Mesh Parameterization

Acknowledgement: Olga Sorkine-Hornung

University of Victoria | Computer Science

Projections

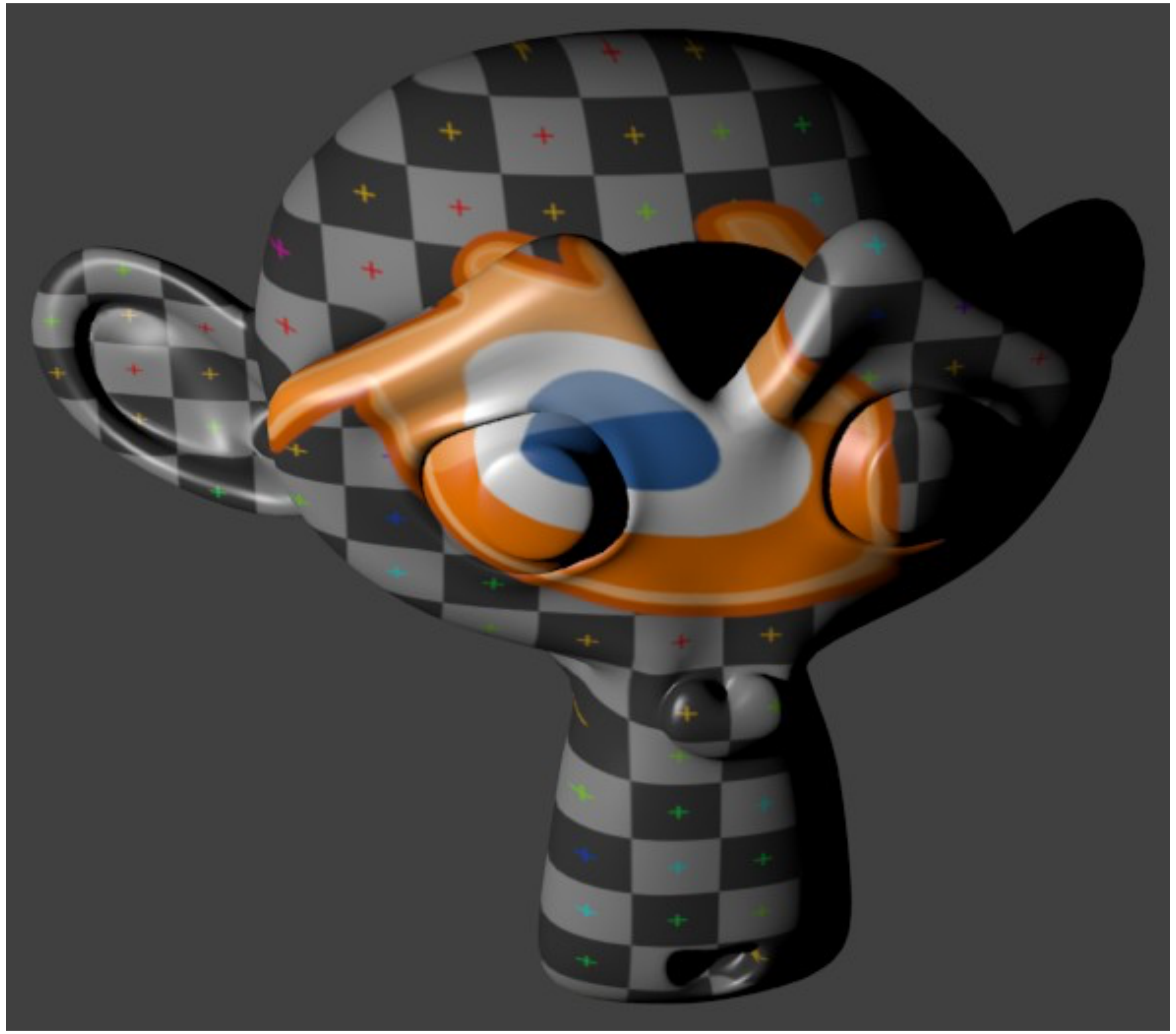
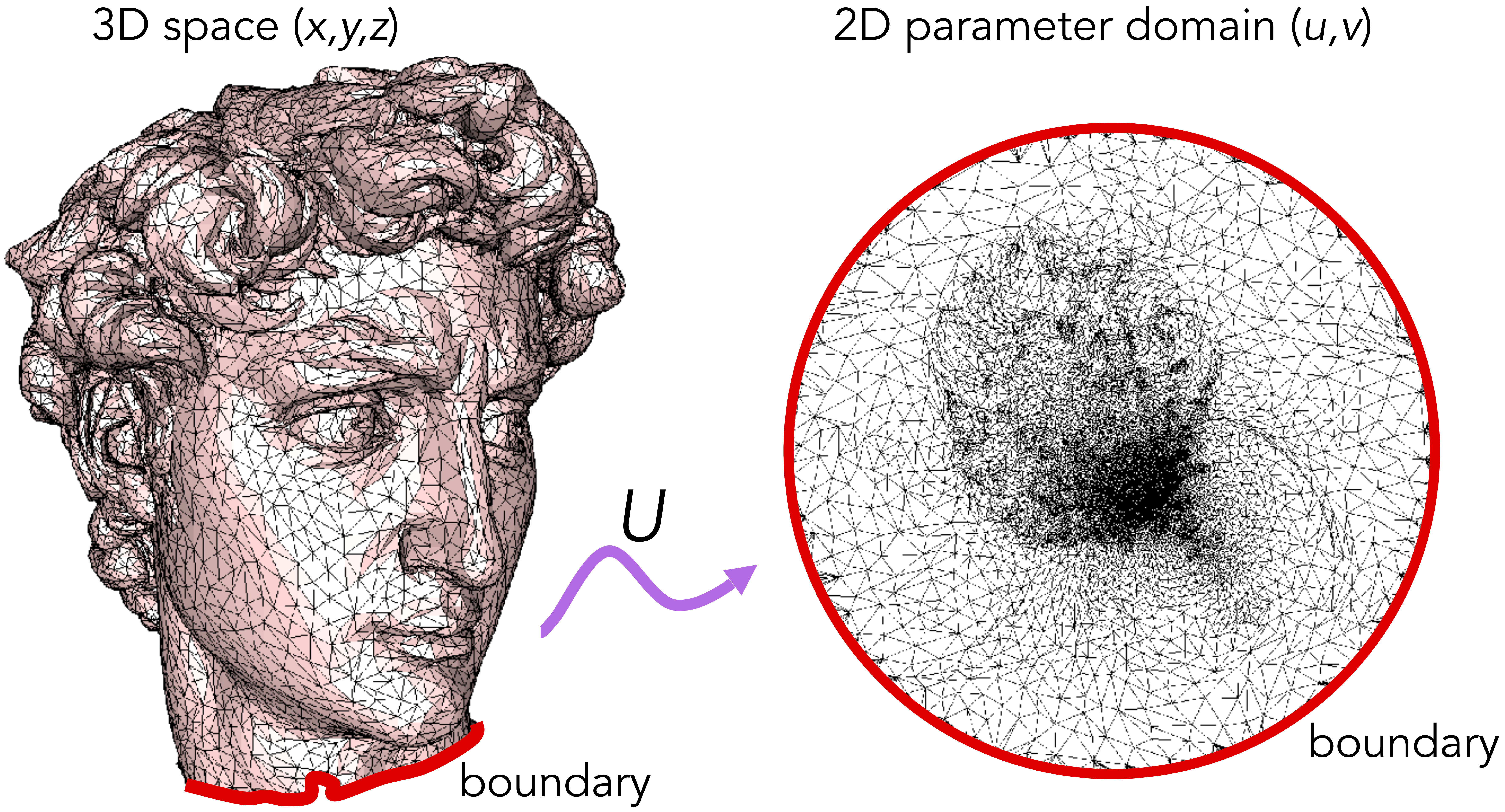


Image Courtesy of Blender

Surface Parameterization

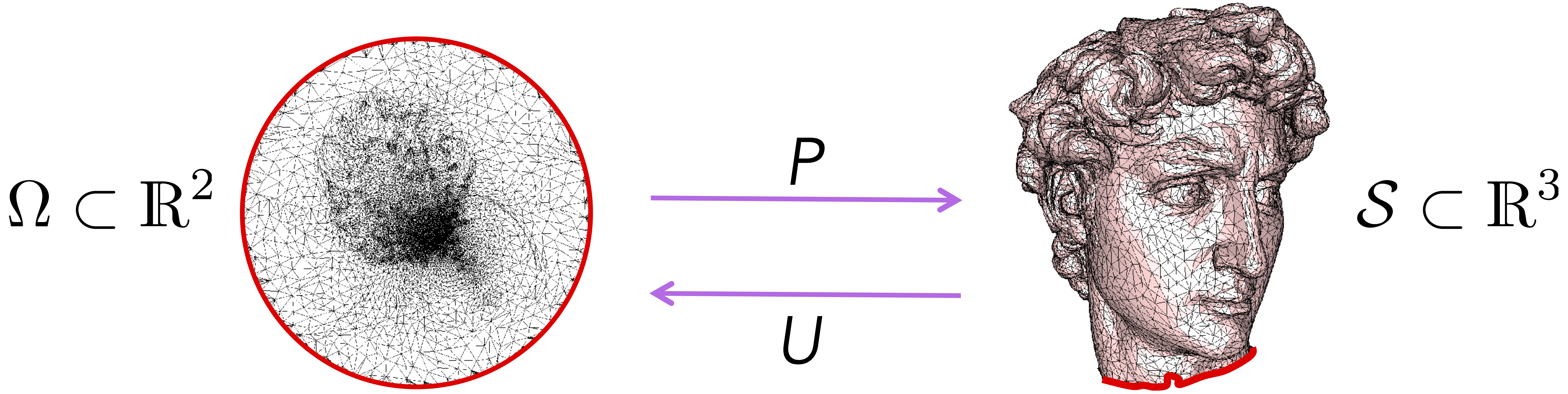


Parameterization – Definition

- Mapping *P* between a 2D domain Ω and the mesh *S* embedded in 3D (the inverse = flattening)
- Each mesh vertex has a corresponding 2D position:

$$U(\mathbf{v}_i) = (u_i, v_i)$$

- Inside each triangle, the mapping is affine (barycentric coordinates)

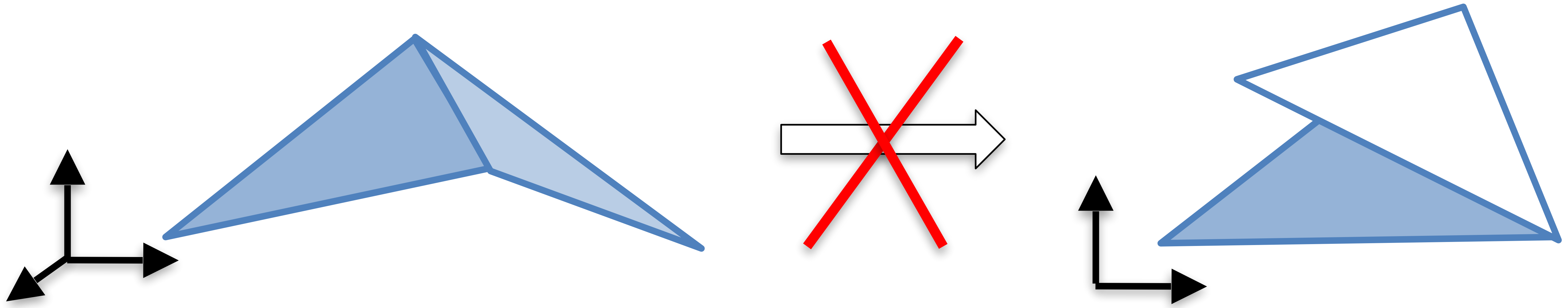


What is a good parametrization?

- It depends on the application, but usually:
  - Bijectivity
  - Number of cuts and charts
  - Geometric distortion

Bijectivity

- Locally bijective (1-1 and onto): No triangles fold over.



- Globally bijective: locally bijective + no “distant” areas overlap

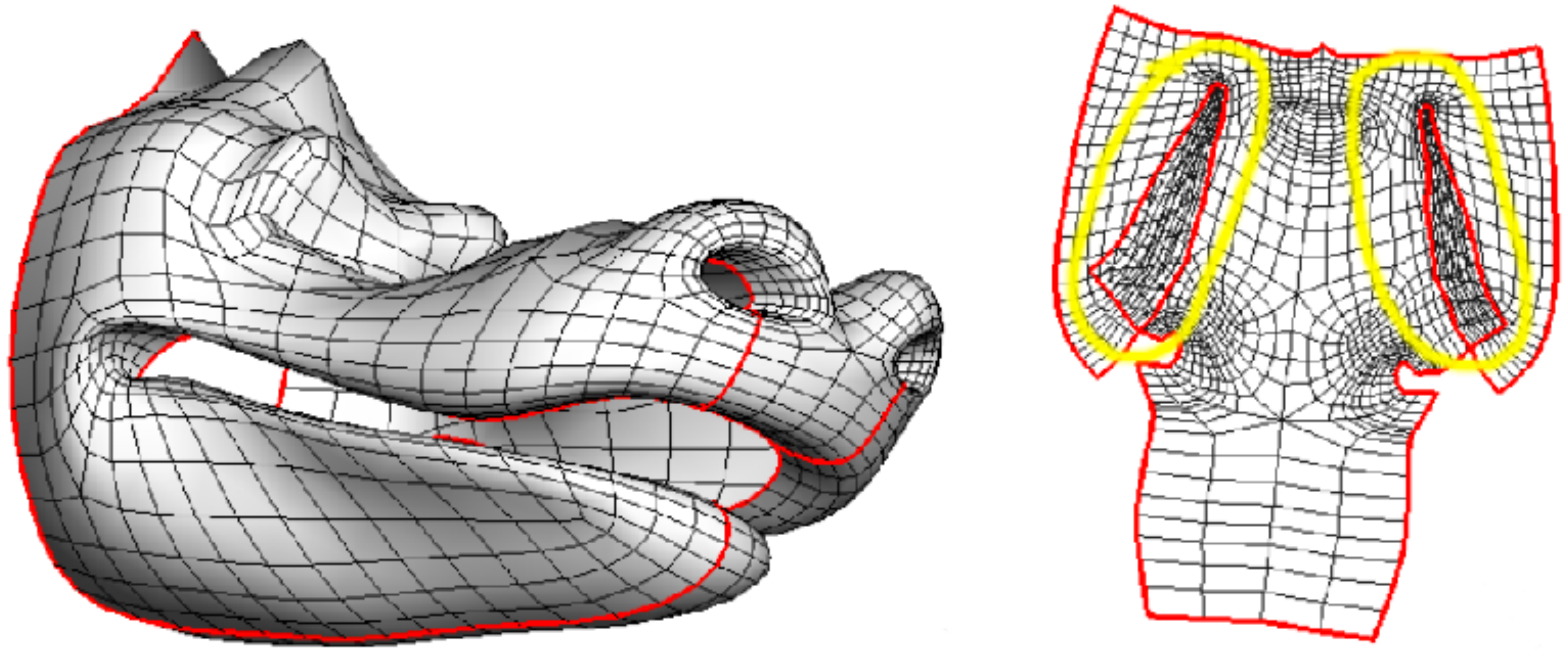
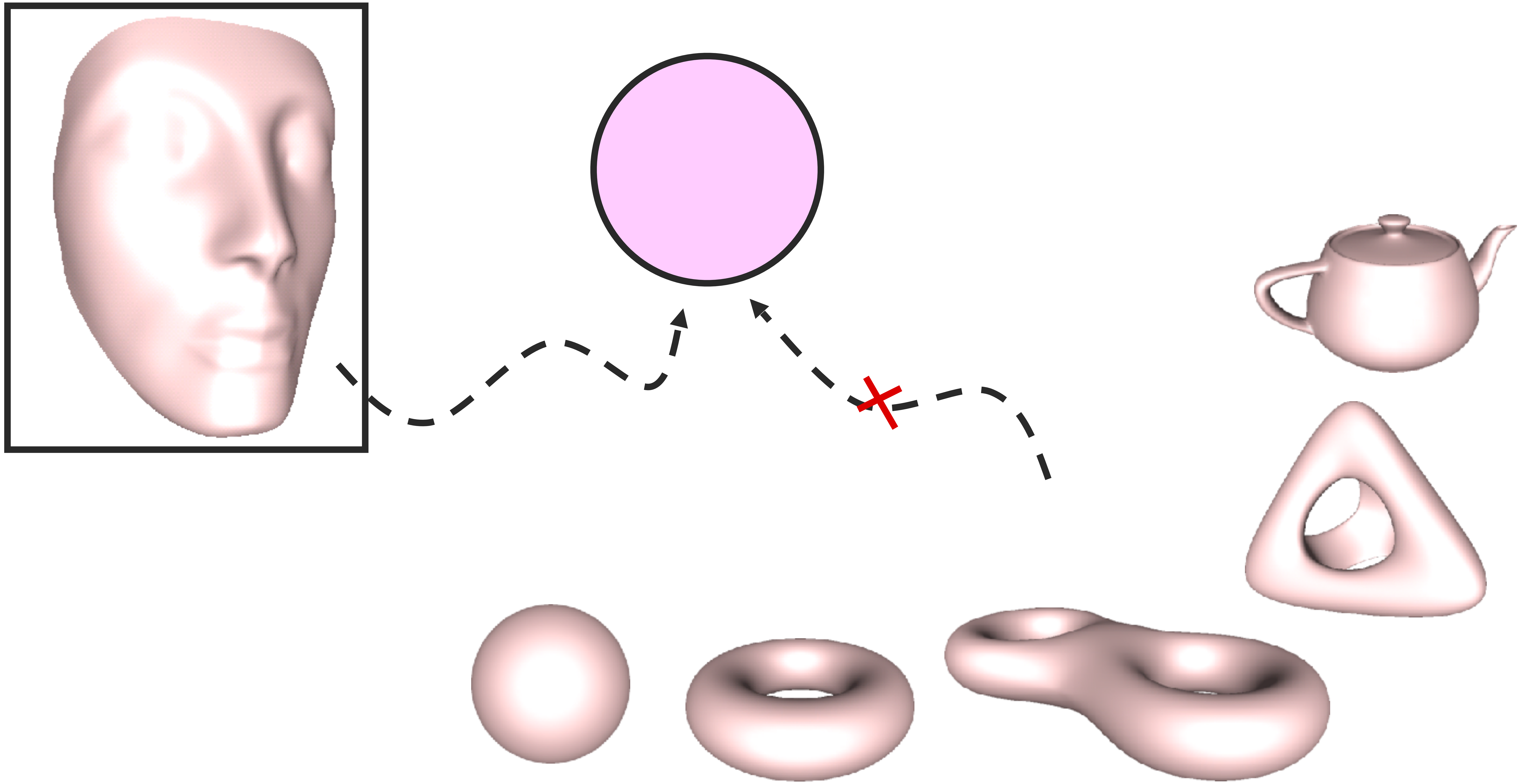
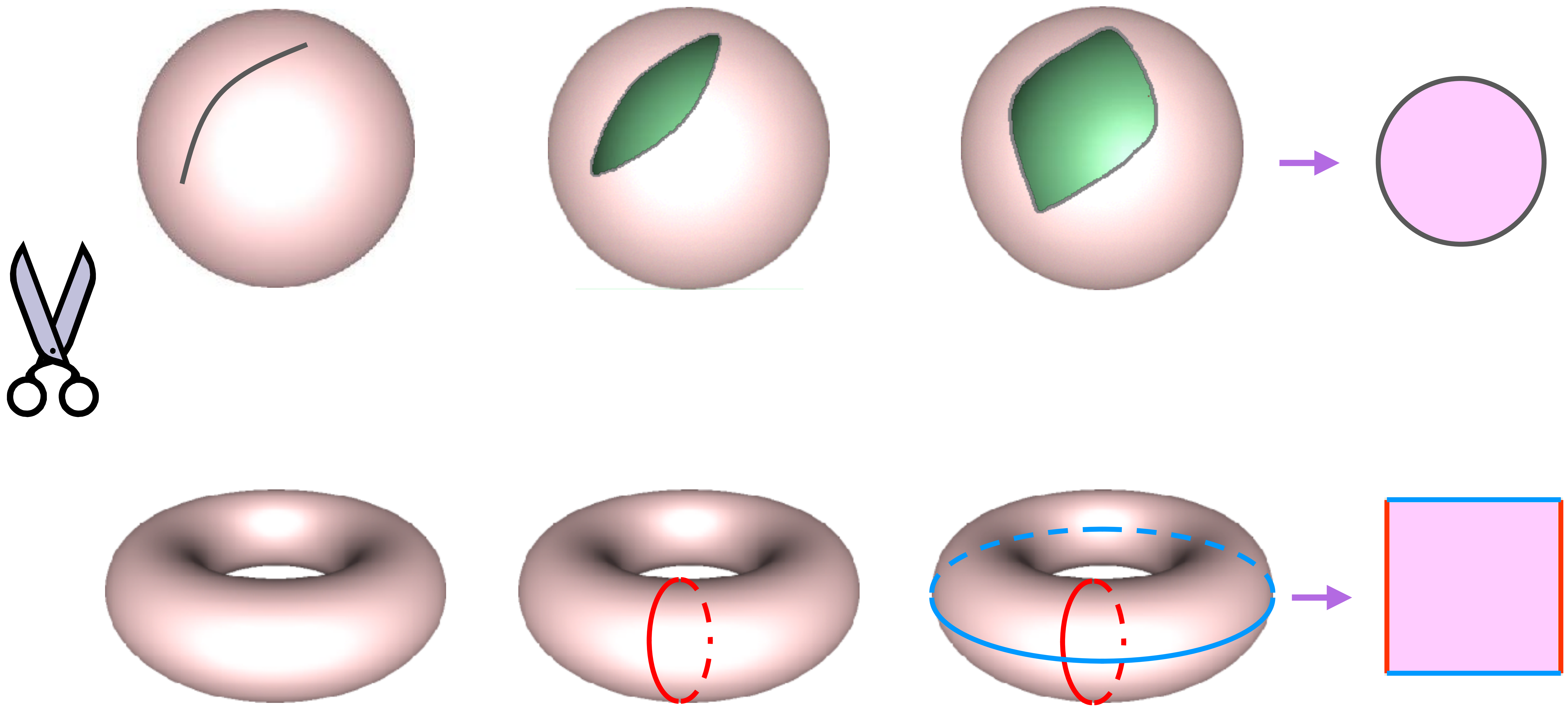


image from “Least Squares Conformal Maps”, Lévy et al., SIGGRAPH 2002

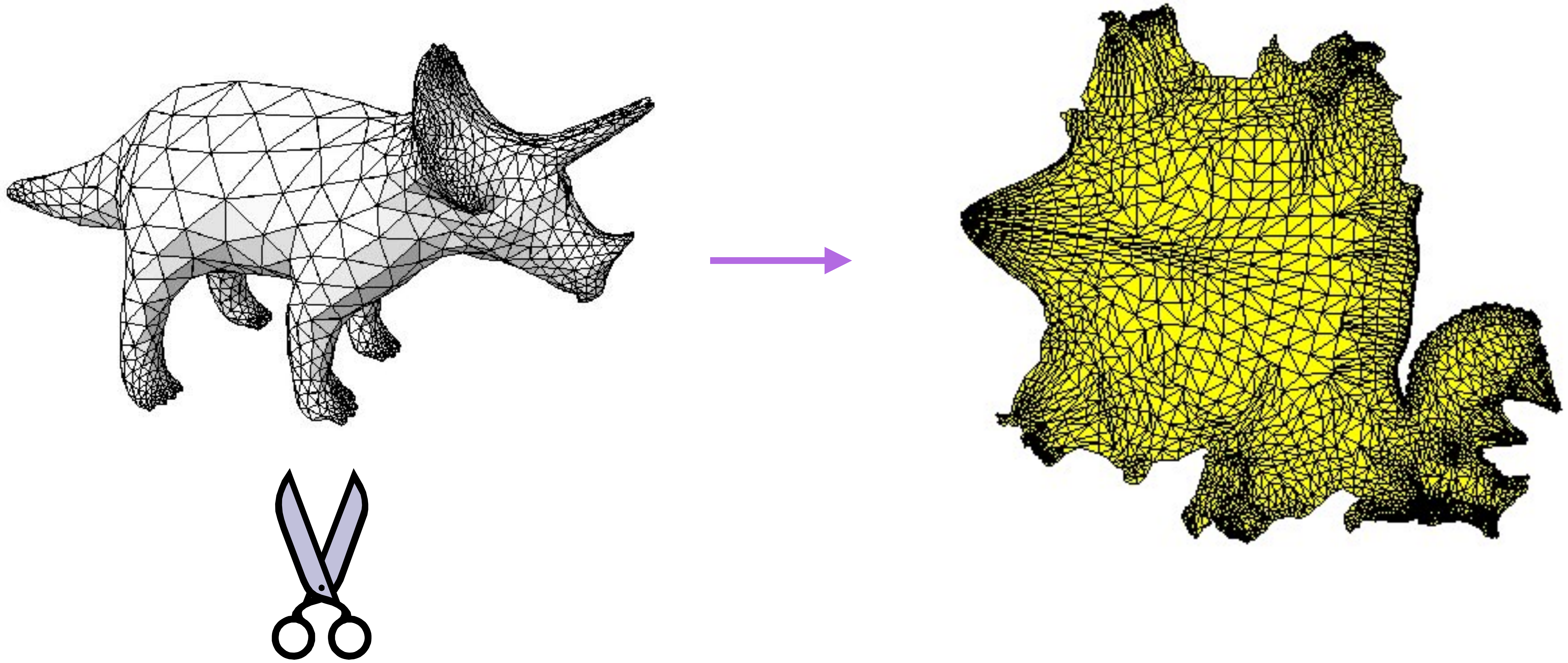
Bijectivity: Non-Disk Domains



Topological Cutting

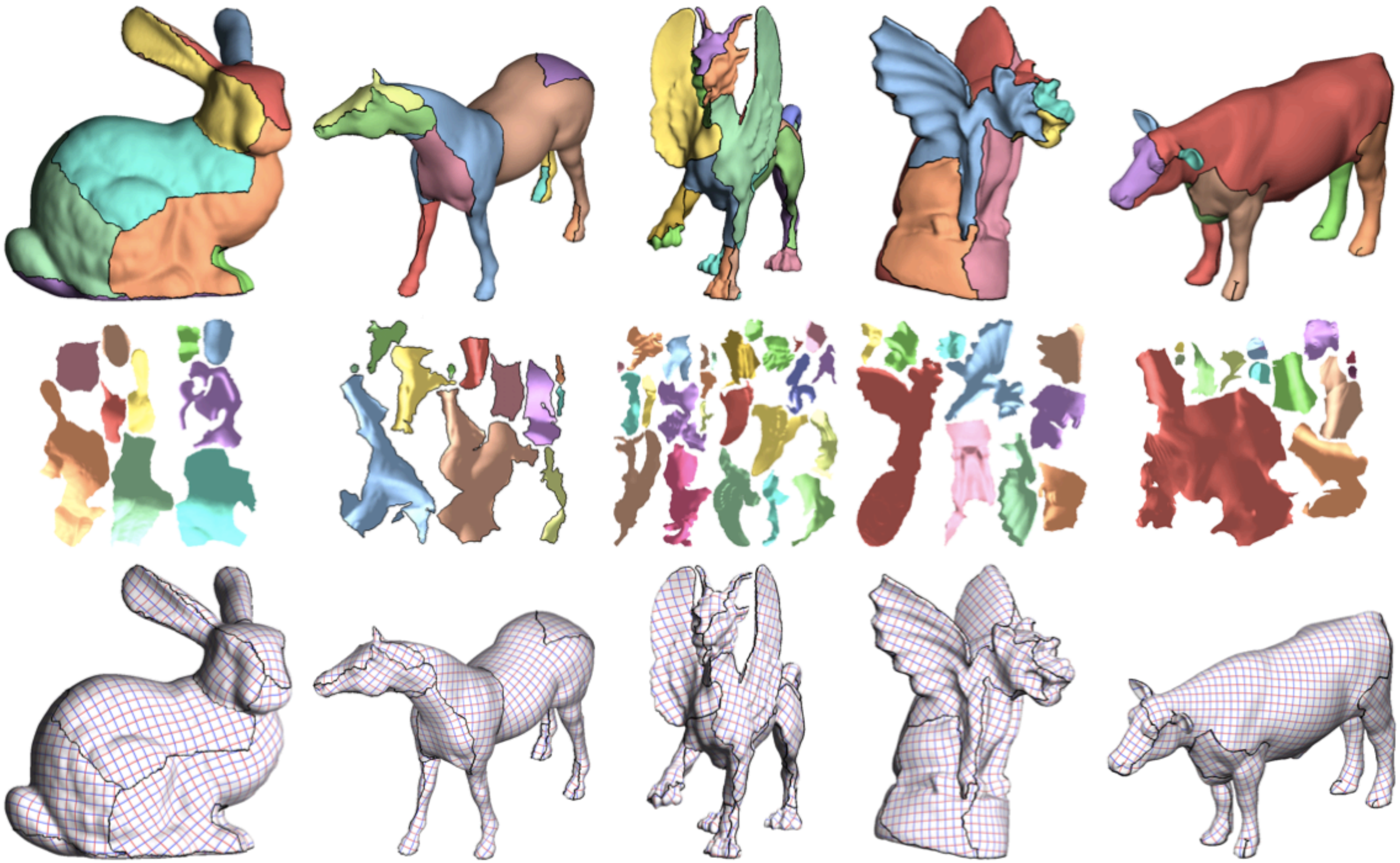


Topological Cutting



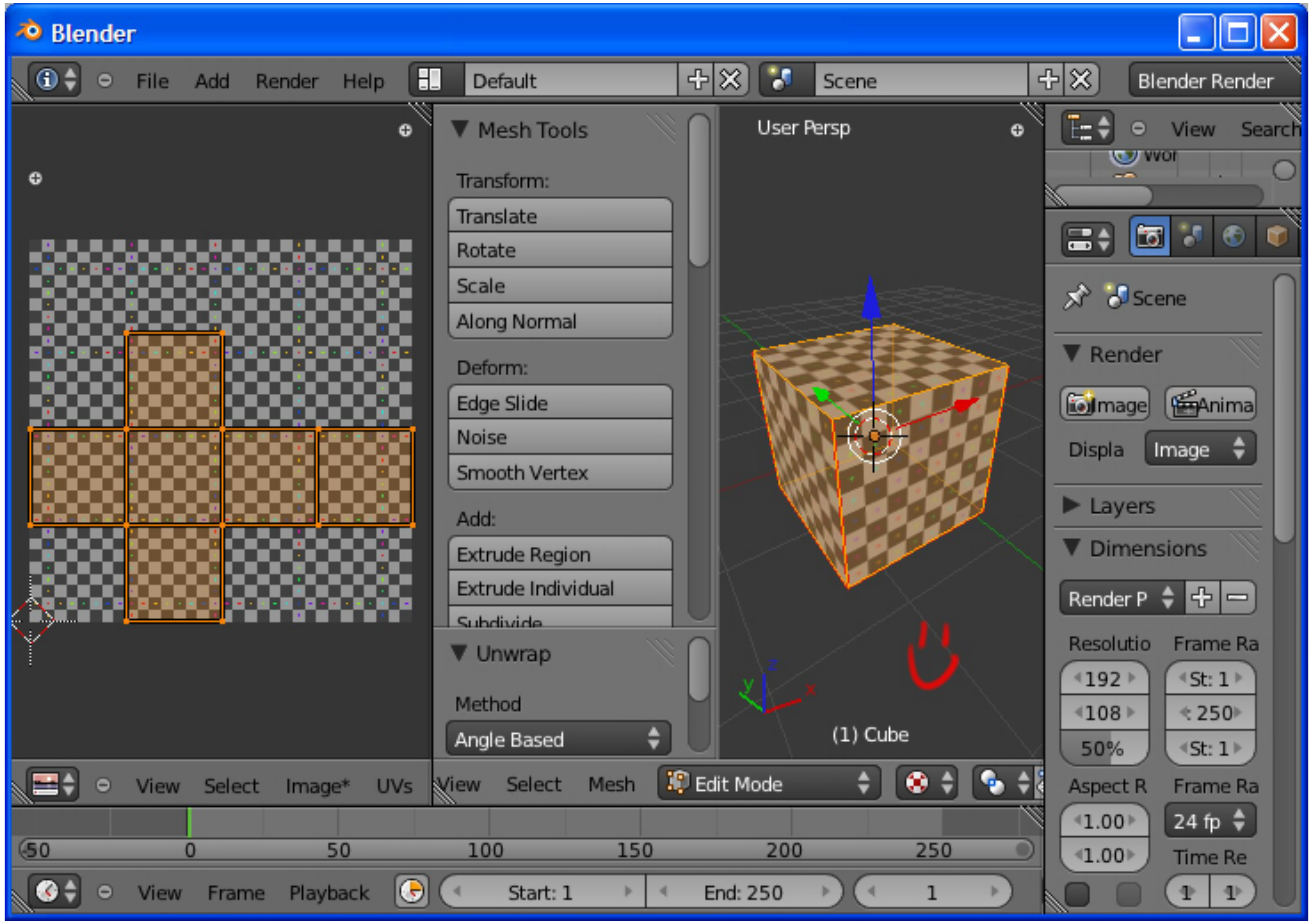
A. Sheffer, J. Hart:
Seamster: Inconspicuous Low-Distortion Texture Seam Layout, IEEE Vis 2002
http://www.cs.ubc.ca/~sheffa/papers/VIS02.pdf

Segmentation



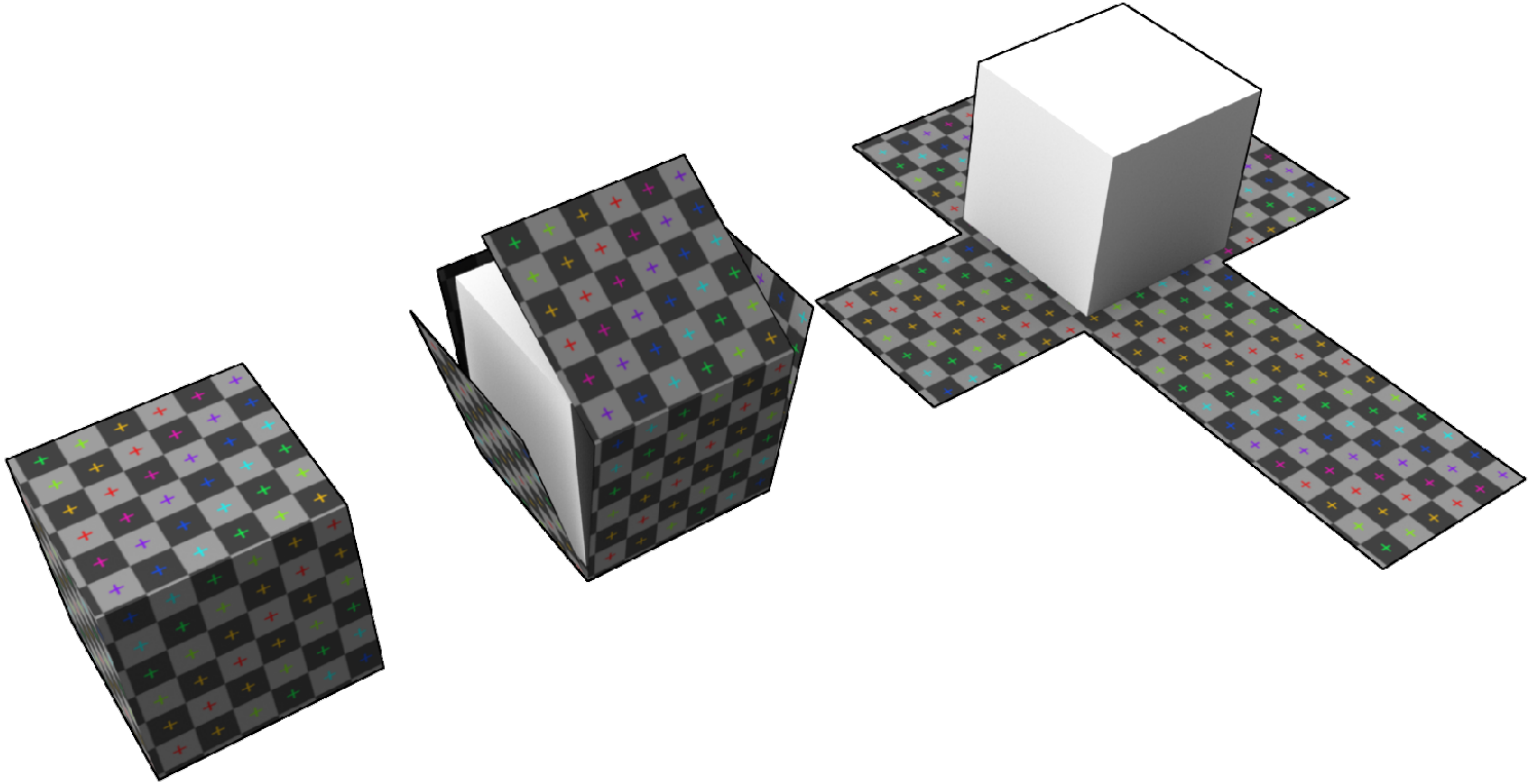
D-Charts: Quasi-Developable Mesh Segmentation,
D. Julius,V. Kraevoy, A. Sheffer, EUROGRAPHICS 2005

Segmentation

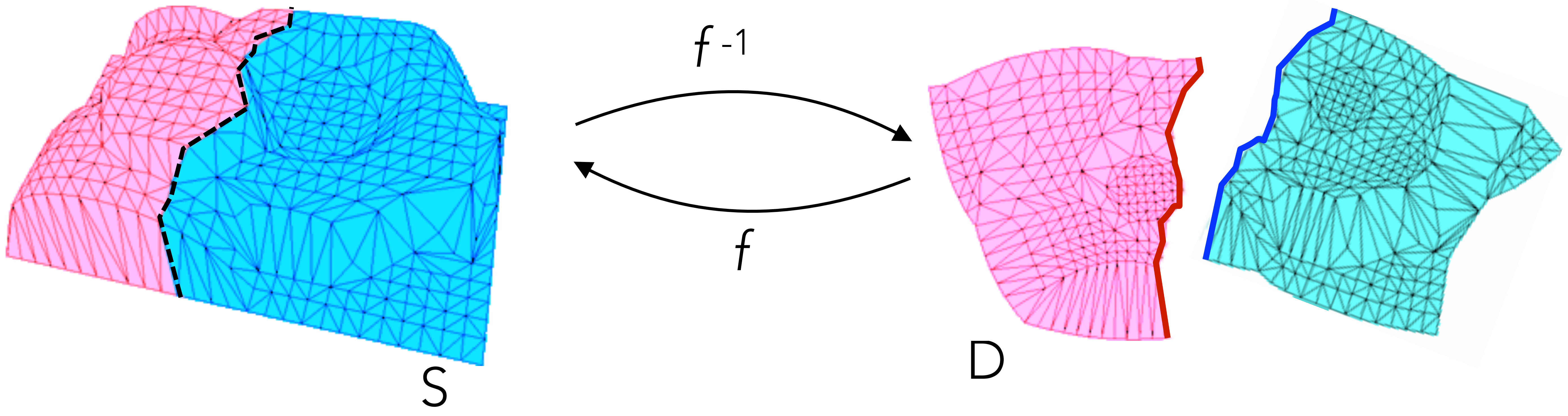


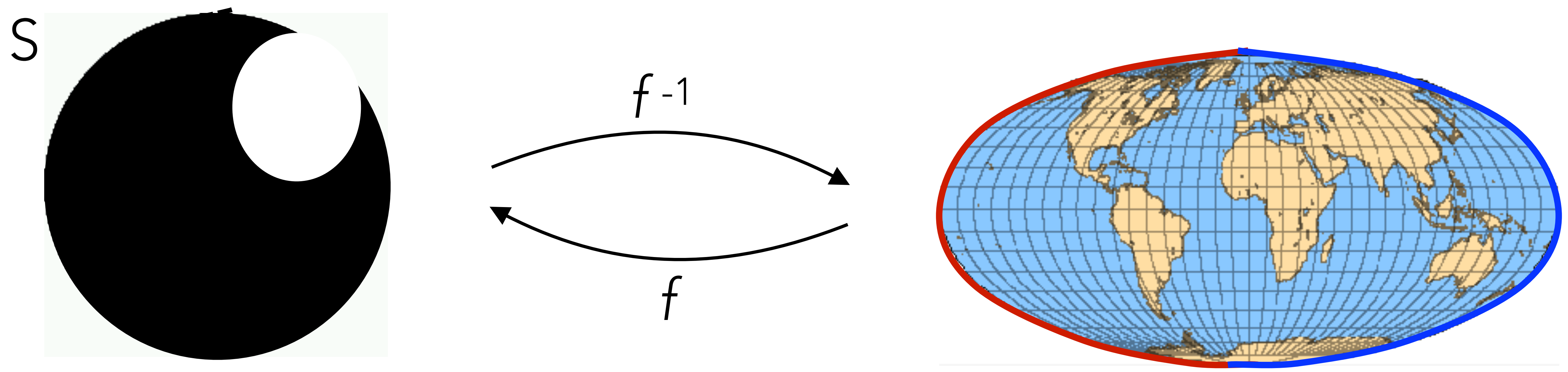
http://sophiehoulden.com/tutorials/blender/unwrapTut.html

Segmentation



Good = “fewer cuts”?



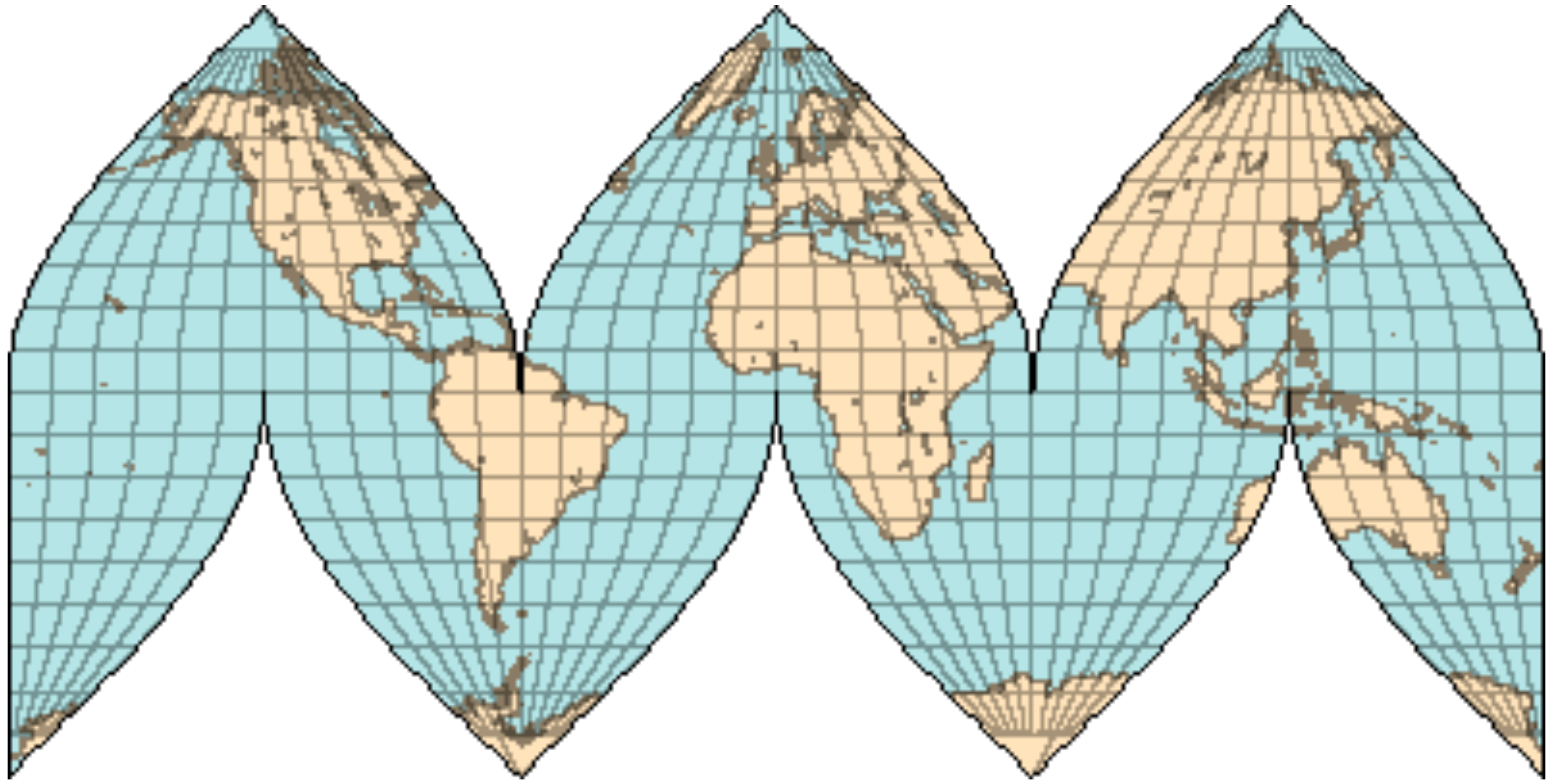


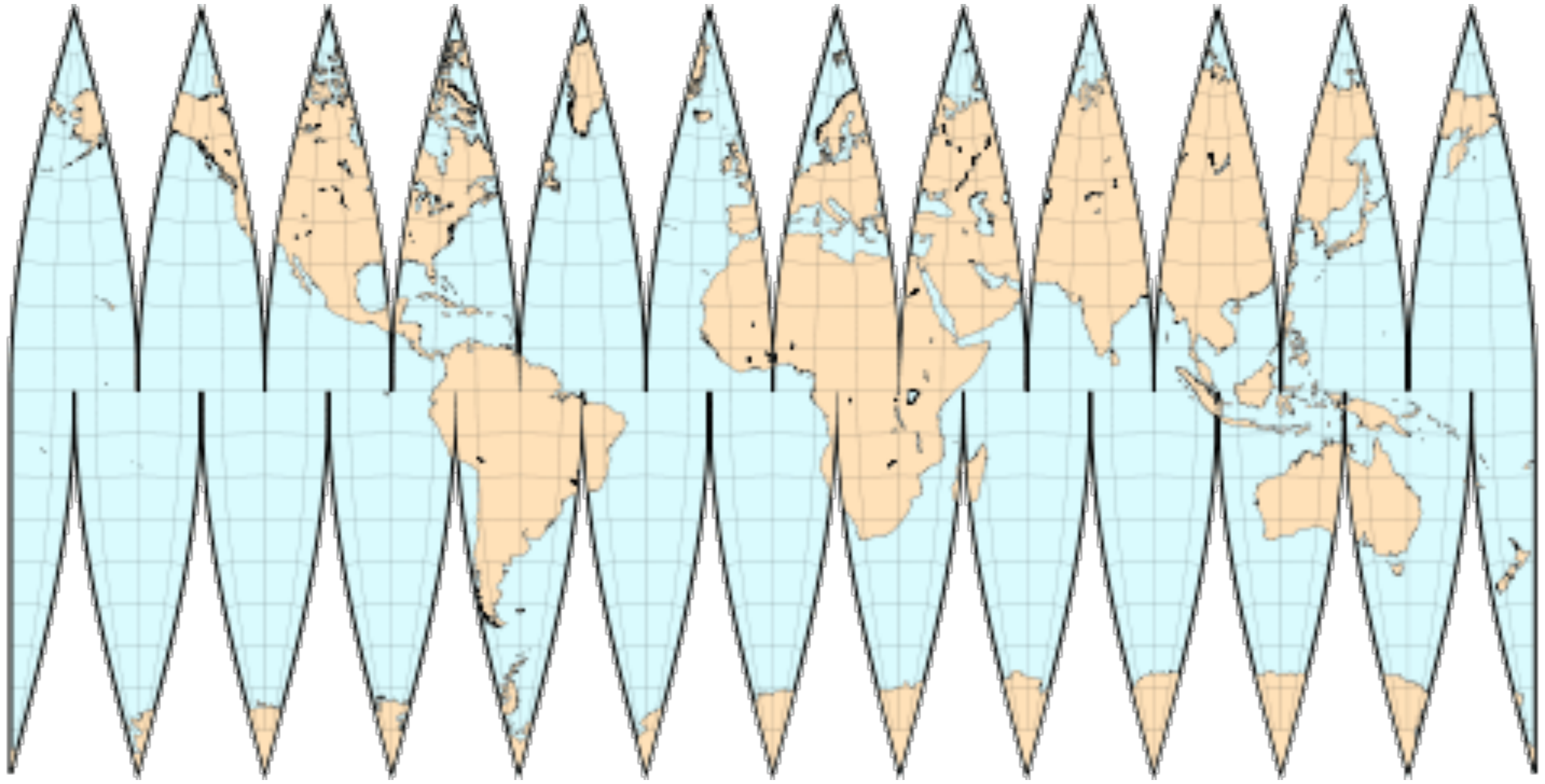
sphere in 3D

2D surface disk

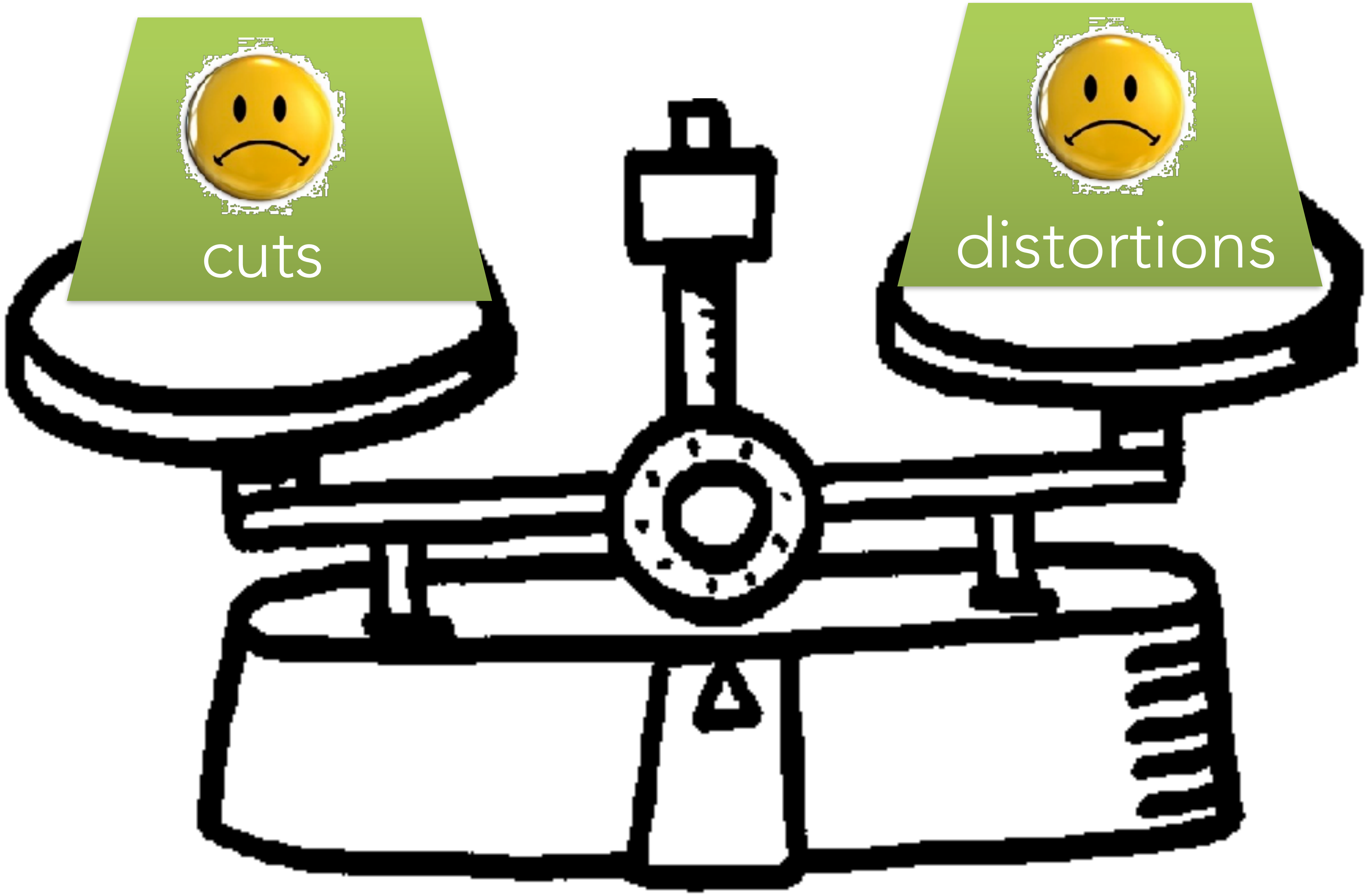
Good = "fewer cuts"?

but… more cuts => less distortion





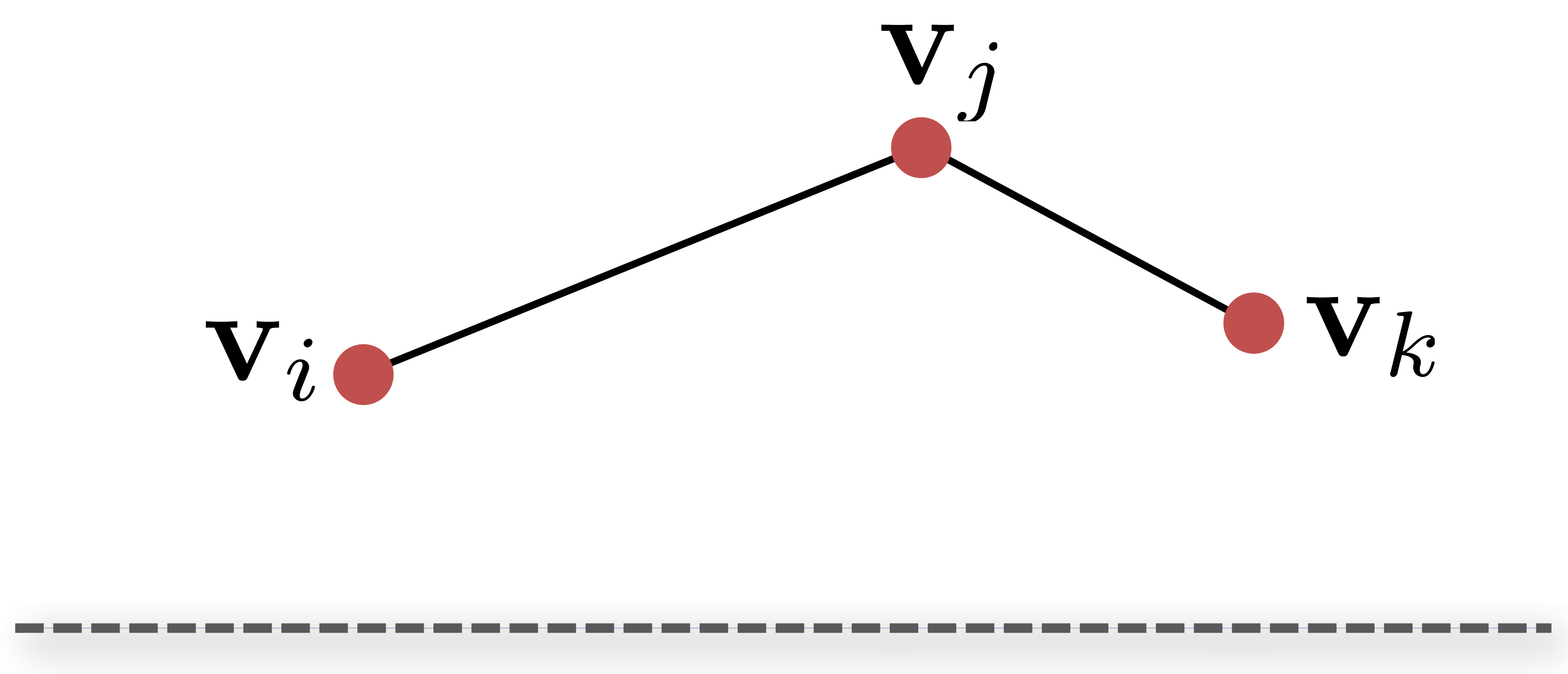
A difficult balance



Harmonic Mapping

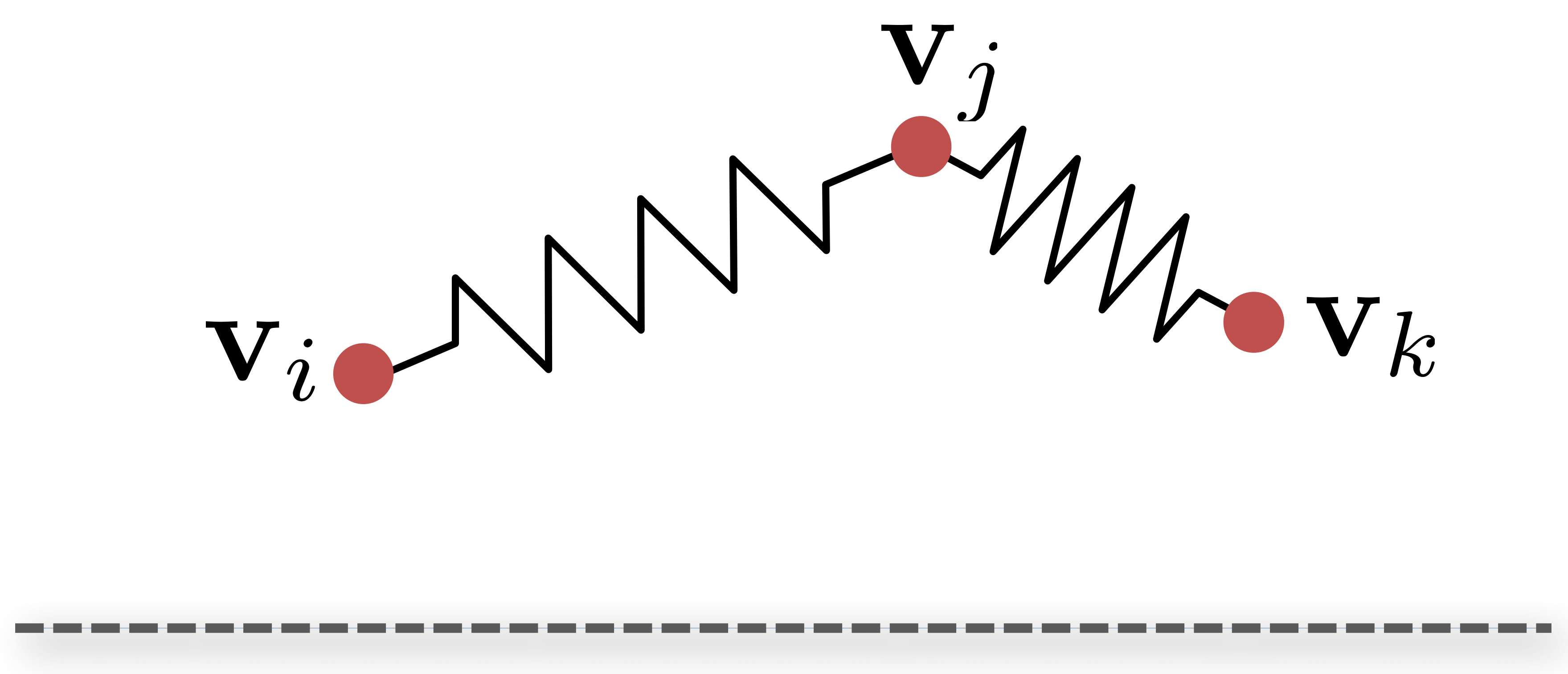
Harmonic Mapping

- Inner mesh edges as springs
- Find minimum-energy state where all vertices lie in the 2D plane



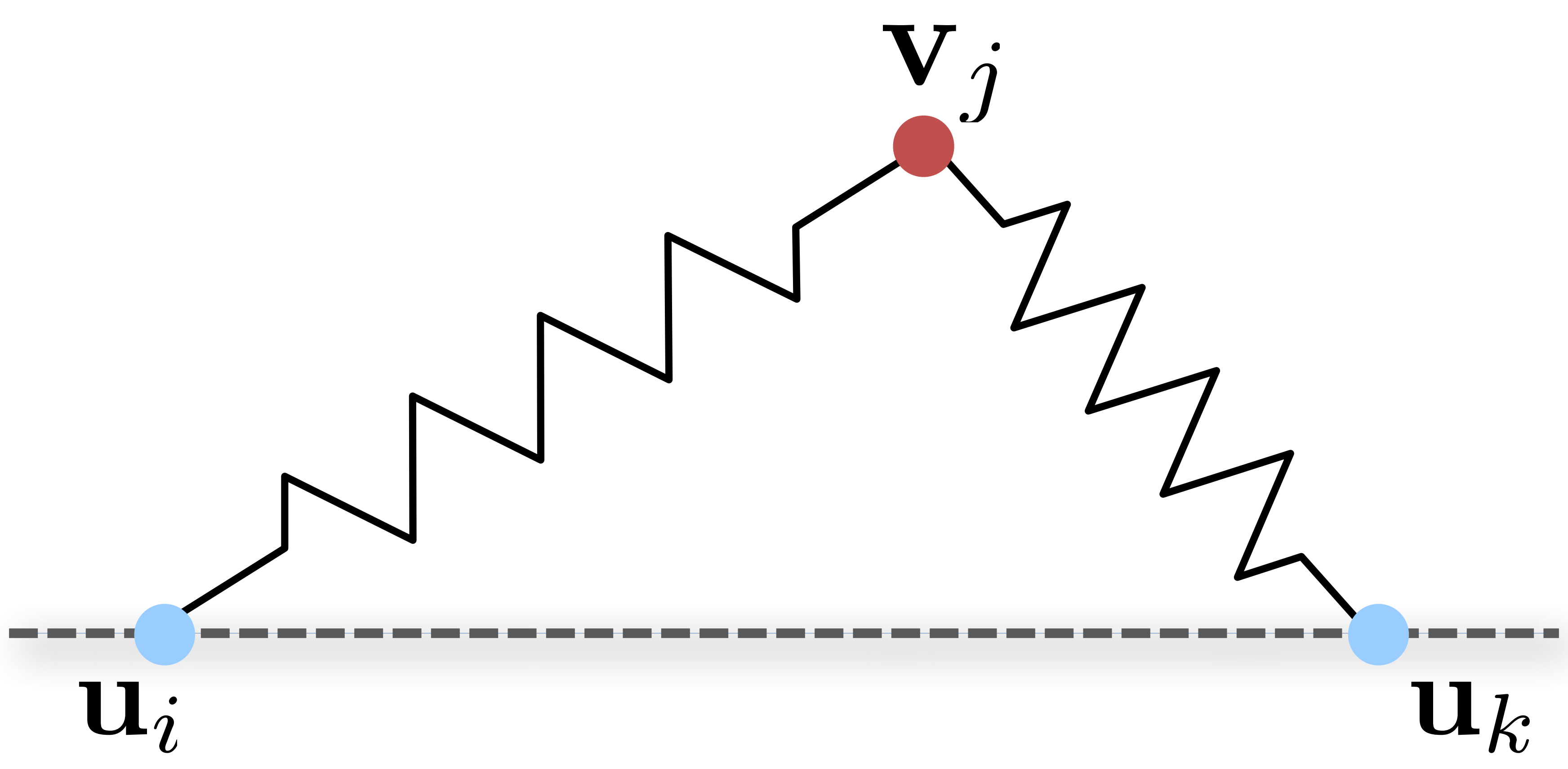
Harmonic Mapping

- Inner mesh edges as springs
- Find minimum-energy state where all vertices lie in the 2D plane



Harmonic Mapping

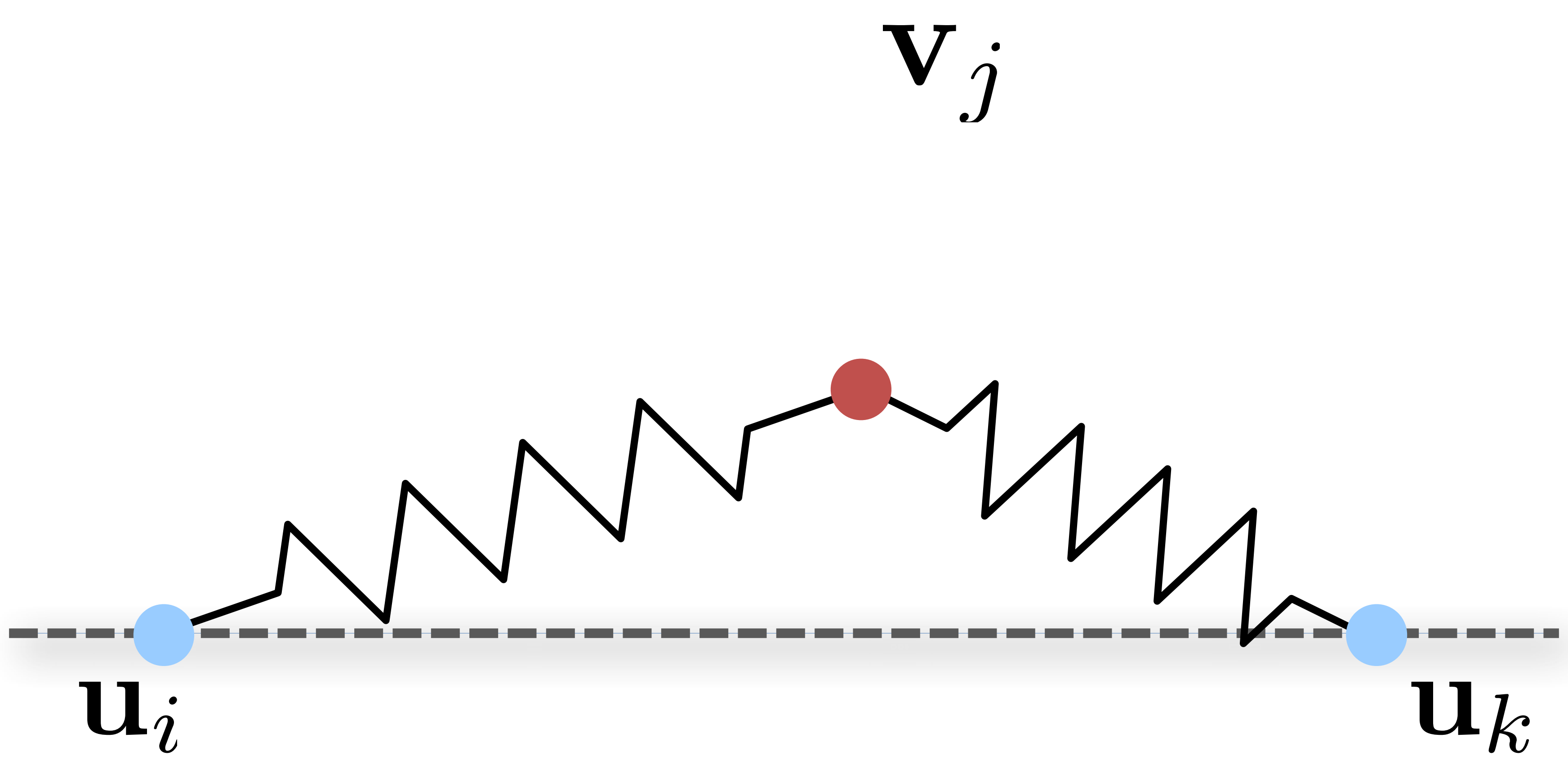
- Inner mesh edges as springs
- Find minimum-energy state where all vertices lie in the 2D plane



University of Victoria | Computer Science

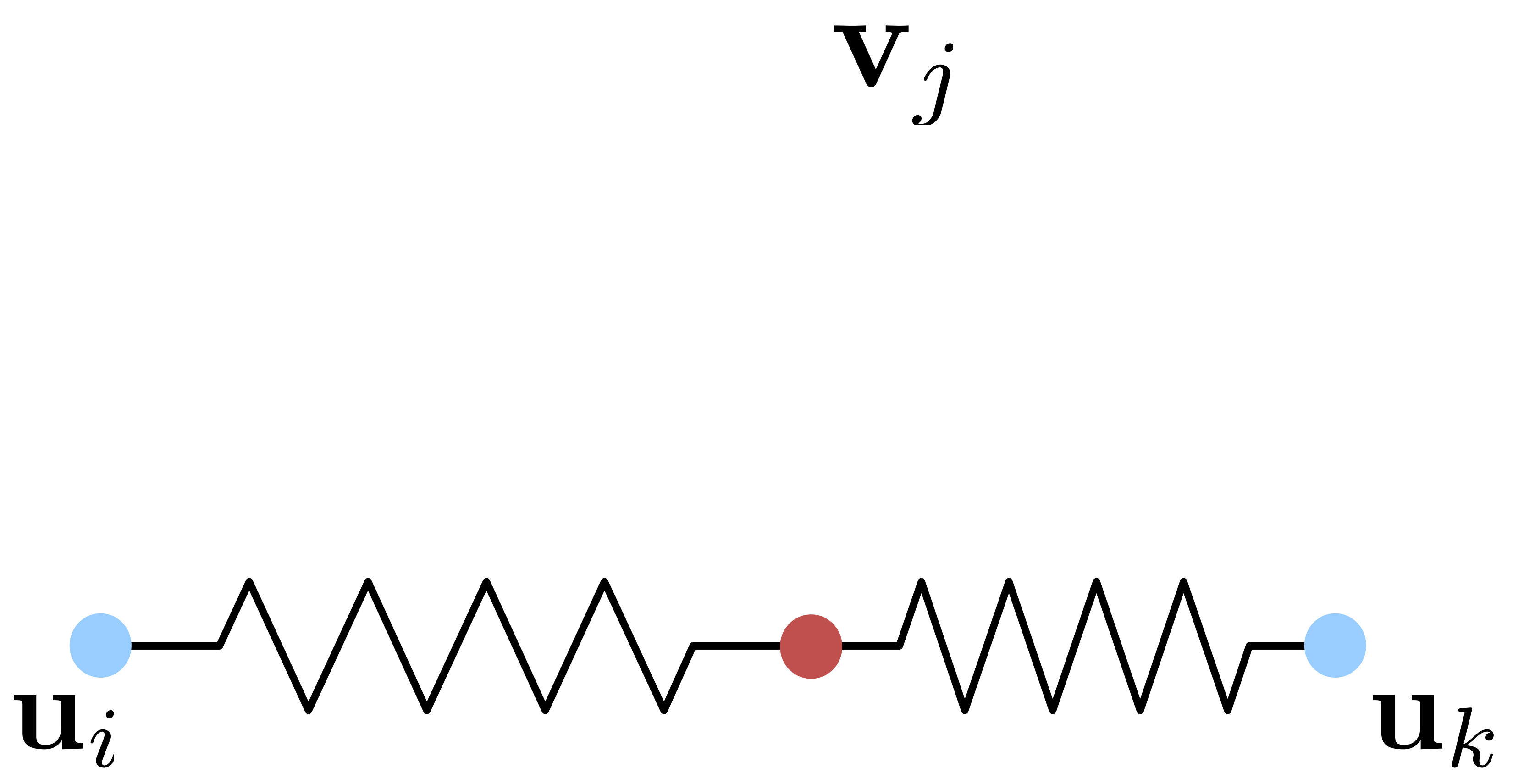
Harmonic Mapping

- Inner mesh edges as springs
- Find minimum-energy state where all vertices lie in the 2D plane



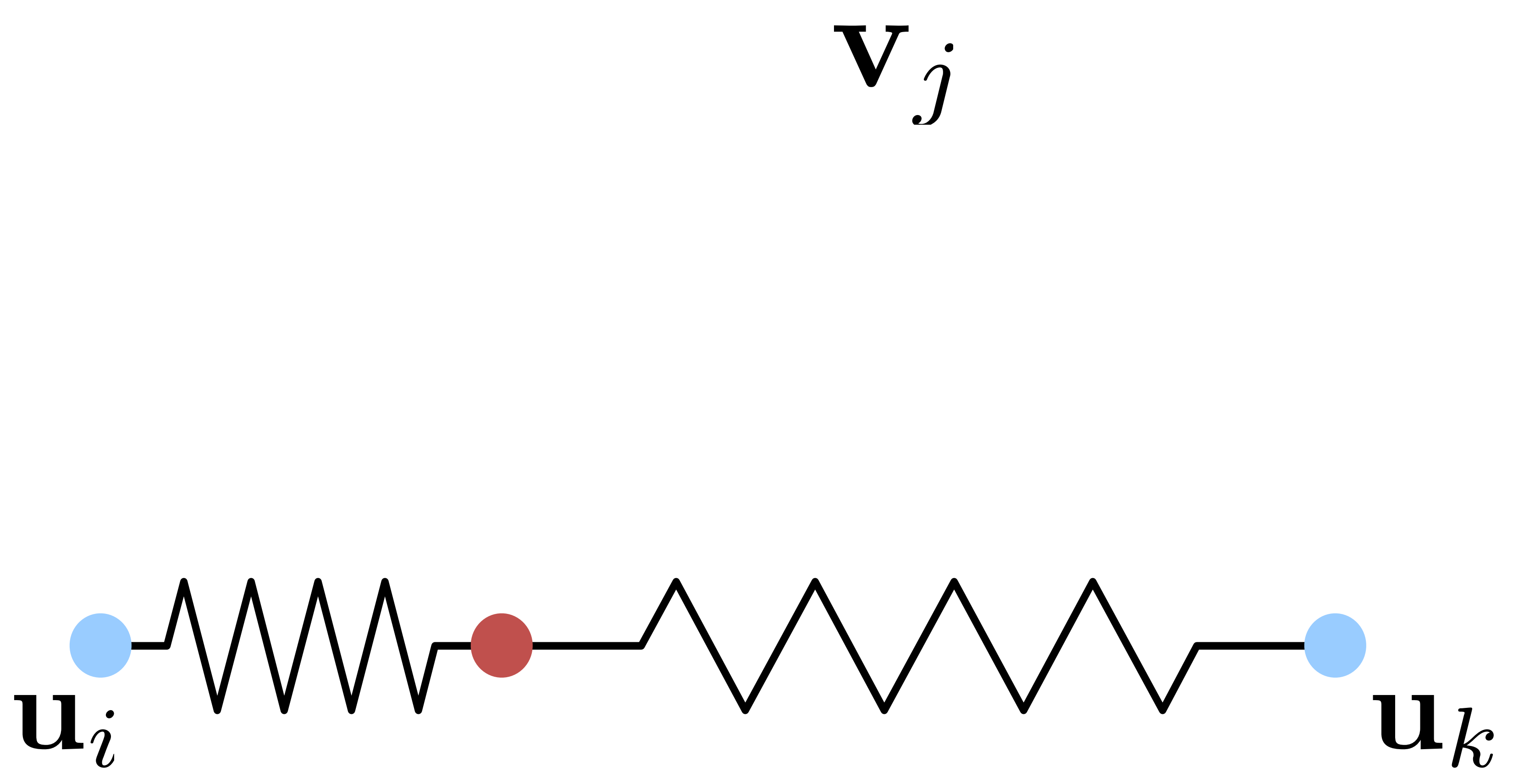
Harmonic Mapping

- Inner mesh edges as springs
- Find minimum-energy state where all vertices lie in the 2D plane



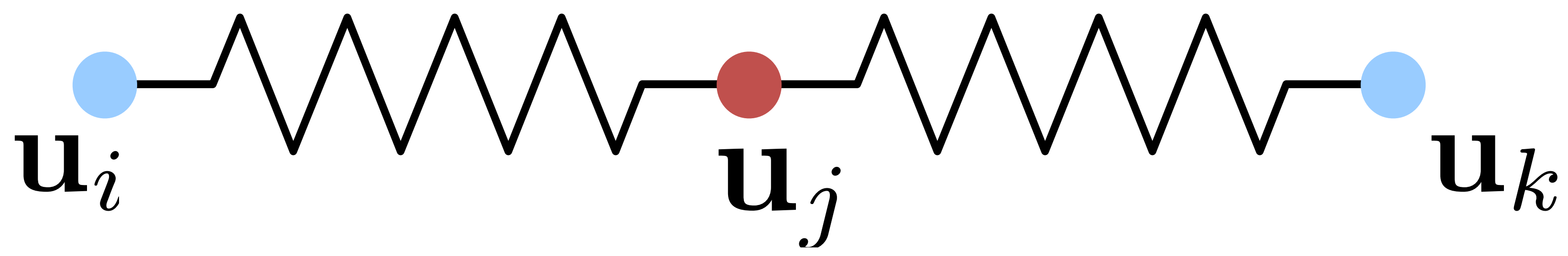
Harmonic Mapping

- Inner mesh edges as springs
- Find minimum-energy state where all vertices lie in the 2D plane



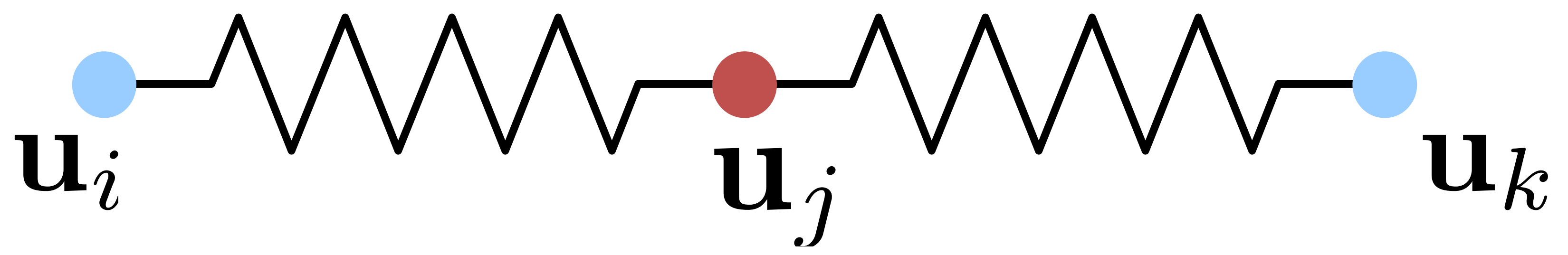
Harmonic Mapping

- Inner mesh edges as springs
- Find minimum-energy state where all vertices lie in the 2D plane



Harmonic Mapping

- Inner mesh edges as springs
- Find minimum-energy state where all vertices lie in the 2D plane
- Spring energy:

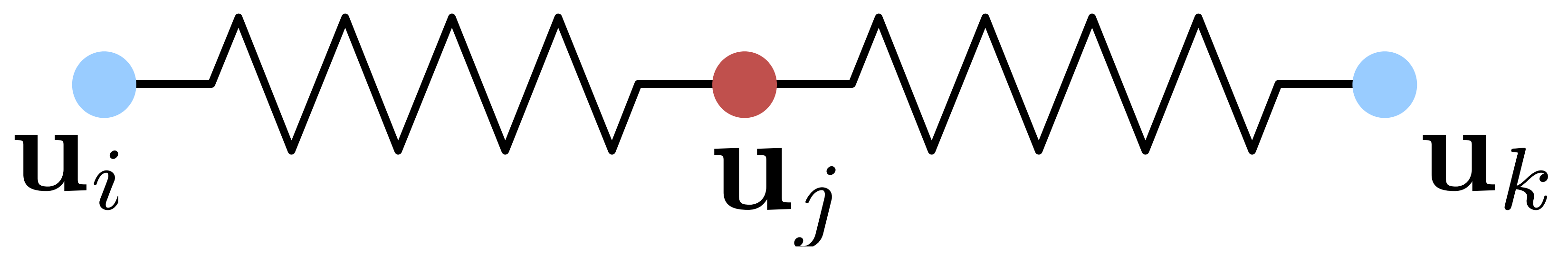


$$\frac{1}{2} k_{i,j} \left\| \mathbf{u}_i - \mathbf{u}_j \right\|^2$$

$$\mathbf{u}_i, \mathbf{u}_j \in \mathbb{R}^2$$

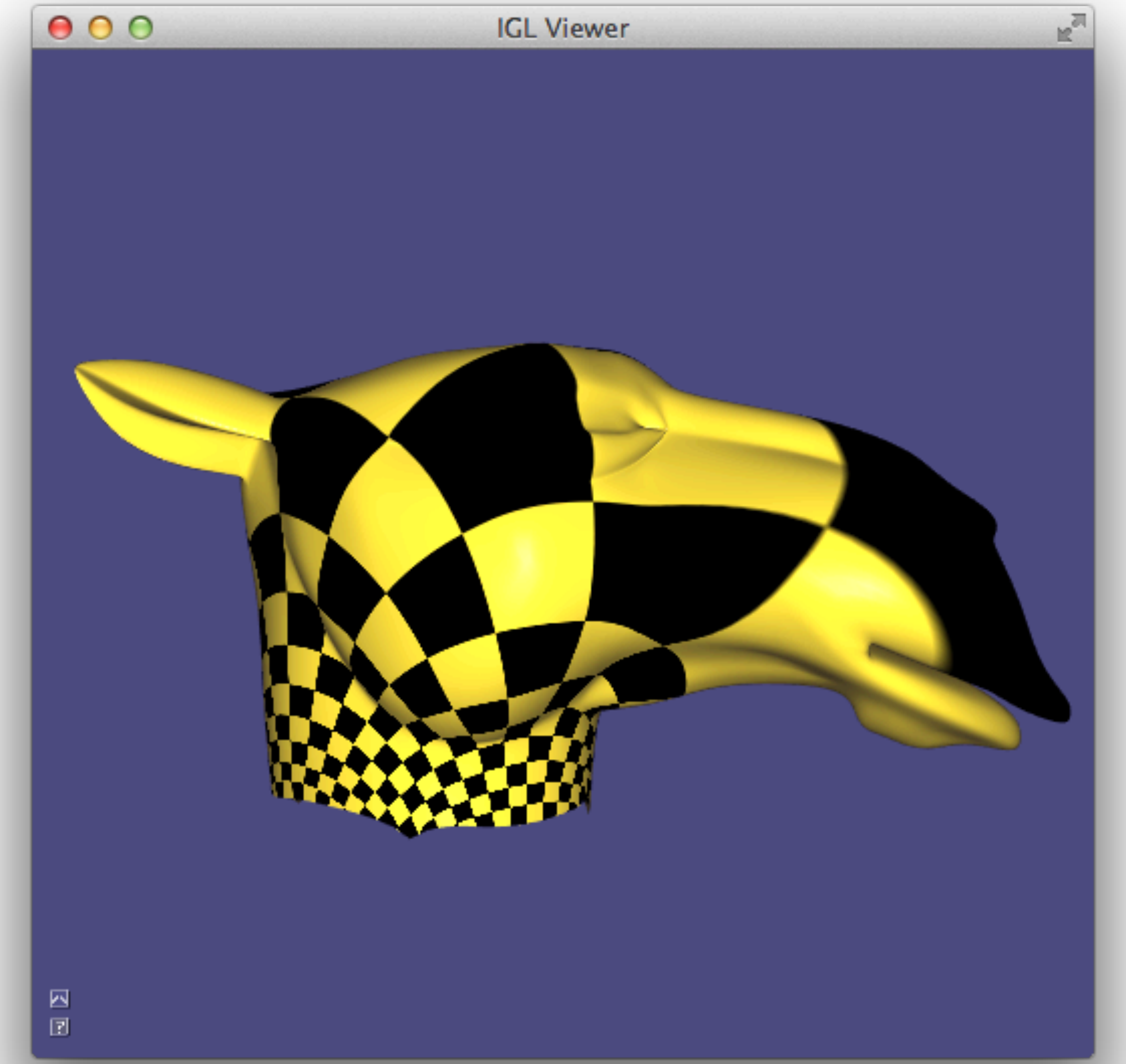
Harmonic Mapping

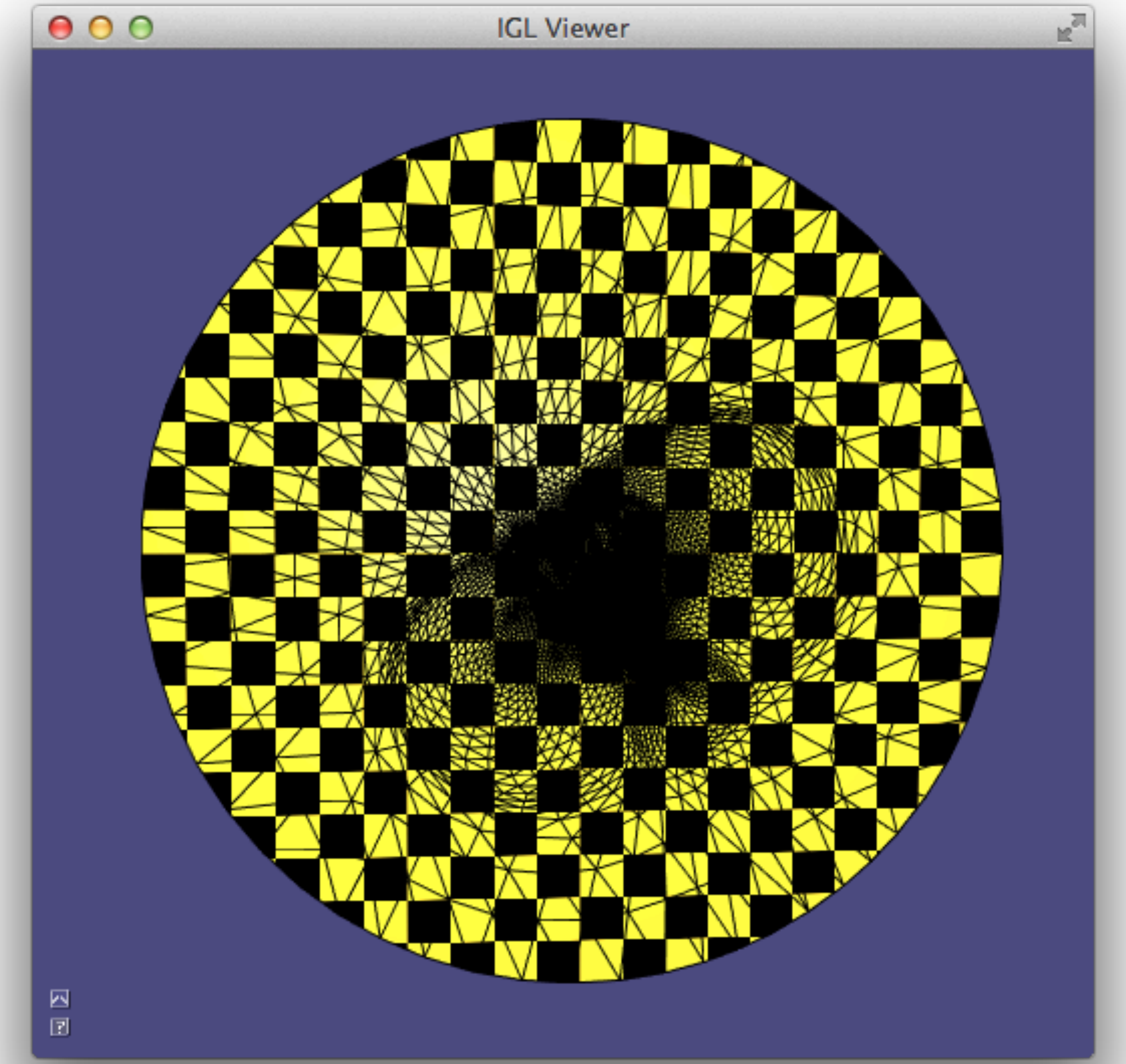
- Inner mesh edges as springs
- Find minimum-energy state where all vertices lie in the 2D plane
- Total spring energy of the flattened mesh:



$$E(\mathbf{u}_1, \ldots, \mathbf{u}_n) = \sum_{(i,j)\in\mathcal{E}} \frac{1}{2} k_{i,j} \|\mathbf{u}_i - \mathbf{u}_j\|^2$$

Demo





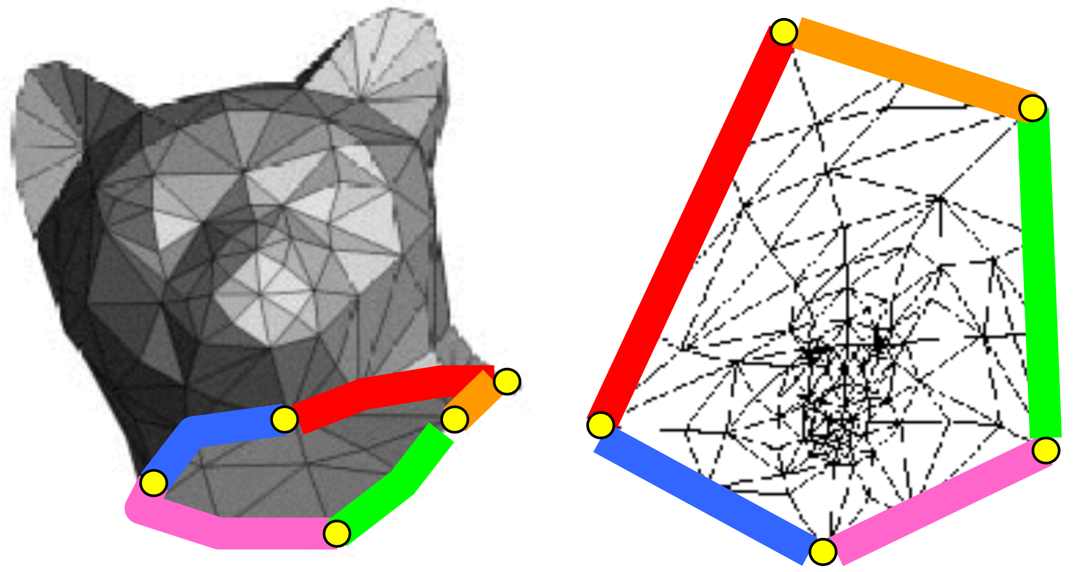
https://libigl.github.io/libigl/tutorial/tutorial.html#harmonicparametrization

Minimizing Spring Energy

$$E(\mathbf{u}_1, \ldots, \mathbf{u}_n) = \sum_{(i,j)\in\mathcal{E}} \frac{1}{2} k_{i,j} \|\mathbf{u}_i - \mathbf{u}_j\|^2$$

$$\frac{\partial E(\mathbf{u}_1, \ldots, \mathbf{u}_n)}{\partial \mathbf{u}_i} = \sum_{j\in\mathcal{N}(i)} k_{i,j}(\mathbf{u}_i - \mathbf{u}_j) = 0$$

$$\sum_{j\in\mathcal{N}(i)\cap\mathcal{B}} k_{i,j}\mathbf{u}_i + \underbrace{\sum_{j\in\mathcal{N}(i)\setminus\mathcal{B}} k_{i,j}(\mathbf{u}_i - \mathbf{u}_j)}_{\text{unknown flat vertex positions}} = \sum_{j\in\mathcal{N}(i)\cap\mathcal{B}} k_{i,j}\underbrace{\mathbf{u}_j}_{\text{known fixed boundary positions}}$$



$\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ – inner vertices

$\mathbf{v}_{n+1}, \ldots, \mathbf{v}_N$ – boundary vertices

Minimizing Spring Energy

- Sparse linear system of *n* equations to solve!

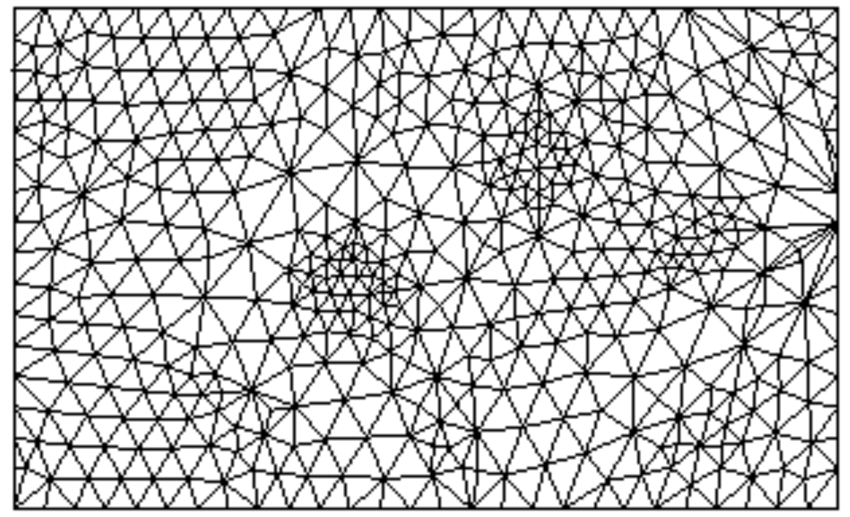
$$\sum_{j\in\mathcal{N}(i)\cap\mathcal{B}} k_{i,j}\mathbf{u}_i + \sum_{j\in\mathcal{N}(i)\setminus\mathcal{B}} k_{i,j}(\mathbf{u}_i - \mathbf{u}_j) = \sum_{j\in\mathcal{N}(i)\cap\mathcal{B}} k_{i,j}\mathbf{u}_j$$

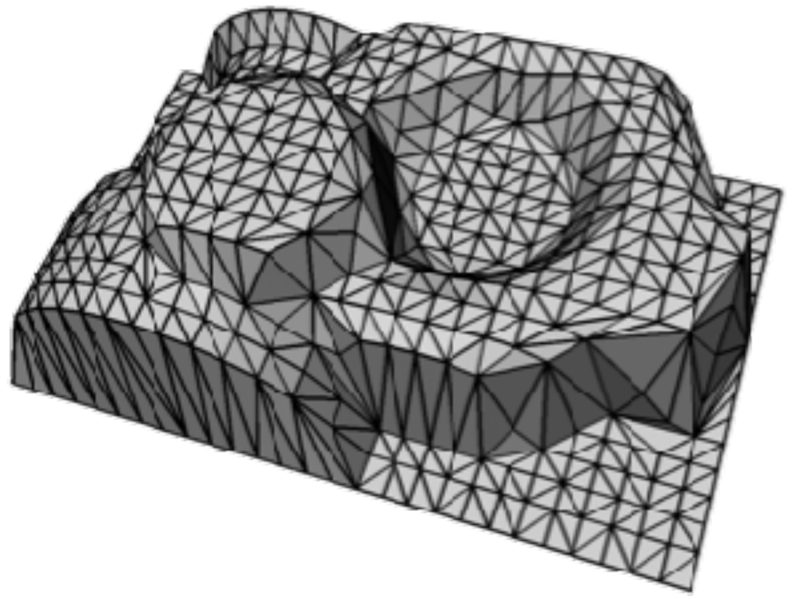
$$\begin{pmatrix} \sum_j k_{i,j} & * & \cdots & -k_{i,j} \\ * & \sum_j k_{i,j} & * & \vdots \\ \vdots & * & \ddots & * \\ -k_{j,i} & \cdots & * & \sum_j k_{i,j} \end{pmatrix} \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_n \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{u}}_1 \\ \bar{\mathbf{u}}_2 \\ \vdots \\ \bar{\mathbf{u}}_n \end{pmatrix}$$

Choice of spring constants $k_{i,j}$

- Uniform

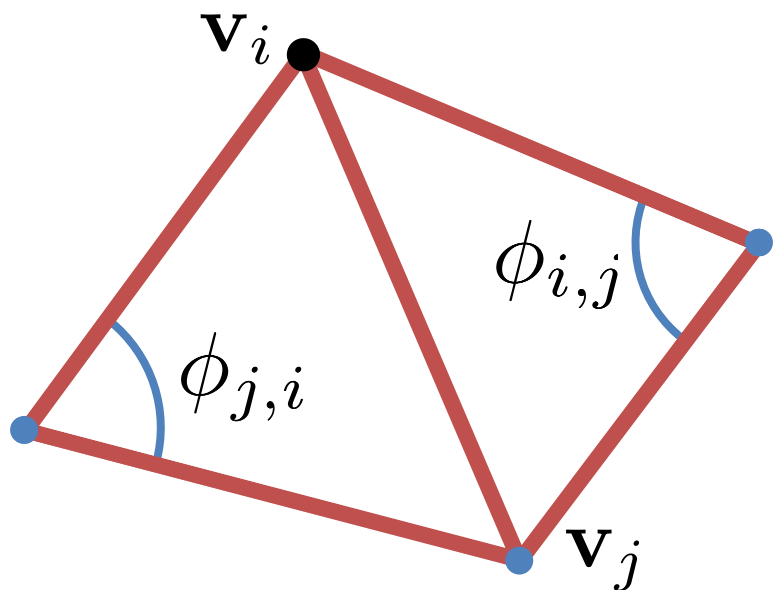
$$k_{i,j} = 1$$

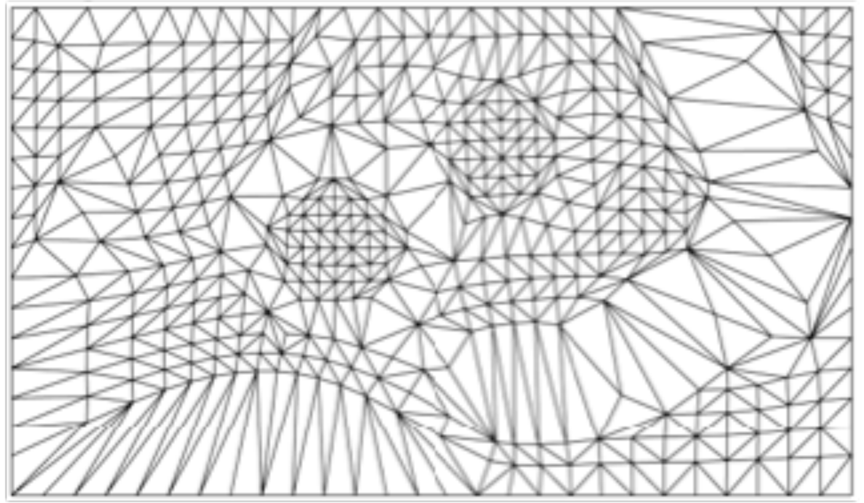




- Cotangent

$$k_{i,j} = \cot \phi_{i,j} + \cot \phi_{j,i}$$



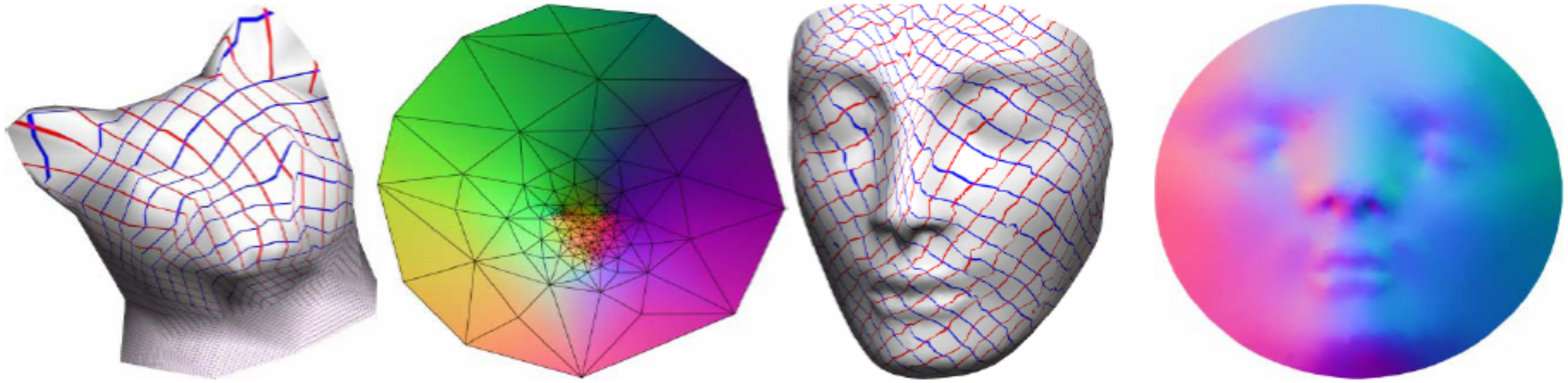


Tutte's Theorem

- If the weights are **nonnegative**, and the boundary is fixed to a **convex** polygon, the parameterization is **bijective**
- (Tutte'63 proved for uniform weights, Floater'97 extended to arbitrary nonnegative weights)
- W.T. Tutte. "How to draw a graph". Proceedings of the London Mathematical Society, 13(3):743-768, 1963.

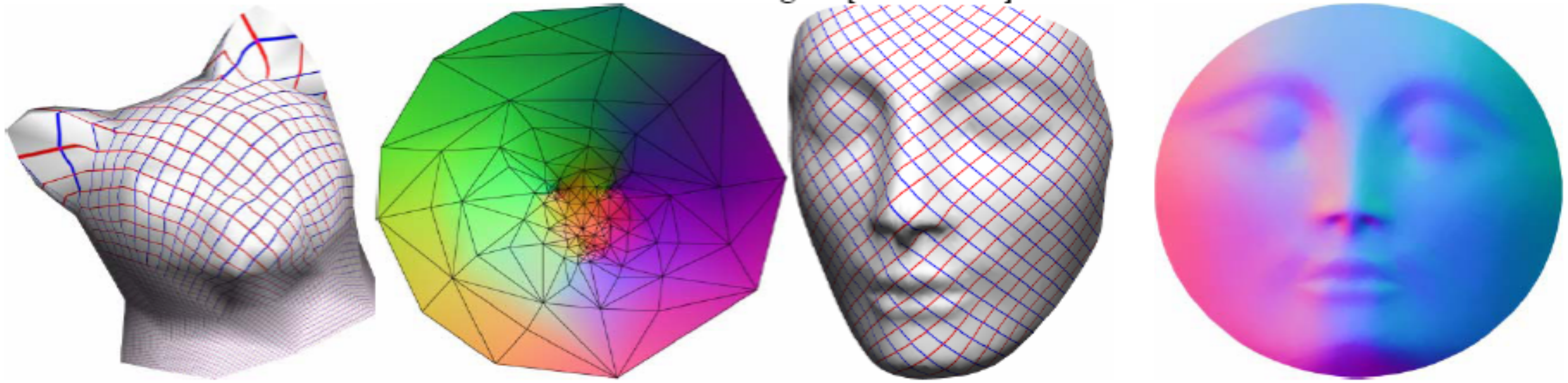
Comparison of Weights

uniform weights



Parameterization with uniform weights [Tutte 1963] on a circular domain.

cotan weights



Parameterization with harmonic weights [Eck et al. 1995] on a circular domain.

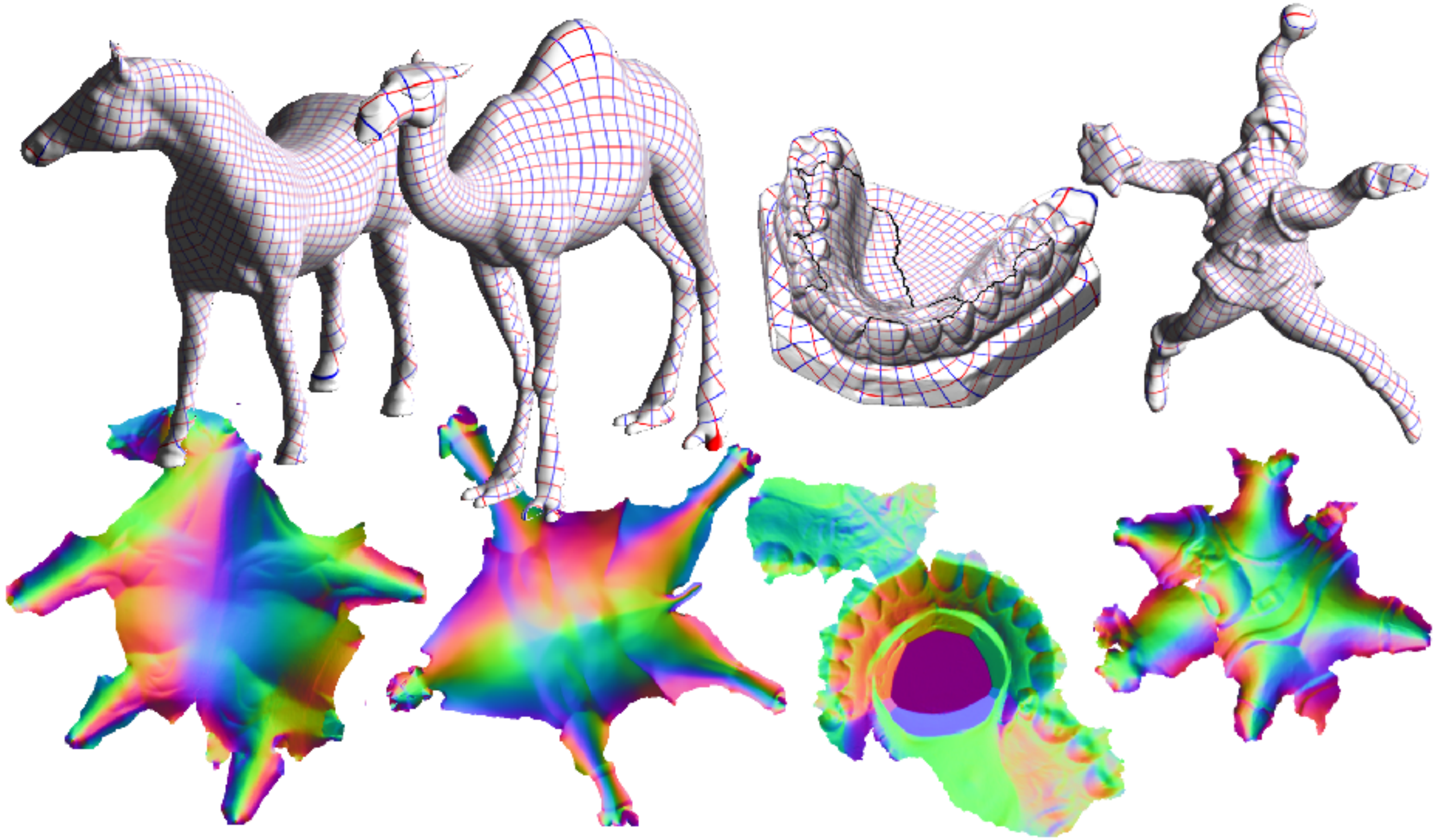
Eck et al. 1995, “Multiresolution analysis of arbitrary meshes”, SIGGRAPH 1995

Discussion

- The results of cotan-weights mapping are *better* than those of uniform convex mapping (local area and angles preservation).
- But: the mapping *is not always legal* (the cotan weights can be negative for badly-shaped triangles…)
- In any case: sparse system to solve. Robust and efficient numerical solvers exist (Eigen Sparse LDLT)

Discussion

- Both mappings have the problem of fixed boundary – it constrains the minimization and causes distortion.
- More advanced methods do not require boundary conditions.



ABF++ method,
Sheffer et al. 2005
http://www.cs.ubc.ca/~sheffa/ABF++/abf.htm

References

Fundamentals of Computer Graphics, Fourth Edition
4th Edition **by Steve Marschner, Peter Shirley**
Chapter 11

https://open.gl

Polygon Mesh Processing
Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, Bruno Levy