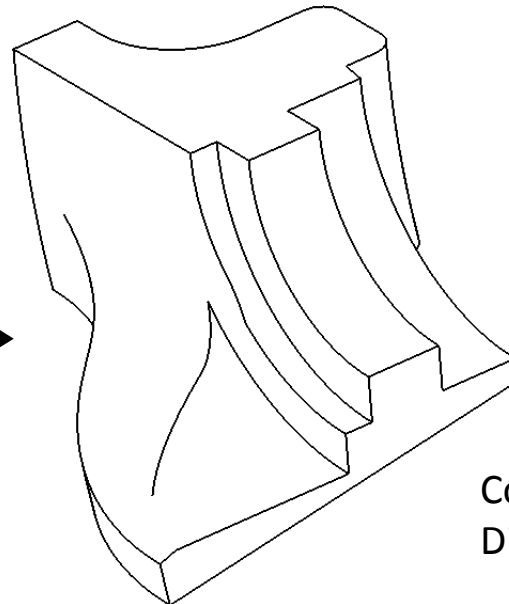
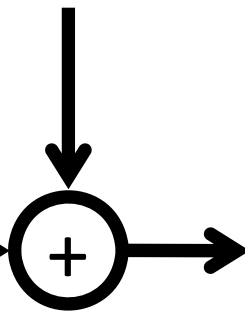
Flat
Shaded



We Use

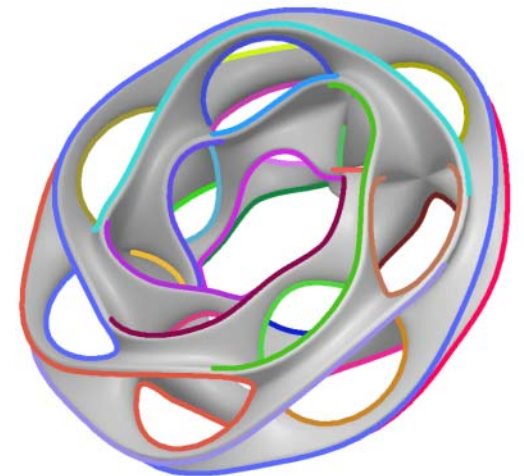
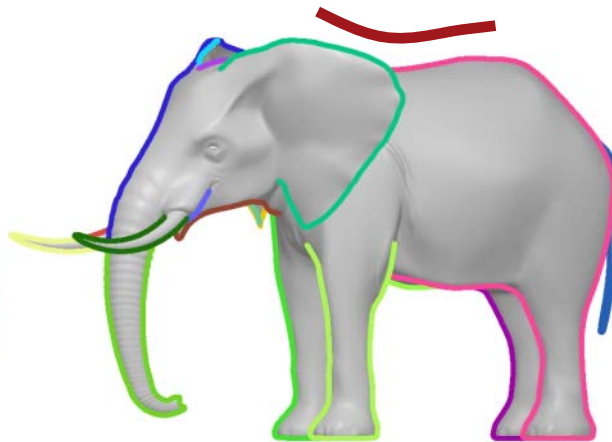
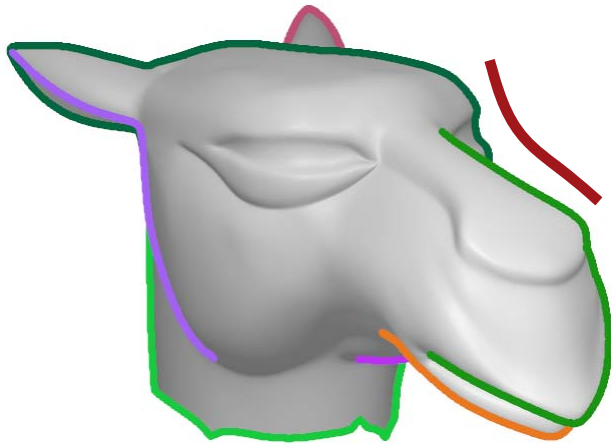
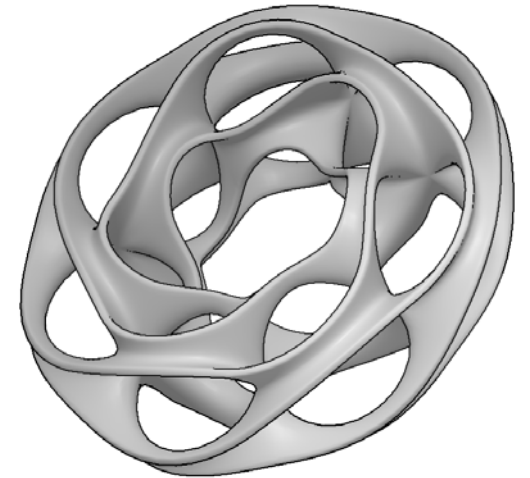
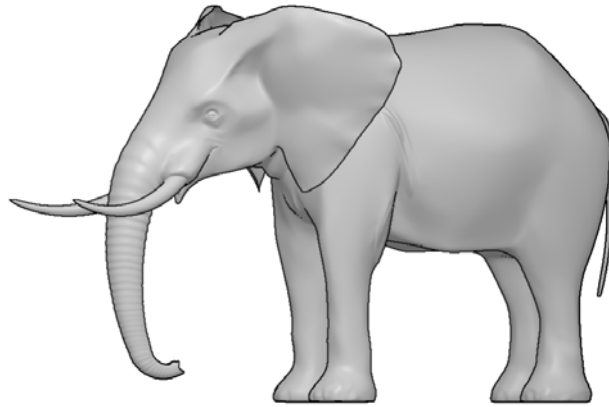
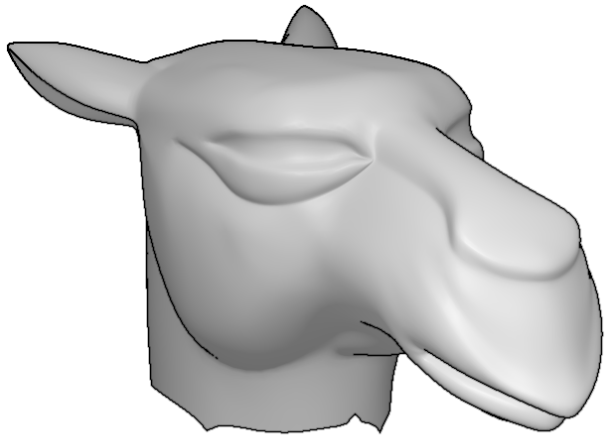


Depth
Discontinuities



Combined
Discontinuities

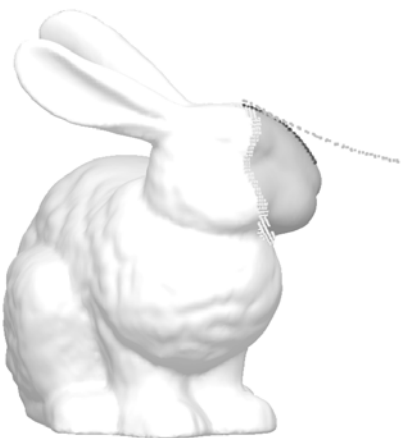
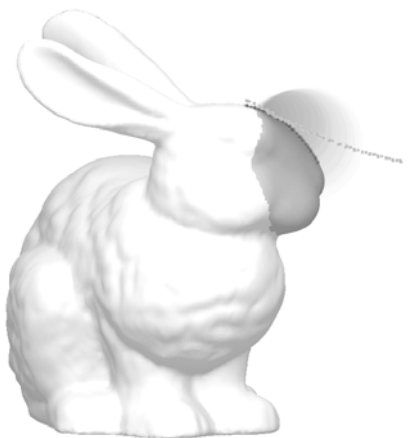
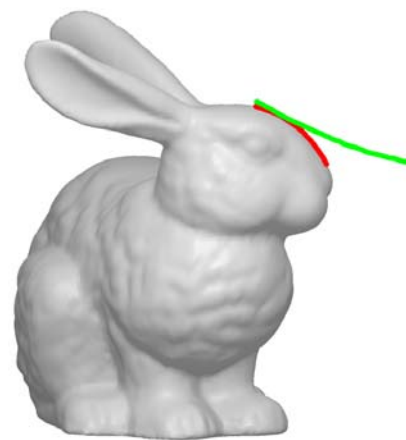
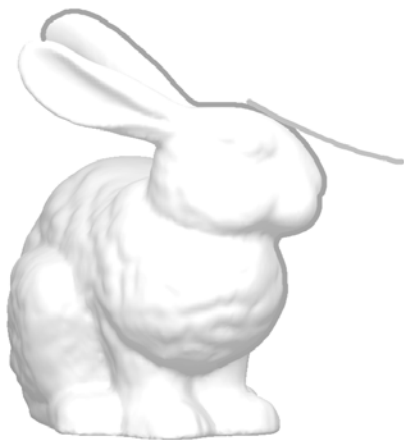
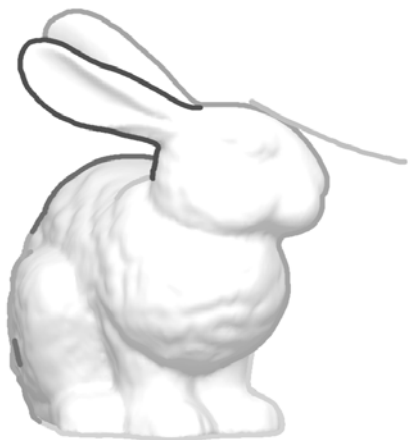
Some Segmentation Results



Resolve depth ambiguity

Partial Matching

Image Space



Partial Matching

Input

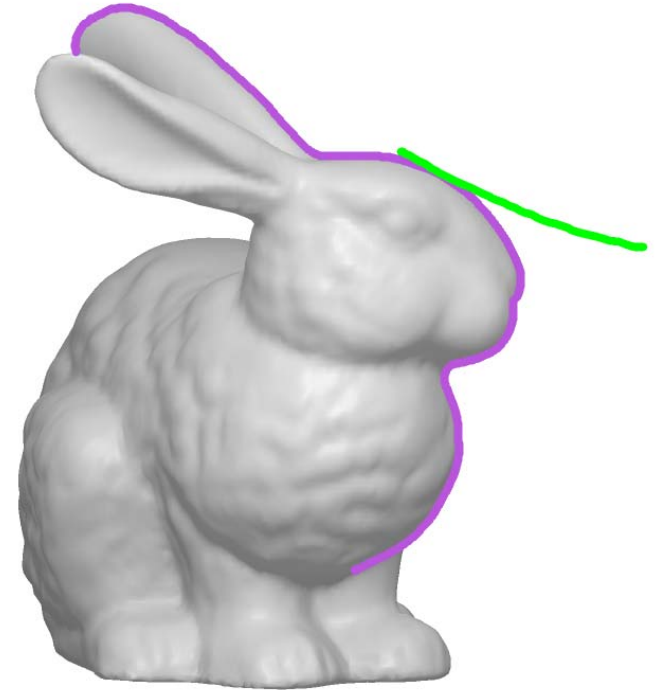
- Silhouette polyline(s)
- User stroke (polyline as well)

Criteria

- Proximity
- Shape similarity

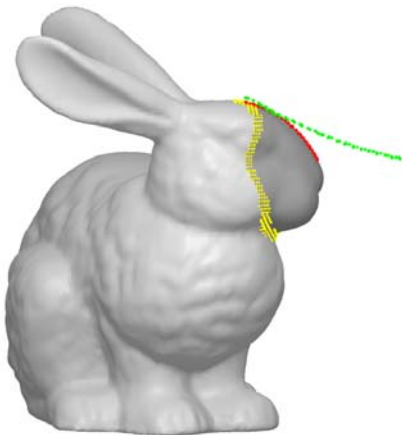
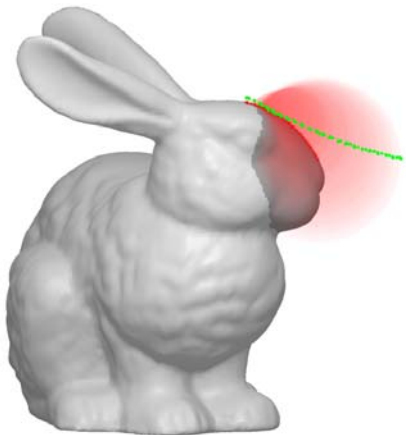
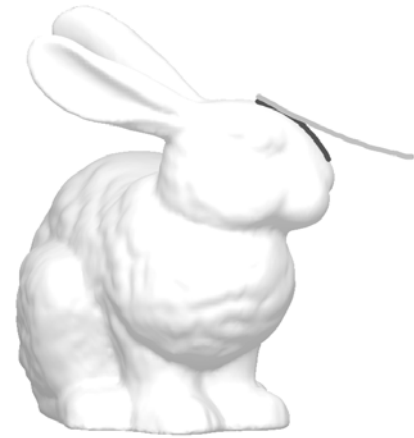
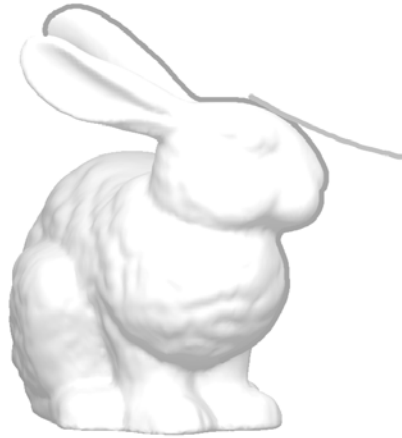
Shape Matching

- Partial Matching of Polylines under Similarity Transformations Cohen and Guibas [1997]



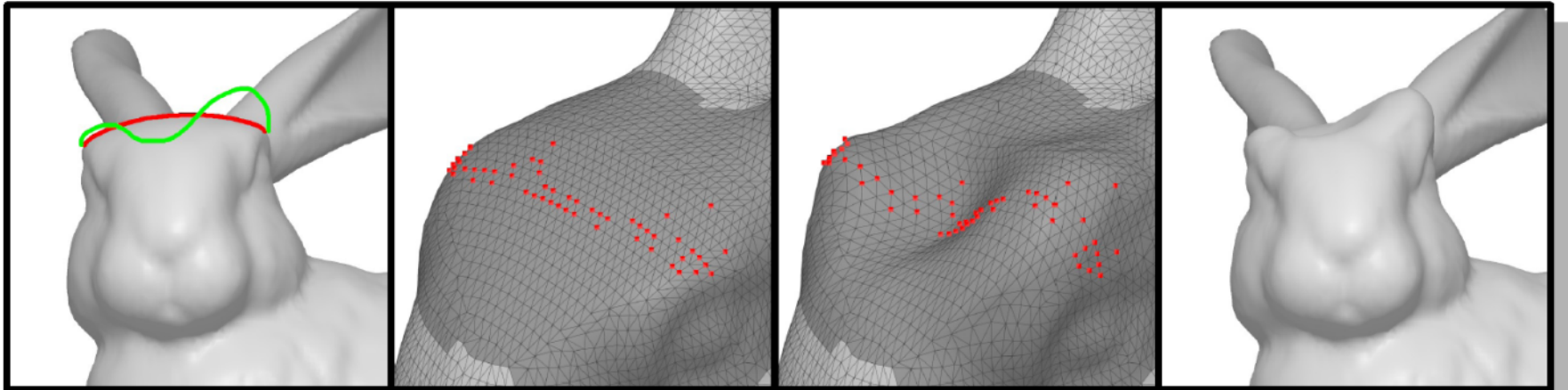
Handle deformation and ROI

Object Space

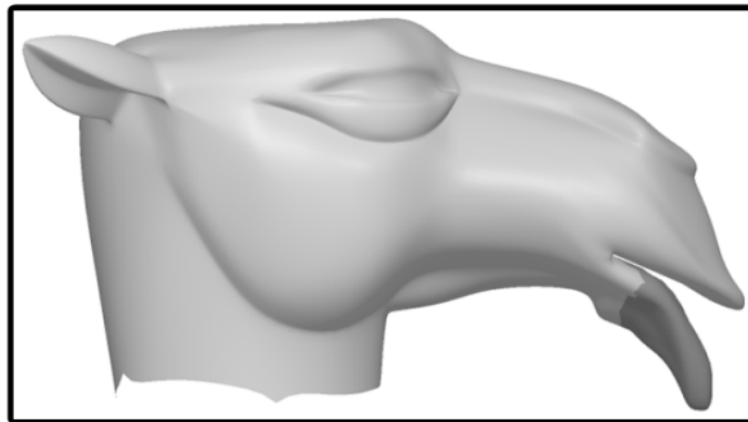
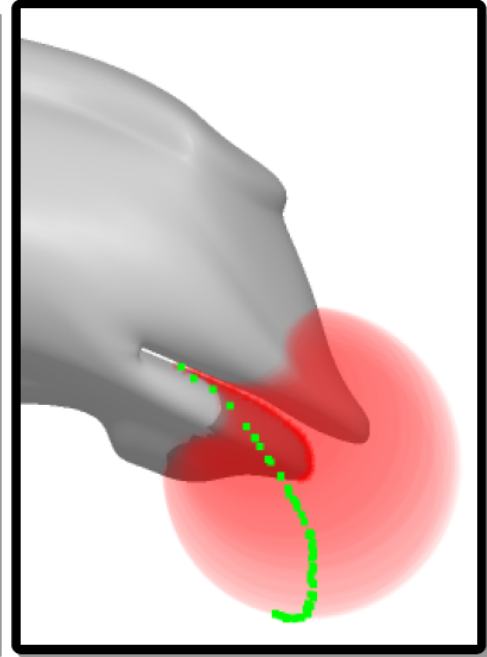
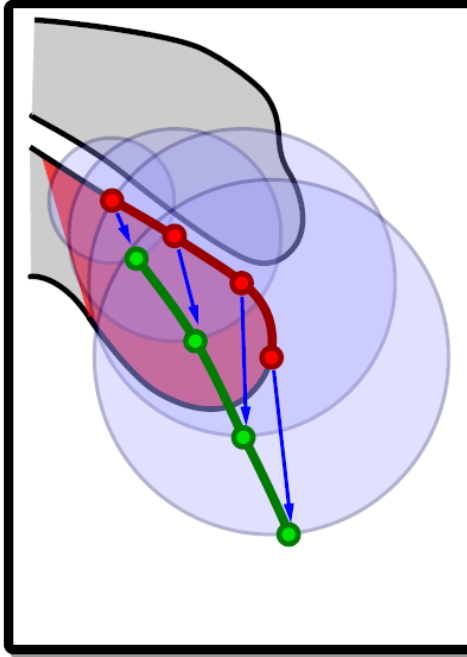
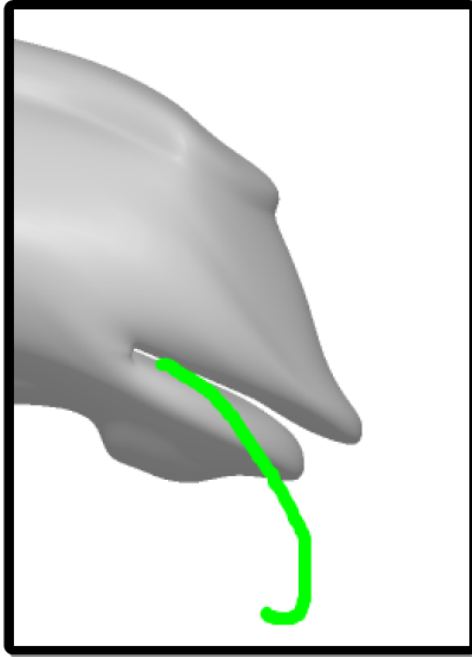


Handle Vertices

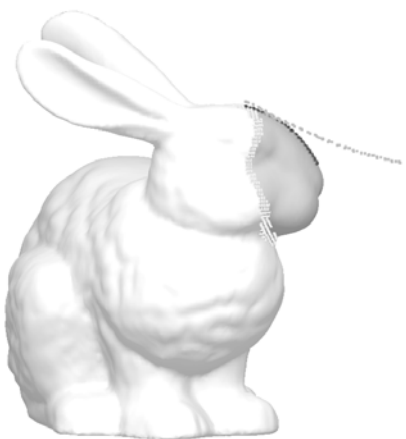
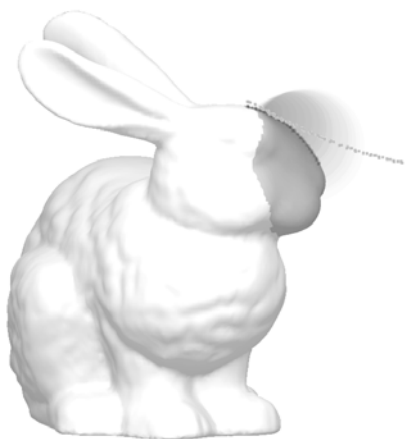
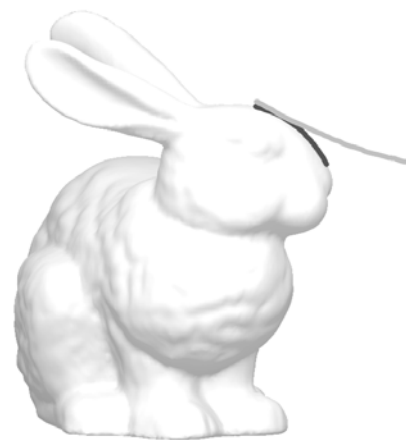
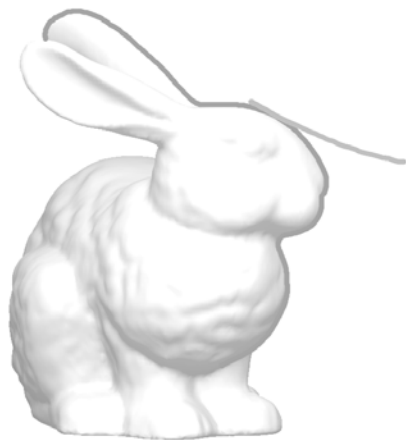
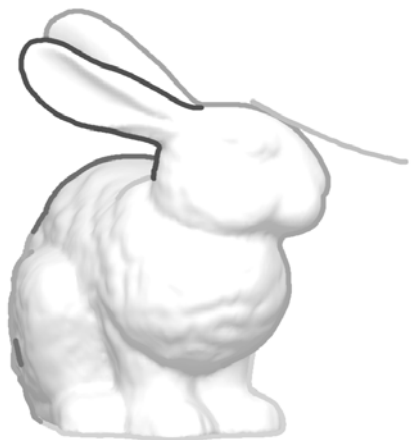
Moving from 2D to 3D



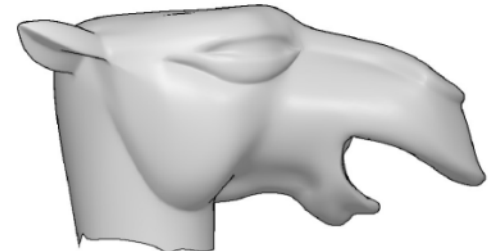
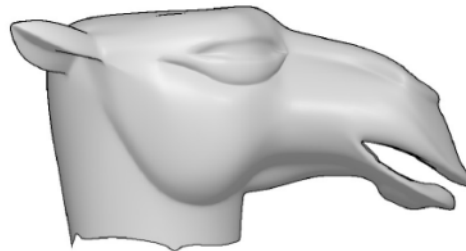
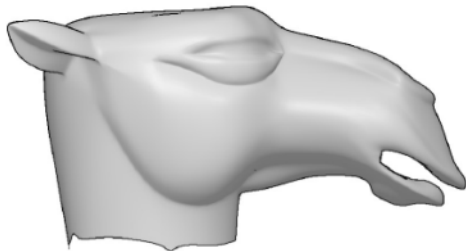
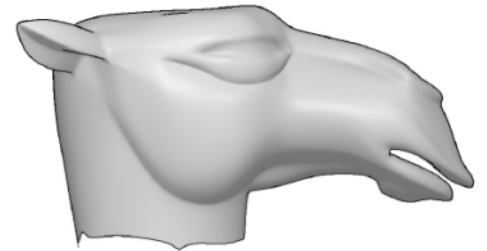
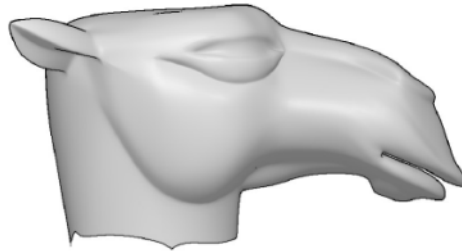
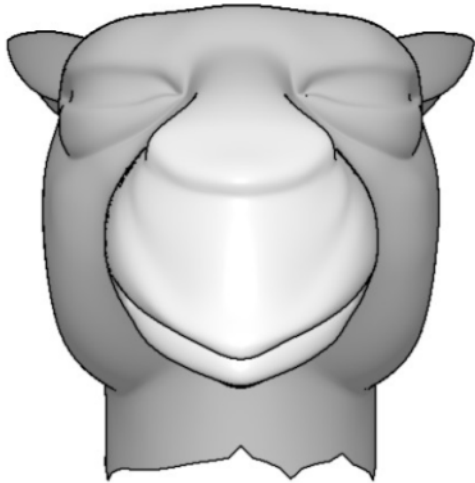
Region of Interest



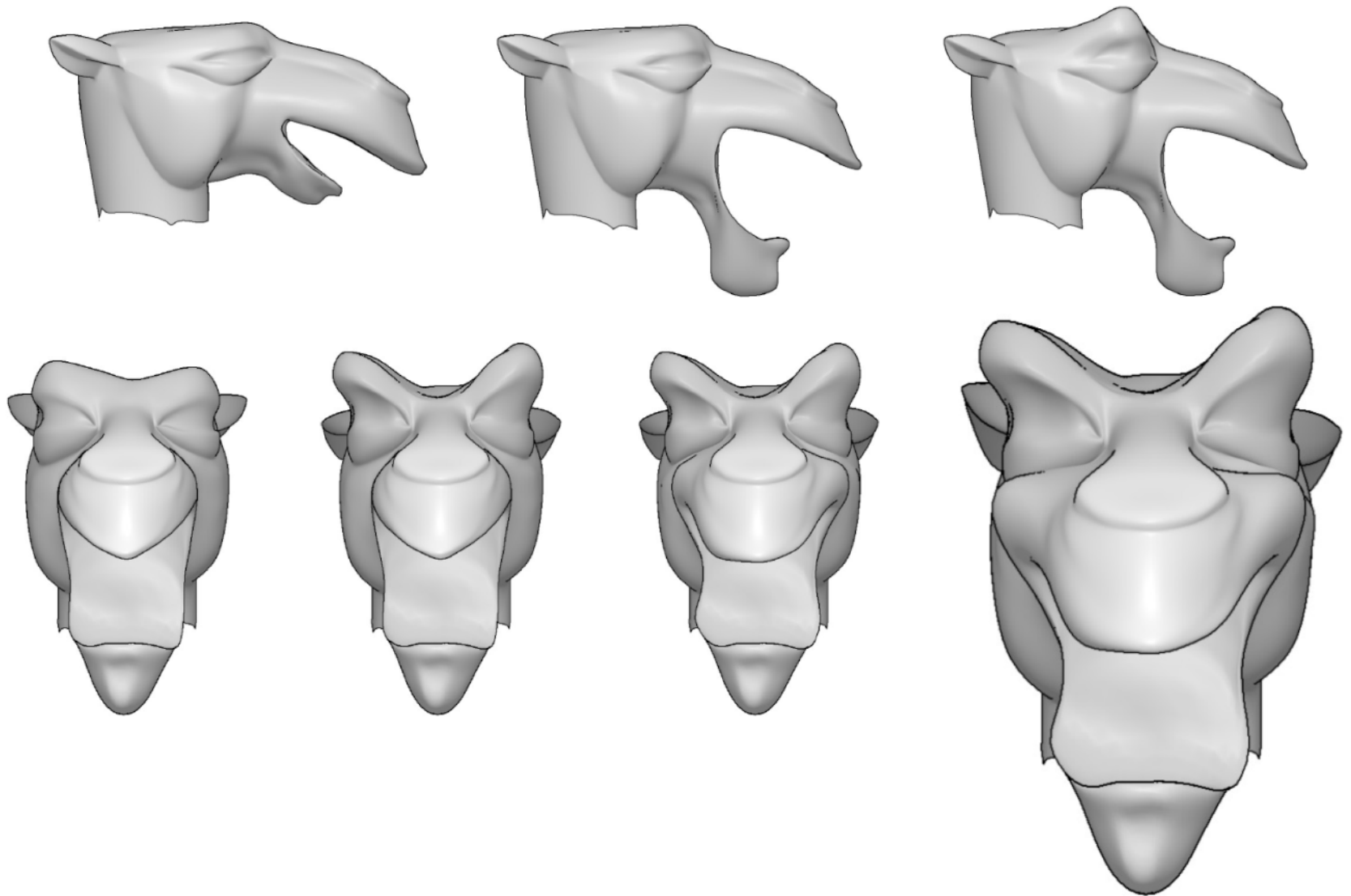
Results



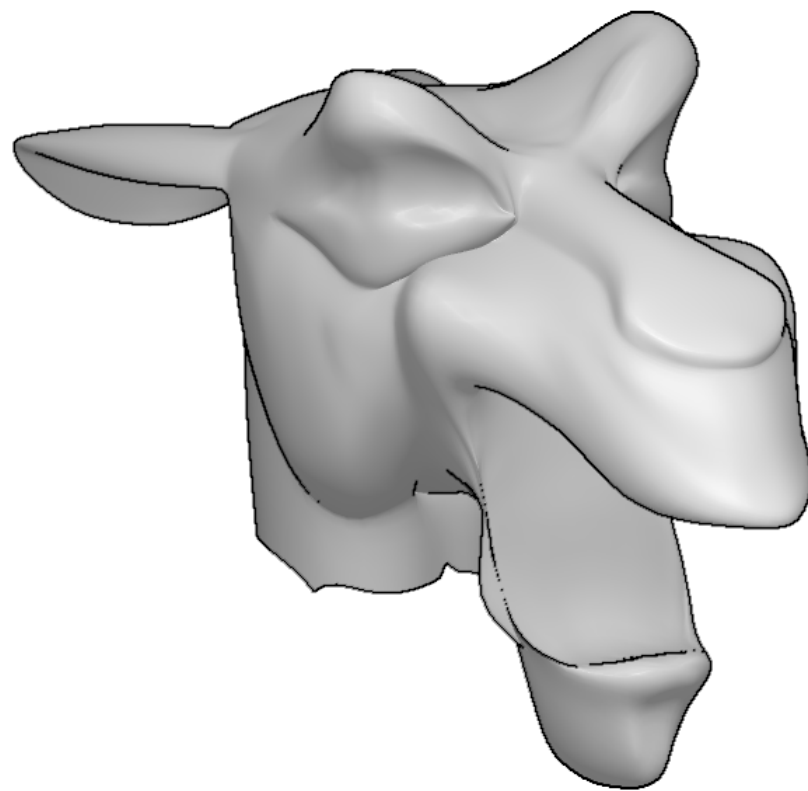
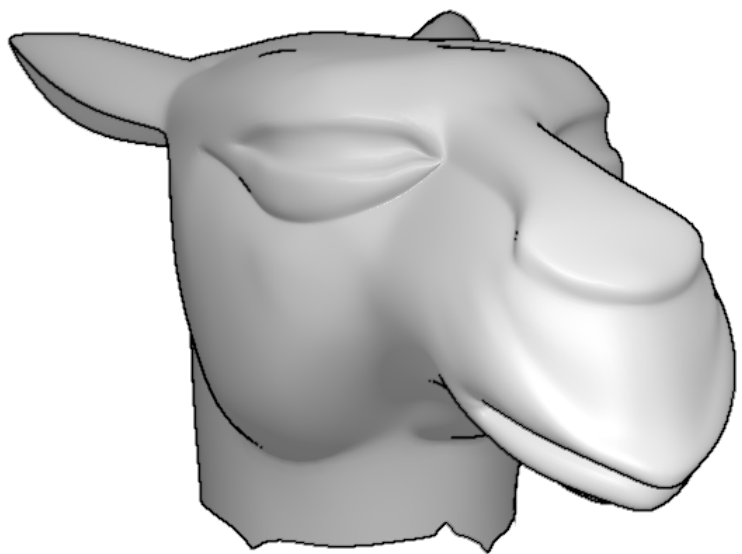
Editing Sequence (1)



Editing Sequence (2)



Editing Sequence Result



SilSketch

Summary

- The human visual system uses silhouettes as the first index into its memory of shapes
 - SilSketch leverages this human “shape database”
 - This enables the creation of diverse shapes for non professionals
- Extends affine handle transformations to general handle warps
- Was added to the content creation pipeline at Disney Animation Studios

Many other works out there...

- Some research

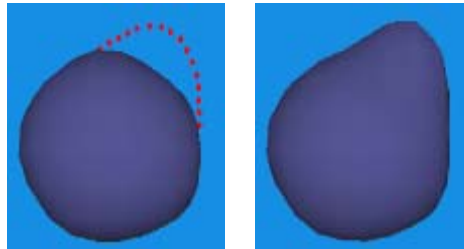
- SKETCH

- [Zelevnik et al. 1996]



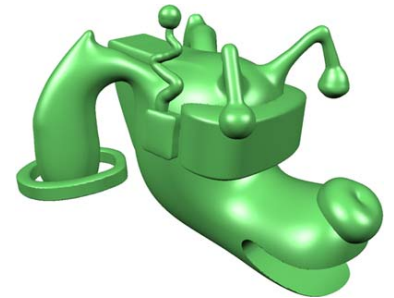
- Variational
implicit

- [Karpenko et al. 2002]



- ShapeShop

- [Schmidt et al. 2005]



Conclusions

- Further step(s) towards a **human “video out”**

