252-0538-00L, Spring 2018

Shape Modeling and Geometry Processing

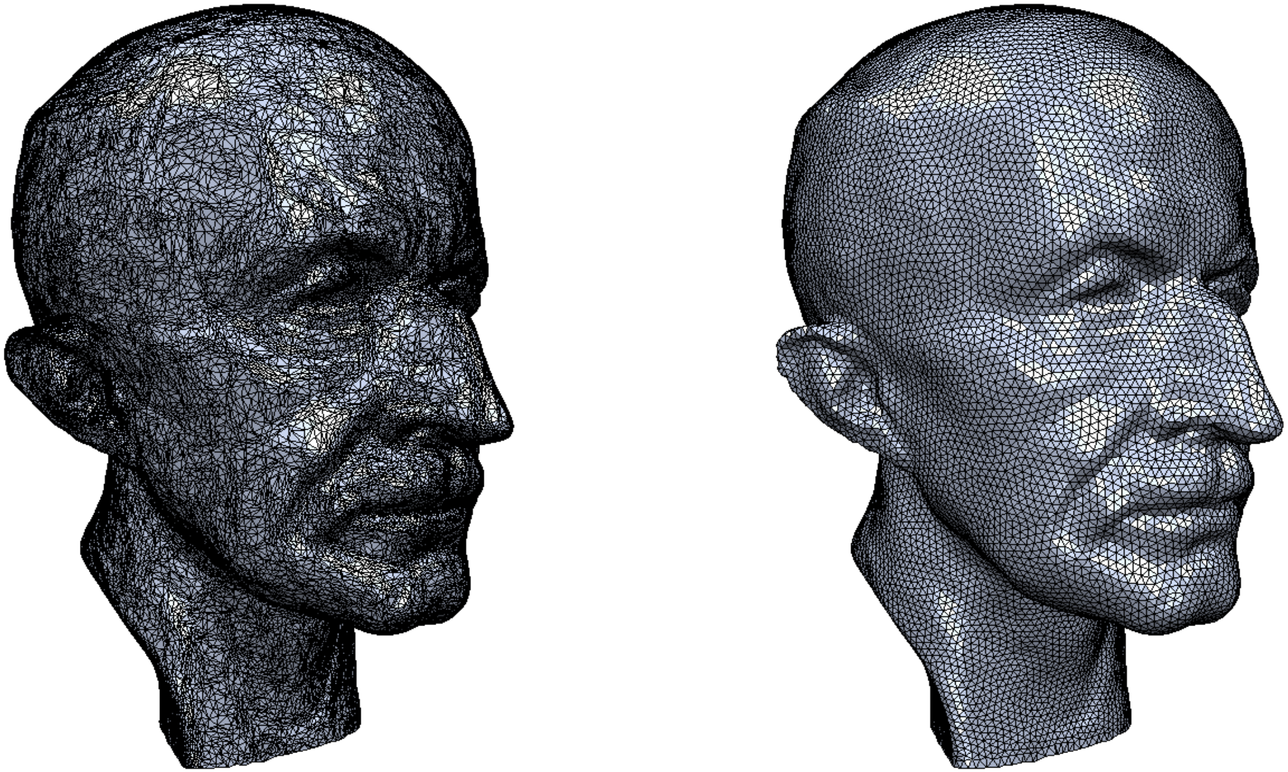
Remeshing and smoothing

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Remeshing

Remeshing

Given a 3D mesh, find a “better” discrete representation of the underlying surface

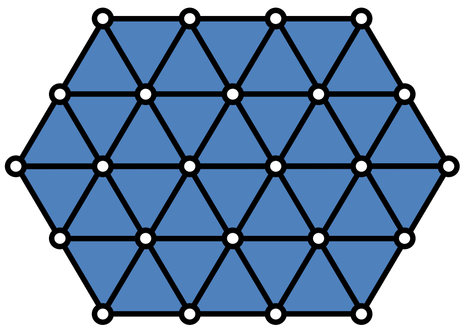


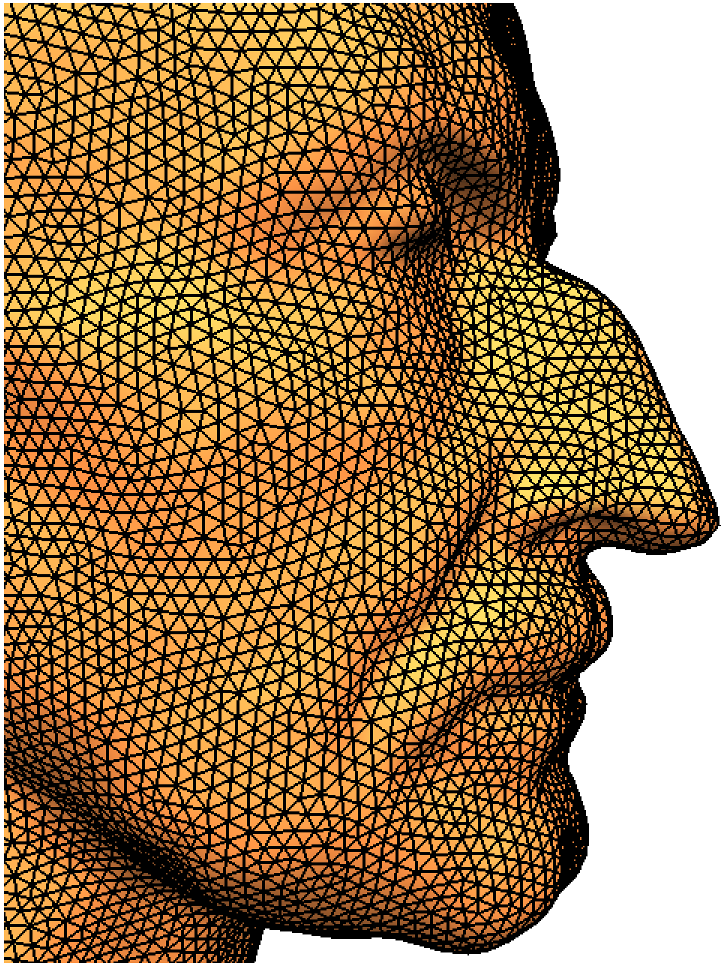
What is a good mesh?

Equal edge lengths

Equilateral triangles

Valence close to 6





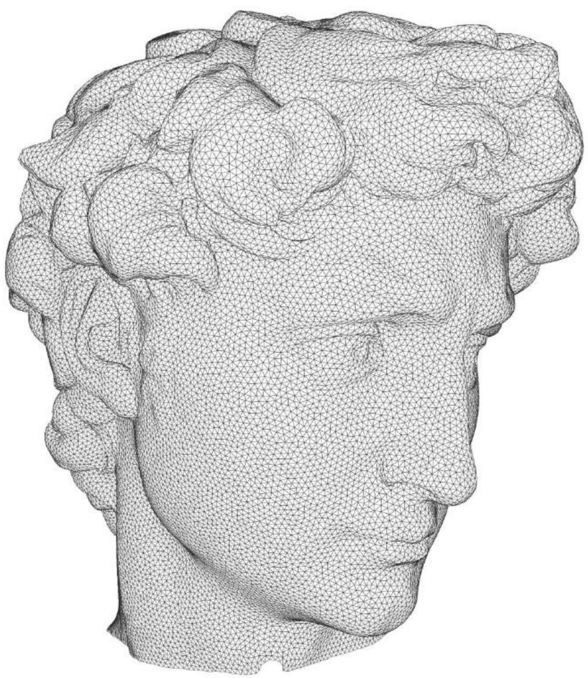
What is a good mesh?

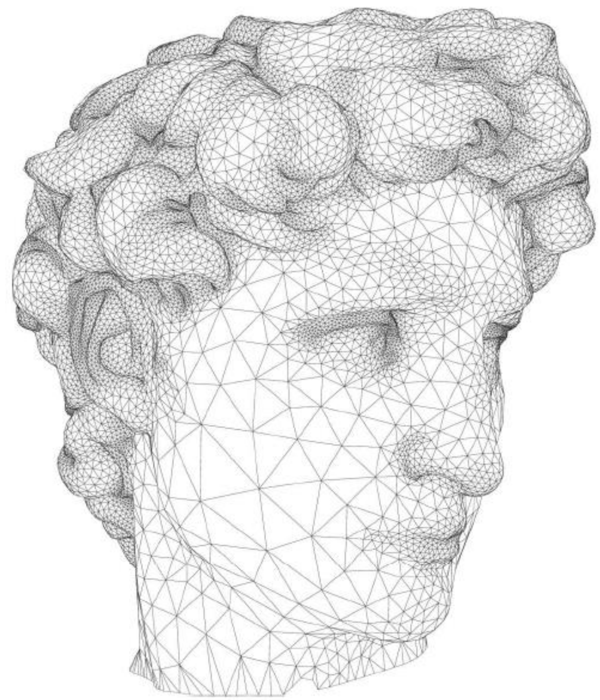
Equal edge lengths

Equilateral triangles

Valence close to 6

Uniform vs. adaptive sampling





What is a good mesh?

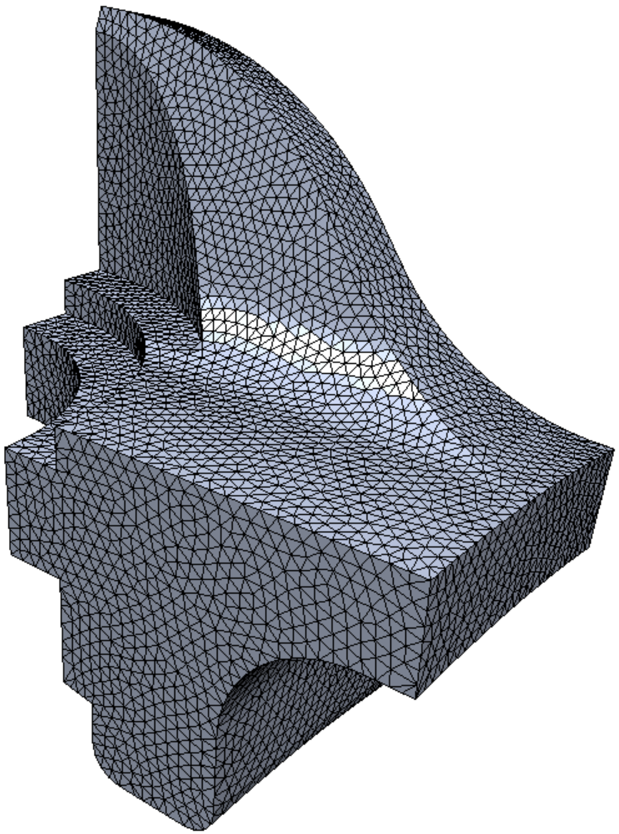
Equal edge lengths

Equilateral triangles

Valence close to 6

Uniform vs. adaptive sampling

Feature preservation



What is a good mesh?

Equal edge lengths

Equilateral triangles

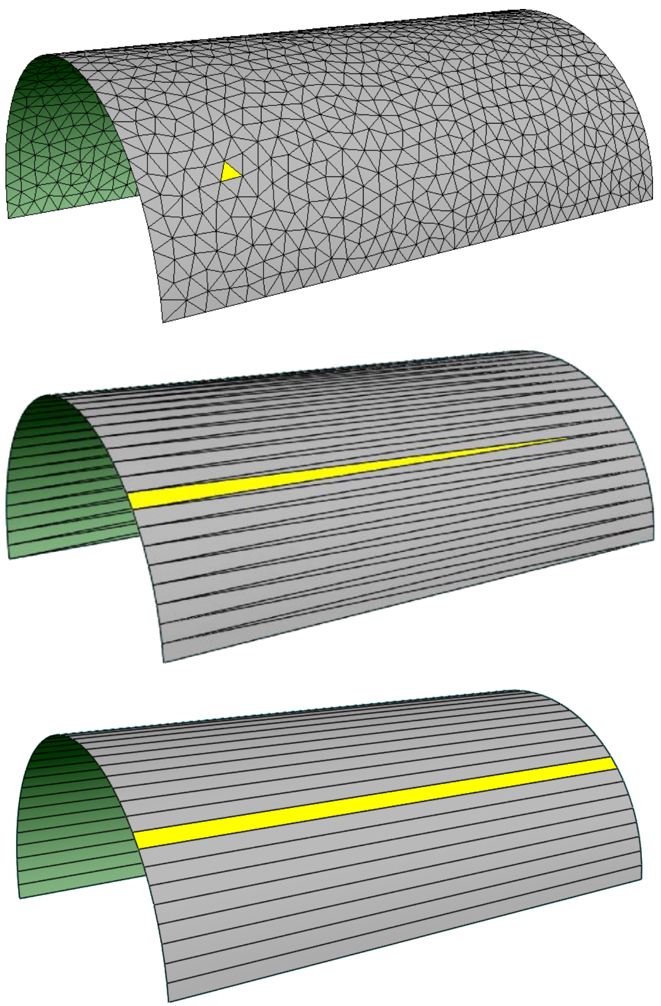
Valence close to 6

Uniform vs. adaptive sampling

Feature preservation

Alignment to curvature lines

Isotropic vs. anisotropic



What is a good mesh?

Equal edge lengths

Equilateral triangles

Valence close to 6

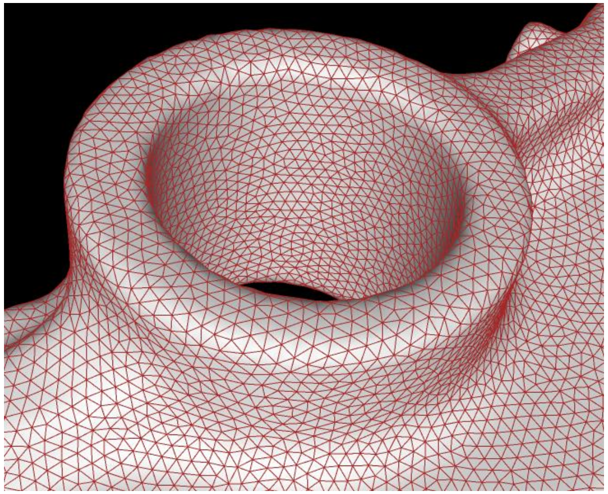
Uniform vs. adaptive sampling

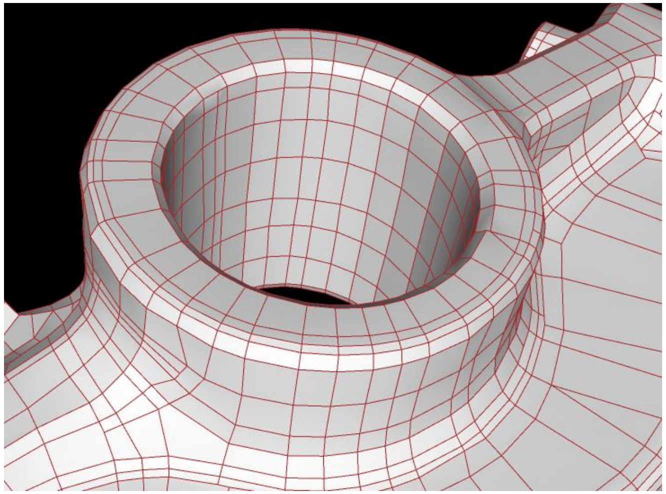
Feature preservation

Alignment to curvature lines

Isotropic vs. anisotropic

Triangles vs. quads

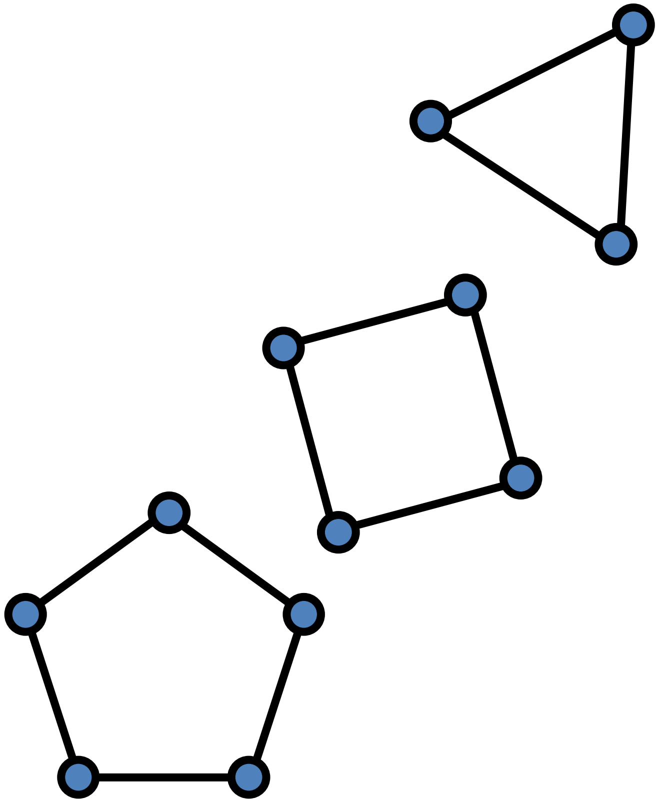




Local Structure

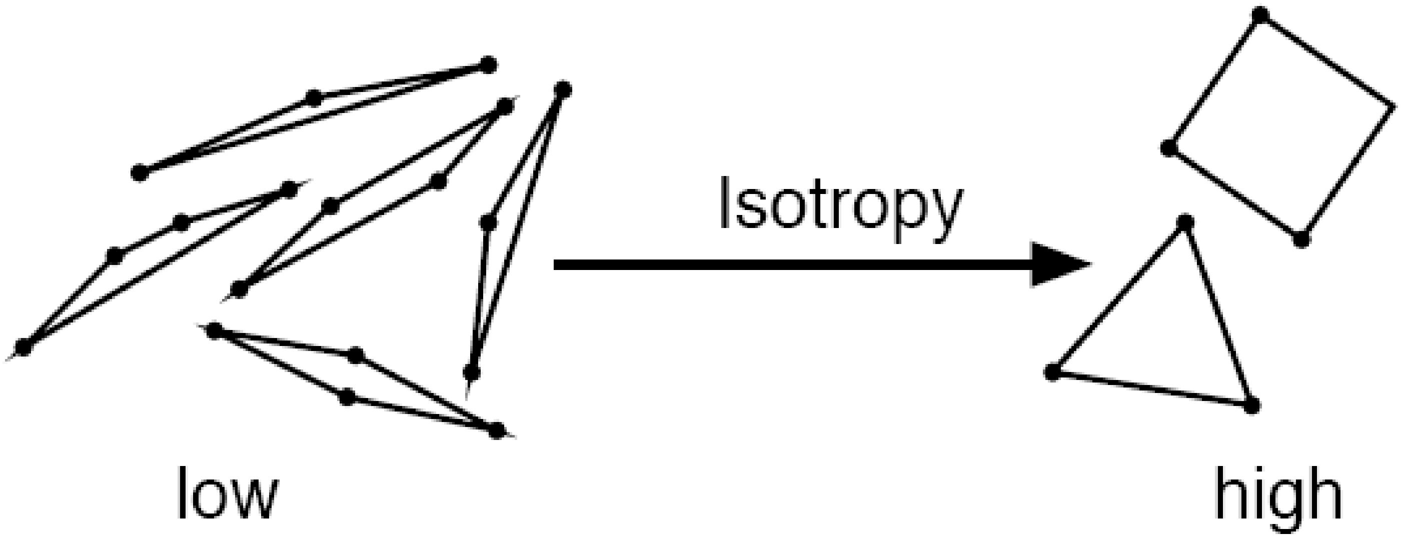
Element **type**

- Triangle
- Quadrangle
- Polygon



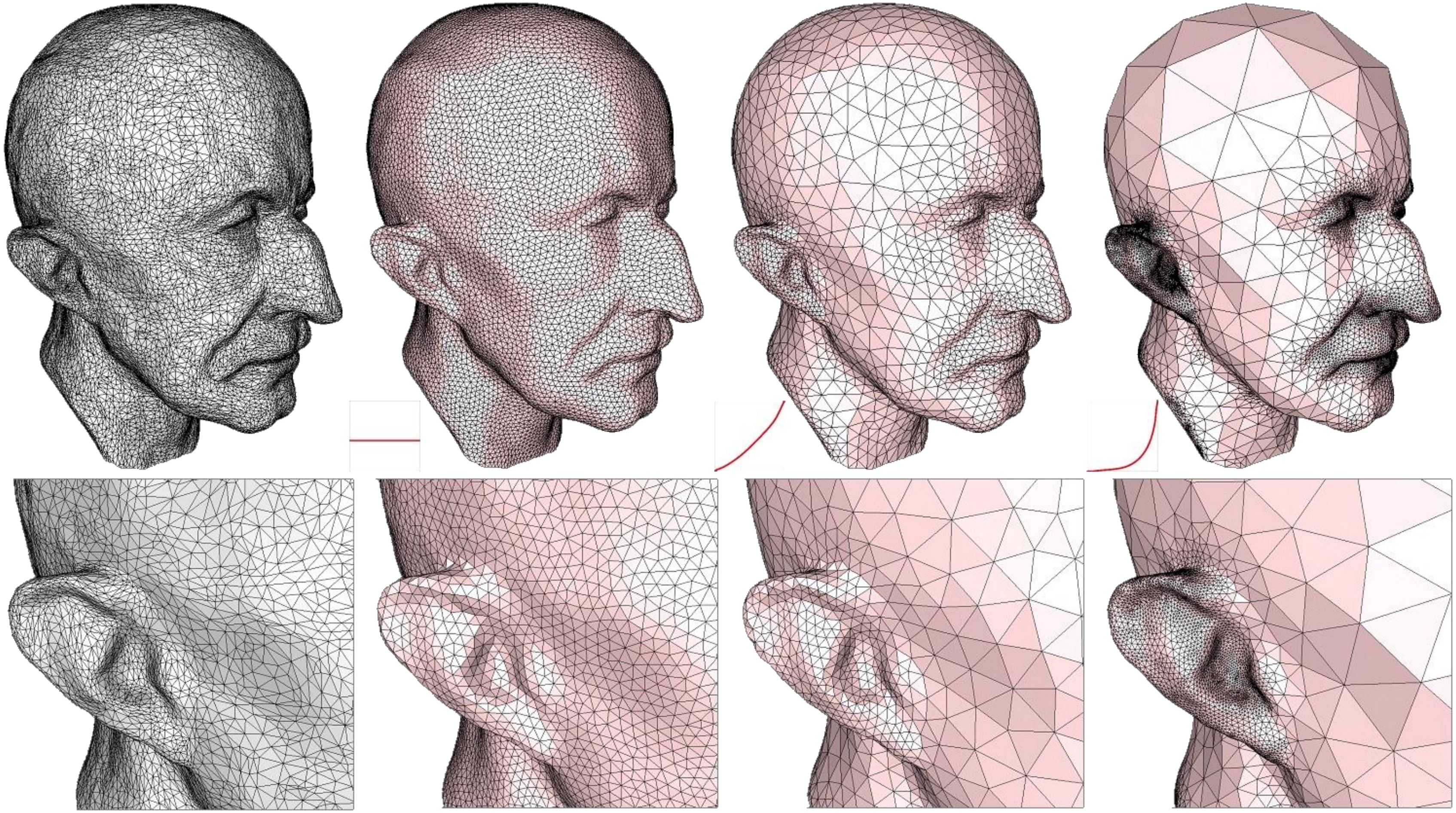
Local Structure

Element **shape** (isotropy vs anisotropy)



Local Structure

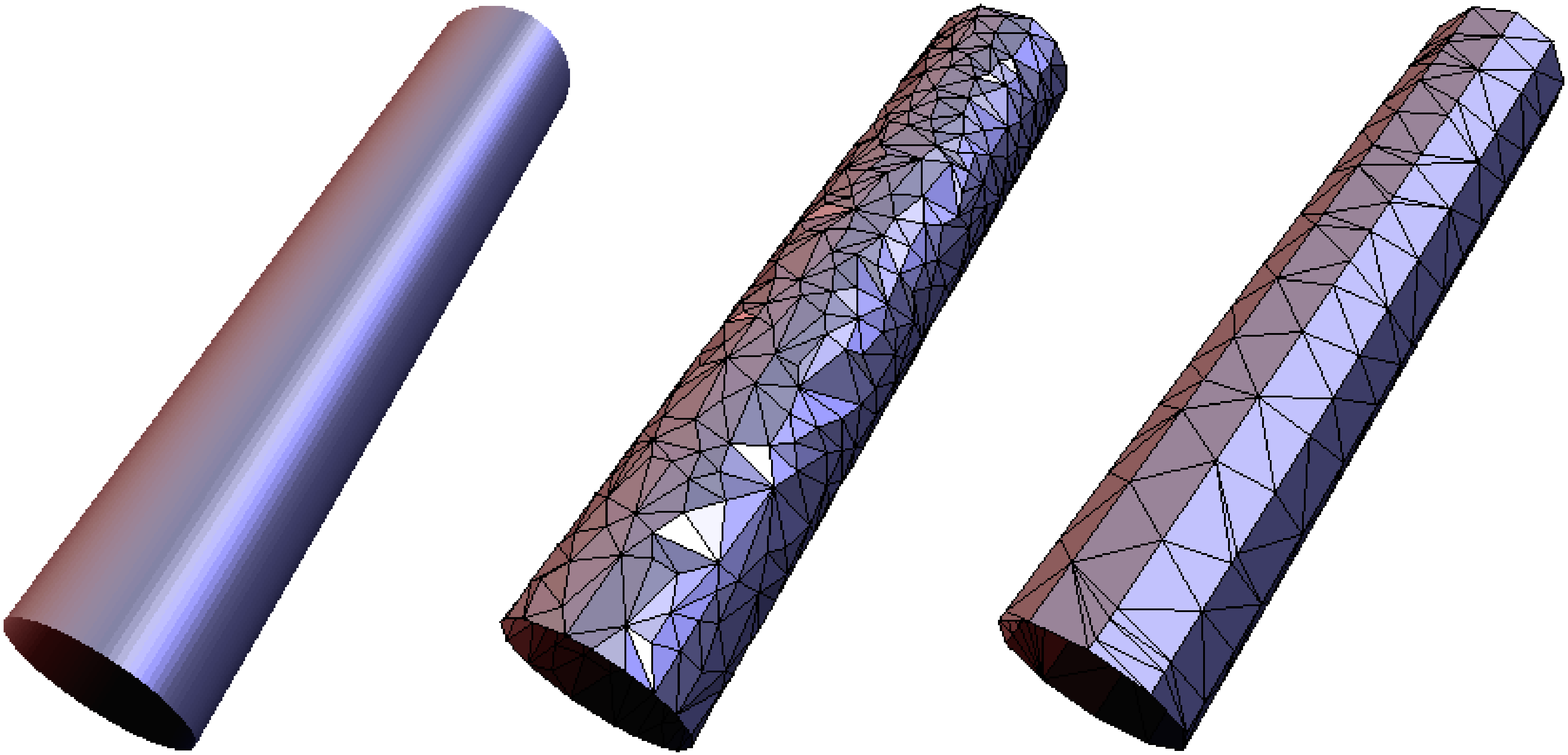
Element distribution (sizing, grading)



Input Uniform Adaptive

Local Structure

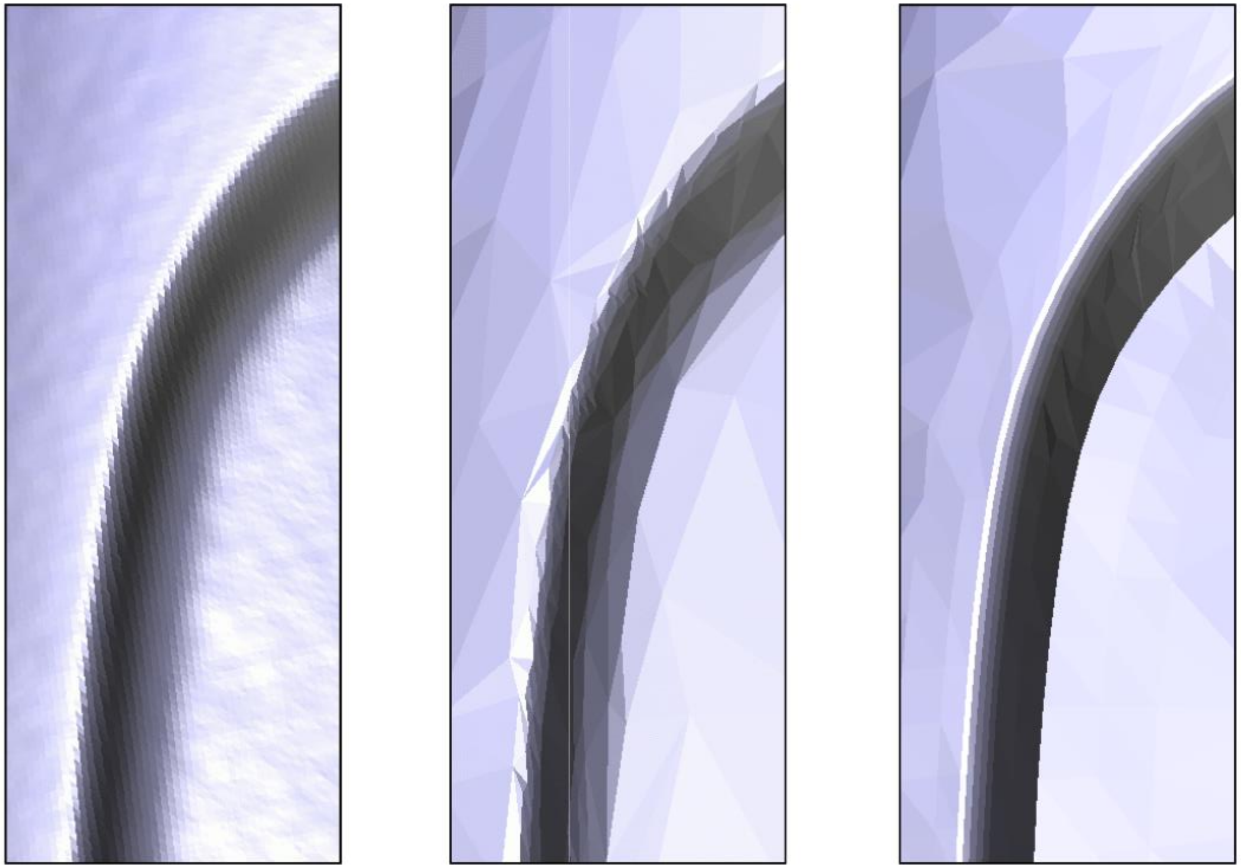
Element orientation



ETH zürich

Local Structure

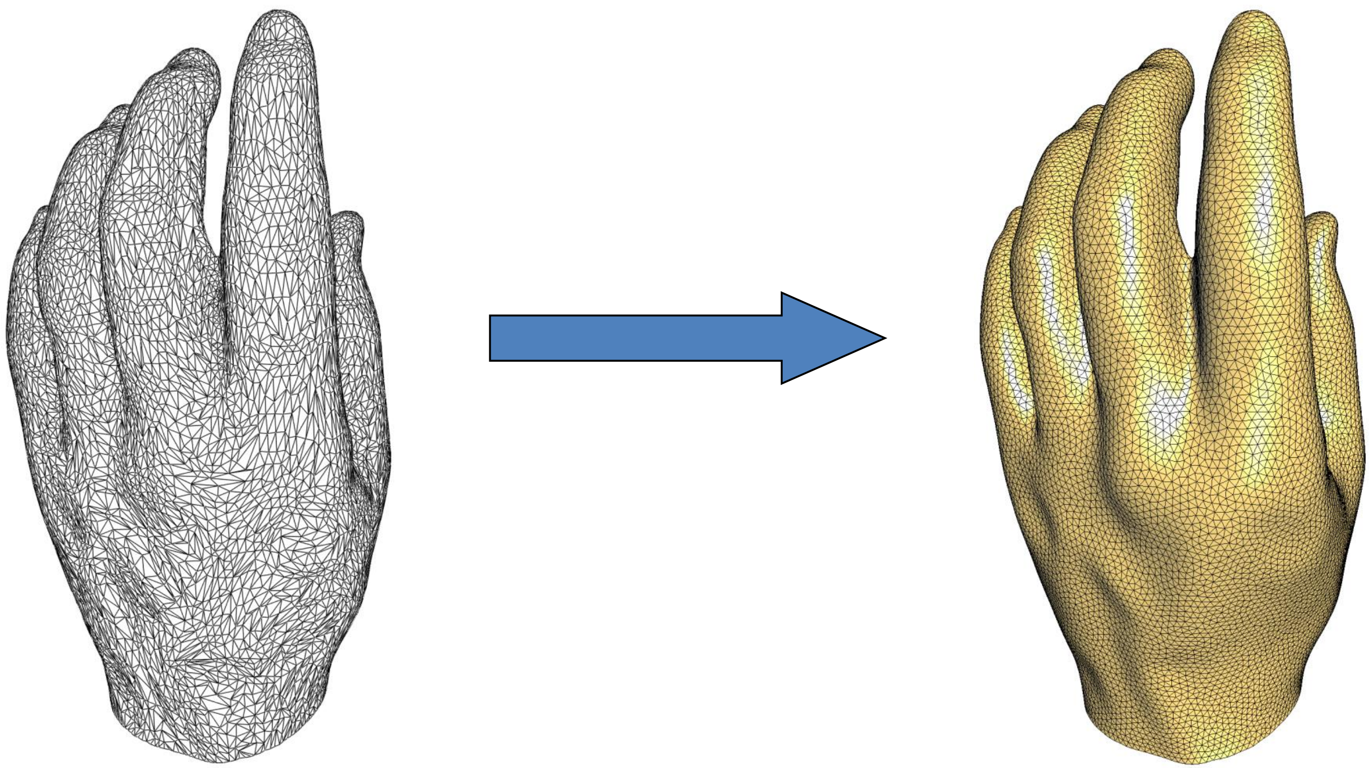
Element orientation



Isotropic Remeshing

Well-shaped elements

- for processing & simulation (numerical stability & efficiency)

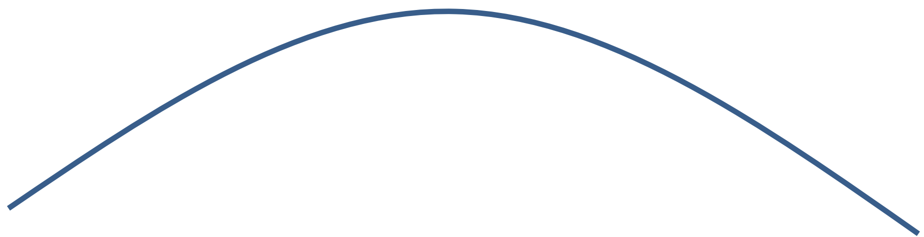


Two Fundamental Approaches

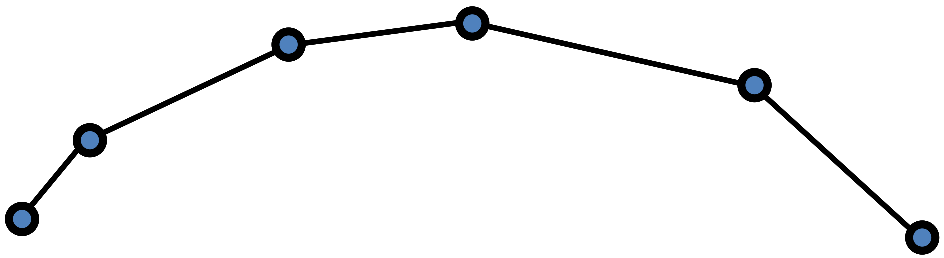
Parameterization-based
- map to 2D domain / 2D problem
- computationally more expensive
- works even for coarse resolution remeshing

Surface-oriented
- operate directly on the surface
- treat surface as a set of points / polygons in space
- efficient for high resolution remeshing

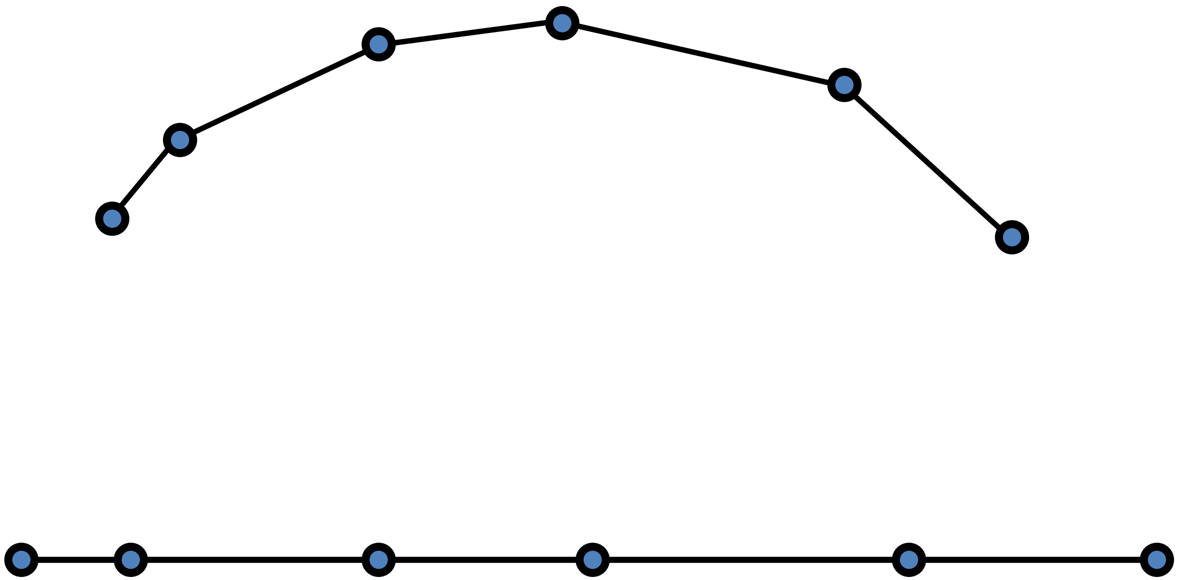
Parameterization Based



Parameterization Based

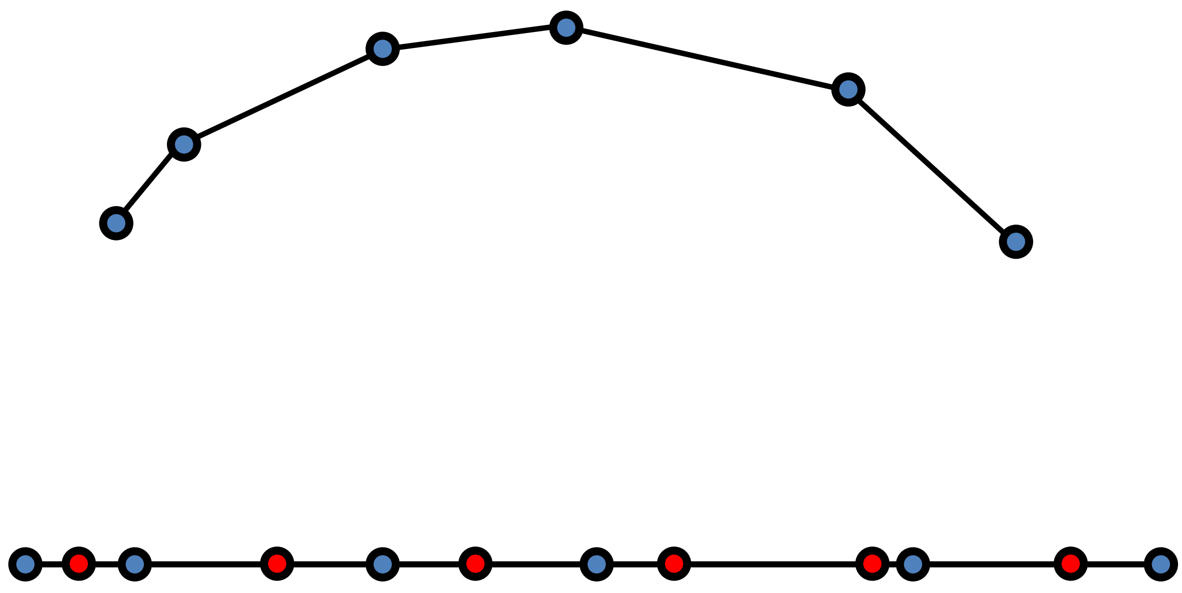


Parameterization Based



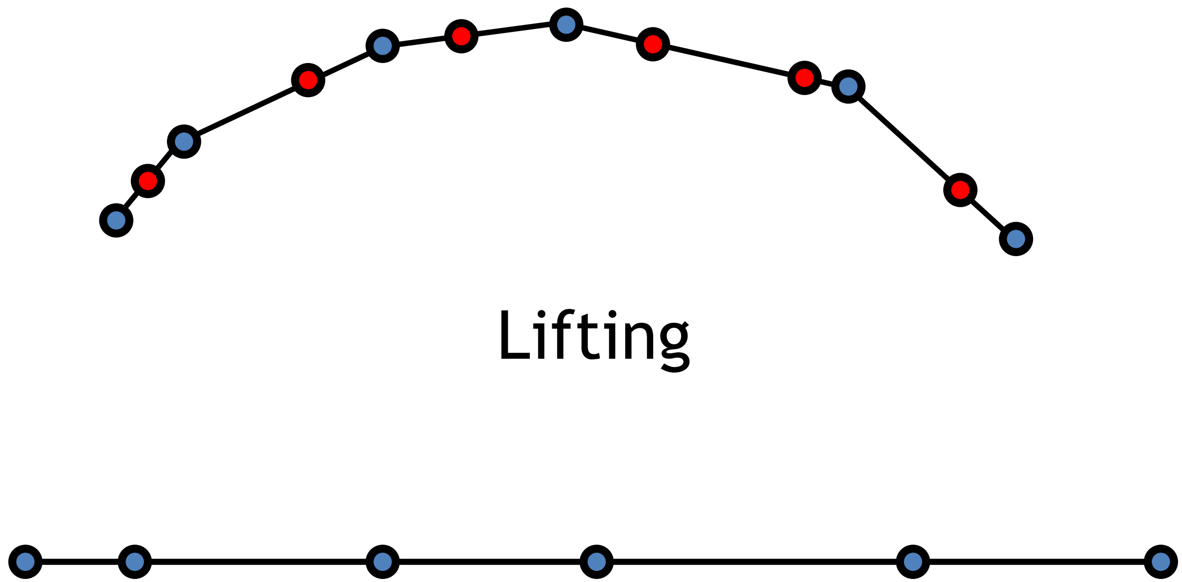
parameterization

Parameterization Based



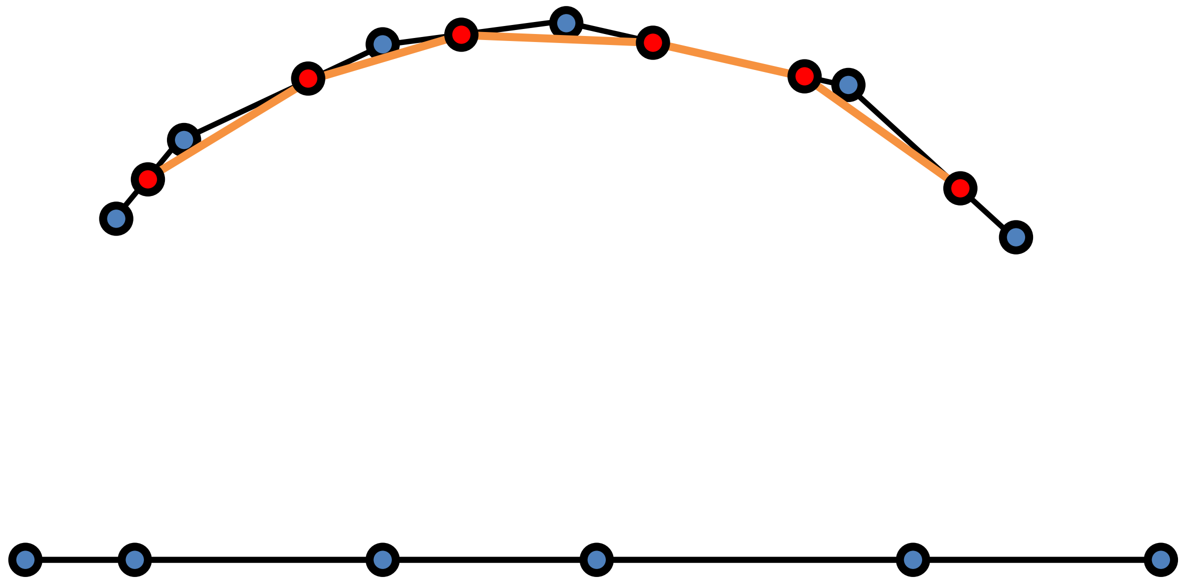
resampling

Parameterization Based



Parameterization Based

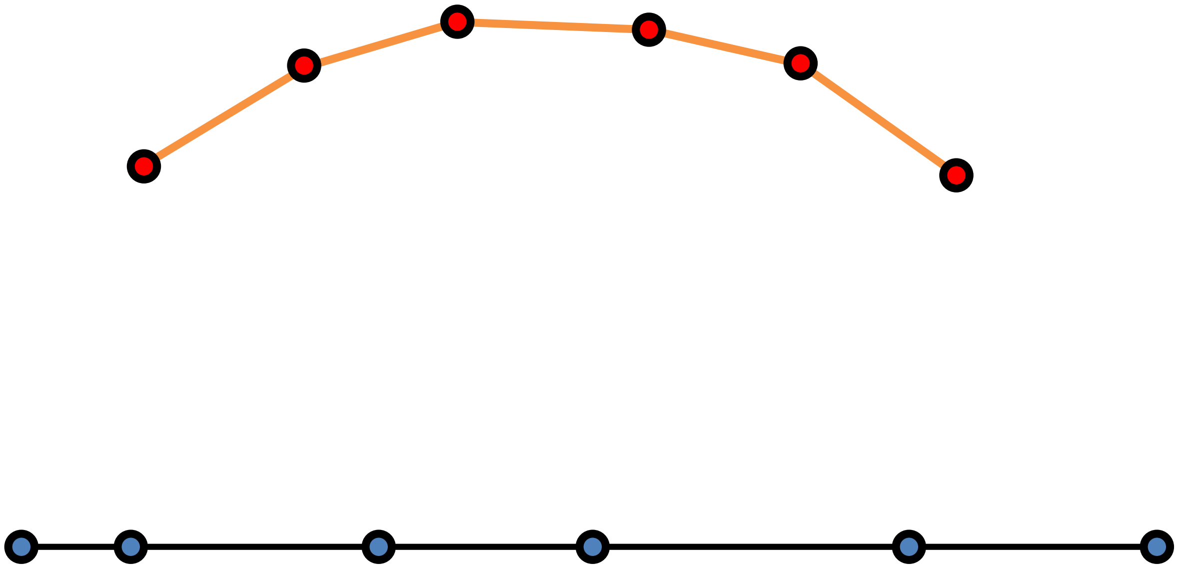
Remeshing



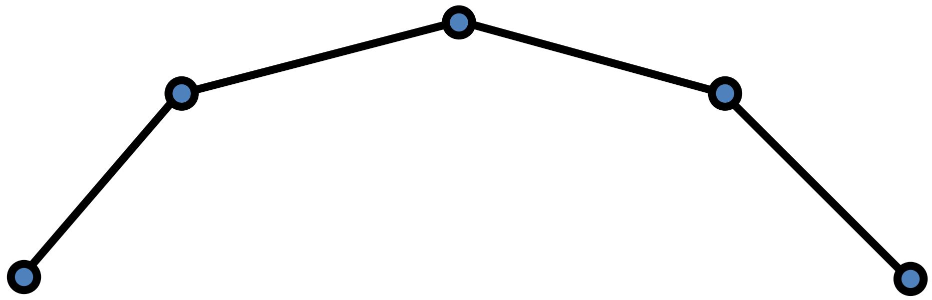
ETH zürich

Parameterization Based

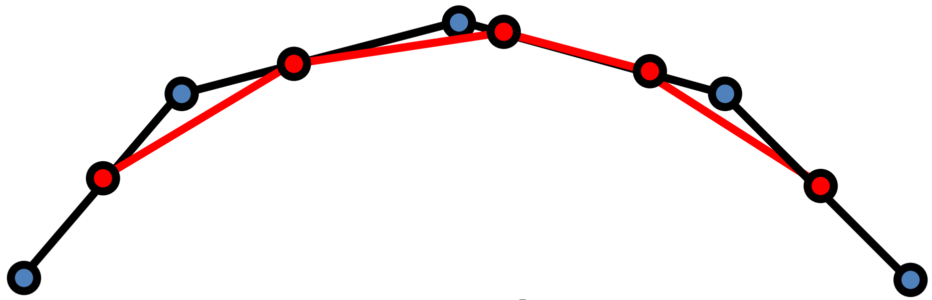
Remeshing



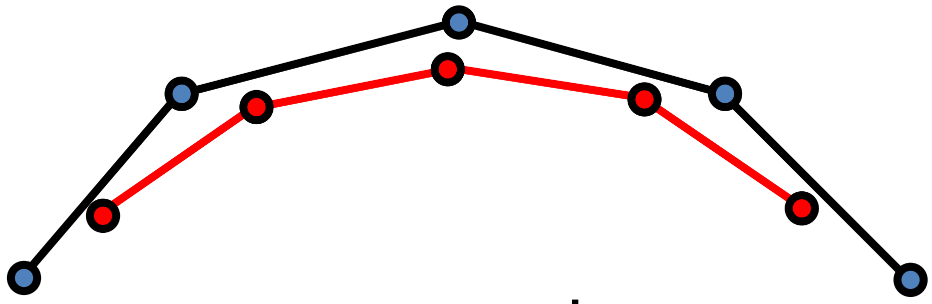
Surface Oriented



Surface Oriented

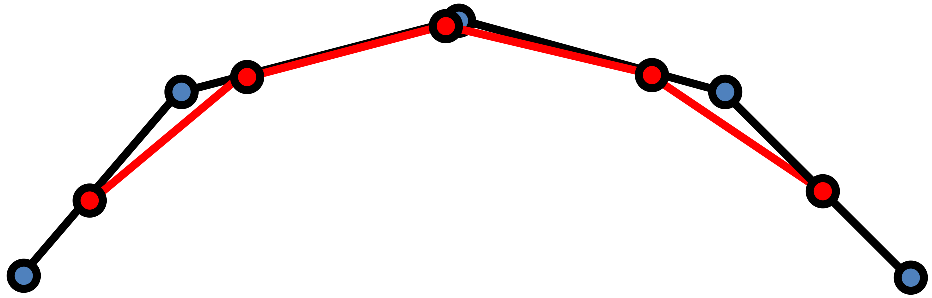


Surface Oriented



smooth

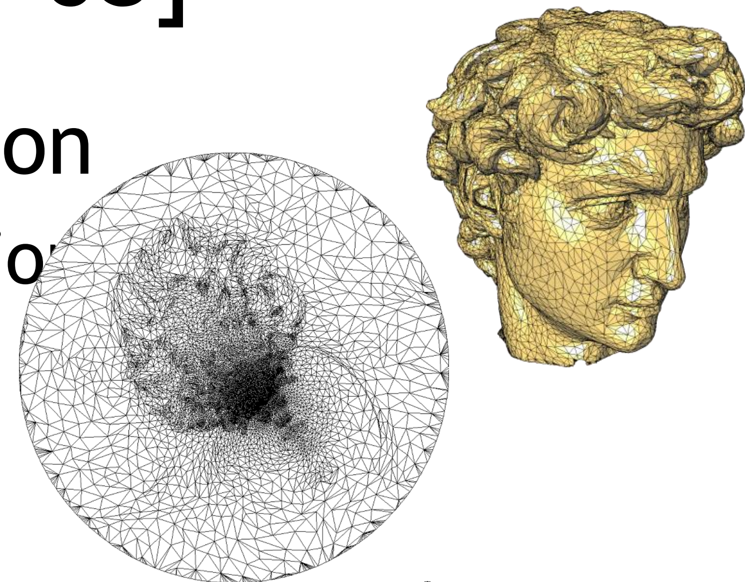
Surface Oriented



Parameterization-Based Remeshing [Alliez et al. '03]

Compute 2D parameterization

Conformal: only area distortio

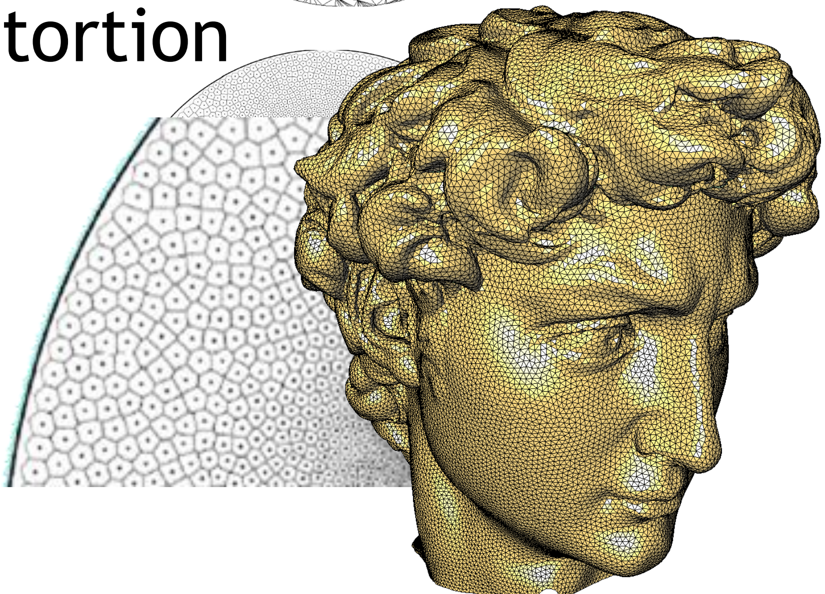


Sample 2D domain

Density based on area distortion

Triangulate

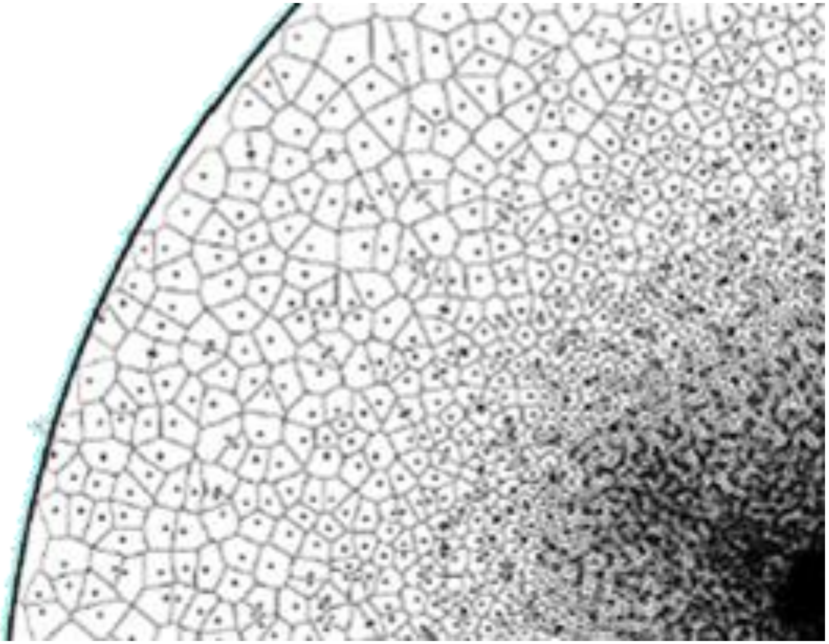
Lift back to 3D

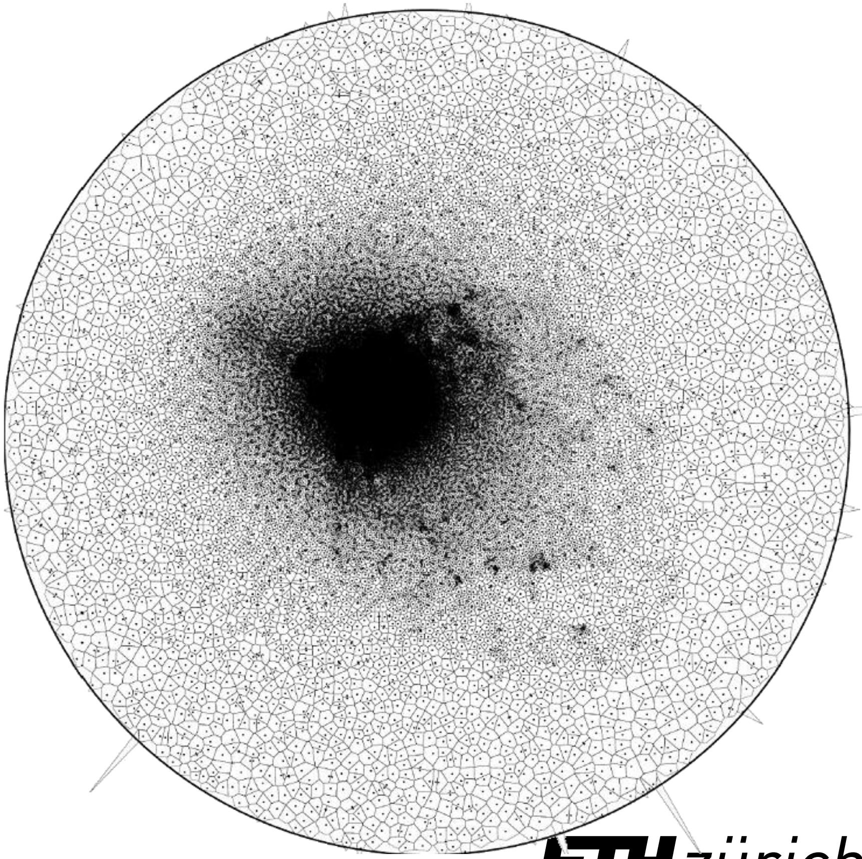


Isotropic 2D Sampling

Density based random sampling

- Does not guarantee uniform distance between samples



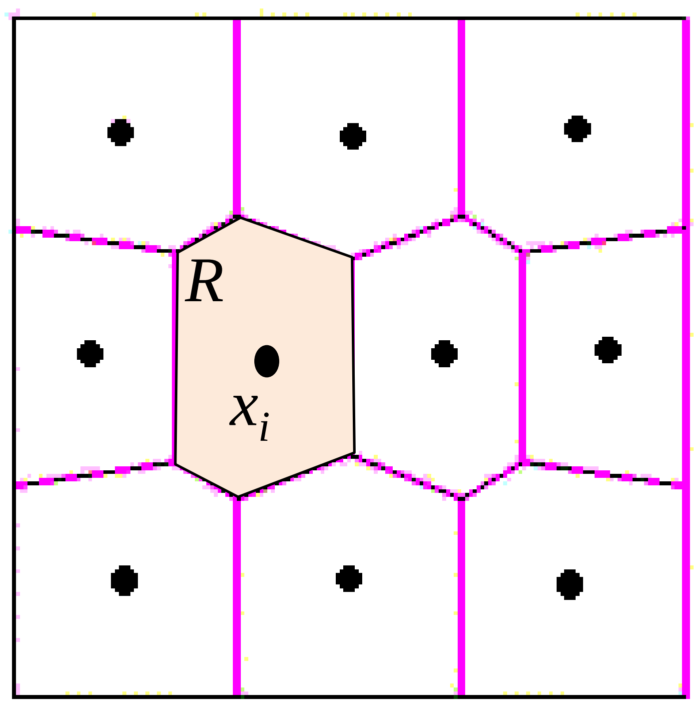


Sampling Energy

Given sites x_i and regions R_i, minimize

$$E(x_1,\ldots,x_k,R_1,\ldots,R_k)=\sum_{i=1}^{k}\int_{x\in R_i}\|x-x_i\|^2$$

Spreads out points



Sampling Energy

Given samples x_i and regions $R_{i,}$ minimize

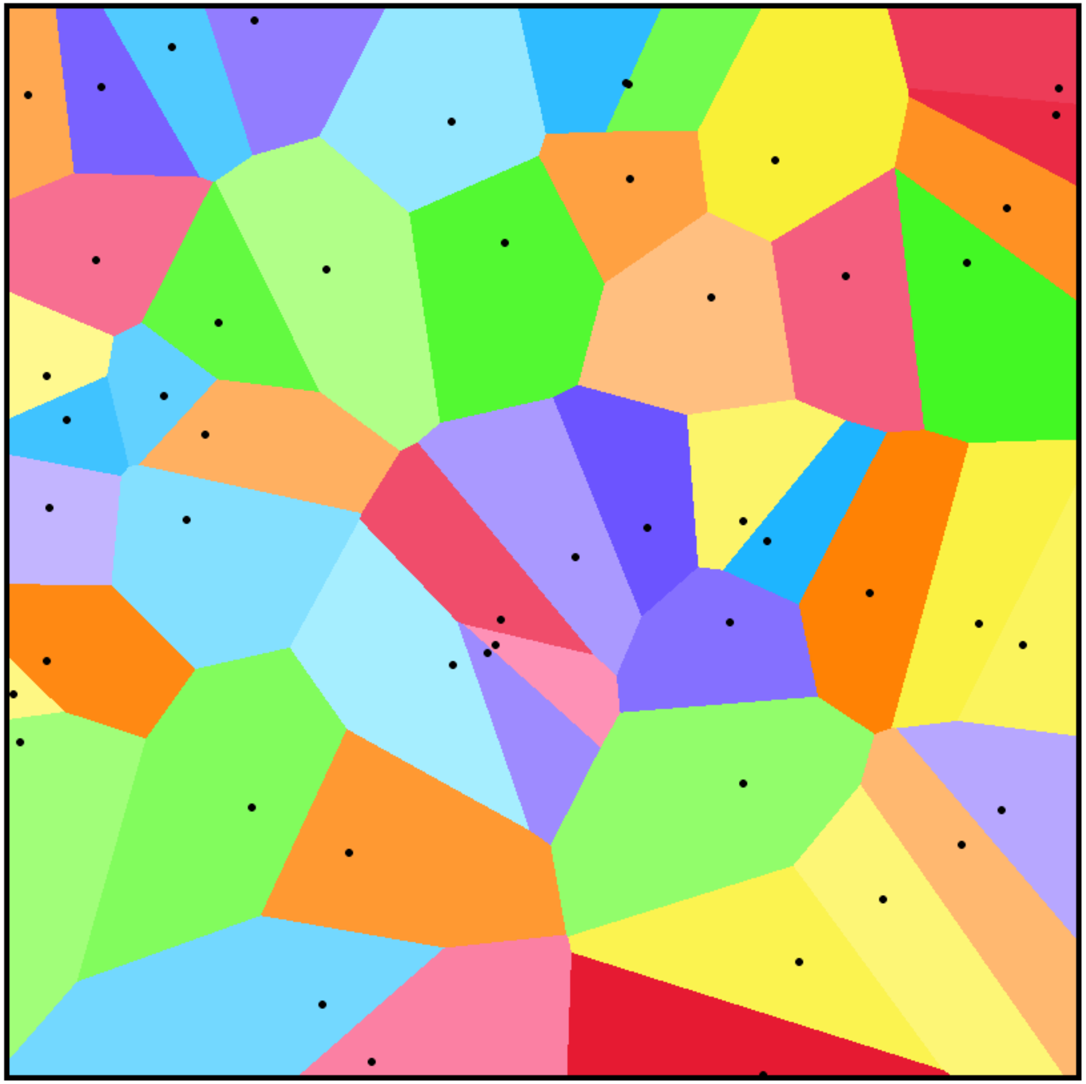
$$E(x_1, \dots, x_k, R_1, \dots, R_k) = \sum_{i=1}^{k} \int \|x - x_i\|^2$$

If x_i are fixed, energy is minimized by the ***Voronoi Tesselation***

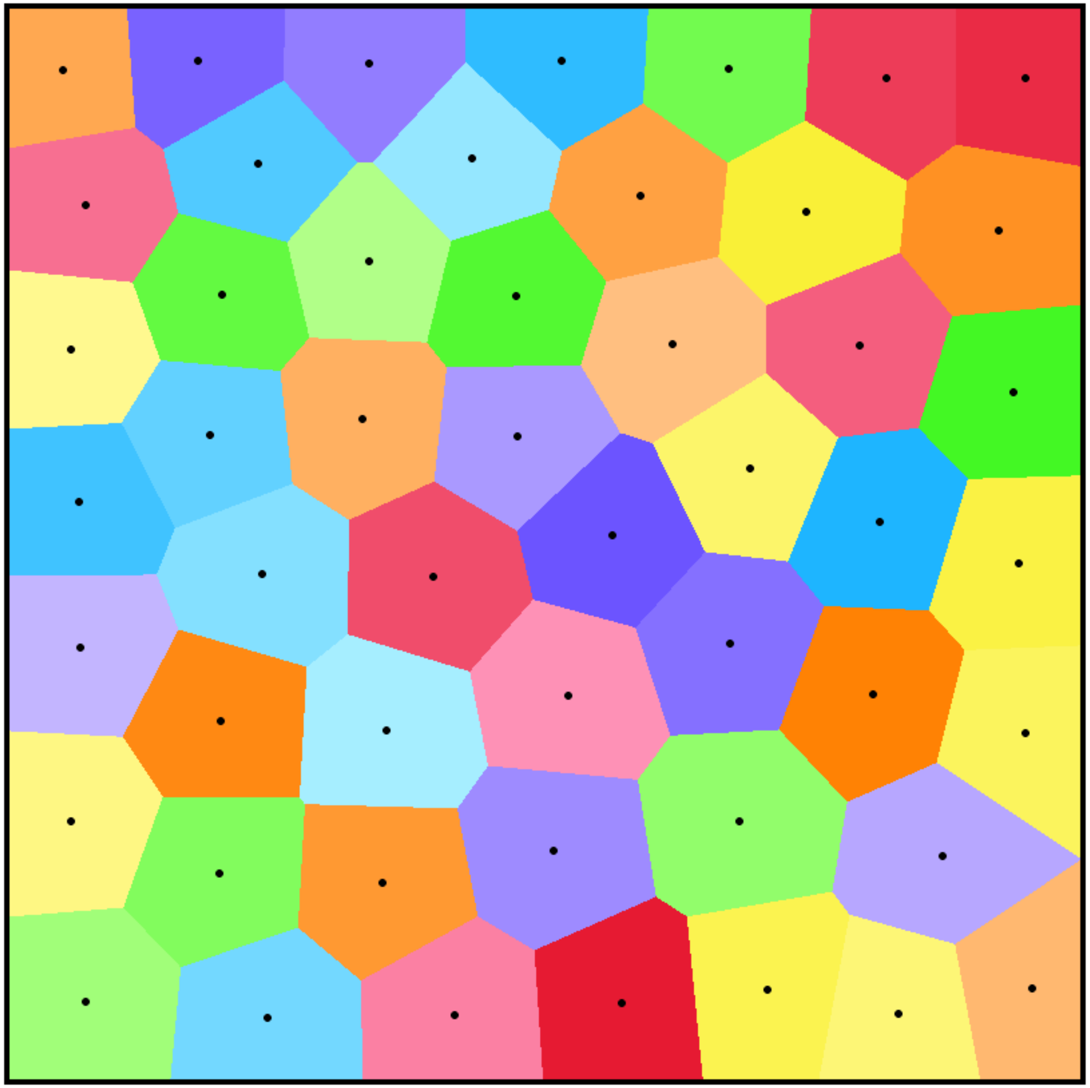
Voronoi cell R_i

= All points closer to x_i than to any other x_j

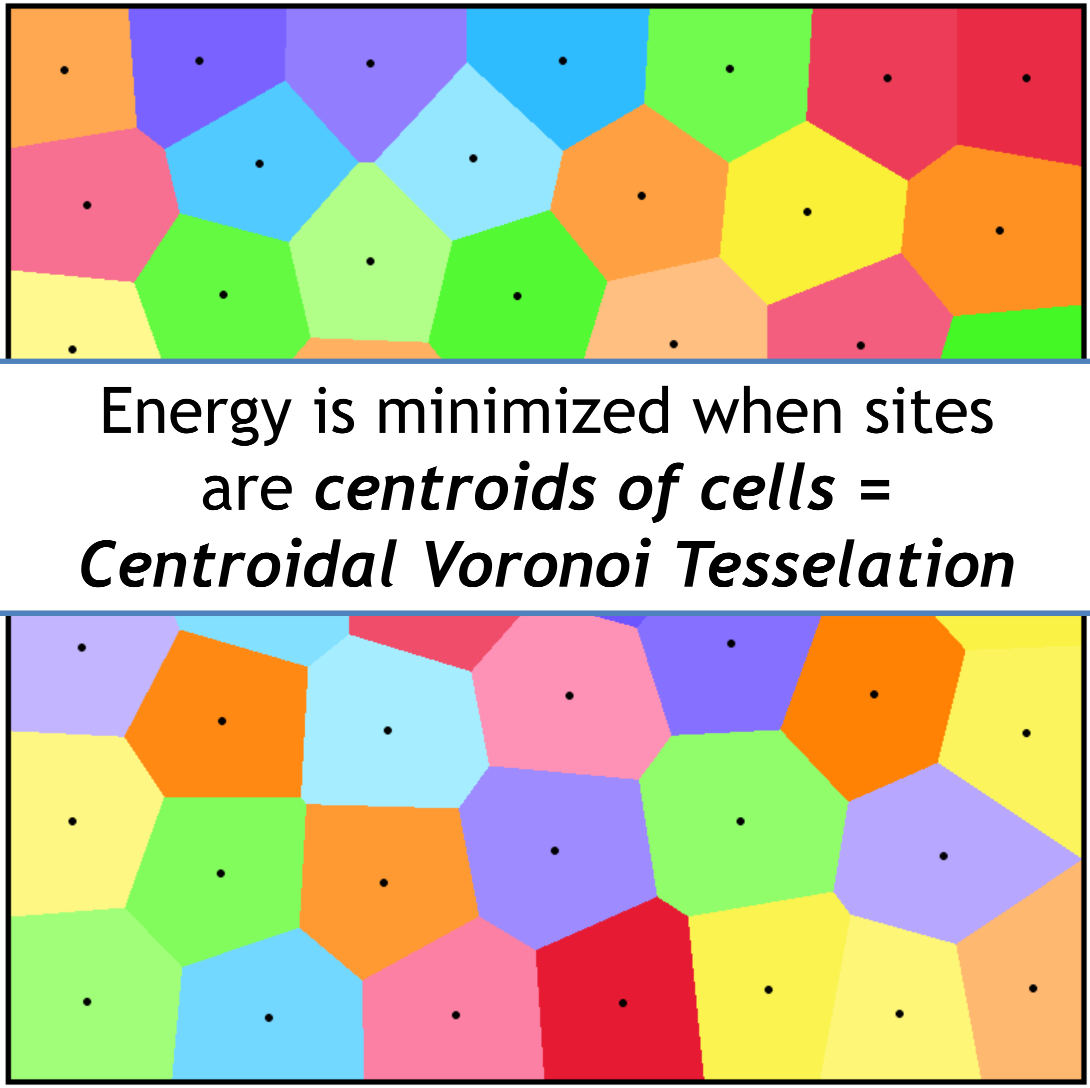
Voronoi Tessellation



Centroidal Voronoi Tessellation



Centroidal Voronoi Tessellation

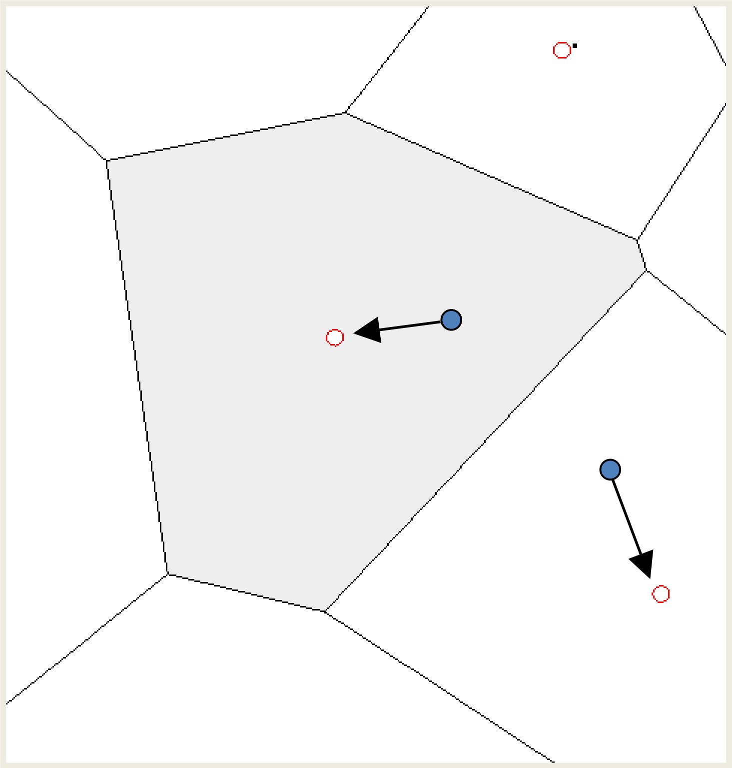


Energy is minimized when sites are ***centroids of cells* = *Centroidal Voronoi Tesselation***

Lloyd Algorithm

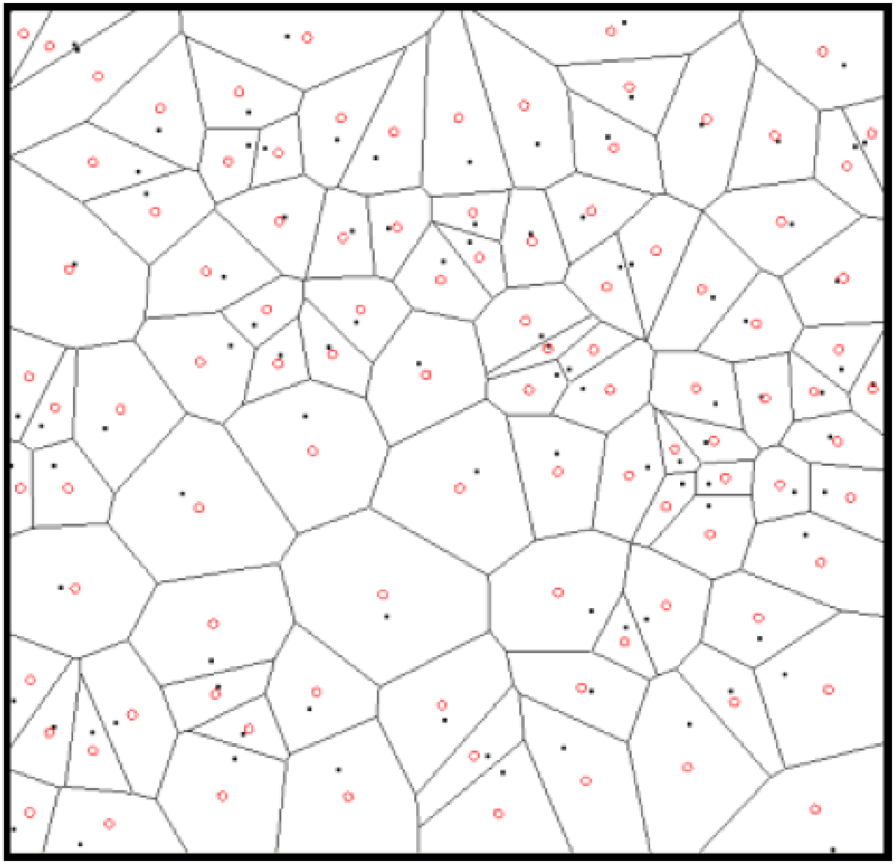
Alternate:

- Voronoi partitioning
- Move sites to respective centroids



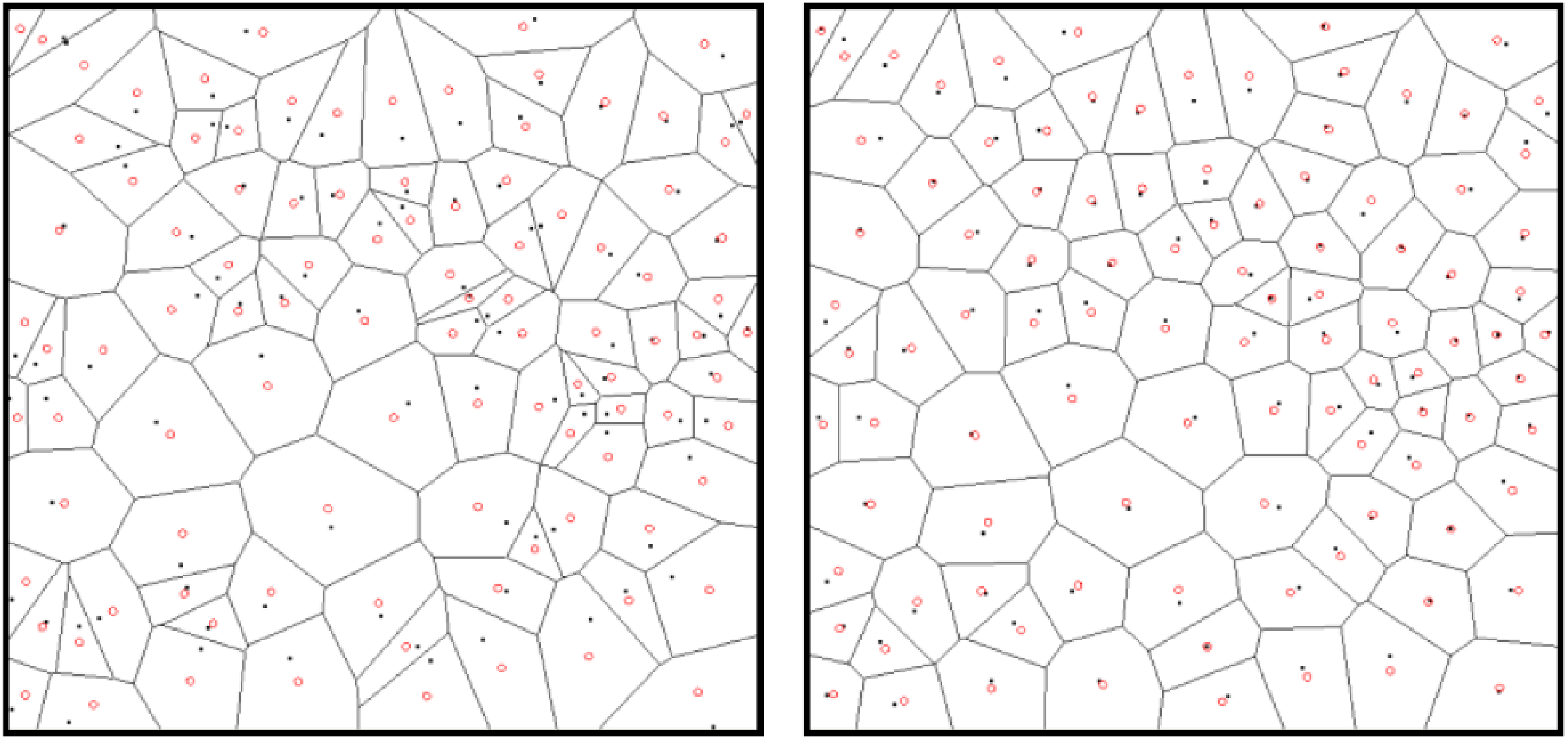
ETH zürich

Centroidal Voronoi Diagrams



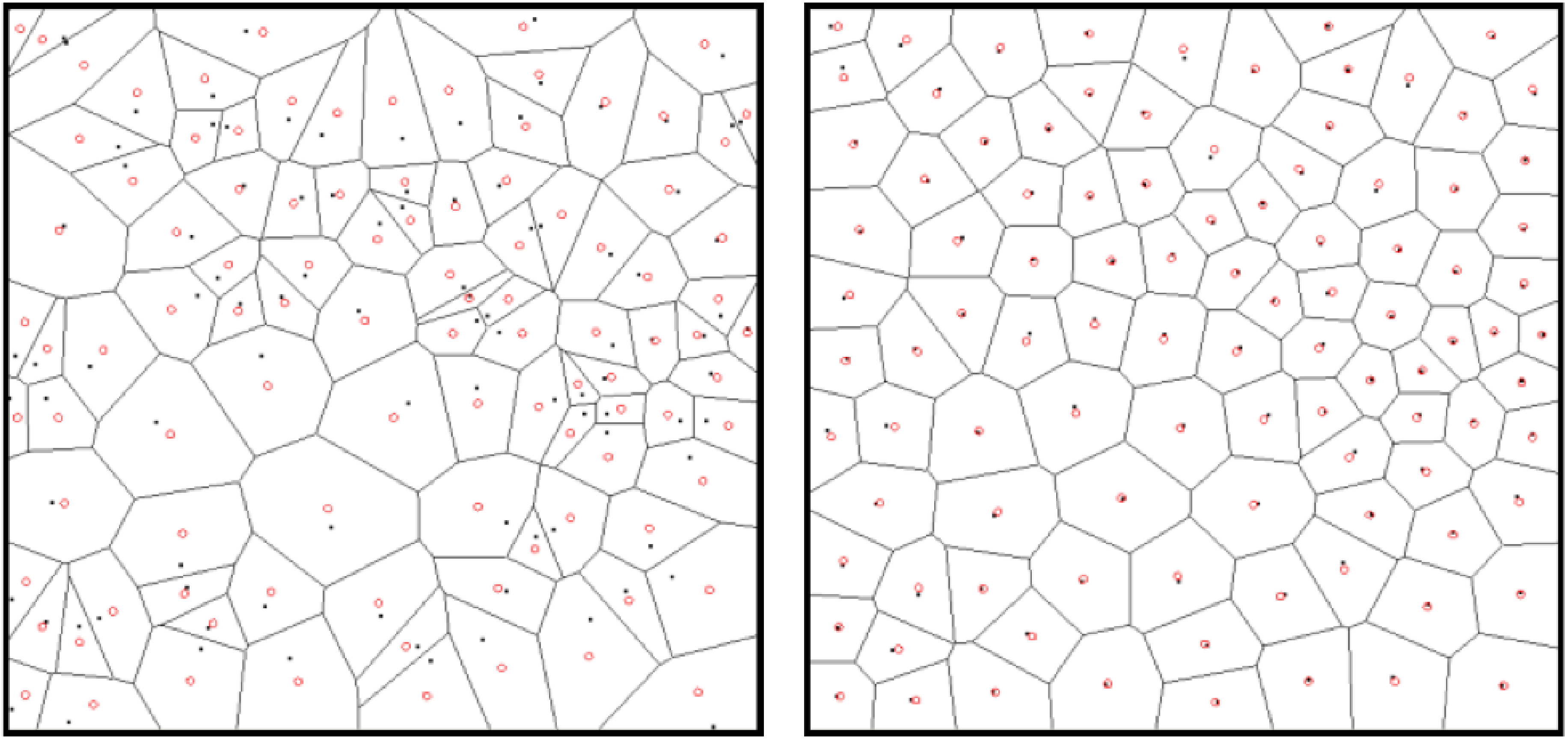
demo

Centroidal Voronoi Diagrams



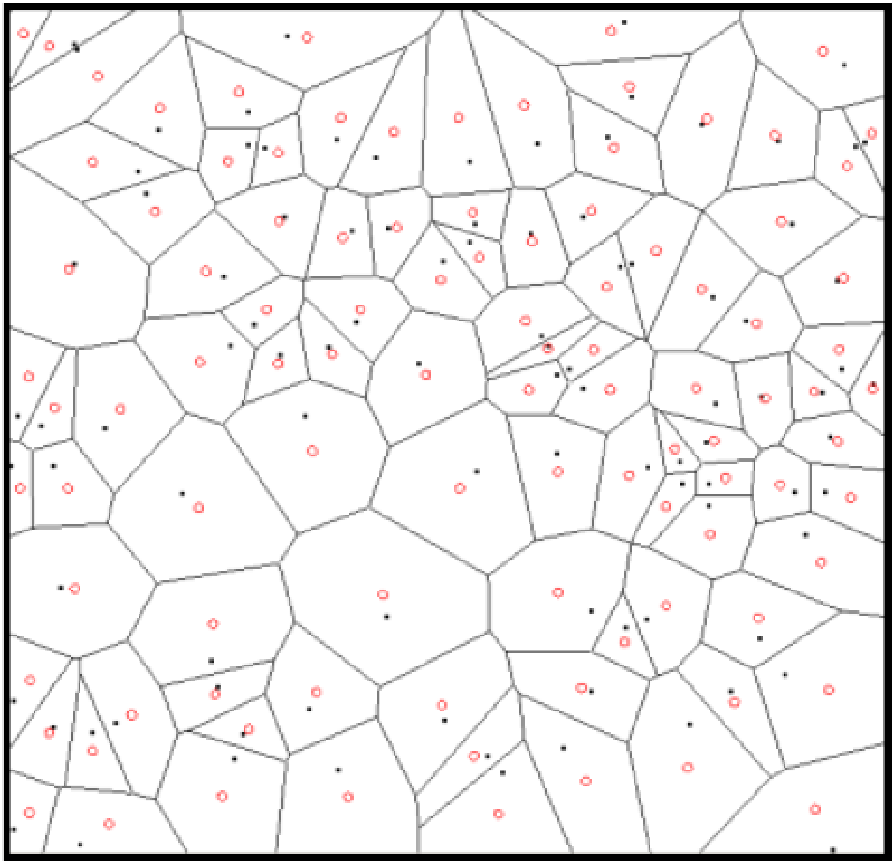
[demo]

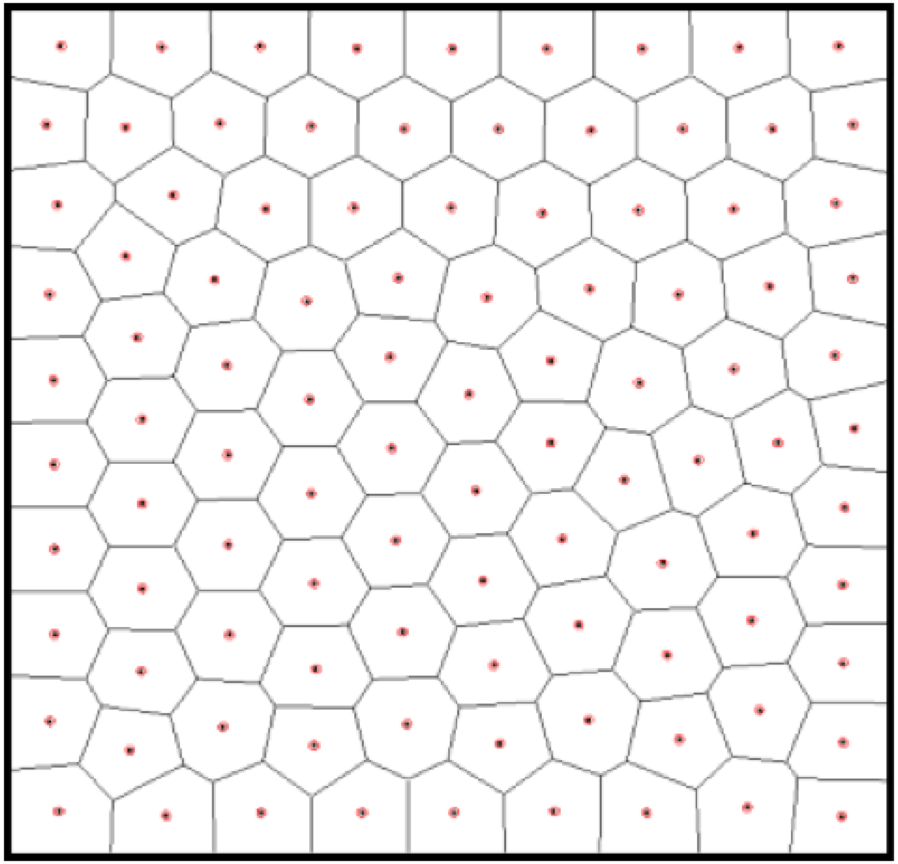
Centroidal Voronoi Diagrams



[demo]

Centroidal Voronoi Diagrams





demo

Centroidal Voronoi Tesselation

Lloyd converges slowly

- Stop when points “stop” moving

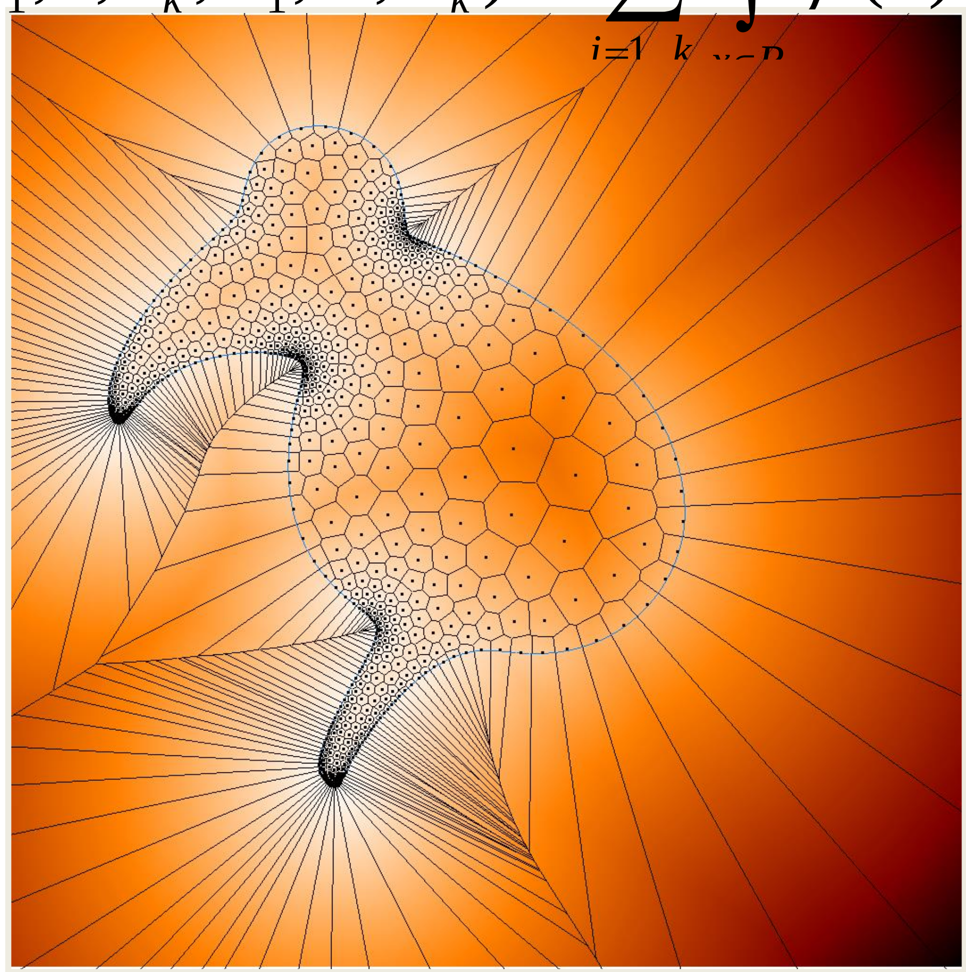
Faster algorithm: direct optimization of the energy using quasi-Newton

“On centroidal voronoi tessellation—energy smoothness and fast computation” [Liu et al., TOG ’09]

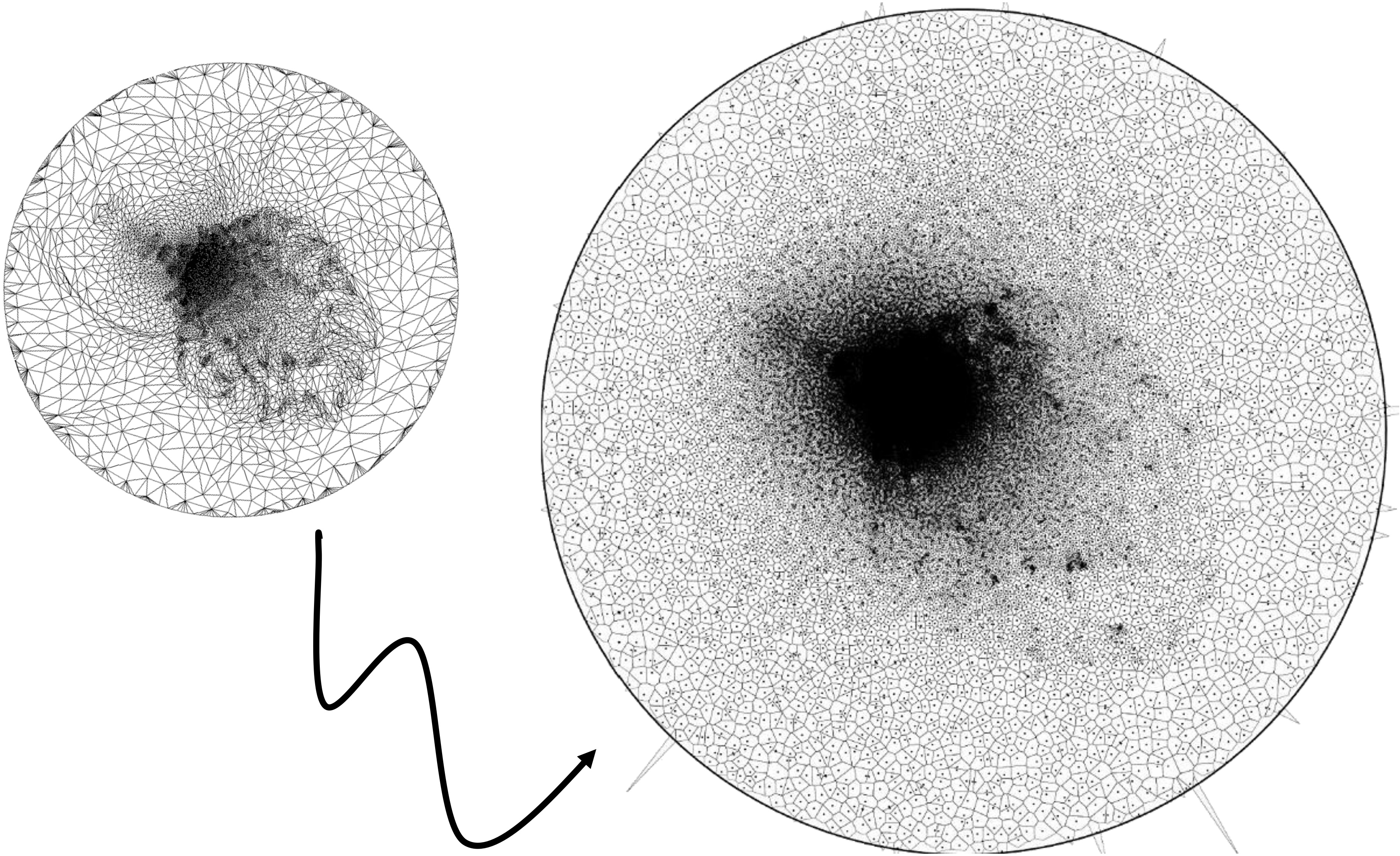
Varying Density

$$E(x_1, \ldots, x_k, R_1, \ldots, R_k) = \sum_{i=1}^{k} \rho(x) \int \|x - x_i\|^2$$

$$E(x_1, .., x_k, R_1, ..., R_k) = \sum_{i=1..k} \int_{x \in R} \rho(x) \| x$$

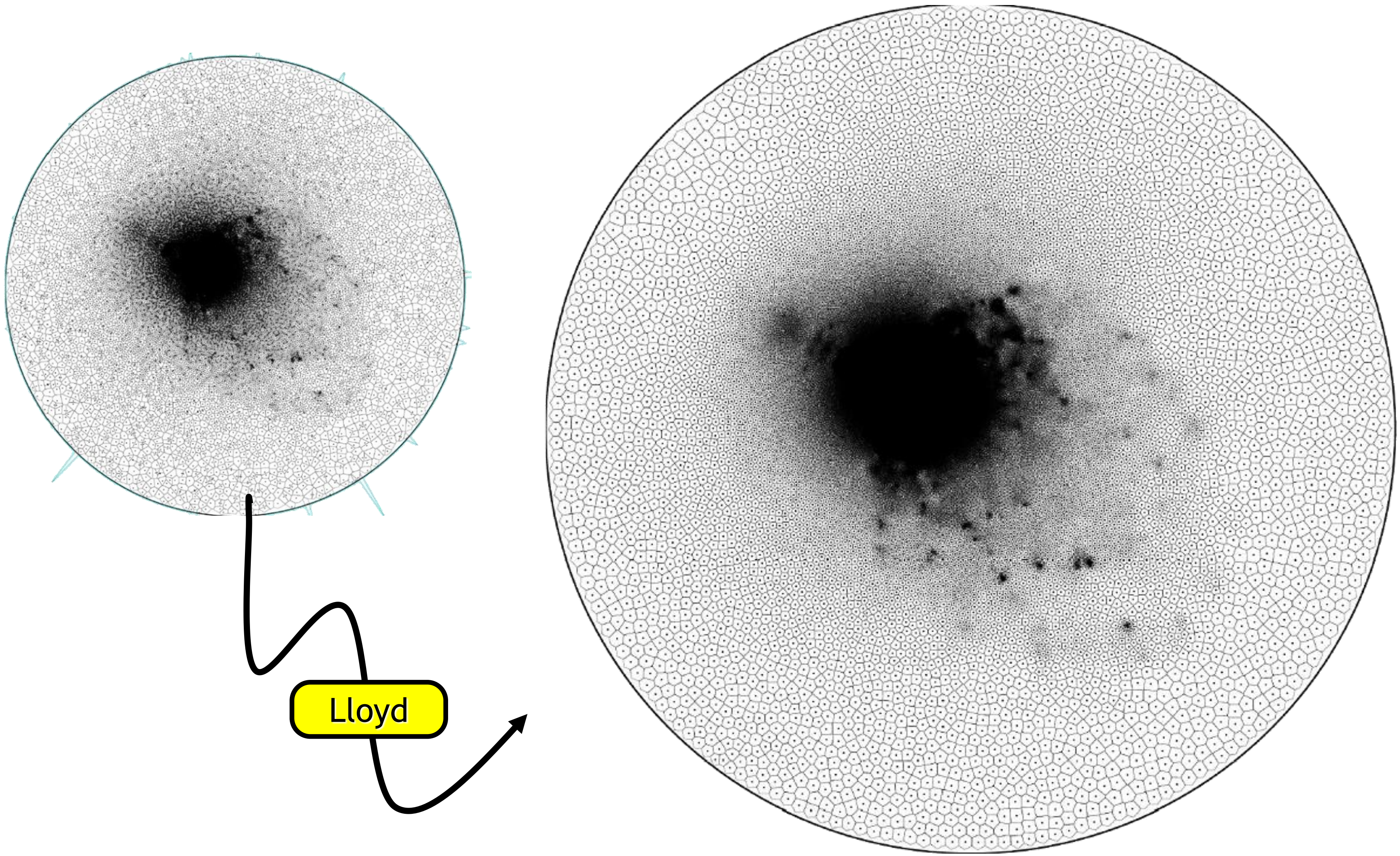


Initial Sample Scatter



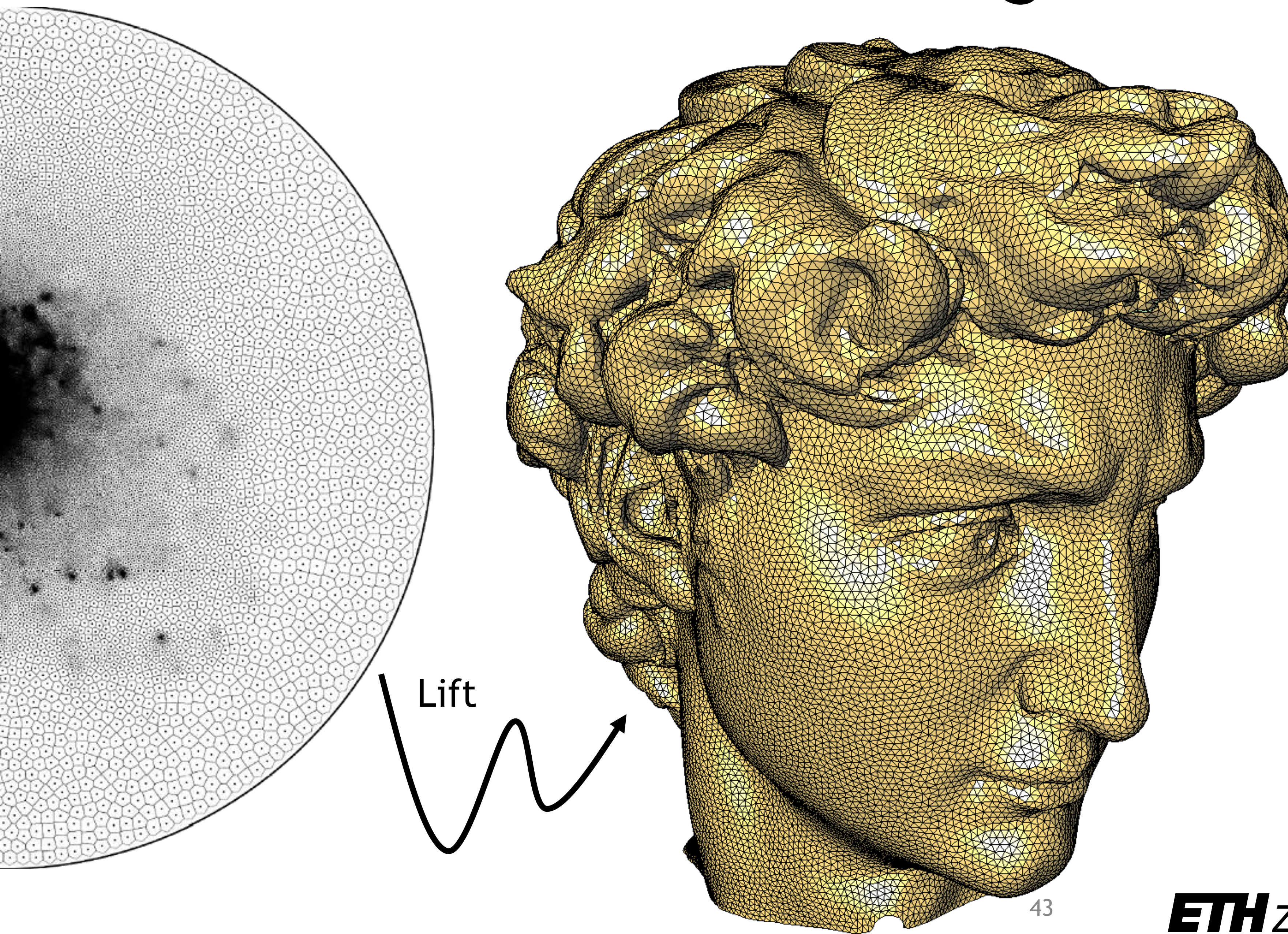
Voronoi tesselation

Optimized Sample Placement

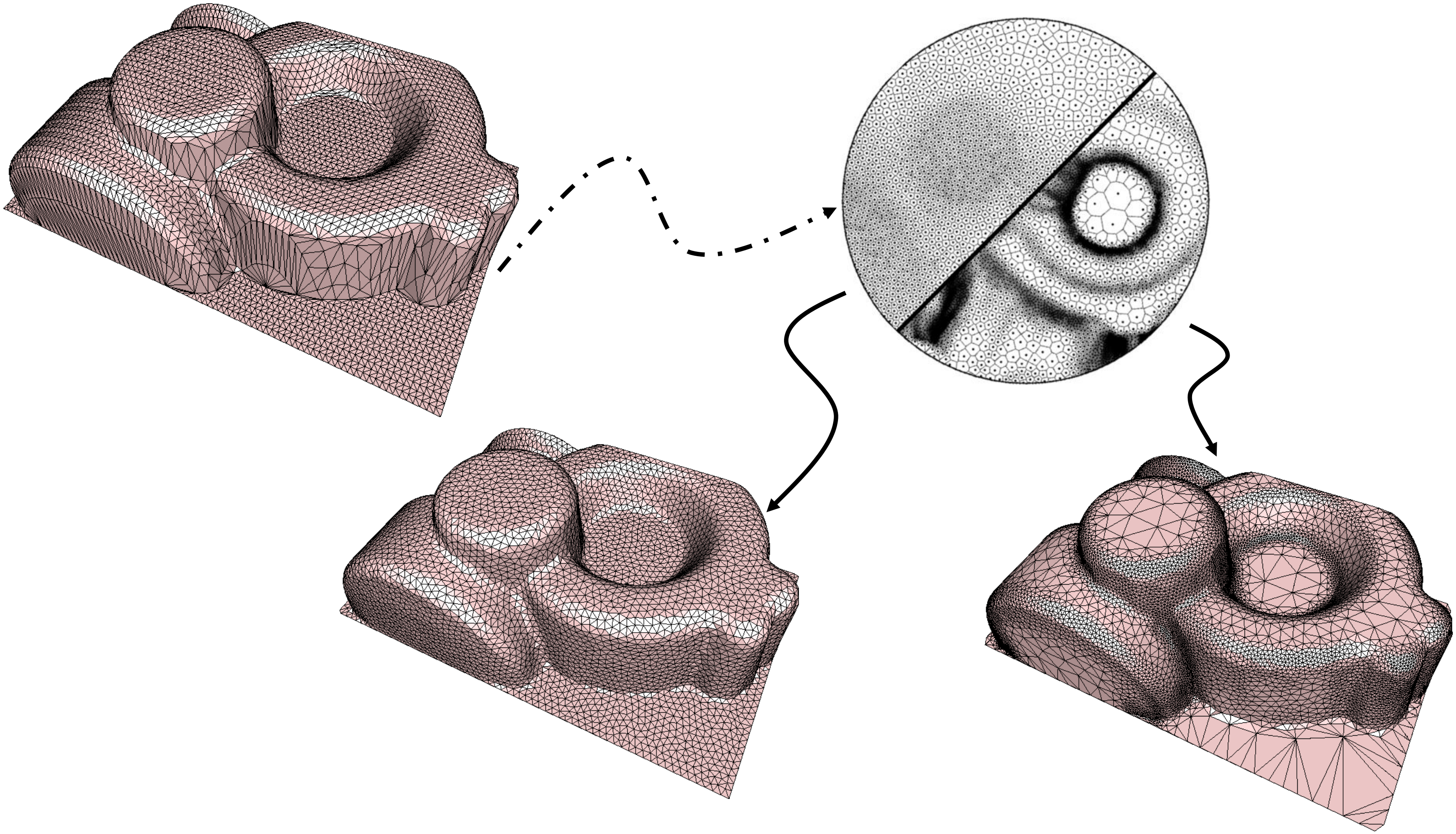


centroidal Voronoi tesselation

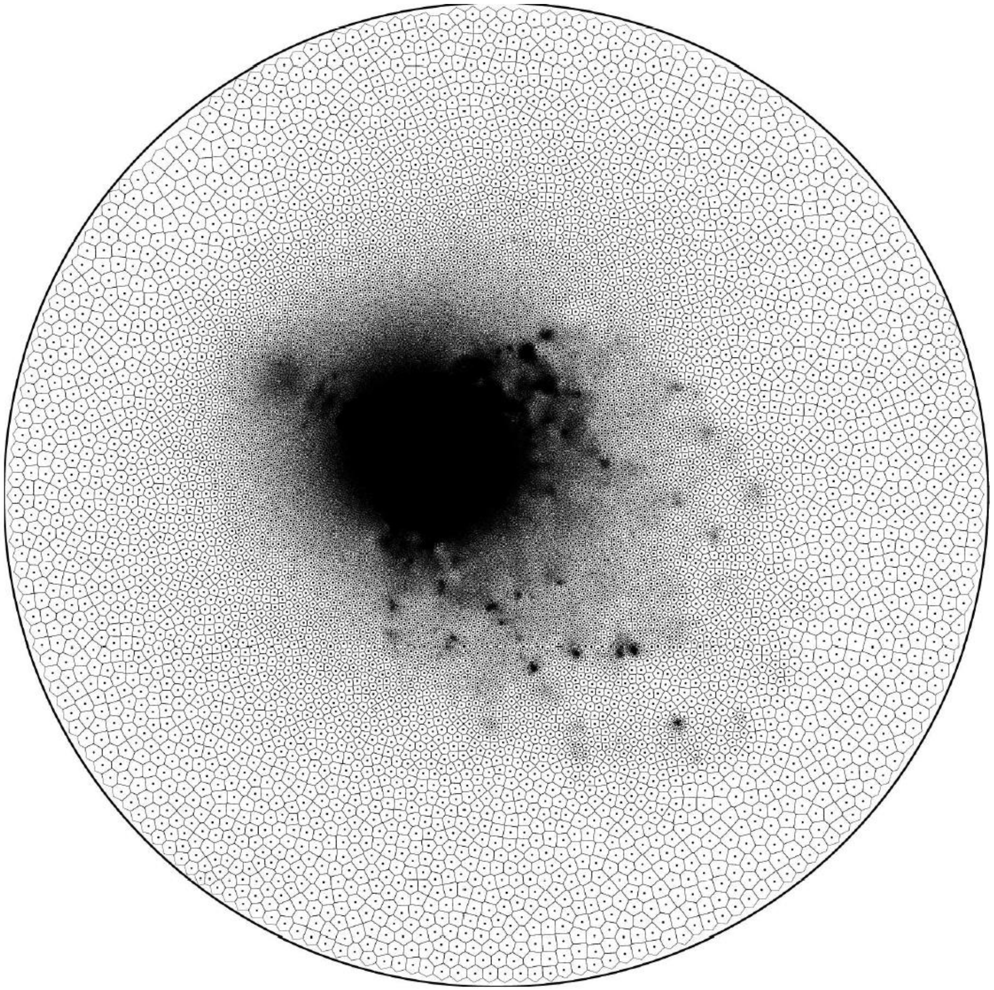
Uniform Remeshing



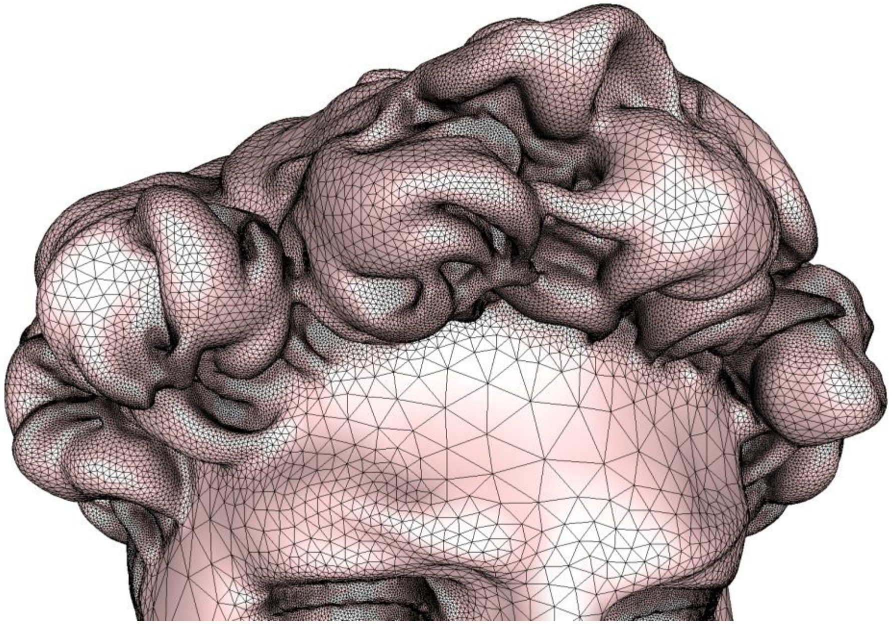
Uniform vs. Adaptive

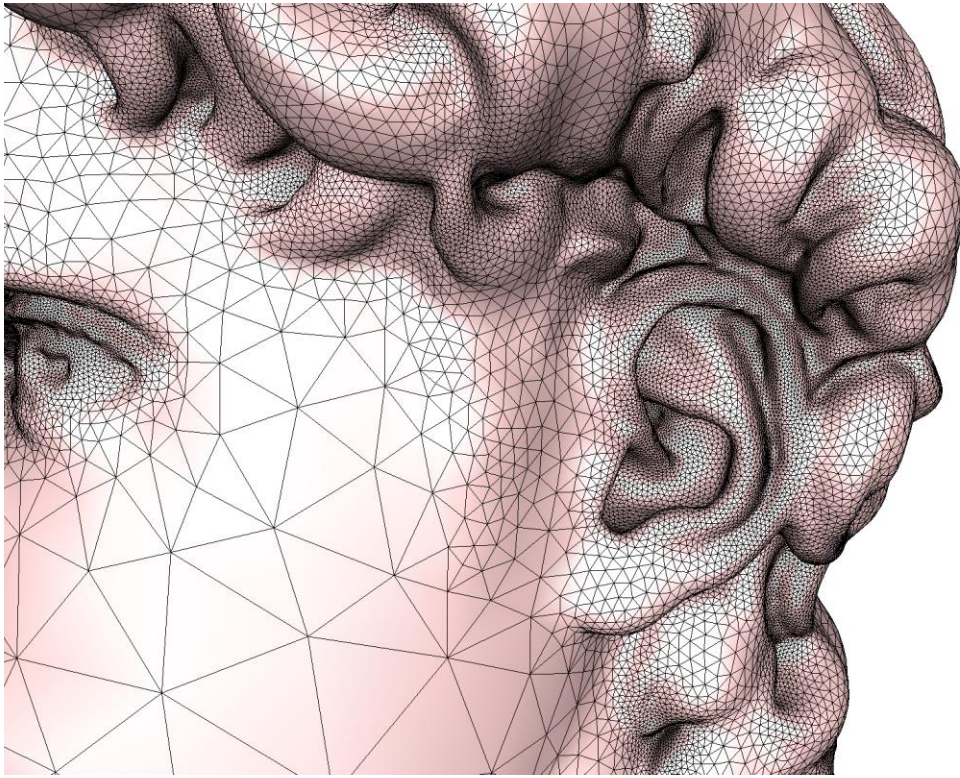


Uniform Sampling



Adaptive Sampling

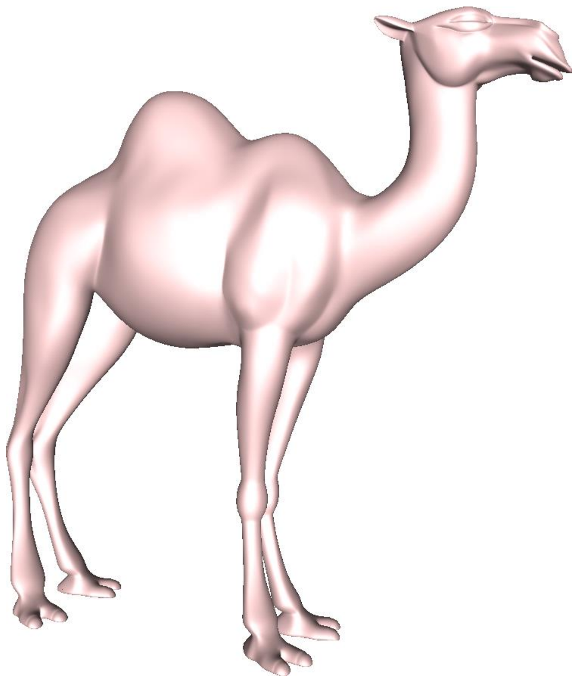




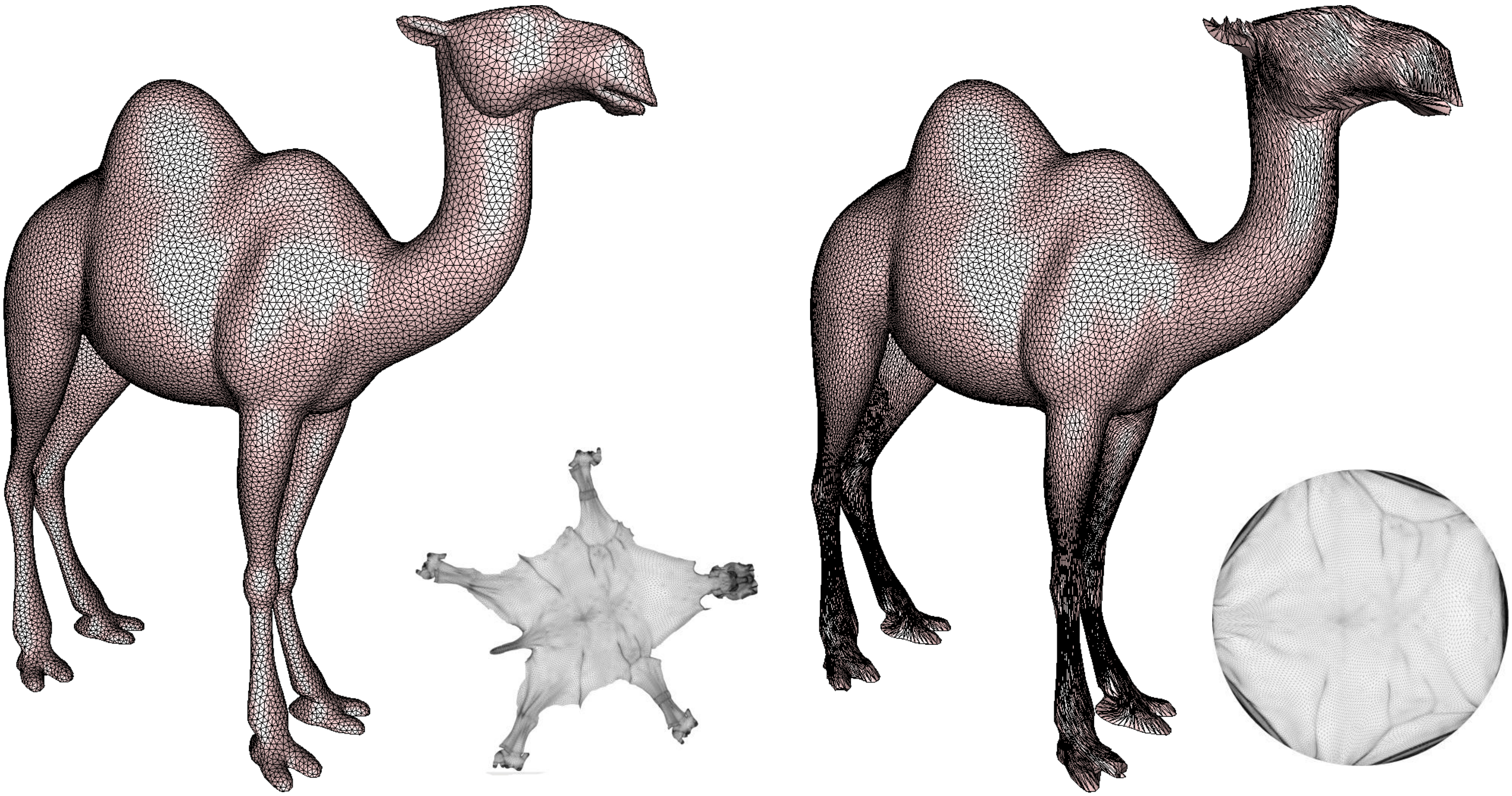
ETH zürich

Limitations

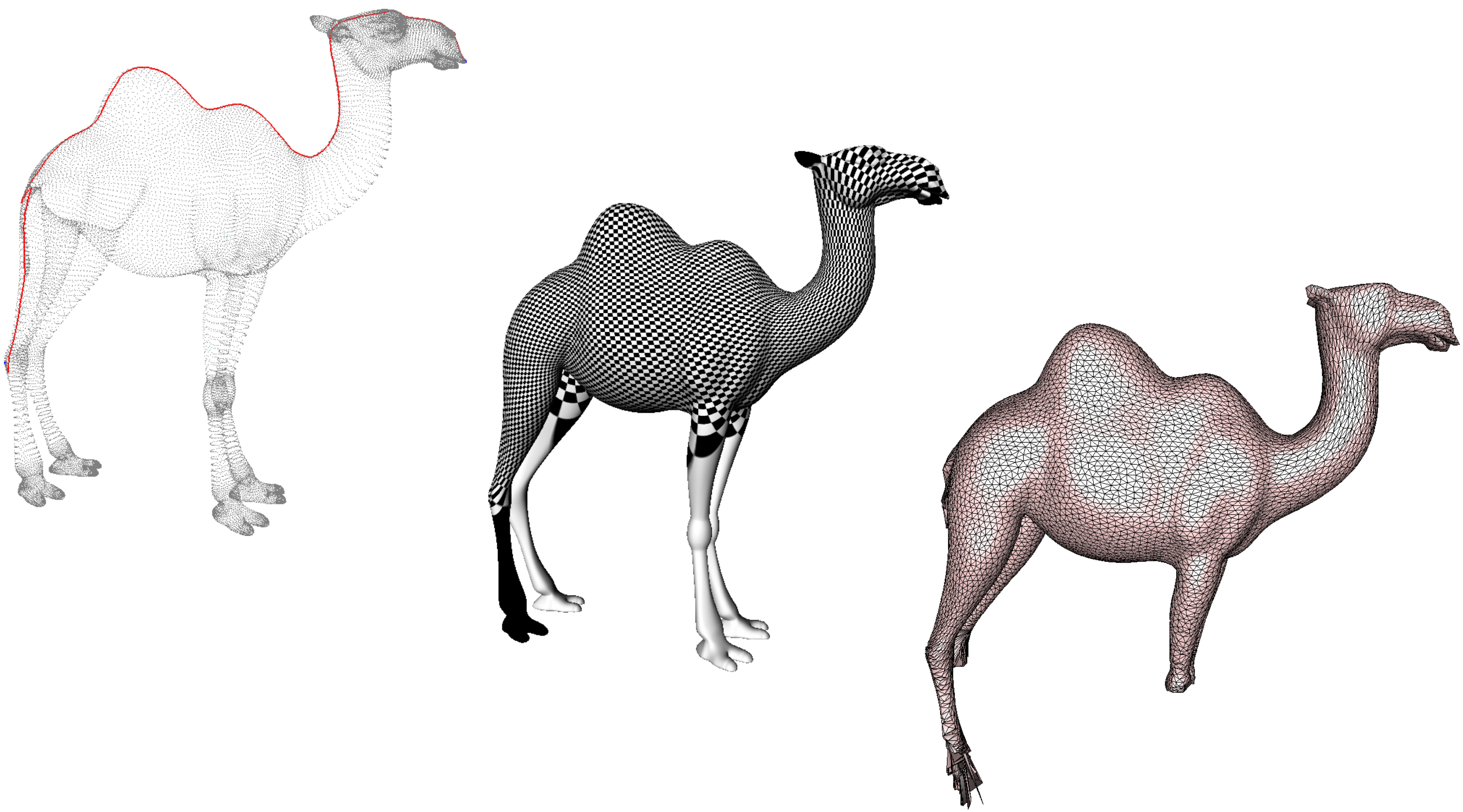
- Closed meshes
  - Need a good cut
  - Free boundary parameteriztion
  - Stitch seams afterwards
- Protruding legs
  - Sampling
  - Numerical problems



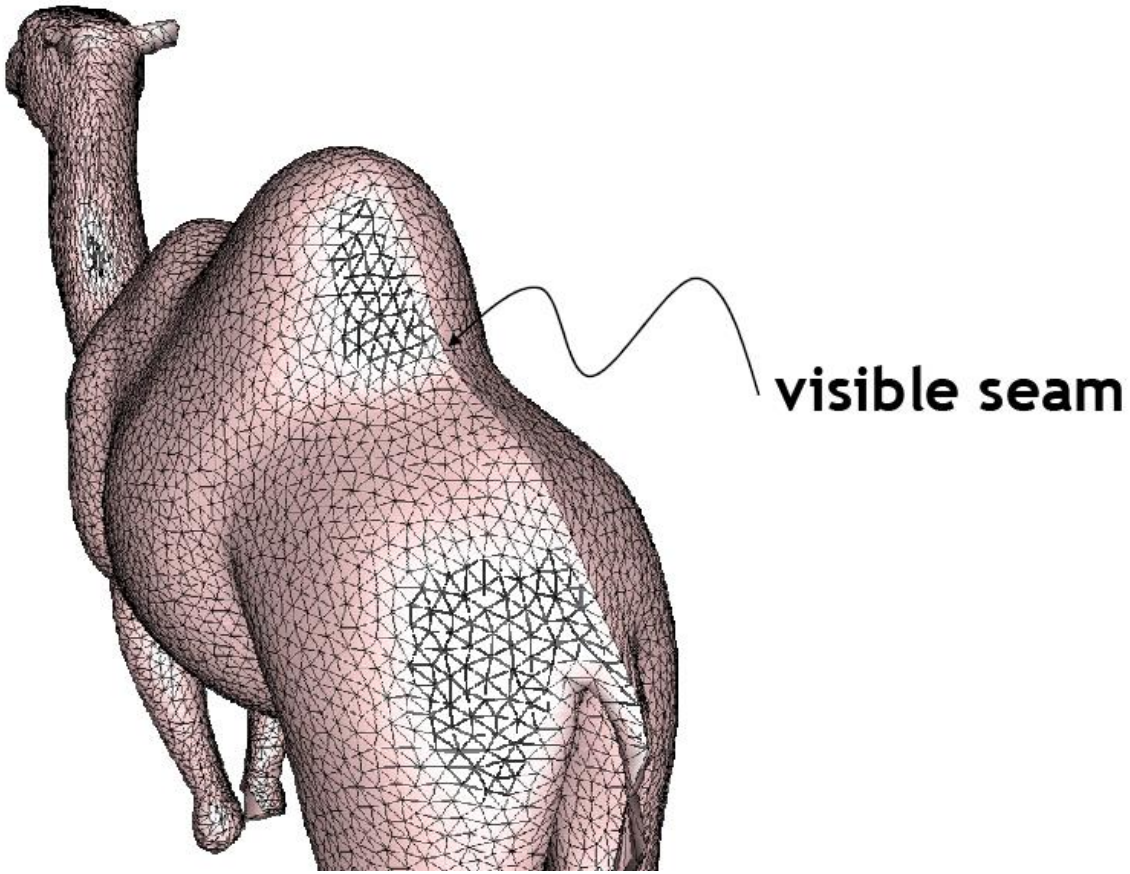
Free vs. Fixed Boundary



Naive Cut, Numerical Problems



Visible Seams



Direct Surface Remeshing

[Botsch et al. ’04]

Avoid global parameterization

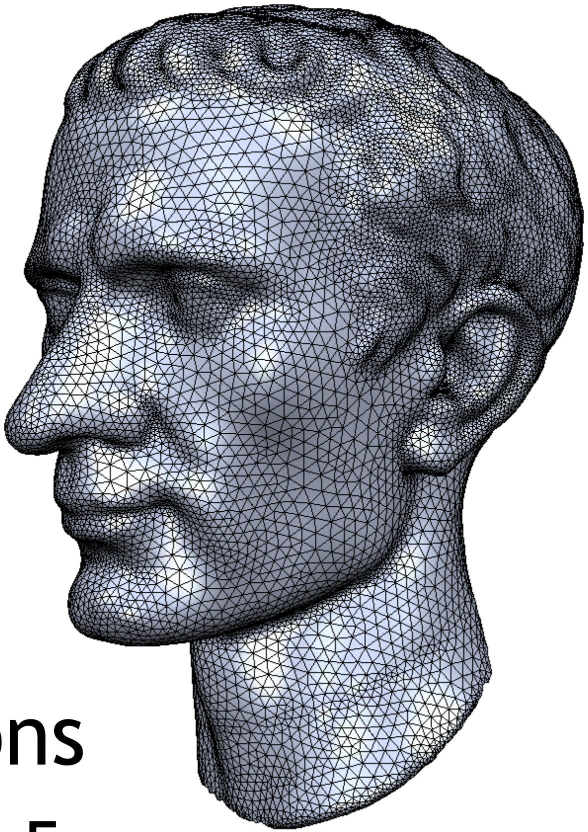
- Numerically very sensitive
- Topological restrictions

Avoid local parameterizations

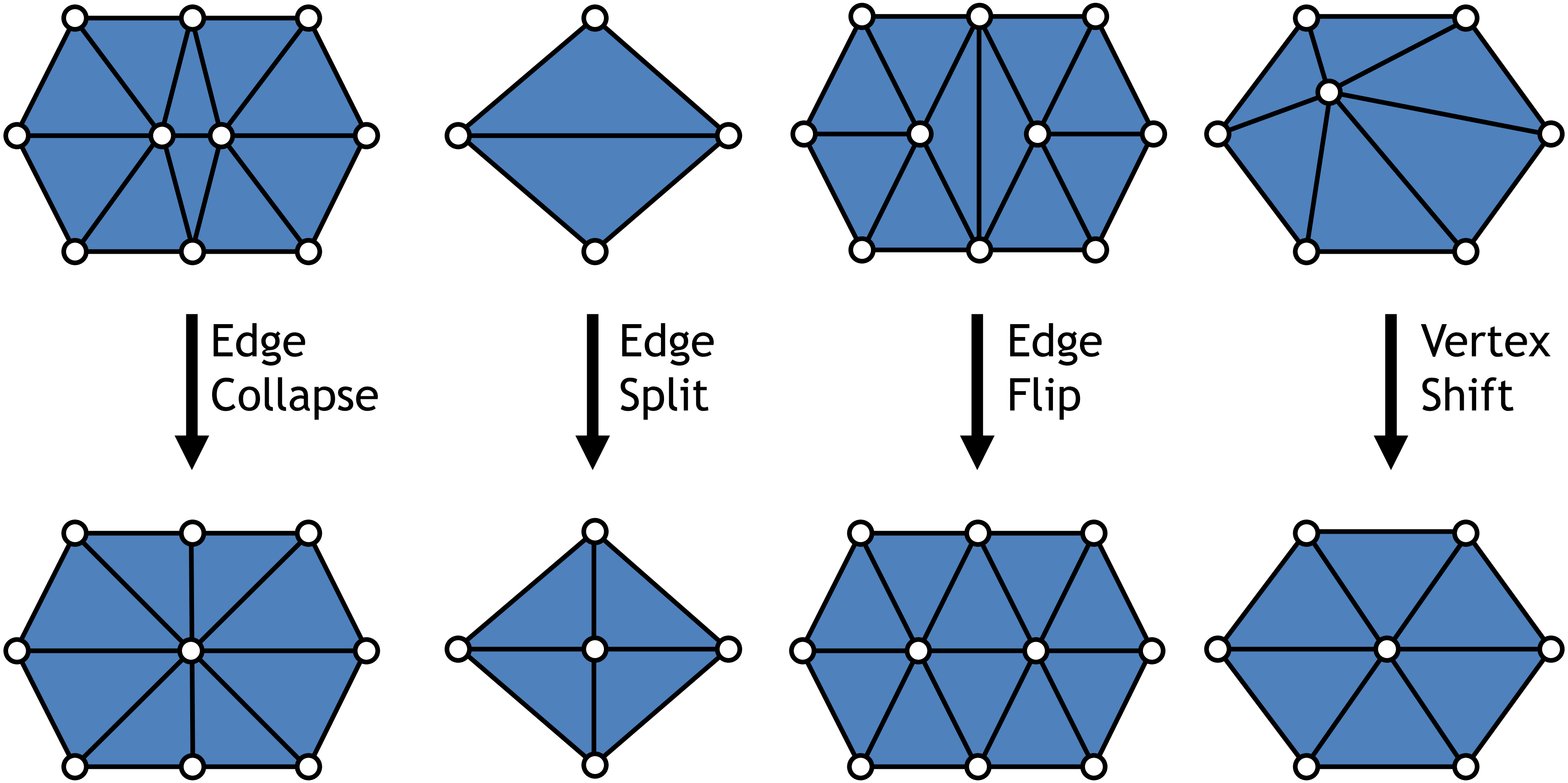
- Expensive computations

Use local operators & projections

- Resampling of 100k triangles in < 5s



Local Remeshing Operators



Isotropic Remeshing

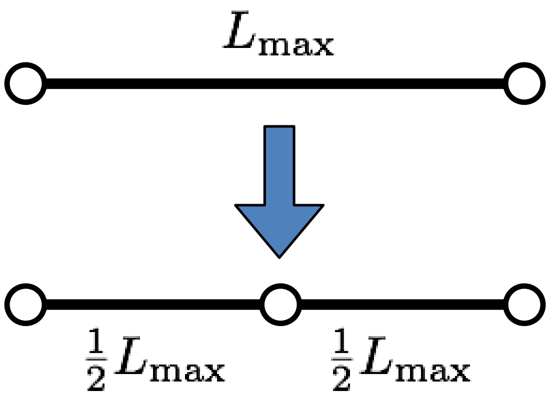
Specify target edge length L

Compute edge length range $[L_{min}, L_{max}]$

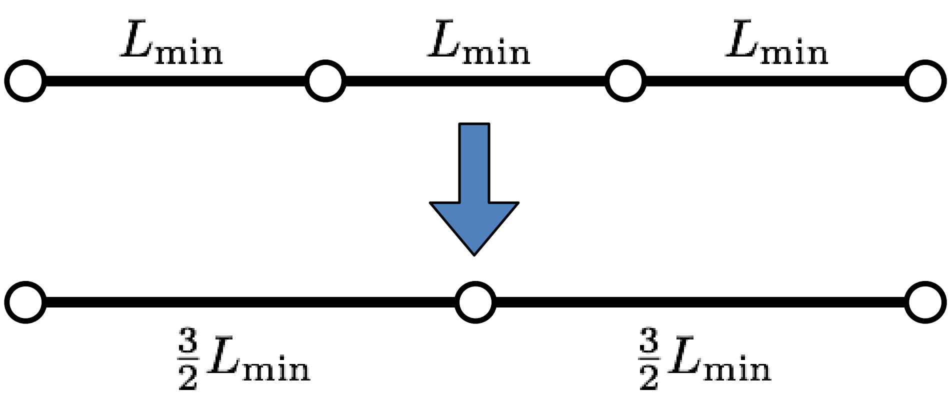
Iterate:

1. Split edges longer than L_{max}
2. Collapse edges shorter than L_{min}
3. Flip edges to get closer to valence 6
4. Vertex shift by tangential relaxation
5. Project vertices onto reference mesh

Edge Collapse / Split



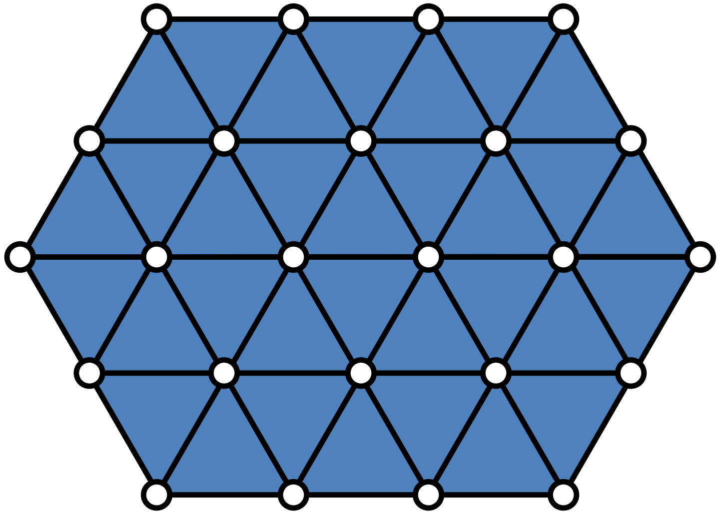
$$\begin{aligned} |L_{\max} - L| &= \left|\frac{1}{2}L_{\max} - L\right| \\ \Rightarrow L_{\max} &= \frac{4}{3}L \end{aligned}$$



$$\begin{aligned} |L_{\min} - L| &= \left|\frac{3}{2}L_{\min} - L\right| \\ \Rightarrow L_{\min} &= \frac{4}{5}L \end{aligned}$$

Edge Flip

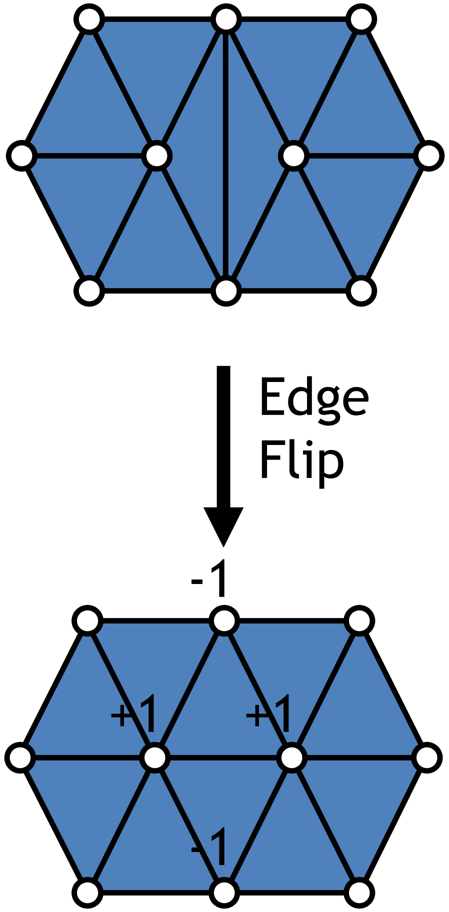
- Improve valences
  - Avg. valence is 6 (Euler)
  - Reduce variation
- Optimal valence is
  - 6 for interior vertices
  - 4 for boundary vertices



Edge Flip

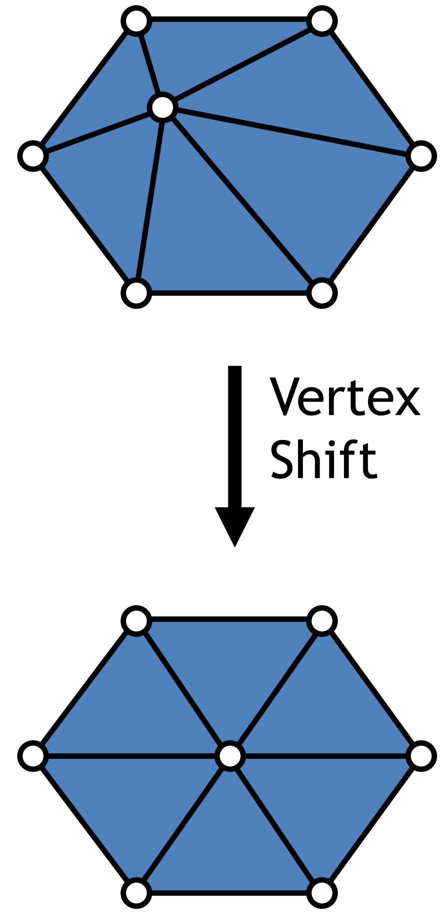
- Improve valences
  - Avg. valence is 6 (Euler)
  - Reduce variation
- Optimal valence is
  - 6 for interior vertices
  - 4 for boundary vertices
- Minimize valence excess

$$\sum_{i=1}^{4}(\text{valence}(v_i) - \text{opt\_valence}(v_i))^2$$



Vertex Shift

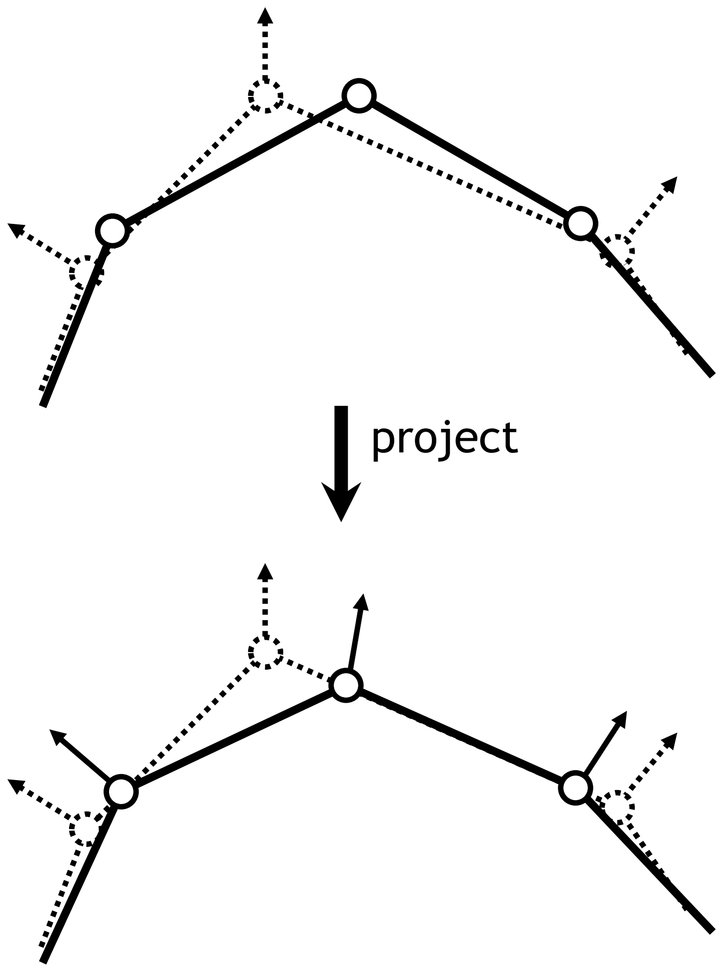
- Local "spring" relaxation
  - Uniform Laplacian smoothing
  - Bary-center of one-ring neighbors



$$\mathbf{c}_i = \frac{1}{\text{valence}(v_i)} \sum_{j \in N(v_i)} \mathbf{p}_j$$

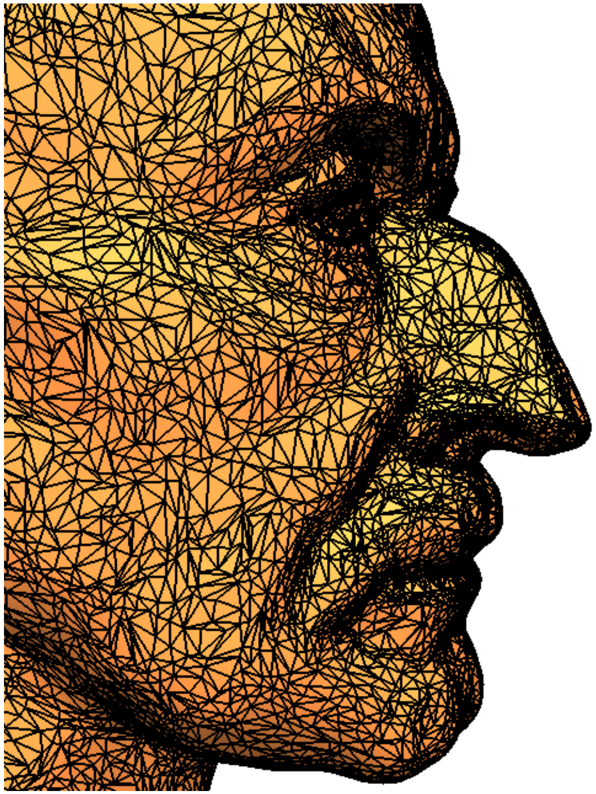
Vertex Projection

- Project vertices onto original reference mesh
- Assign position & interpolated normal

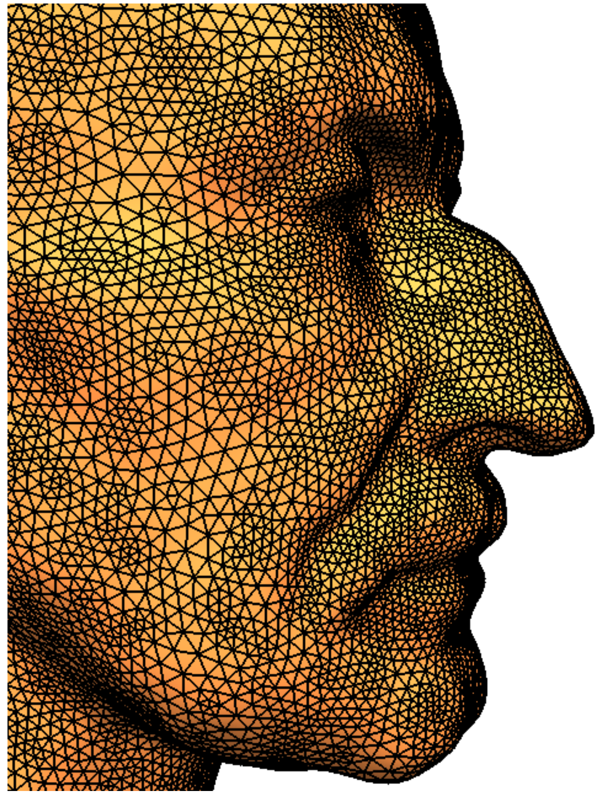


ETH zürich

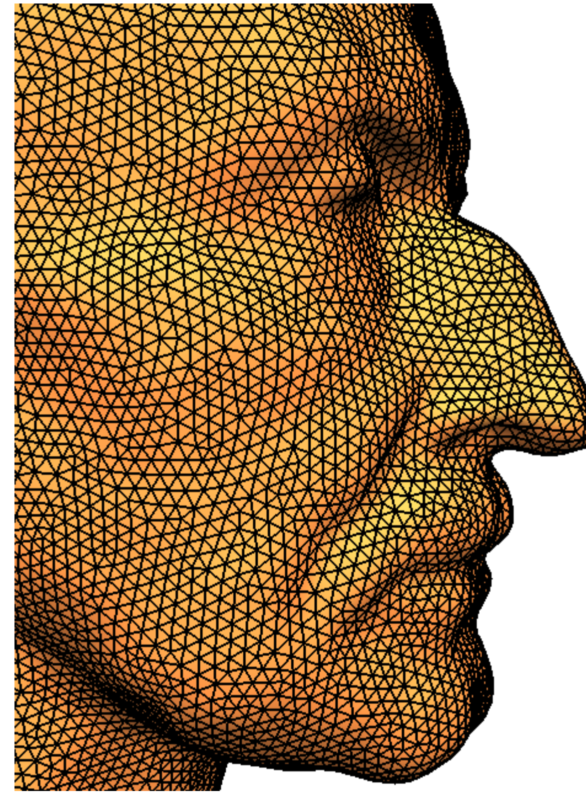
Remeshing Results



Original

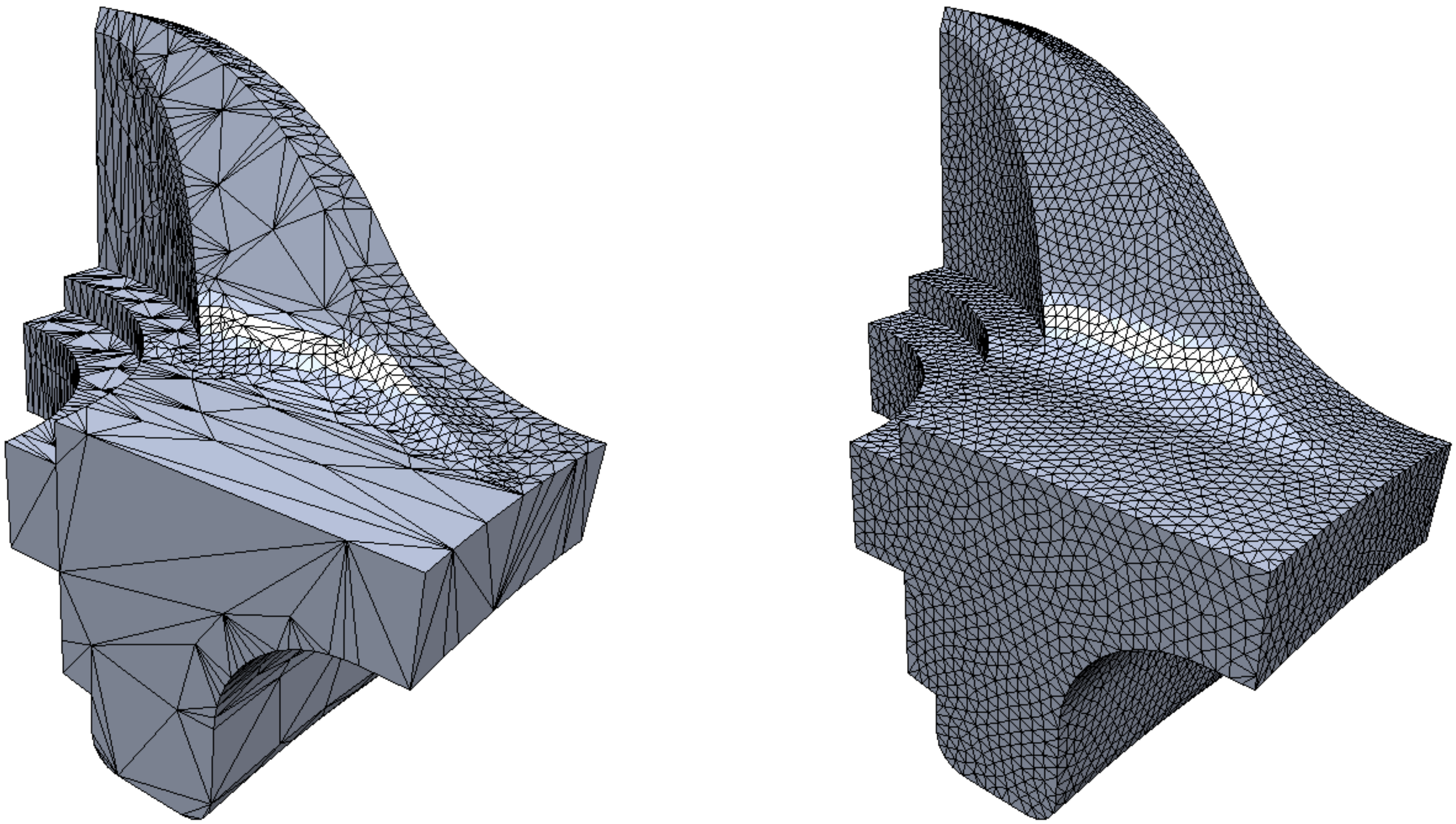


$\left(\frac{1}{2}, 2\right)$



$\left(\frac{4}{5}, \frac{4}{3}\right)$

Feature Preservation?



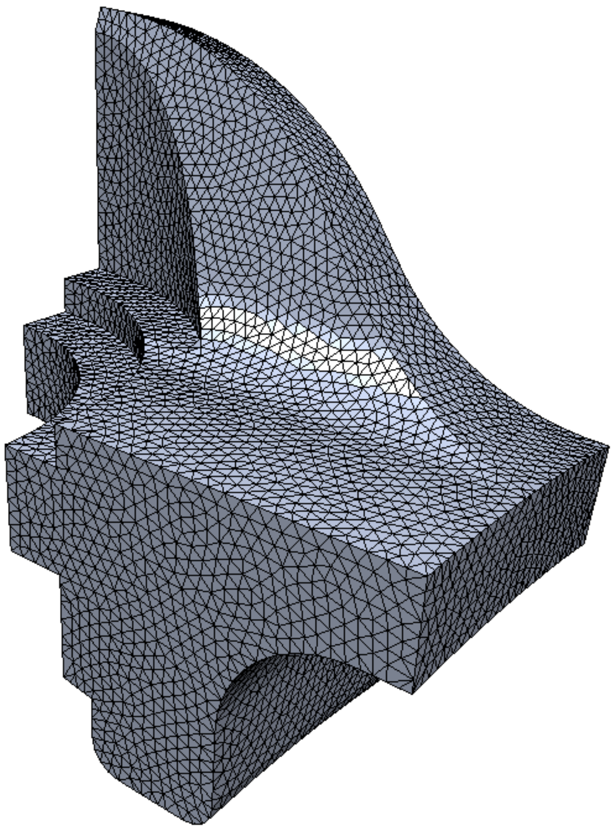
Feature Preservation

Define features

- Sharp edges
- Material boundaries

Adjust local operators

- Don't move corners
- Collapse only along features
- Don't flip feature edges
- Project to feature curves

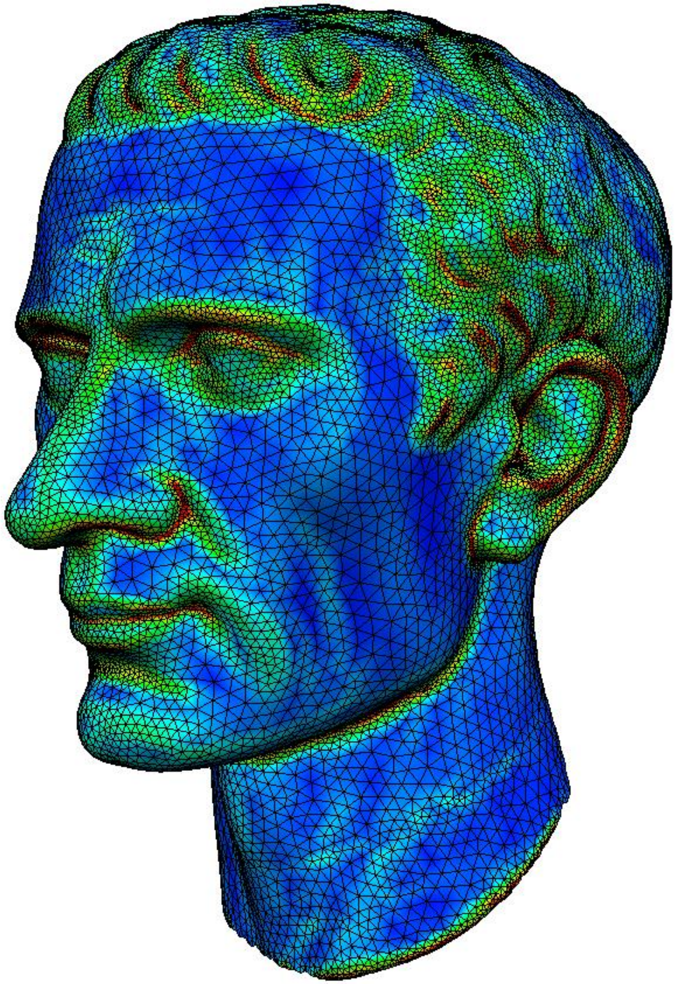


Adaptive Remeshing

Precompute max. curvature on reference mesh

Target edge length locally determined by curvature

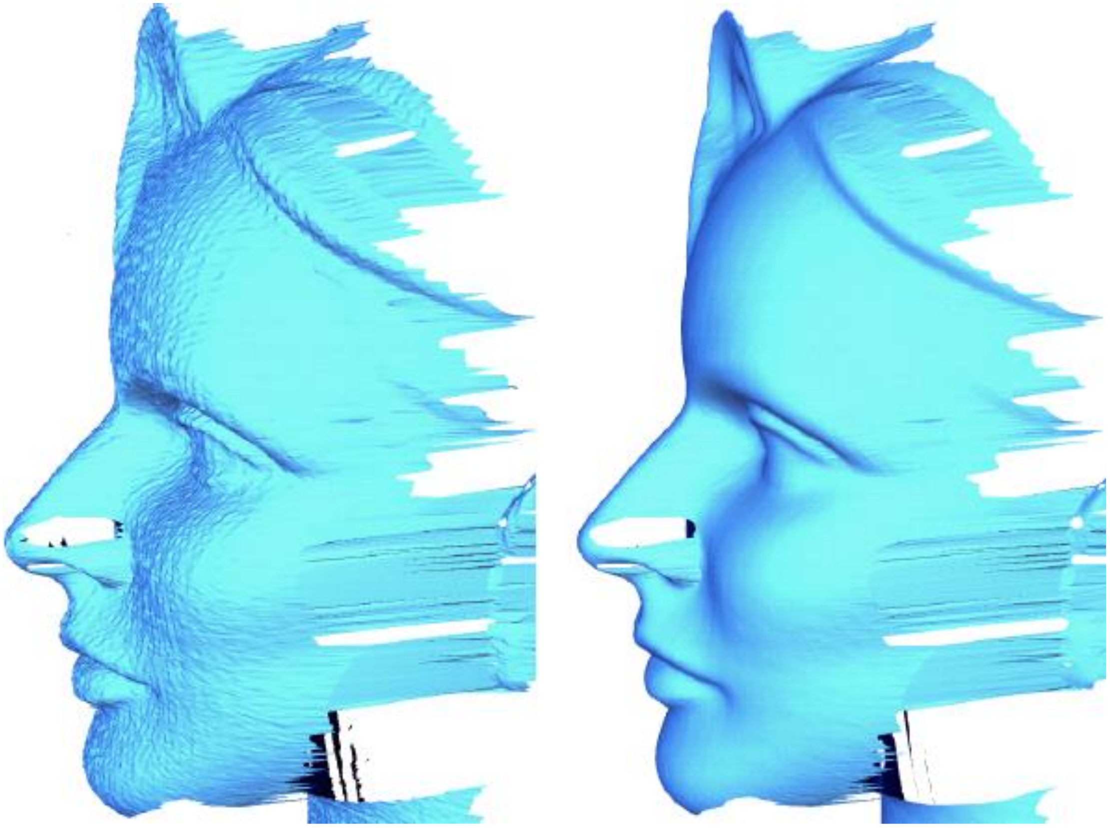
Adjust split / collapse criteria



Smoothing

Surface Smoothing – Motivation

Scanned surfaces can be noisy

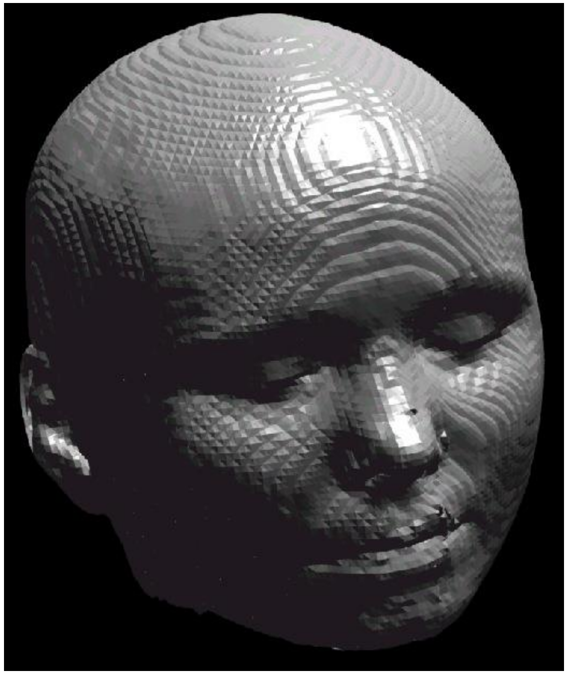


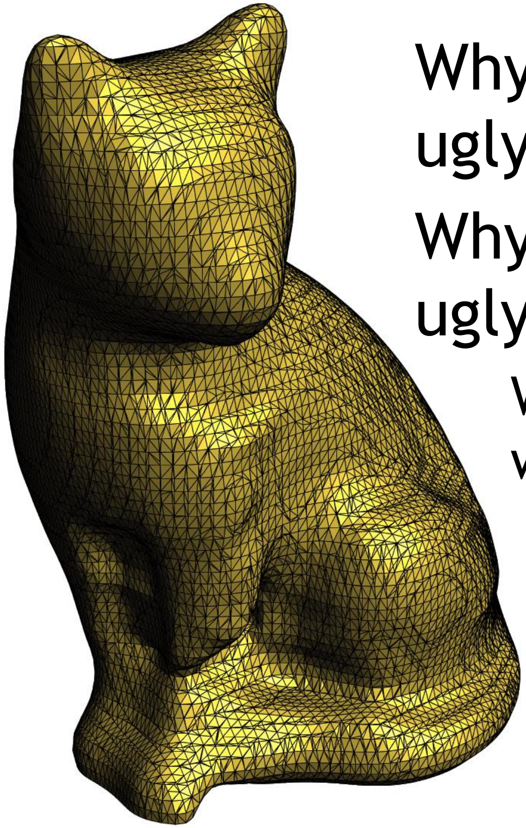
Surface Smoothing – Motivation

Scanned surfaces can be noisy

Surface Fairing – Motivation

Marching Cubes meshes are ugly!





Why is the left mesh ugly?

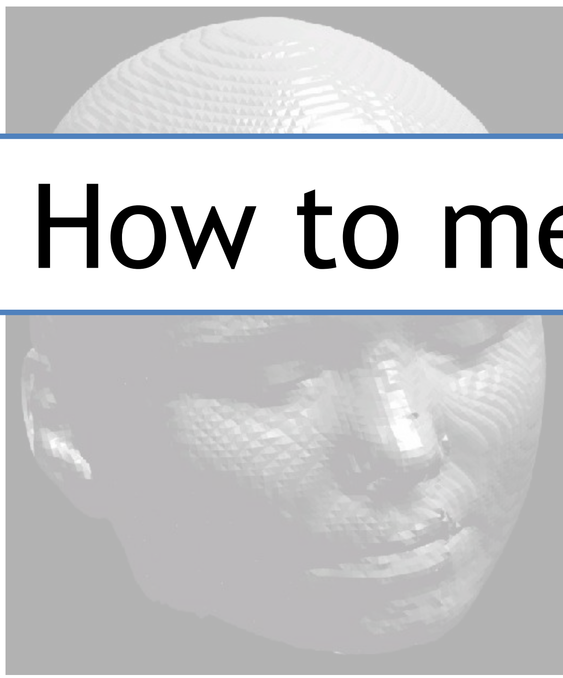
Why is the right mesh ugly?

- What is the problem with such triangles?

ETH zürich

Surface Fairing – Motivation

Marching Cubes meshes are ugly!



Why is the left mesh ugly?

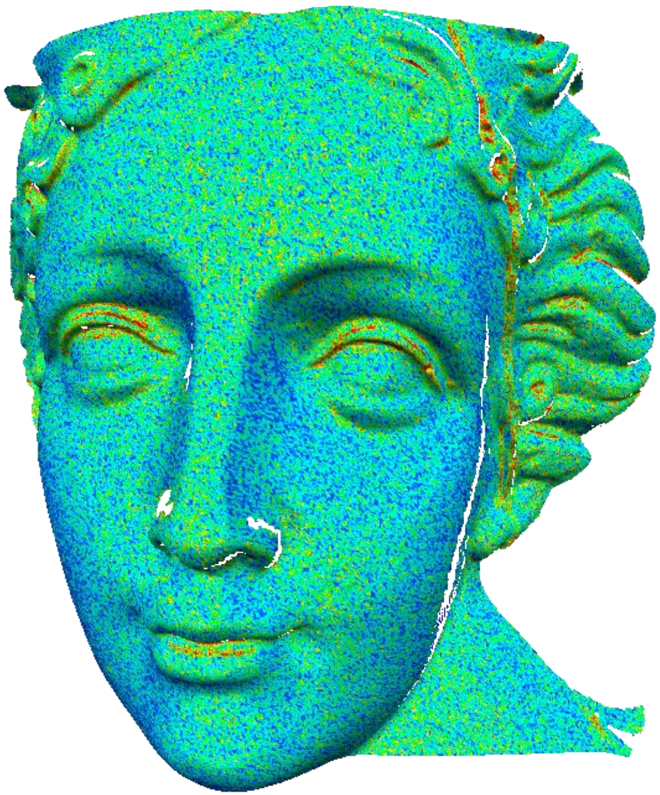
esh

How to measure smoothness?

What is the problem with such triangles?

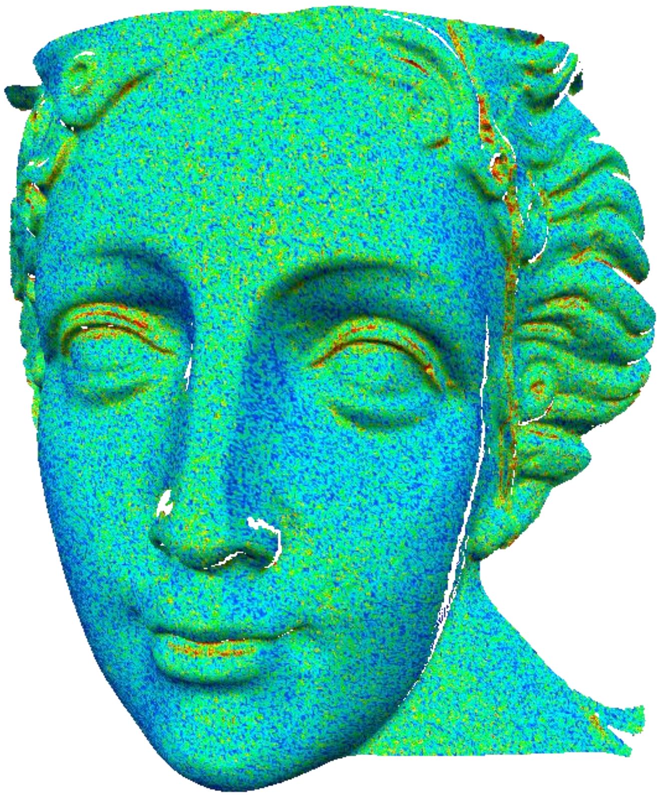
Curvature and Smoothness

Curvature and Smoothness

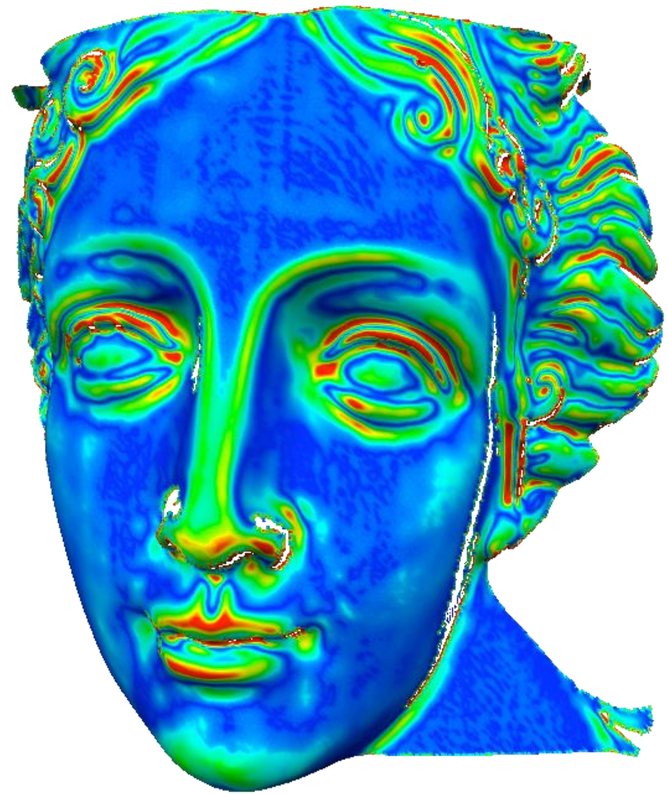


mean curvature plot

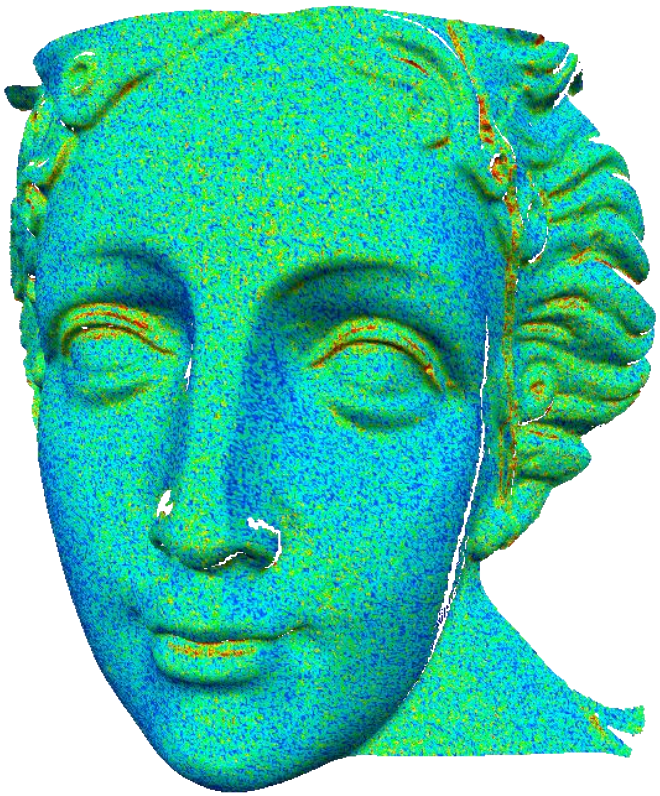
Curvature and Smoothness



mean curvature plot

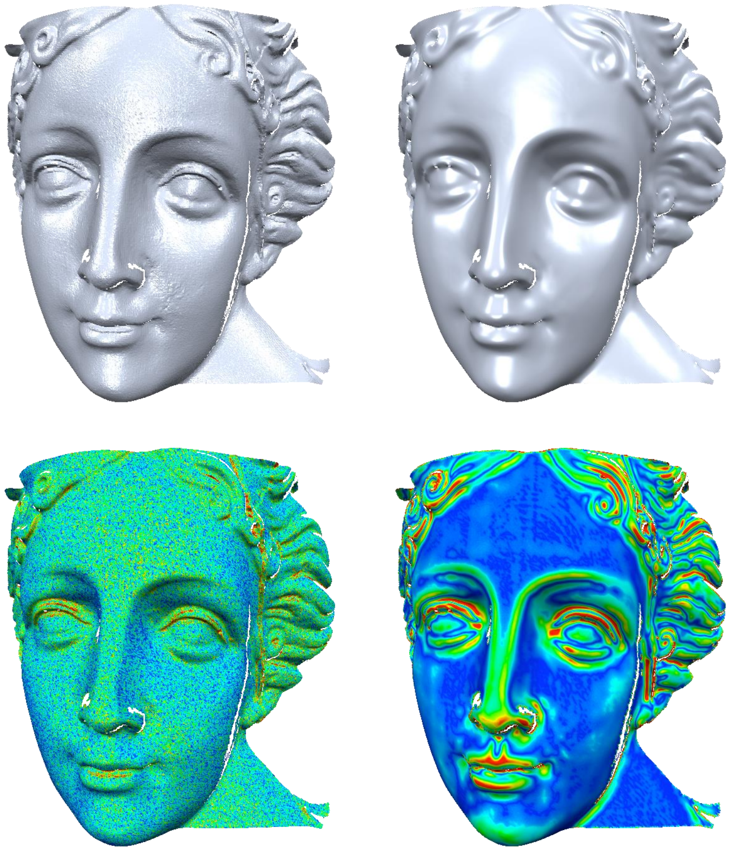


Curvature and Smoothness



mean curvature plot

Curvature and Smoothness



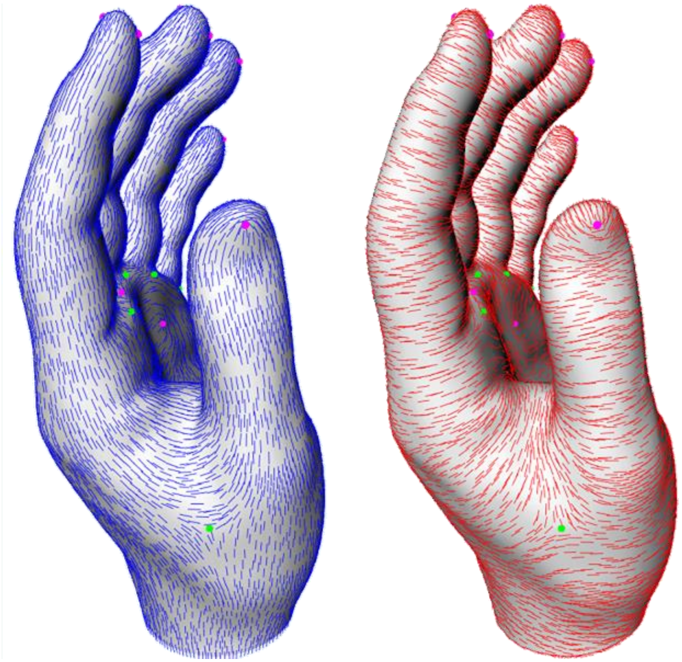
Is smoothing =
reducing curvature?

Is smoothing =
make curvature change less?

Which curvature?

Principal curvatures $\kappa_{\min}, \kappa_{\max}$

- Nonlinear and "discontinuous" operator in the definition (min, max)



principal directions

Which curvature?

Principal curvatures $\kappa_{\min}, \kappa_{\max}$

Nonlinear and "discontinuous" operator in the definition (min, max)

Gauss curvature K

Intrinsic-only, insensitive to embedding in $\mathbb{R}^3$

Which curvature?

Principal curvatures $\kappa_{\min}, \kappa_{\max}$

Nonlinear and "discontinuous" operator in the definition (min, max)

Gauss curvature K

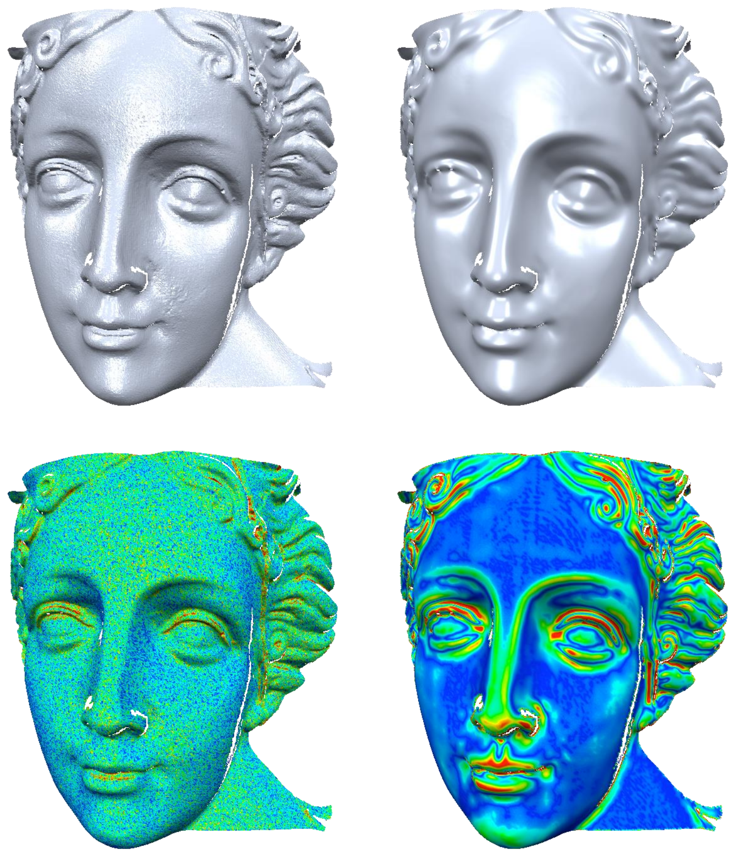
Intrinsic-only, insensitive to embedding in $\mathbb{R}^3$

Mean curvature H

Relatively simple to extract on meshes via Laplace-Beltrami:

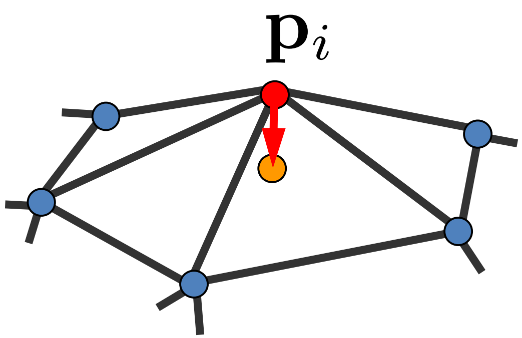
$$\Delta_{\mathcal{M}} \mathbf{p} = -2H\mathbf{n}$$

goal: $H = 0$ or $H = \text{const}$



$\boxed{\Delta_{\mathcal{M}}\mathbf{p} = -2H\mathbf{n}}$

Recap: Laplace-Beltrami



$$\Delta_{\mathcal{M}}(\mathbf{p}_i) = \delta_i = \frac{1}{W_i} \sum_{j \in \mathcal{N}(i)} w_{ij}(\mathbf{p}_j - \mathbf{p}_i)$$

The direction of δ_i approximates the normal

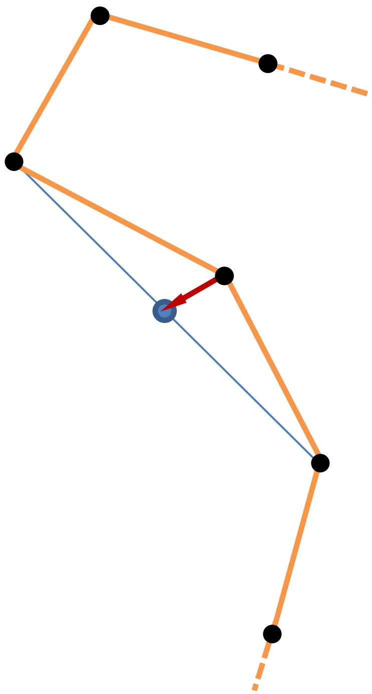
The size approximates the mean curvature

Smoothing by flowing

Example – smoothing curves

- Laplace in 1D = second derivative:

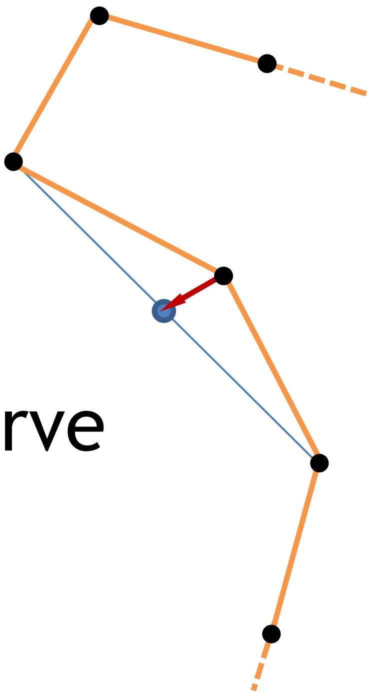
$$L(\mathbf{p}_i) = \frac{1}{2}(\mathbf{p}_{i-1} - \mathbf{p}_i) + \frac{1}{2}(\mathbf{p}_{i+1} - \mathbf{p}_i)$$



Example – smoothing curves

- Laplace in 1D = second derivative:

$$L(\mathbf{p}_i) = \frac{1}{2}(\mathbf{p}_{i-1} - \mathbf{p}_i) + \frac{1}{2}(\mathbf{p}_{i+1} - \mathbf{p}_i)$$



- In matrix-vector form for the whole curve

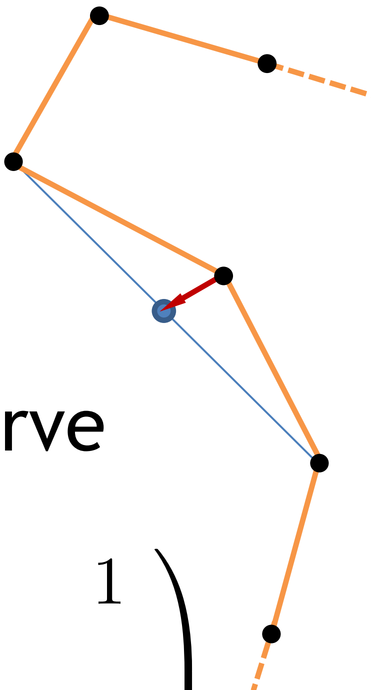
$$L\mathbf{p}$$

$$\mathbf{p} = [\mathbf{x}\ \mathbf{y}] \in \mathbb{R}^{n\times 2}$$

Example – smoothing curves

- Laplace in 1D = second derivative:

$$L(\mathbf{p}_i) = \frac{1}{2}(\mathbf{p}_{i-1} - \mathbf{p}_i) + \frac{1}{2}(\mathbf{p}_{i+1} - \mathbf{p}_i)$$



- In matrix-vector form for the whole curve

$$L\mathbf{p}$$

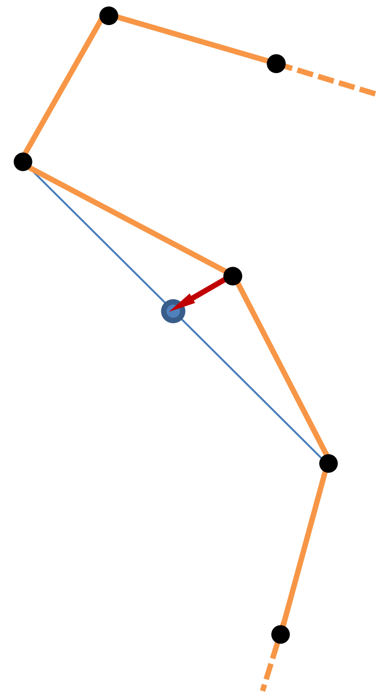
$$\mathbf{p} = [\mathbf{x}\ \mathbf{y}] \in \mathbb{R}^{n \times 2}$$

$$L = \frac{1}{2}\begin{pmatrix} -2 & 1 & & & 1 \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ 1 & & & 1 & -2 \end{pmatrix}$$

Example – smoothing curves

- Flow to reduce curvature:

$$\tilde{\mathbf{p}}_i = \mathbf{p}_i + \lambda \frac{d^2}{ds^2}(\mathbf{p}_i)$$



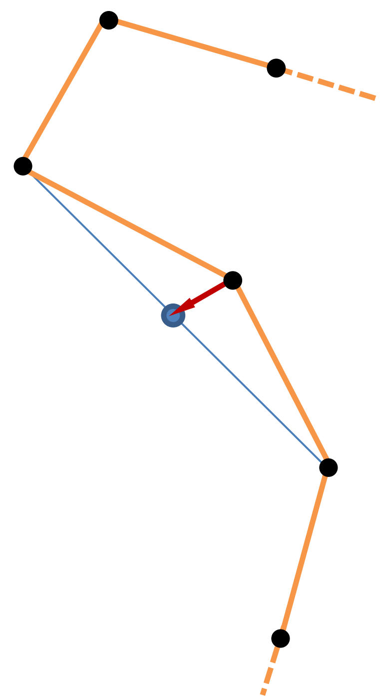
- Scale factor $0 < \lambda < 1$
- Matrix-vector form:

$$\tilde{\mathbf{p}} = \mathbf{p} + \lambda L \mathbf{p}, \quad \mathbf{p} \in \mathbb{R}^{n \times 2}$$

Example – smoothing curves

- Flow to reduce curvature:

$$\tilde{\mathbf{p}}_i = \mathbf{p}_i + \lambda L(\mathbf{p}_i) = \mathbf{p}_i + \lambda \frac{d^2}{ds^2}(\mathbf{p}_i)$$



- Scale factor $0 < \lambda < 1$
- Matrix-vector form:

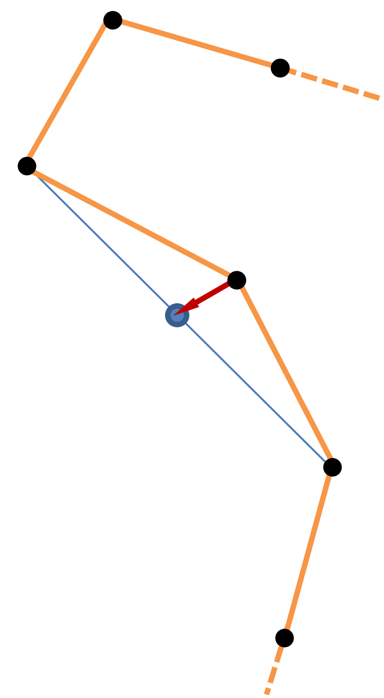
$$\tilde{\mathbf{p}} = \mathbf{p} + \lambda L \mathbf{p}, \quad \mathbf{p} \in \mathbb{R}^{n \times 2}$$

- Drawbacks?

Example – smoothing curves

- Flow to reduce curvature:

$$\tilde{\mathbf{p}}_i = \mathbf{p}_i + \lambda L(\mathbf{p}_i) = \mathbf{p}_i + \lambda \frac{d^2}{ds^2}(\mathbf{p}_i)$$

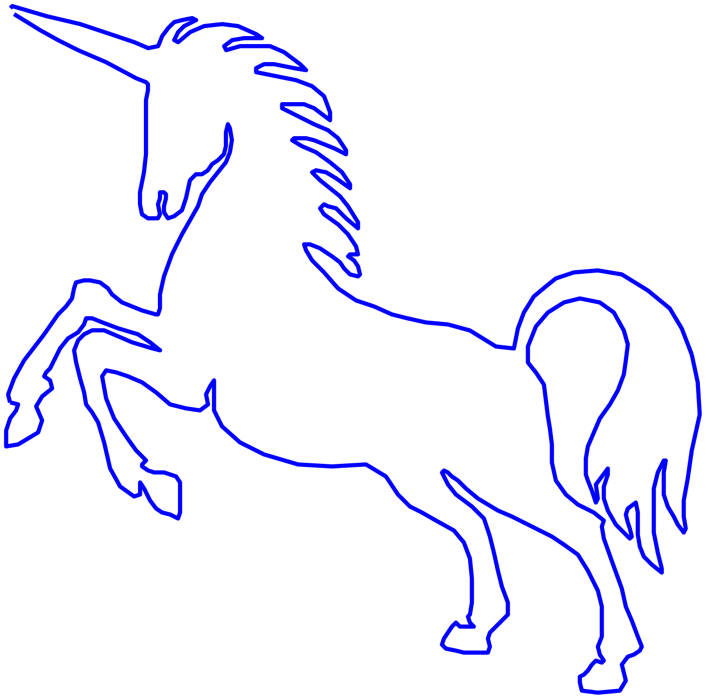


- Scale factor $0 < \lambda < 1$
- Matrix-vector form:

$$\tilde{\mathbf{p}} = \mathbf{p} + \lambda L \mathbf{p}, \quad \mathbf{p} \in \mathbb{R}^{n \times 2}$$

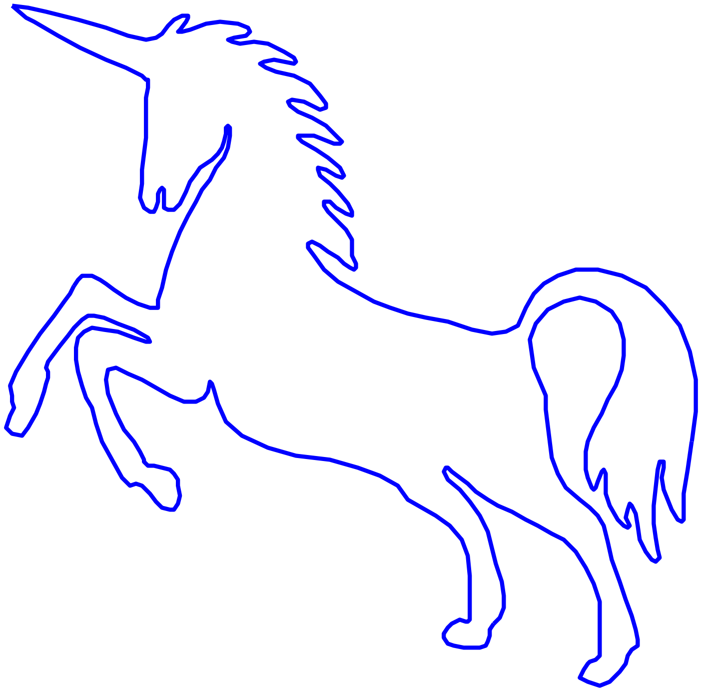
- May shrink the shape; can be slow

Filtering Curves



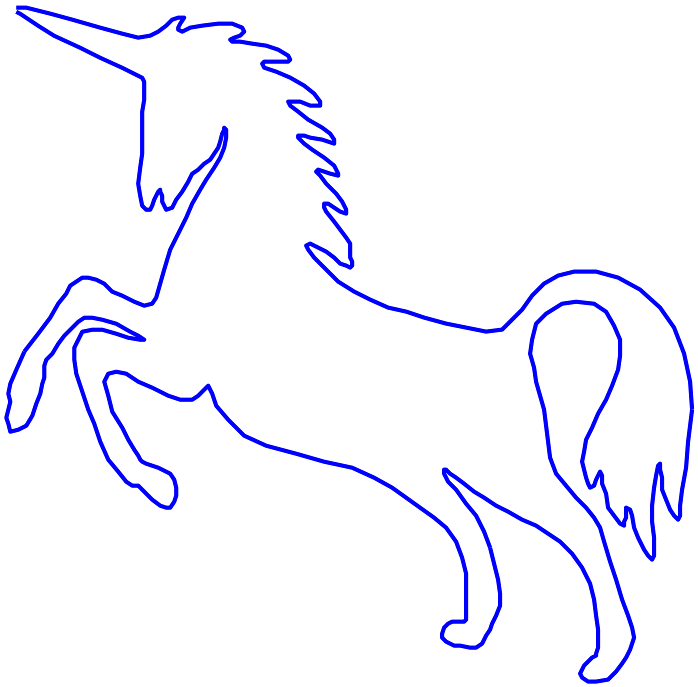
Original curve

Filtering Curves



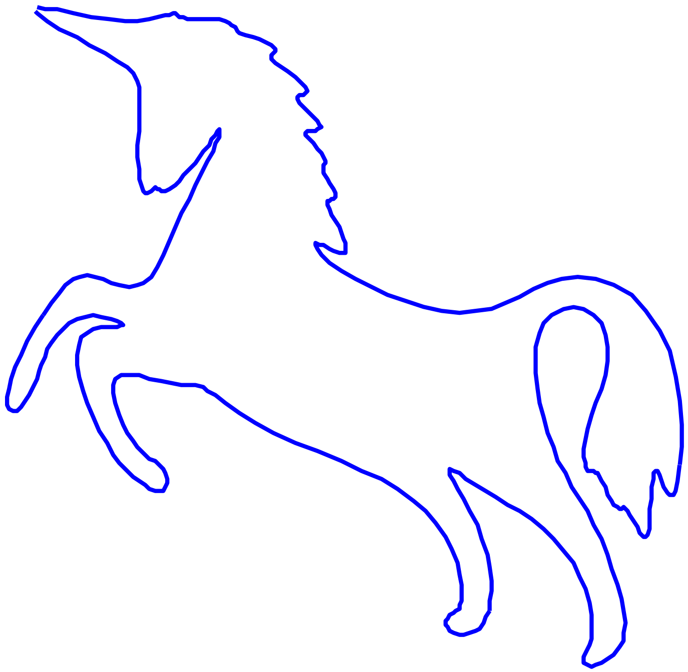
1st iteration; $\lambda=0.5$

Filtering Curves



2nd iteration; λ=0.5

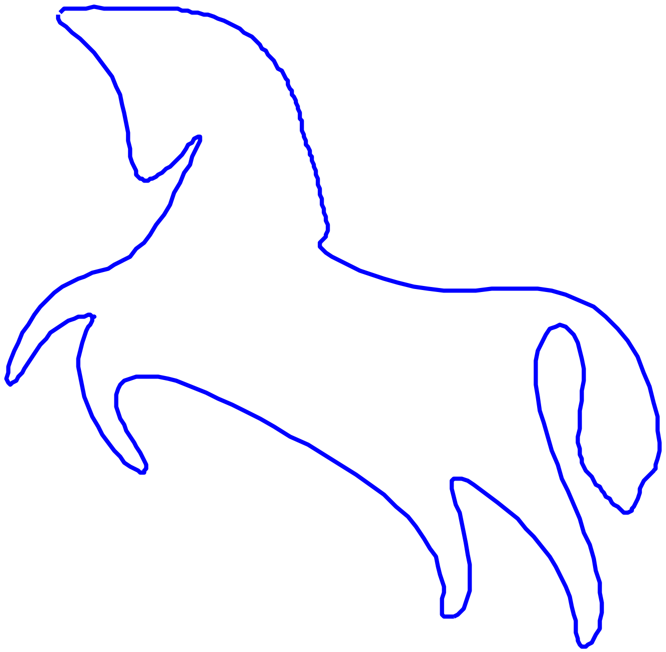
Filtering Curves



8th iteration; λ=0.5

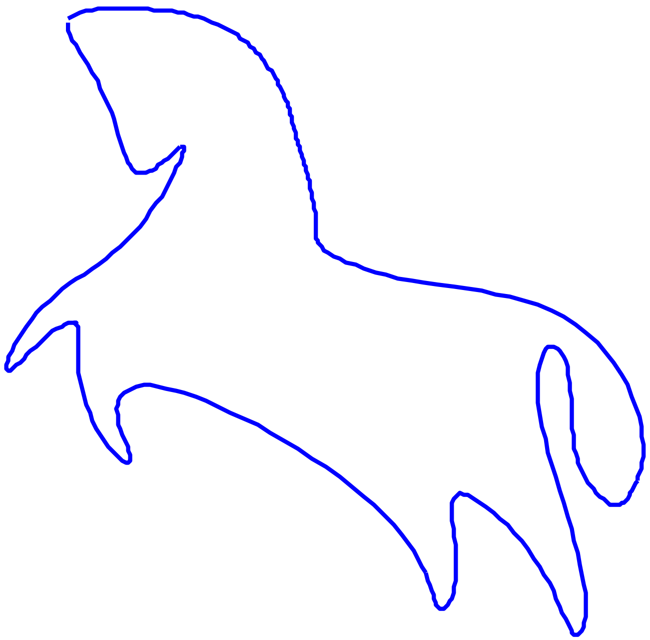
ETHzürich

Filtering Curves



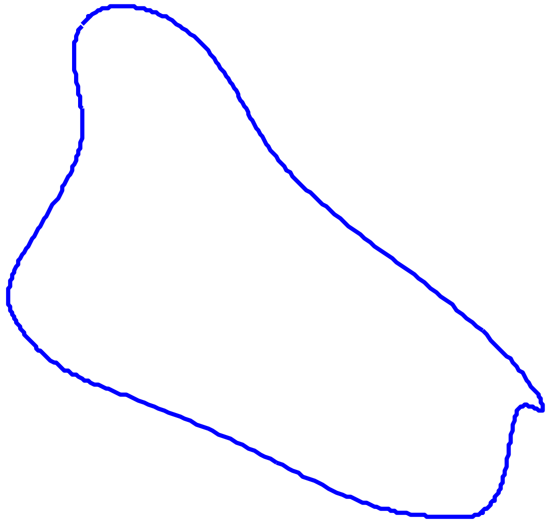
27th iteration; λ=0.5

Filtering Curves



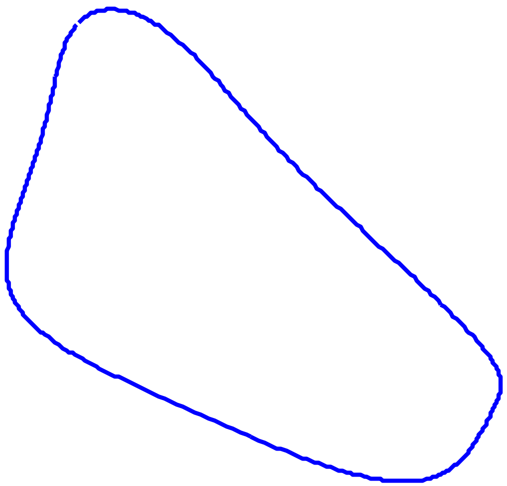
50th iteration; λ=0.5

Filtering Curves



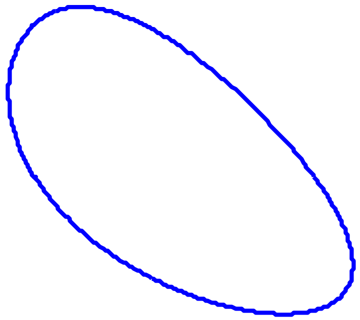
500th iteration; $\lambda=0.5$

Filtering Curves



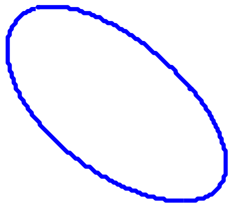
1000th iteration; $\lambda=0.5$

Filtering Curves



5000th iteration; $\lambda=0.5$

Filtering Curves



10000th iteration; $\lambda=0.5$

Filtering Curves

50000th iteration; $\lambda=0.5$

On meshes: smoothing as mean curvature flow

- Model smoothing as a diffusion process

$$\boxed{\frac{\partial \mathbf{p}}{\partial t} = \lambda \Delta \mathbf{p}} = -2\lambda H \mathbf{n}$$

On meshes: smoothing as mean curvature flow

- Model smoothing as a diffusion process

$$\frac{\partial \mathbf{p}}{\partial t} = \lambda \Delta \mathbf{p} = -2\lambda H \mathbf{n}$$

- Discretize in time, forward differences:

$$\frac{\mathbf{p}^{(n+1)} - \mathbf{p}^{(n)}}{dt} = \lambda L \mathbf{p}^{(n)}$$

On meshes: smoothing as mean curvature flow

- Model smoothing as a diffusion process

$$\boxed{\frac{\partial \mathbf{p}}{\partial t} = \lambda \Delta \mathbf{p}} = -2\lambda H \mathbf{n}$$

- Discretize in time, forward differences:

$$\frac{\mathbf{p}^{(n+1)} - \mathbf{p}^{(n)}}{dt} = \lambda L \mathbf{p}^{(n)}$$

$$\mathbf{p}^{(n+1)} - \mathbf{p}^{(n)} = dt\,\lambda L \mathbf{p}^{(n)}$$

On meshes: smoothing as mean curvature flow

- Model smoothing as a diffusion process

$$\boxed{\frac{\partial \mathbf{p}}{\partial t} = \lambda \Delta \mathbf{p}} = -2\lambda H \mathbf{n}$$

- Discretize in time, forward differences:

$$\frac{\mathbf{p}^{(n+1)} - \mathbf{p}^{(n)}}{dt} = \lambda L \mathbf{p}^{(n)}$$

$$\mathbf{p}^{(n+1)} - \mathbf{p}^{(n)} = dt\, \lambda L \mathbf{p}^{(n)}$$

$$\boxed{\mathbf{p}^{(n+1)} = (I + dt\, \lambda L)\mathbf{p}^{(n)}}$$

Explicit integration! Unstable unless time step dt is small

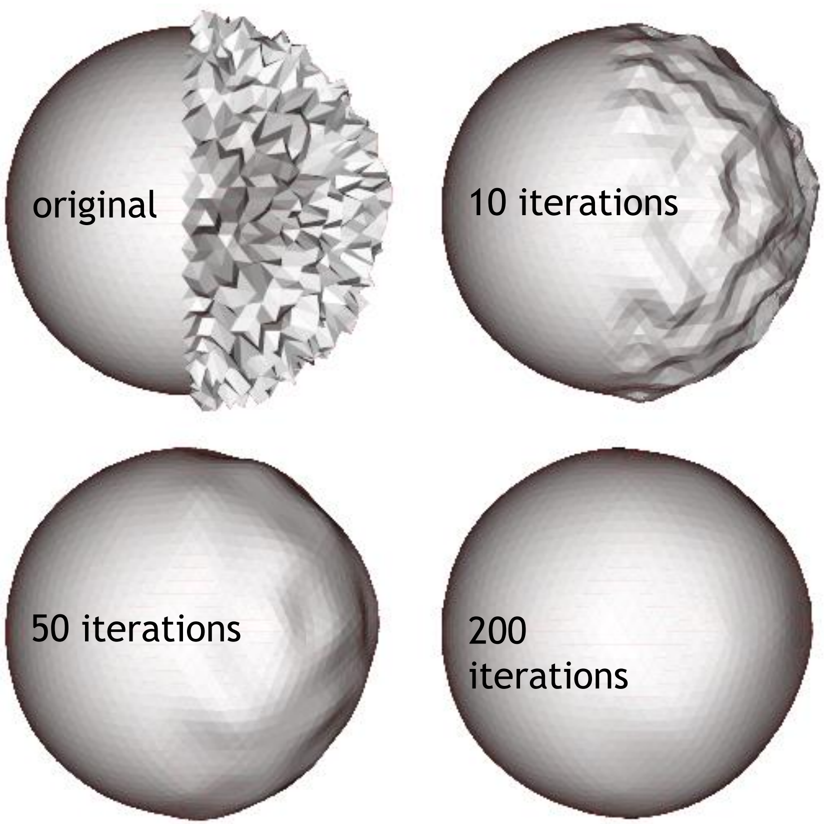
Taubin Smoothing: Explicit Steps

- Iterate:

$$\tilde{\mathbf{p}} = \mathbf{p} + \lambda L \mathbf{p} = (I + \lambda L)\mathbf{p}$$

$$\tilde{\mathbf{p}} = \mathbf{p} + \mu L \mathbf{p} = (I + \mu L)\mathbf{p}$$

- $\lambda > 0$ to smooth;
  $\mu < 0$ to inflate
- Originally proposed with uniform Laplacian weights



A Signal Processing Approach to Fair Surface Design
Gabriel Taubin
ACM SIGGRAPH 95

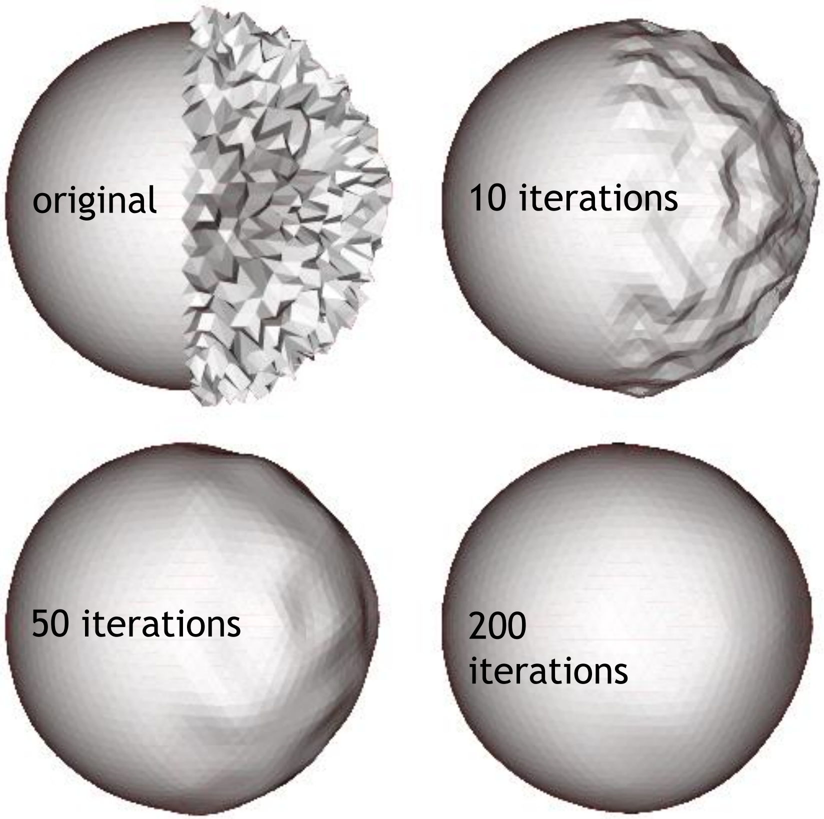
Taubin Smoothing: Explicit Steps

- Per-vertex iterations

$$\tilde{\mathbf{p}}_i = \mathbf{p}_i + \lambda L(\mathbf{p}_i)$$

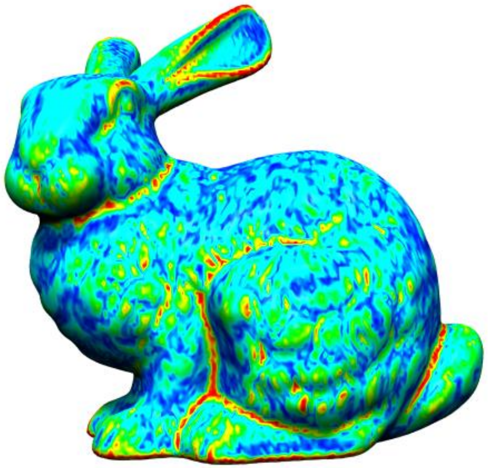
$$\tilde{\mathbf{p}}_i = \mathbf{p}_i + \mu L(\mathbf{p}_i)$$

- Simple to implement
- Requires many iterations
- Need to tweak μ and λ

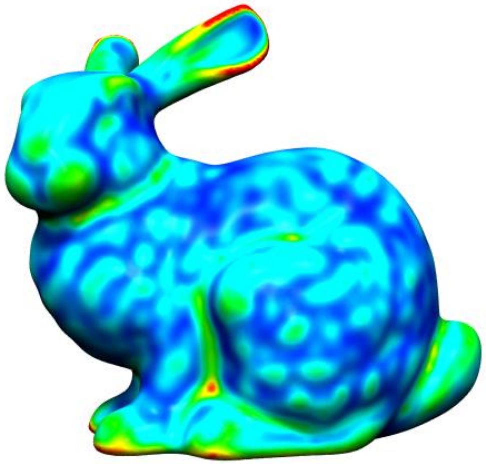


A Signal Processing Approach to Fair Surface Design
Gabriel Taubin
ACM SIGGRAPH 95

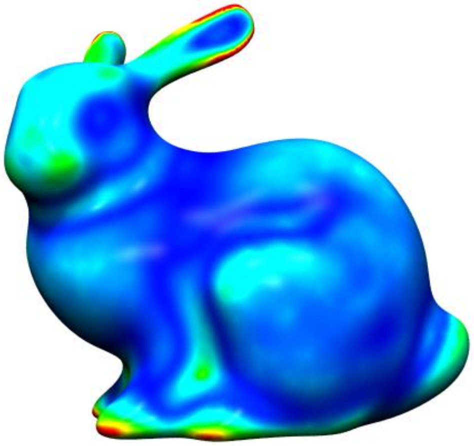
Example



0 iterations



10 iterations



100 iterations

Smoothing as Mean Curvature Flow

- Model smoothing as a diffusion process

$$\boxed{\frac{\partial \mathbf{p}}{\partial t} = \lambda \Delta \mathbf{p}} = -2\lambda H \mathbf{n}$$

- Backward Euler for unconditional stability

$$\frac{\mathbf{p}^{(n+1)} - \mathbf{p}^{(n)}}{dt} = \lambda L \mathbf{p}^{\boxed{(n+1)}}$$

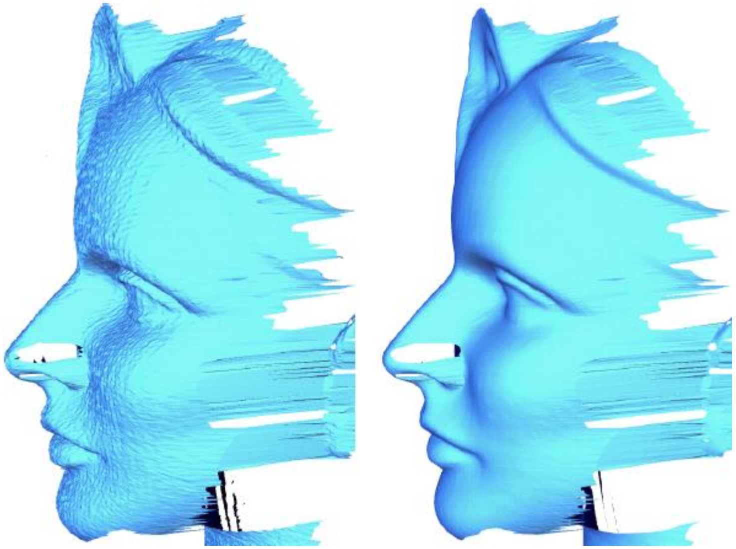
$$\mathbf{p}^{(n+1)} - \mathbf{p}^{(n)} = dt\,\lambda L \mathbf{p}^{(n+1)}$$

$$\boxed{(I - dt\,\lambda L)\mathbf{p}^{(n+1)} = \mathbf{p}^{(n)}}$$

Implicit Fairing: Implicit Euler Steps

- In each iteration, solve for the smoothed $\tilde{\mathbf{p}}$:

$$(I - \tilde{\lambda} L)\tilde{\mathbf{p}} = \mathbf{p}$$



Implicit fairing of irregular meshes using diffusion and curvature flow
M. Desbrun, M. Meyer, P. Schroeder, A. Barr
ACM SIGGRAPH 99

Implicit Fairing

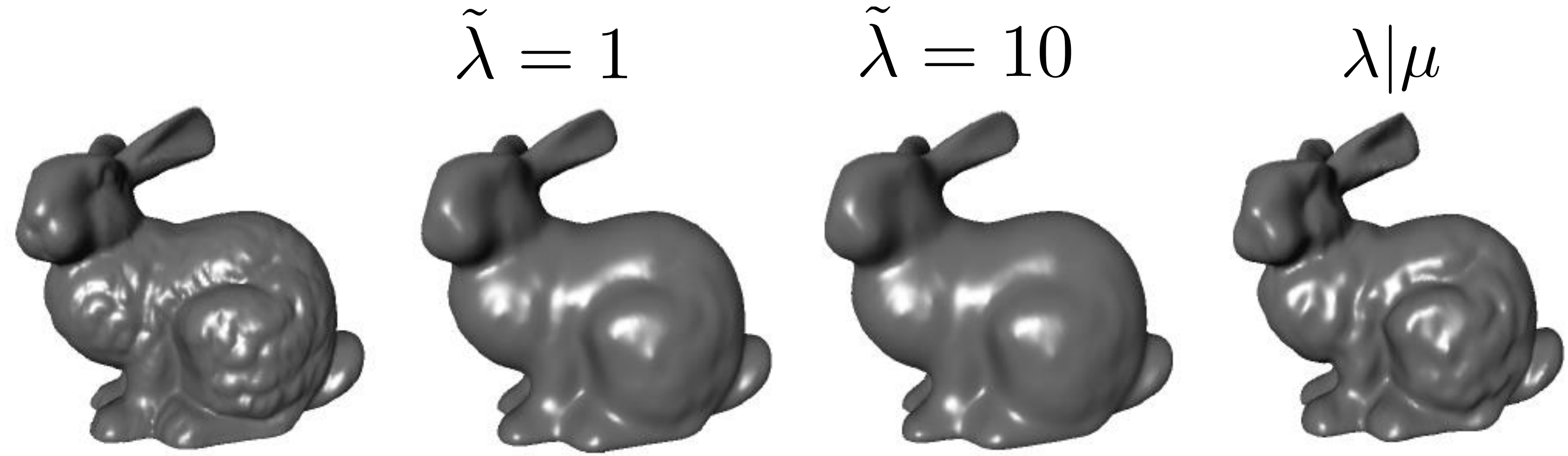
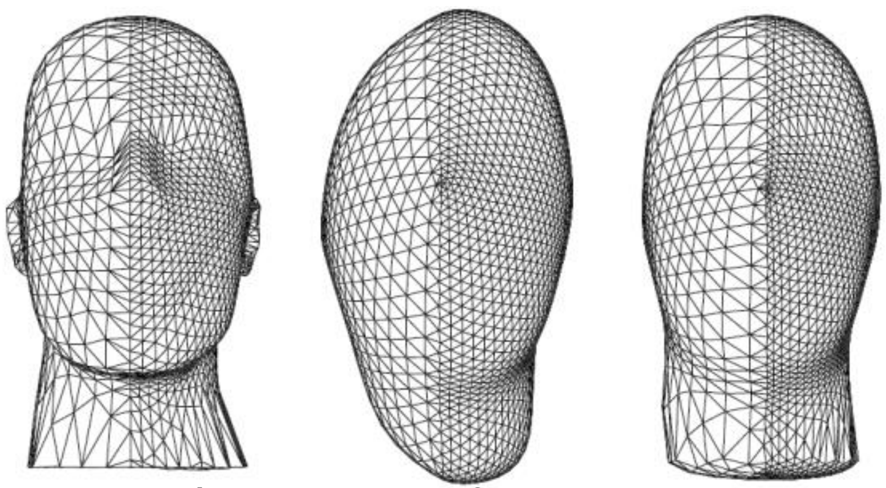


Figure 4: *Stanford bunnies: (a) The original mesh, (b) 10 explicit integrations with* $\lambda dt = 1$*, (c) 1 implicit integration with* $\lambda dt = 10$ *that takes only 7 PBCG iterations (30% faster), and (d) 20 passes of the* $\lambda|\mu$ *algorithm, with* $\lambda = 0.6307$ *and* $\mu = -0.6732$*. The implicit integration results in better smoothing than the explicit one for the same, or often less, computing time. If volume preservation is called for, our technique then requires many fewer iterations to smooth the mesh than the* $\lambda|\mu$ *algorithm.*

Implicit fairing of irregular meshes using diffusion and curvature flow
M. Desbrun, M. Meyer, P. Schroeder, A. Barr
ACM SIGGRAPH 99

Mesh Independence

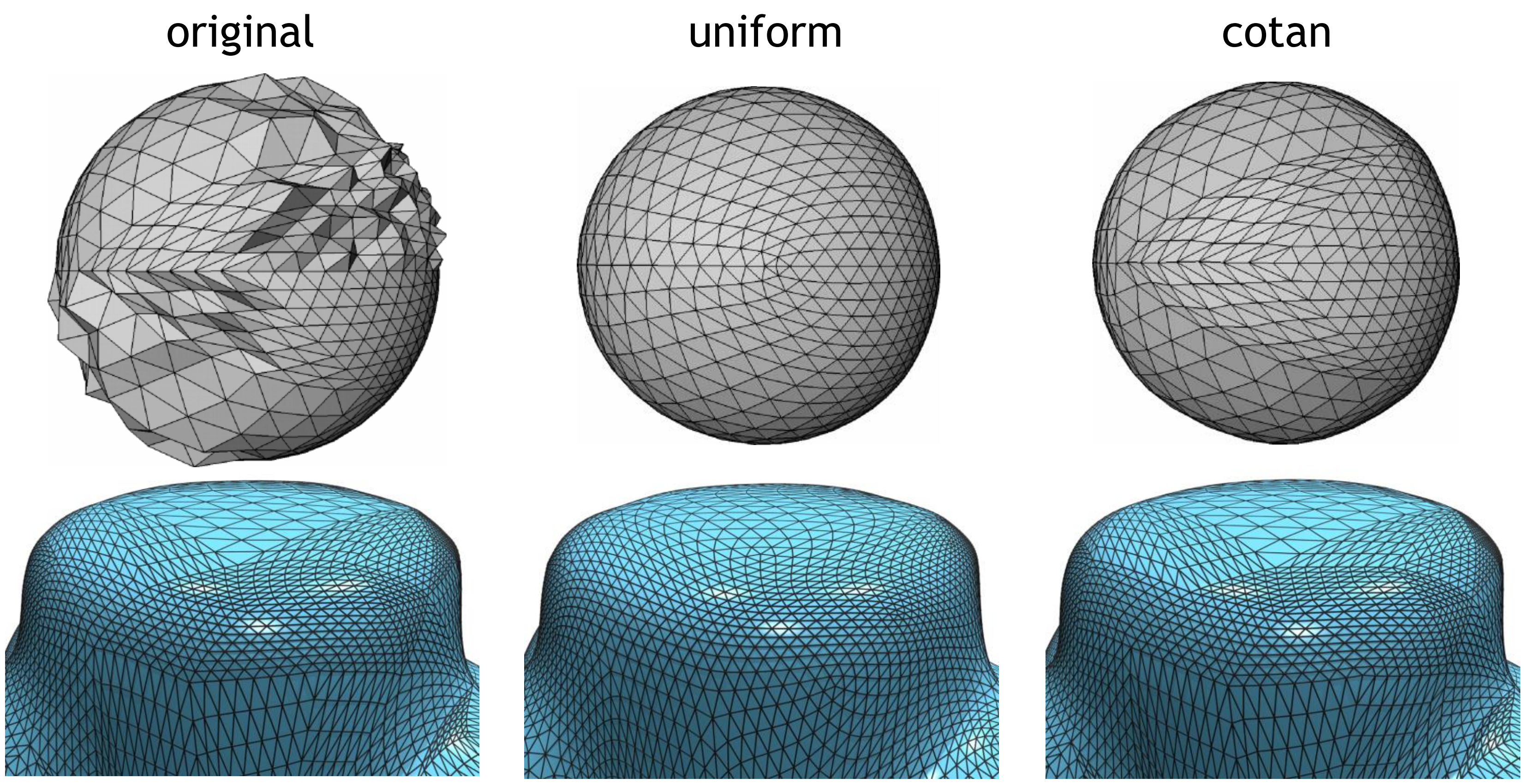
- Result of smoothing with uniform Laplacian depends on triangle density and shape
  - Why?
- Asymmetric results although underlying geometry is symmetric



original uniform cotan

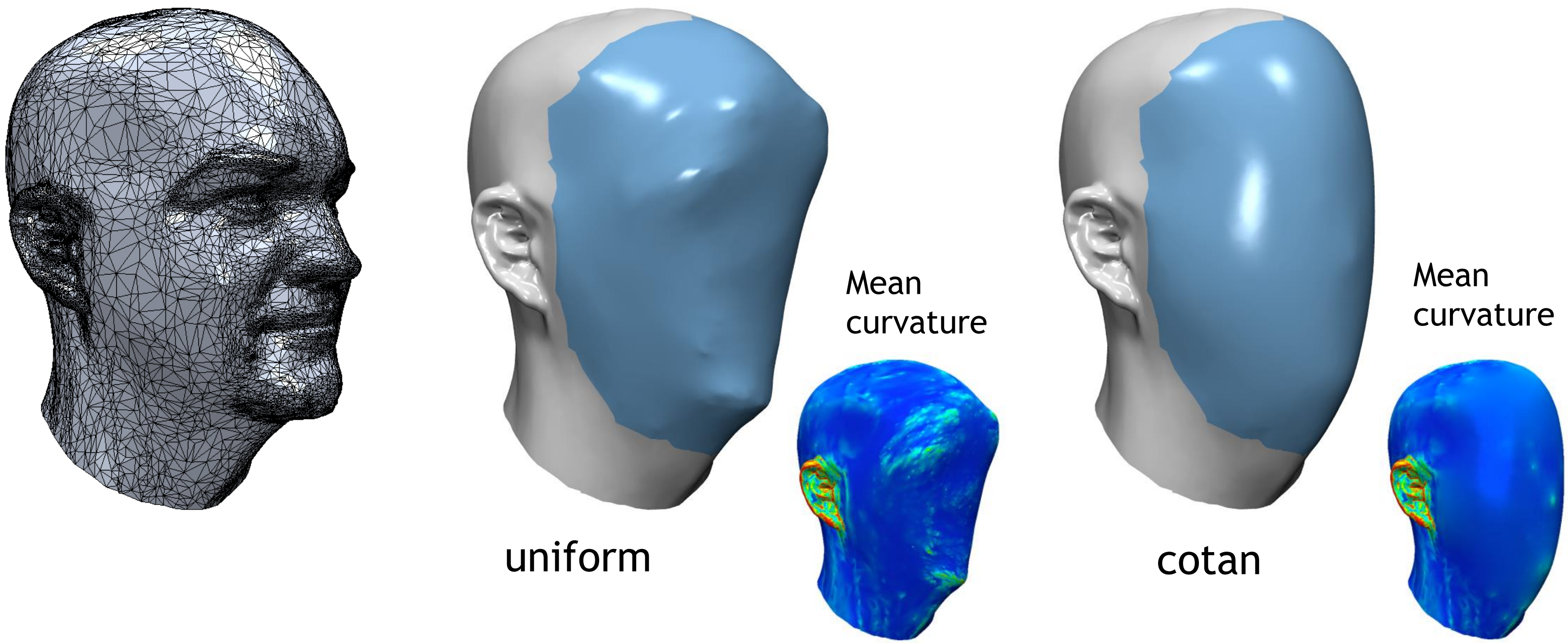
Comparison of the weights

- Explicit flow with different weights:



Implicit Fairing

- The importance of using the right weights



Thank You

Acknowledgment: some slides were adapted from Prof. Mario Botsch with his kind permission

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich