MVA

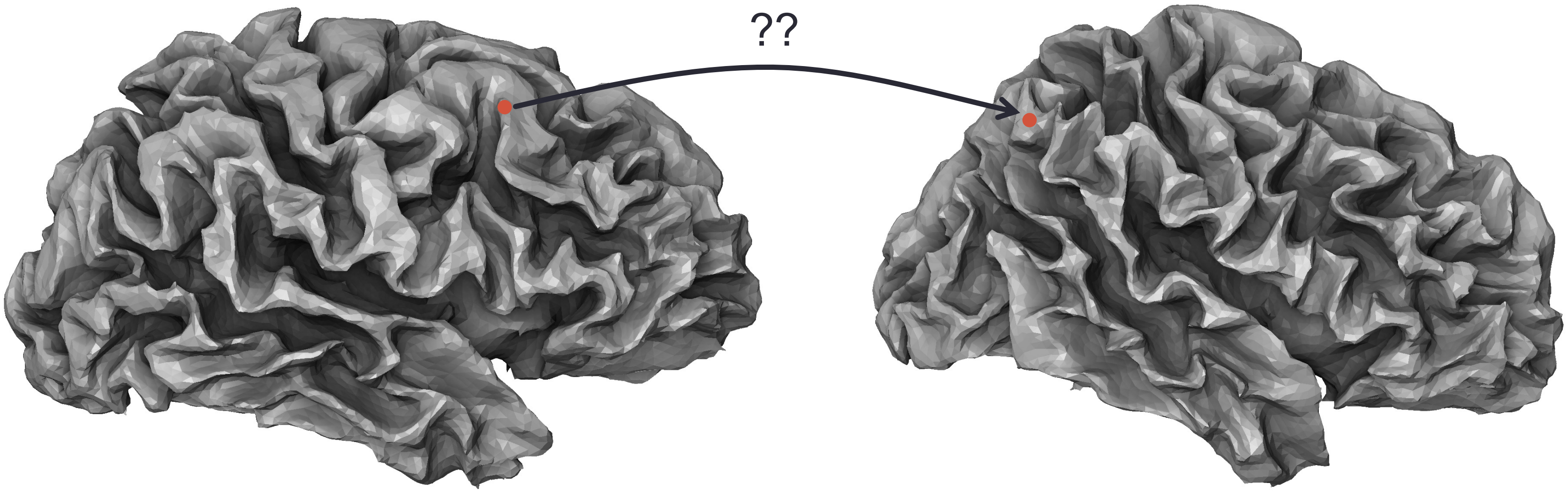
Geometry Processing and Geometric Deep Learning

Today

- Last week: geometric characterization of surfaces
- Optimization of geometric energies for shape matching
  - The matching problem
  - Topology
  - Surface parametrization
  - Surface deformation

Surface Correspondence Problem

- Which points on one object correspond to points on another?

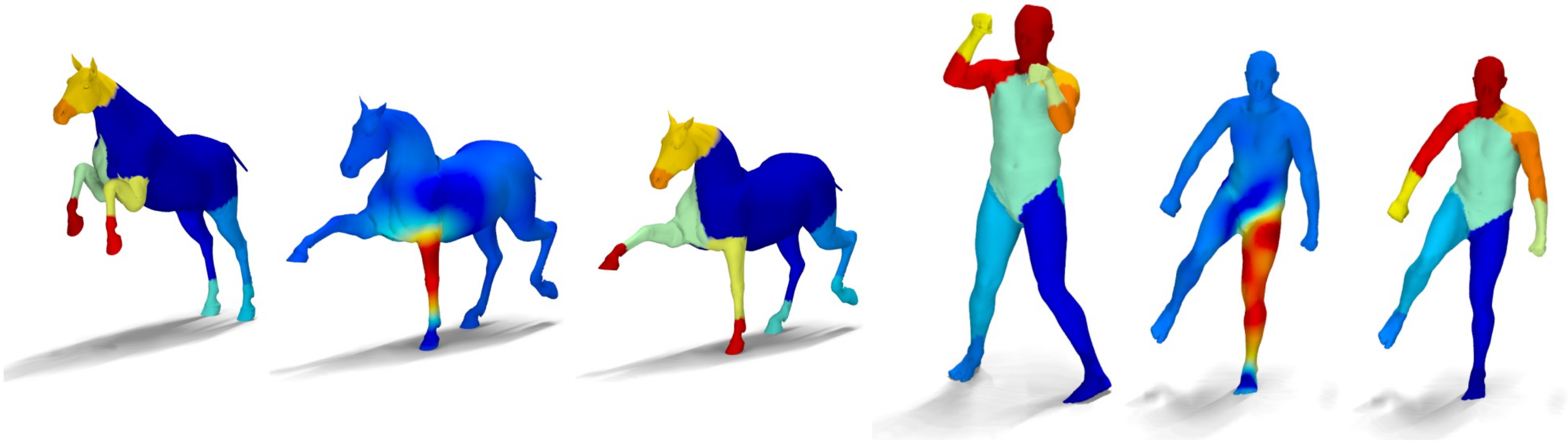


- Two approaches:
  1. Look for shared geometric structure
  2. Seek best alignment

Deformation Transfer

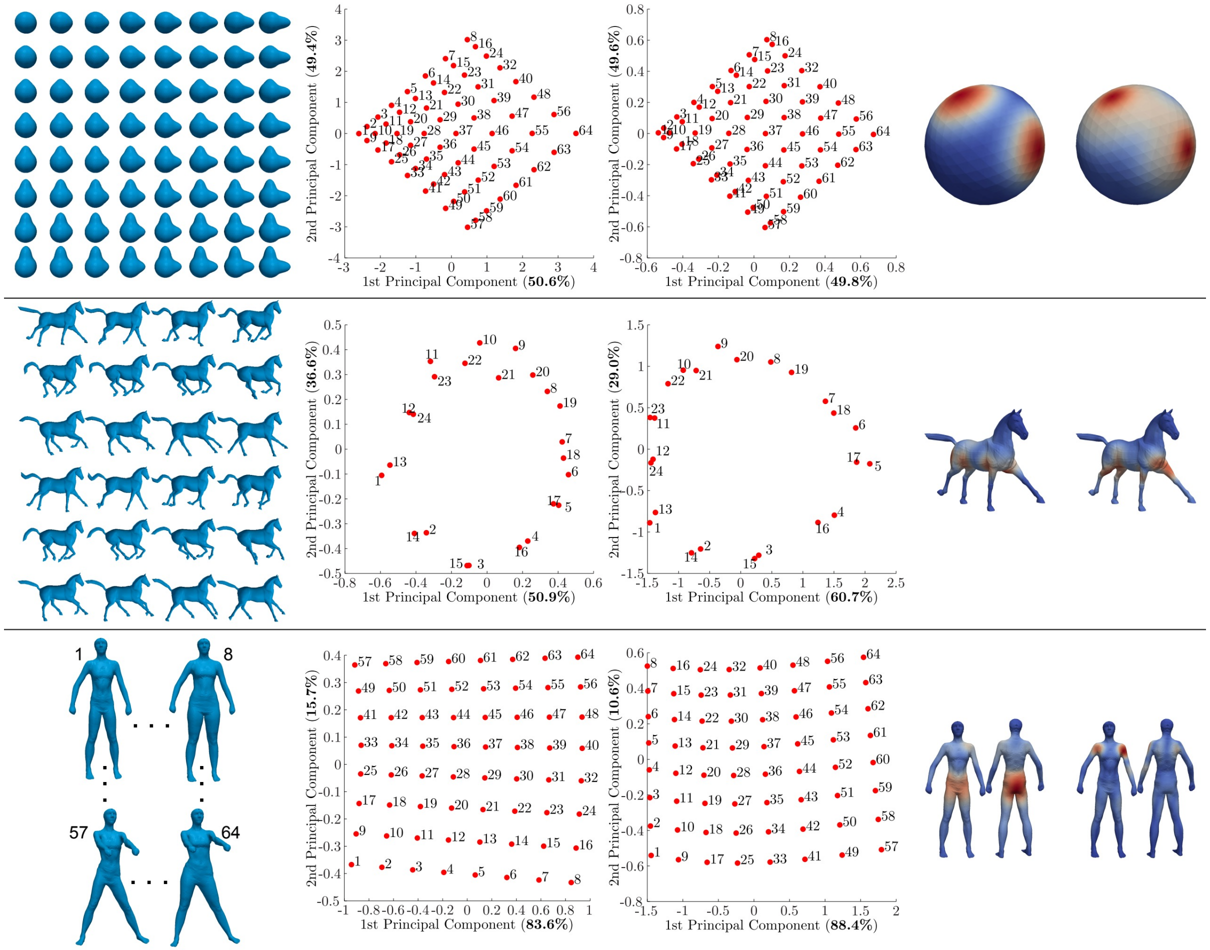
https://www.youtube.com/watch?v=9SDWKIVHnUM

Segmentation Transfer



Functional Maps: A Flexible Representation of Maps Between Shapes

Statistical Shape Analysis



Shapes in dataset: numbered left to right, top to bottom

PCA on area-based shape differences

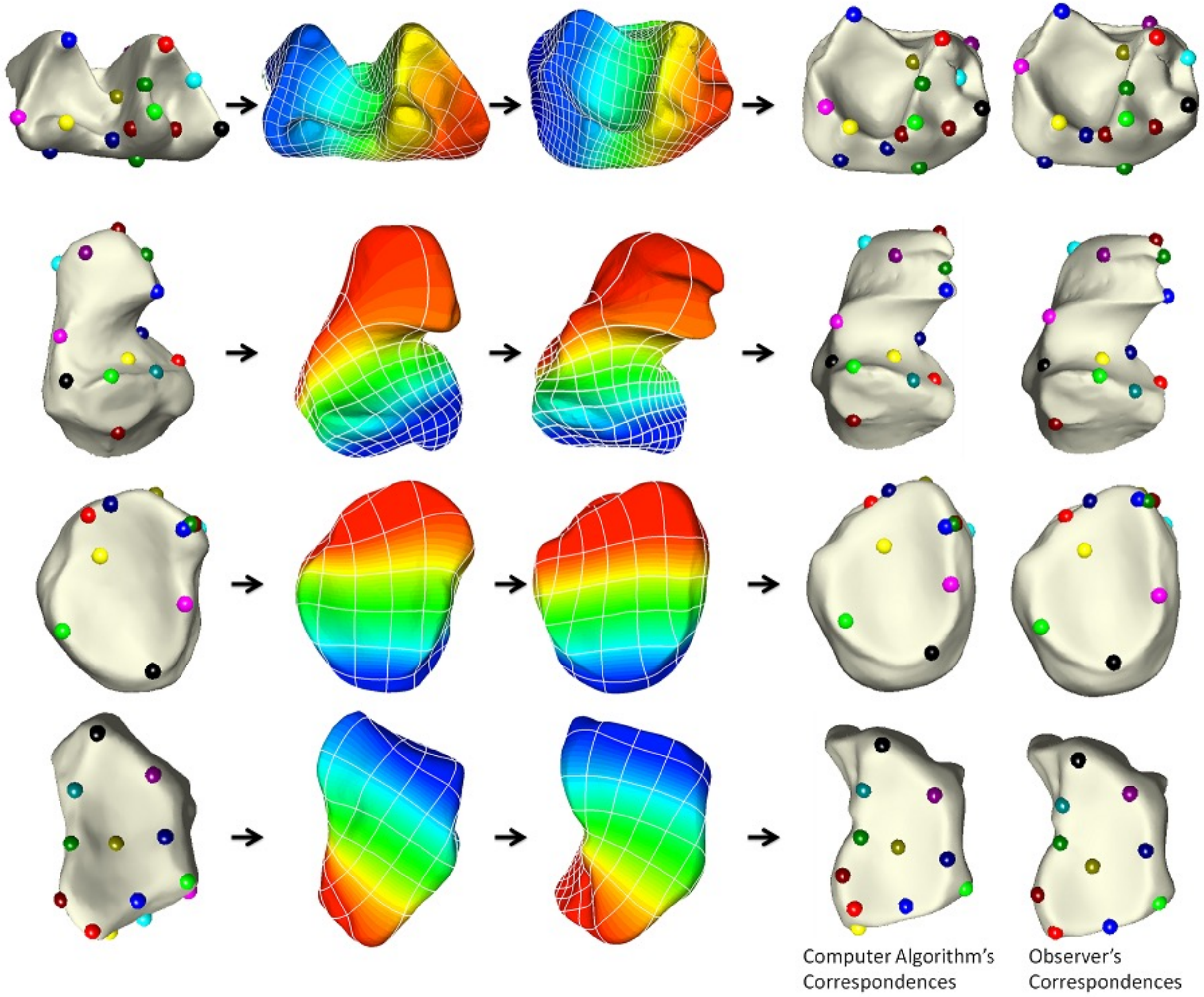
PCA on conformal shape differences

Variability localization for area

Variability localization for conformal

Map-Based Exploration of Intrinsic Shape Differences and Variability

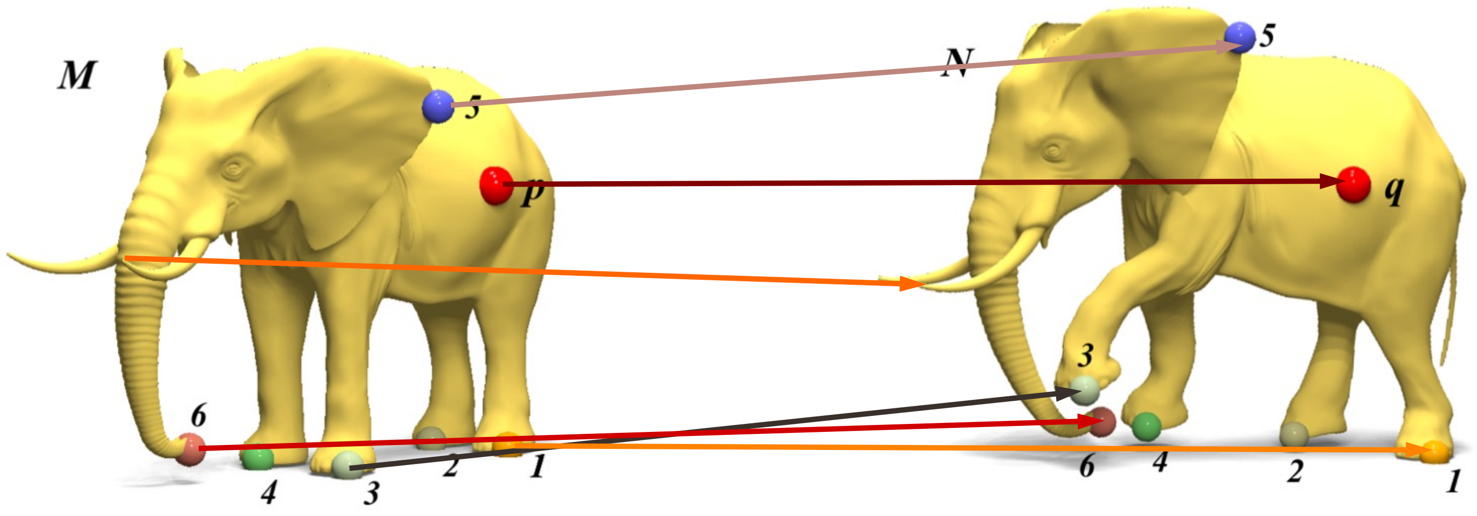
Paleontology



Algorithms to Automatically Quantify the Geometric Similarity of Anatomical Surfaces

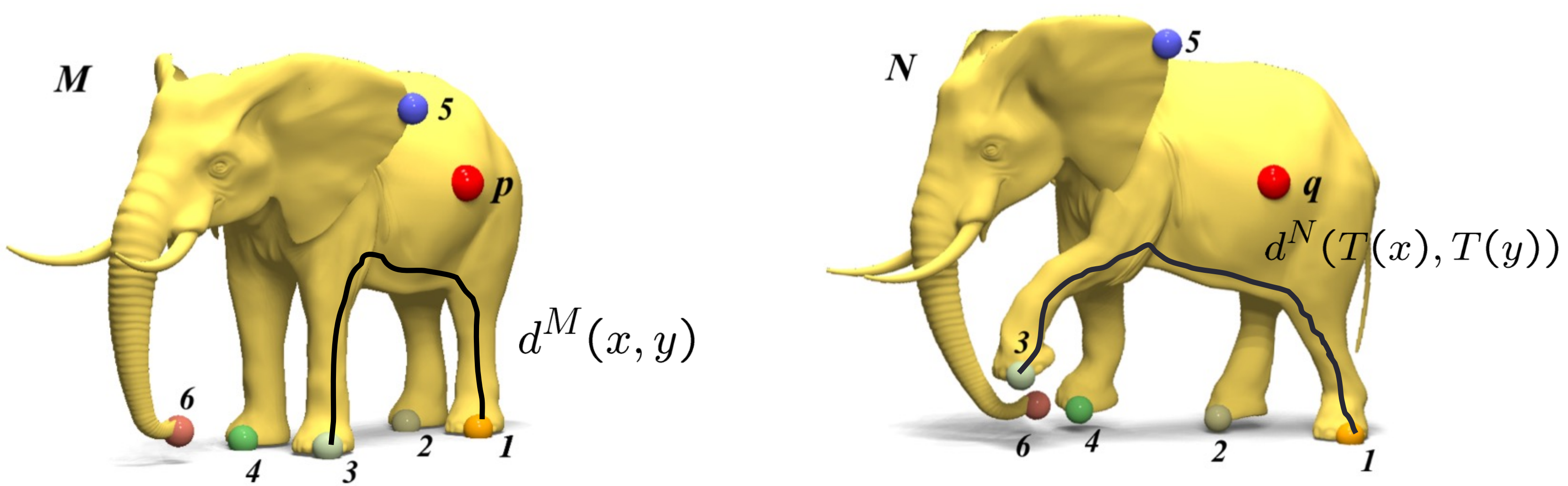
Mapping Problem

- Given a pair of shapes, find corresponding points
- An ideal map:
  - Preserves important features
  - Is fast to compute



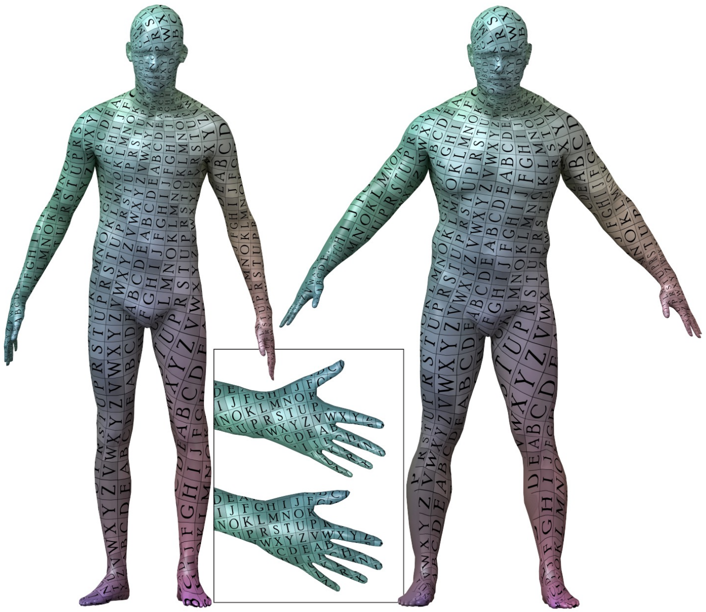
Mapping Problem

- Given a pair of shapes, find corresponding points
- An ideal map:
  - Preserves important features
  - Is fast to compute
  - Has low distortion (preserves geodesic distances)

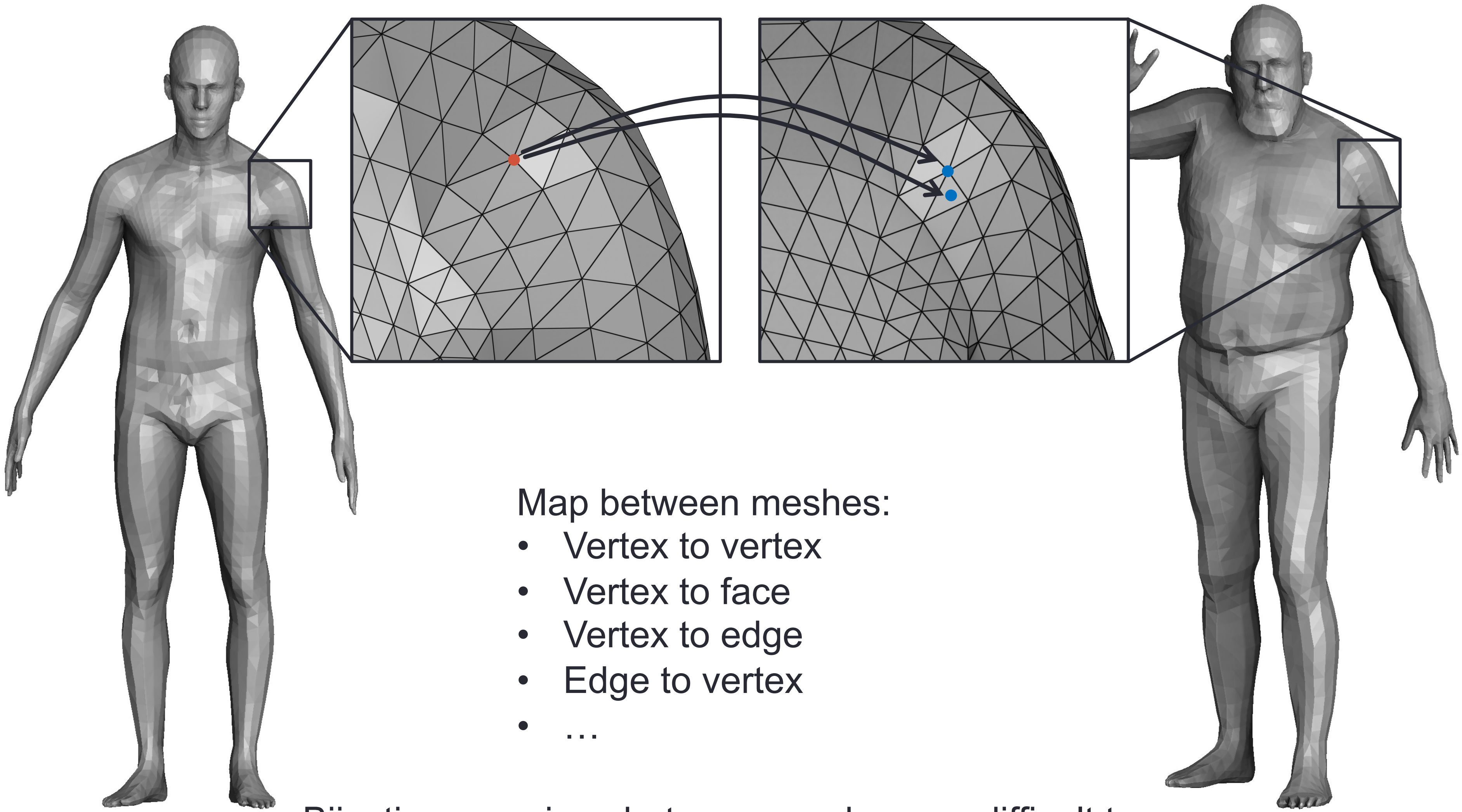


Mapping Problem

- Given a pair of shapes, find corresponding points
- An ideal map:
  - Preserves important features
  - Is fast to compute
  - Has low distortion (preserves geodesic distances)
  - Is *continuous* and bijective



Surface to Surface Map On Meshes



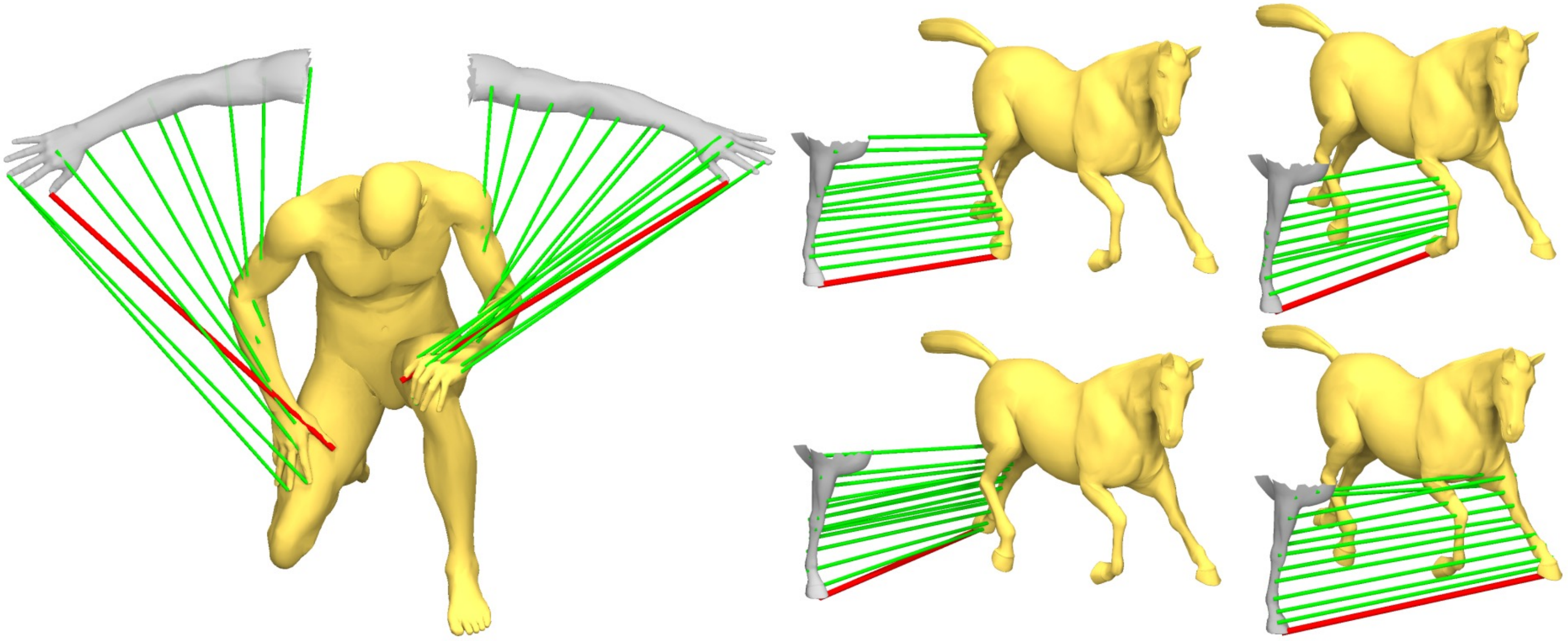
Map between meshes:

- Vertex to vertex
- Vertex to face
- Vertex to edge
- Edge to vertex
- …

Bijective mappings between meshes are difficult to parse

Vertex-To-Vertex Map

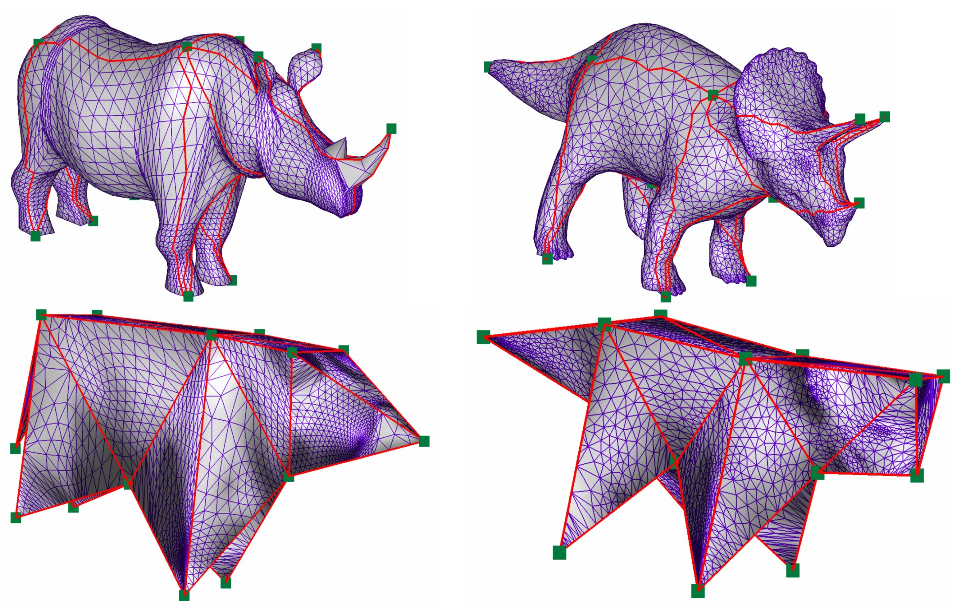
- Nearest neighbors on HKS and heat diffusion
  - Partial matching
  - No topological matching
  - Low cost
  - No continuity



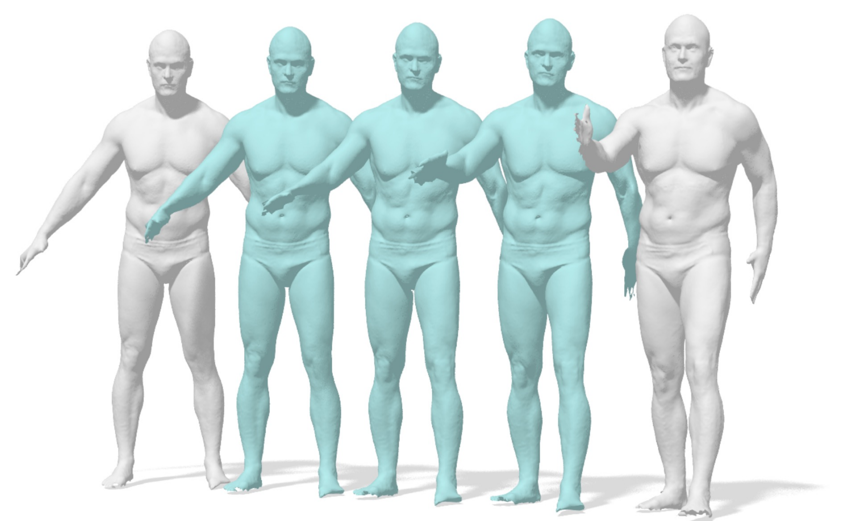
One Point Isometric Matching with the Heat Kernel

Common Methods For Computing Maps

- **Spectral methods**: Laplacian eigenfunctions
  - Fast and very flexible but no guaranties
- **Cross parametrization**: find correspondences in a common domain
  - Slow but bijective and continuous
- **Deformations**: non-rigid alignment of surfaces
  - Slow but does not guaranty continuity

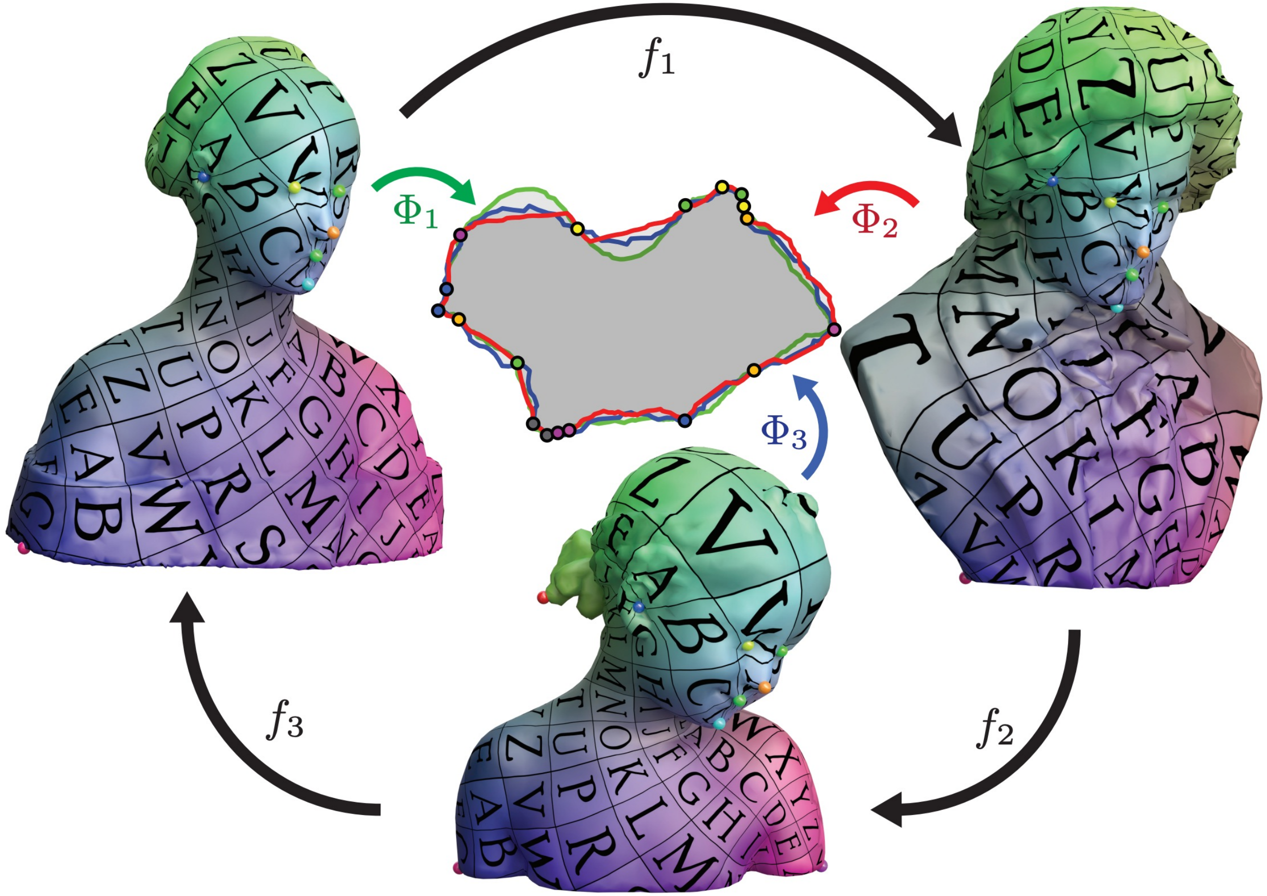


Cross-Parameterization and Compatible Remeshing of 3D Models



Divergence-Free Shape Interpolation and Correspondence

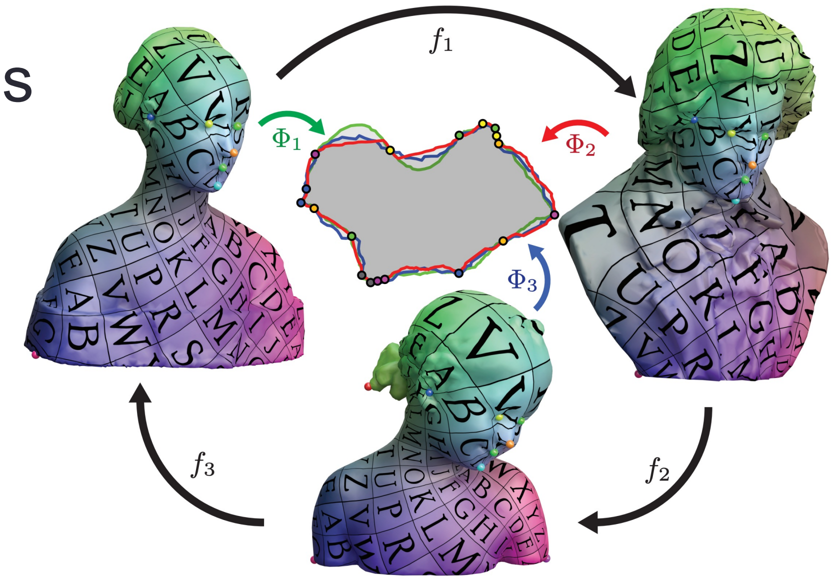
Parameterization for Matching



Seamless Surface Mappings

Cross Parameterization for **Continuous** Maps

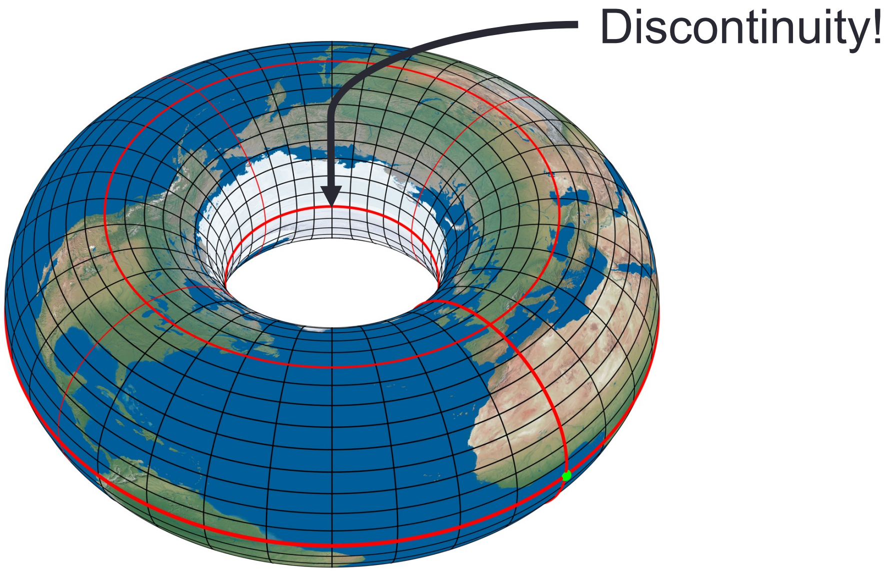
- Topological obstruction to the computation of maps
- Triangle mesh parametrization
  - Tutte embedding
  - Conformal mappings
  - And more…
- Computing correspondences



Seamless Surface Mappings

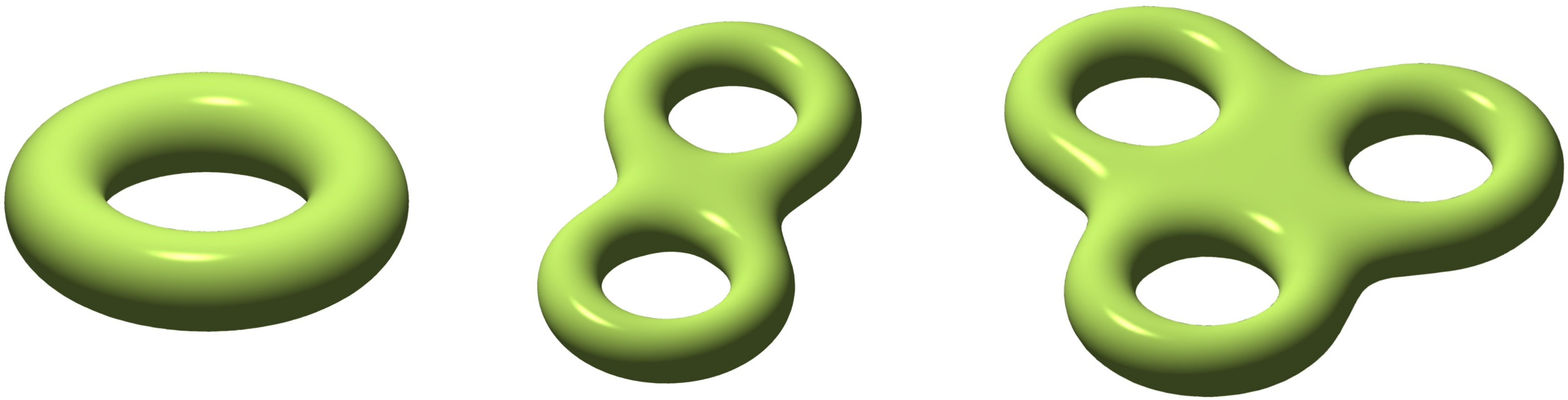
Topological Obstruction

- There is no **continuous bijective** map between these two shapes



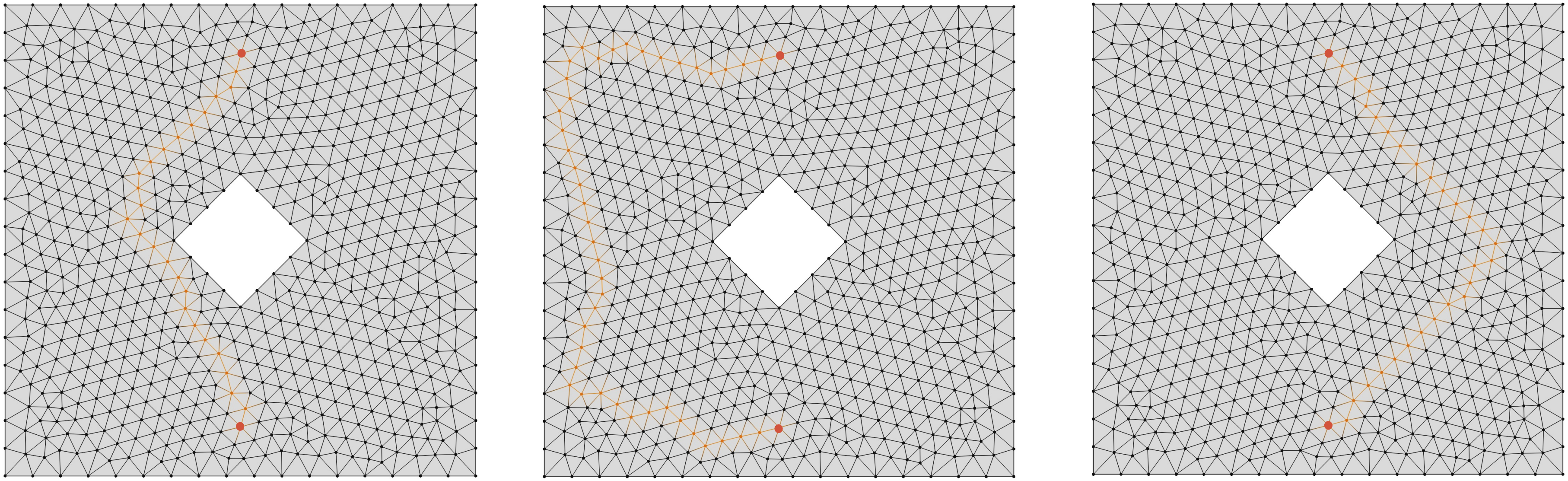
Topology

- Local geometry fully determines a surface (cf. first lecture)
- Topology studies **global** characteristics



Topology

- **Equivalent paths**: equal up to a continuous deformation.
- Two paths are **independent** if they are not related by a continuously deformation

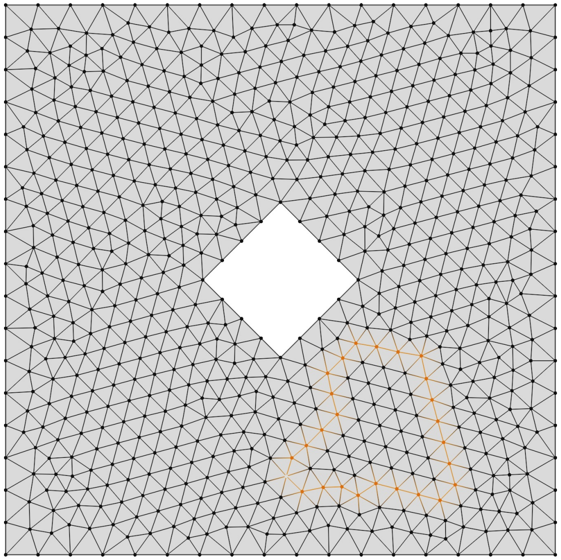


Equivalent

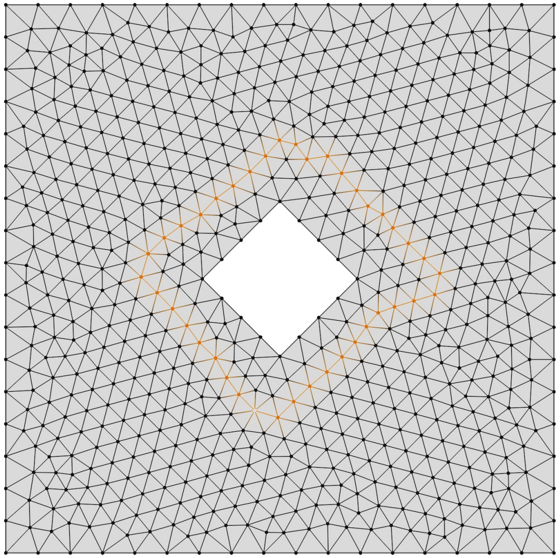
Independent

Topology

- **Contractible loop**: closed path that can be continuously contracted to a point.
- Otherwise, **non-contractible** loops.



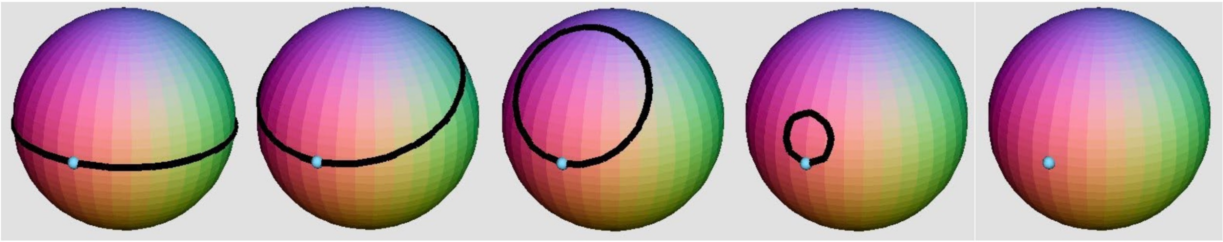
Contractible



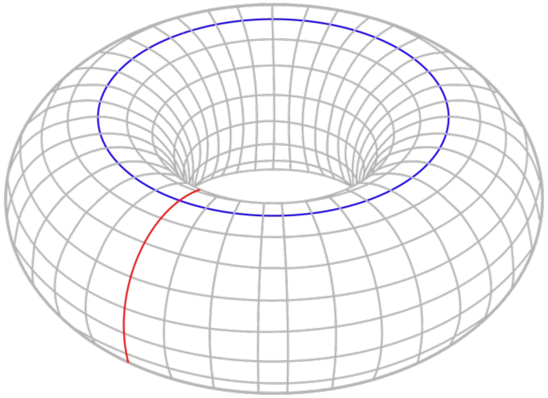
Non-contractible

Topology

- **Contractible loop**: closed path that can be contracted to a single vertex.
- Otherwise, **non-contractible** loops.



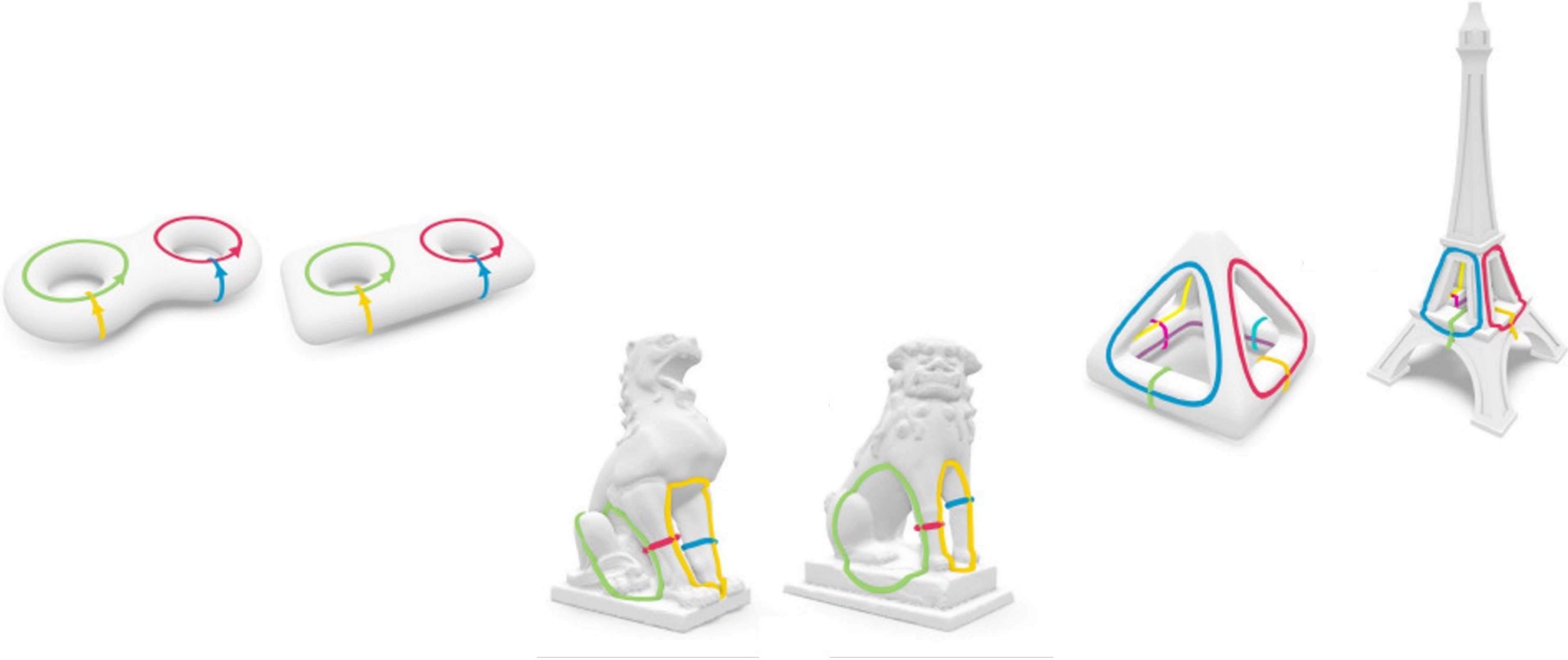
All loops are contractible



Some loops are non-contractible

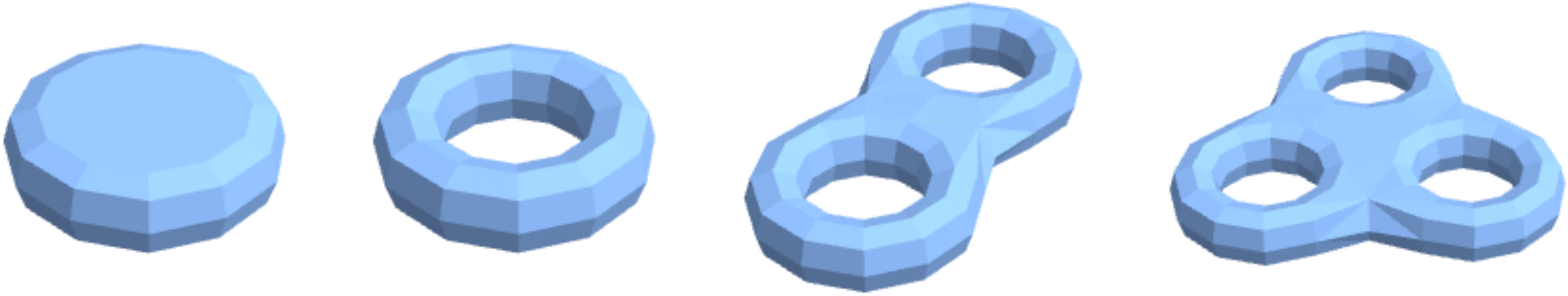
Topology

- **Contractible loop**: closed path that can be contracted to a single vertex.
- Otherwise, **non-contractible** loops.



Topology

- **Genus**: number of independent loops divided by 2 (“number of holes”)



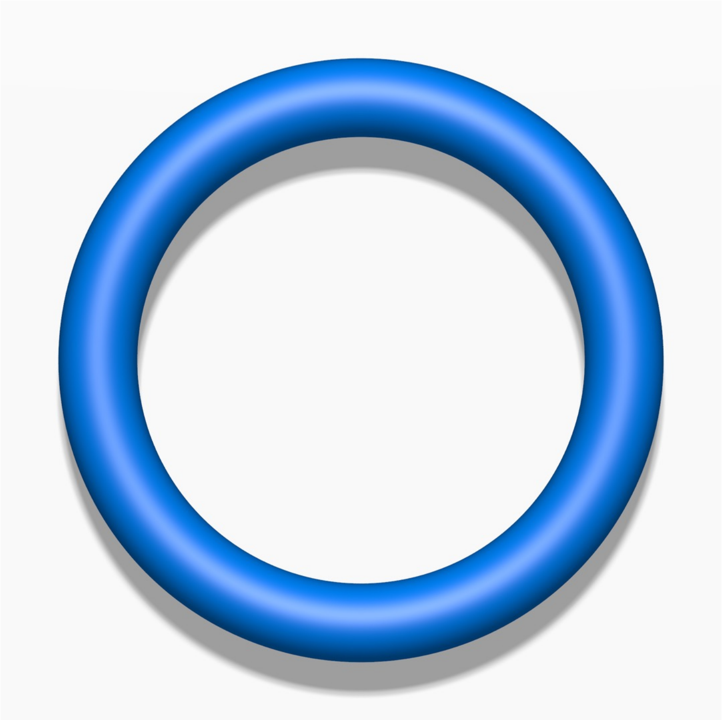
genus 0 genus 1 genus 2 genus 3

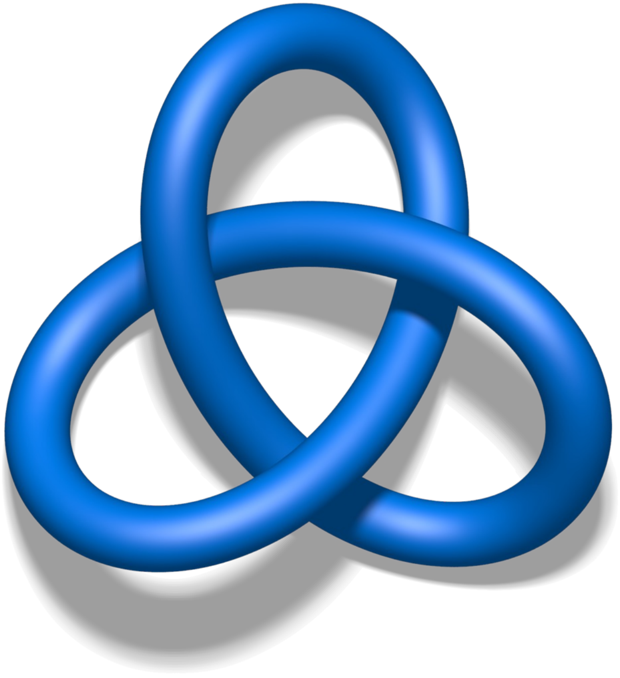
Topology and Continuous Deformation

- There exists a continuous and bijective **map** between any two surfaces with same genus

Topology and Continuous Deformation

- There exists a continuous and bijective **map** between any two surfaces with same genus
- It does not mean that there exits a bijective and continuous **deformation**

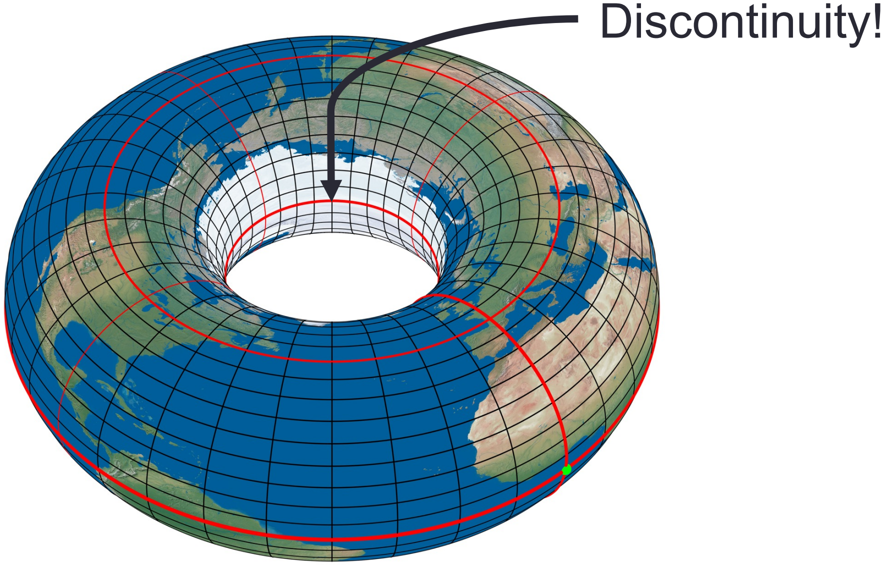




Topology and Continuous Deformation

- There is no **continuous bijective** map between these two shapes

Genus 0



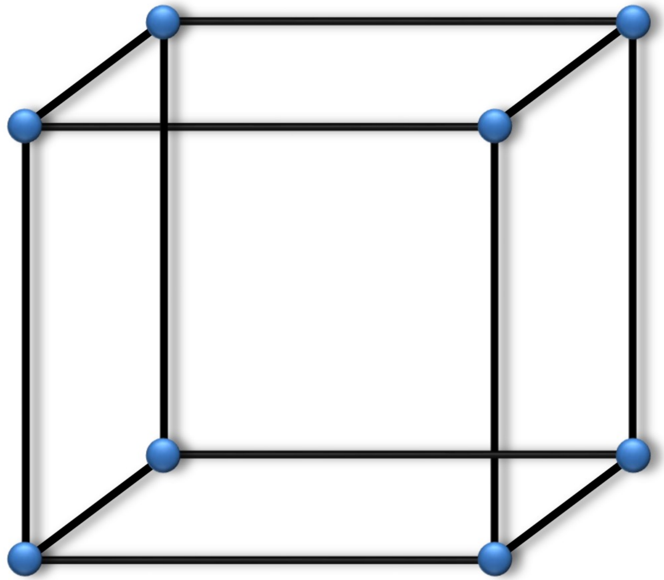
Genus 1

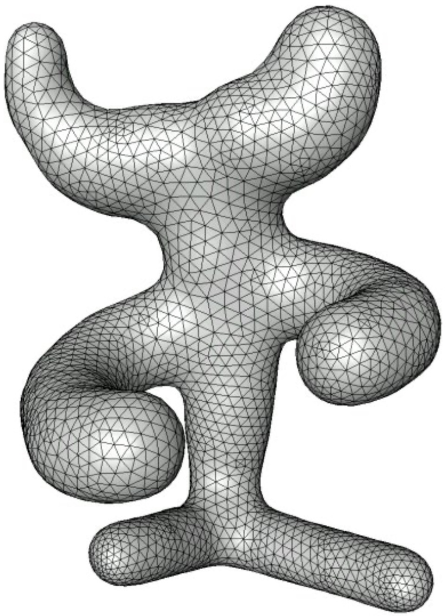
Manifold Meshes without Boundaries

- Euler-Poincaré formula $F - E + V = 2 - 2g$

edges   genus
faces   # vertices

[20 proofs]





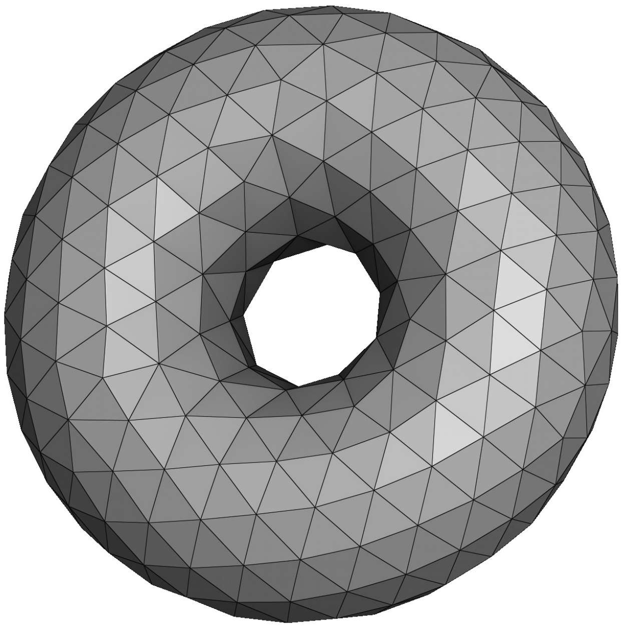
F = 6
E = 12
V = 8
2-2g = 6-12+8=2

F = 7776
E = 11664
V = 3890
g = 0

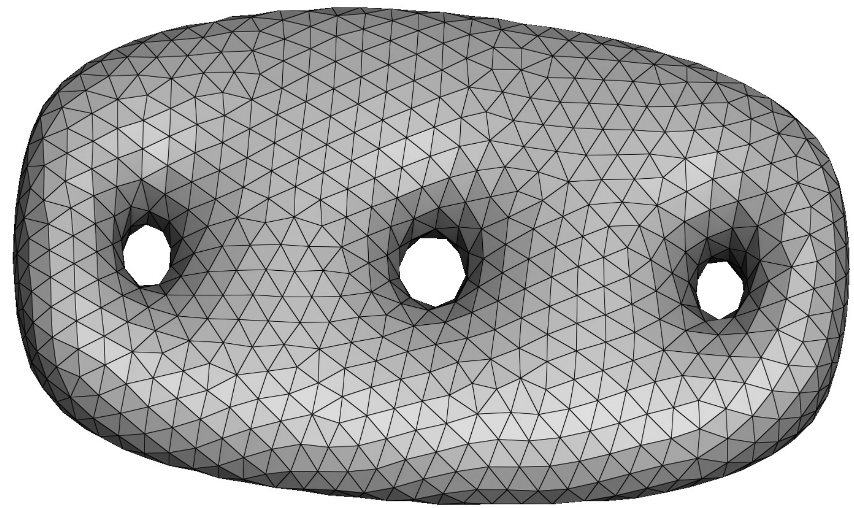
Manifold Meshes without Boundaries

- Euler-Poincaré formula $F - E + V = 2 - 2g$

20 proofs



F = 600
E = 900
V = 300
g = 1



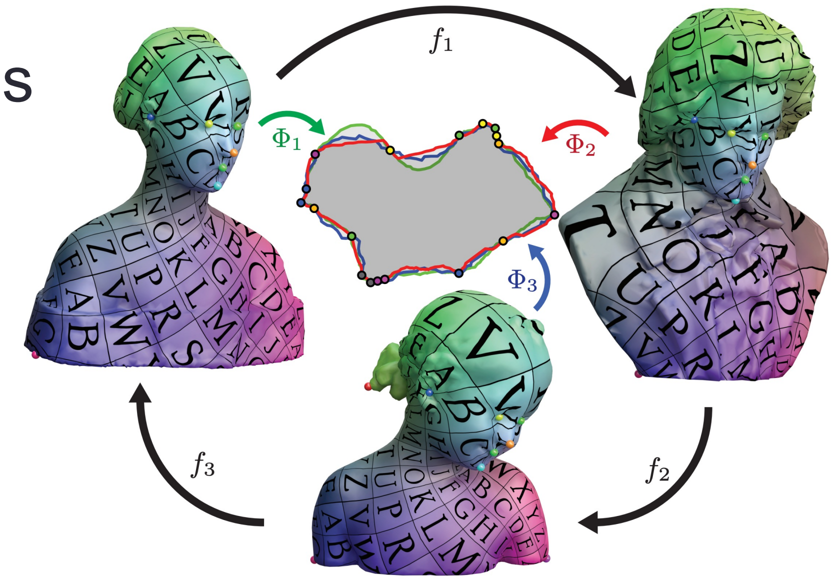
F = 2408
E = 3612
V = 1200
g = 3

Topological Conclusion

- The genus is a global invariant of a surface
- The genus is easily computed with Euler-Poincaré formula
- Surfaces with same genus can continuously mapped into each other

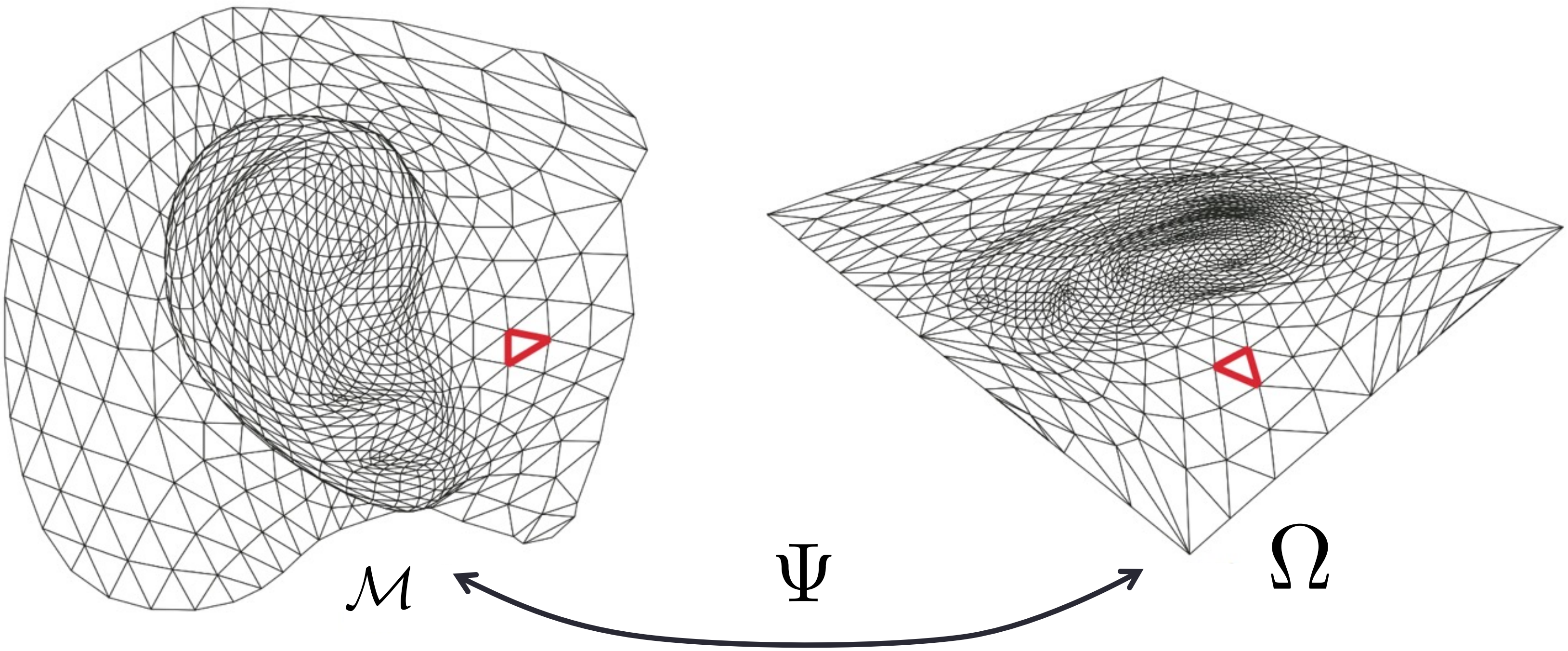
Cross **Parameterization** for Continuous Maps

- Topological obstruction to the computation of maps
  - Triangle mesh parametrization
    - Tutte embedding
    - Conformal mappings
    - And more…
- Computing correspondences



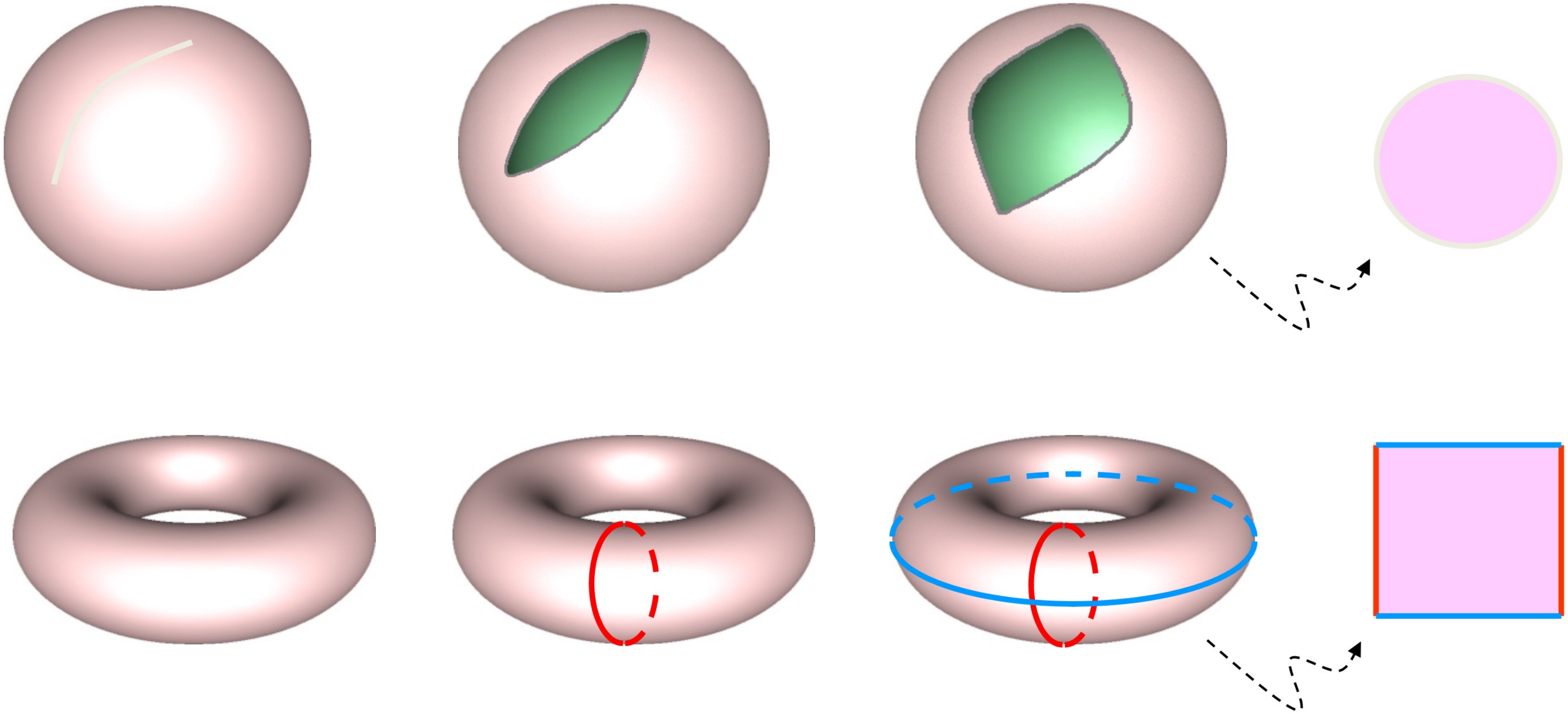
Parameterization Problem

Given a surface (mesh) $\mathcal{M}$ in R^3 and a planar domain Ω:
Find a bijective map $\Psi : \Omega \longleftrightarrow \mathcal{M}$.

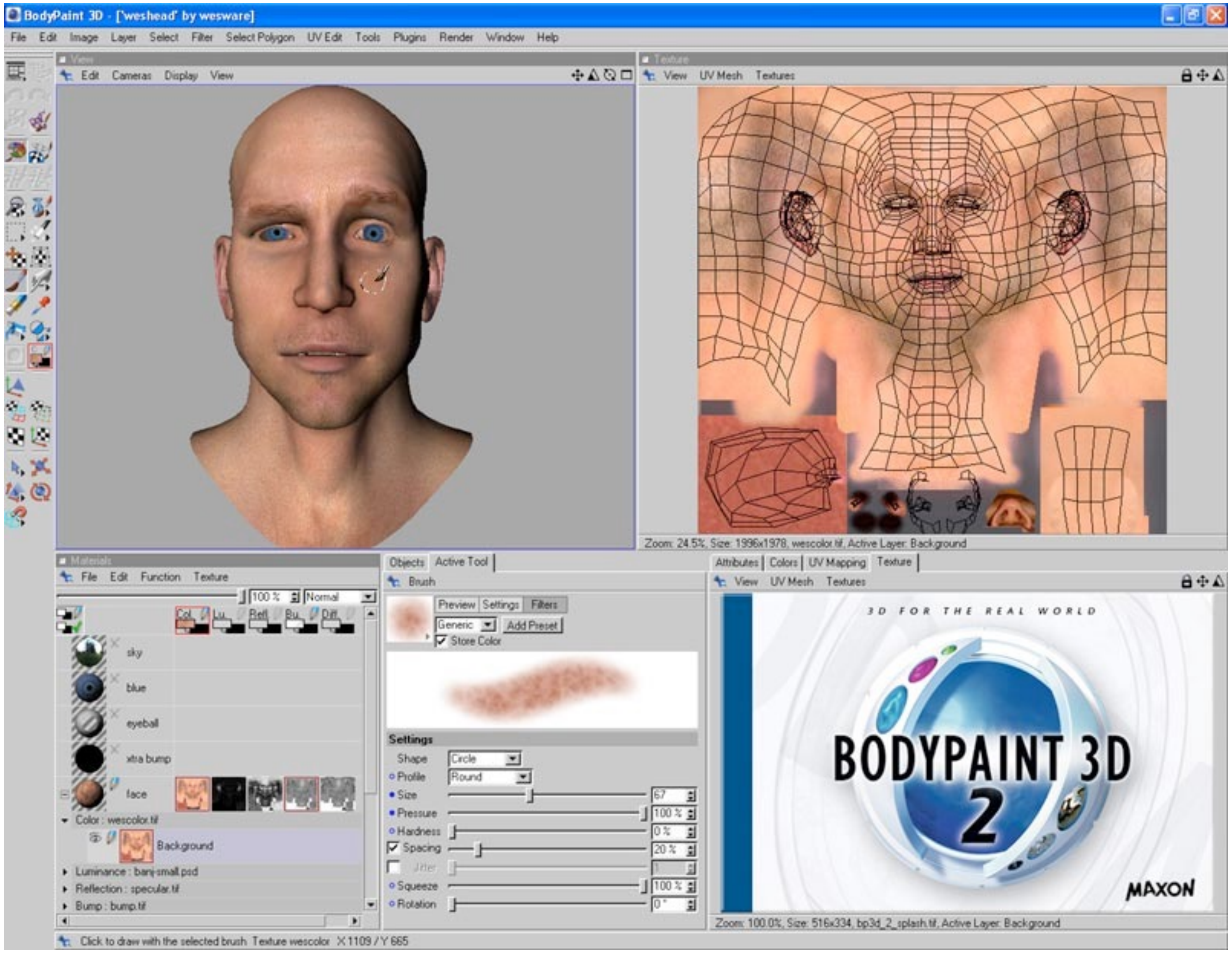


Cutting To a Disk

- For non-disk topology: need to creates artificial boundaries
- For high genus, the cut graph is constructed from non-contractible loops



Parameterization for Texture Mapping



Bodypaint 3D

Parameterization – Applications

Mesh simplification:

- Approximate the geometry using few triangles

Idea:

- Decouple geometry from appearance

~600k triangles

~600 triangles

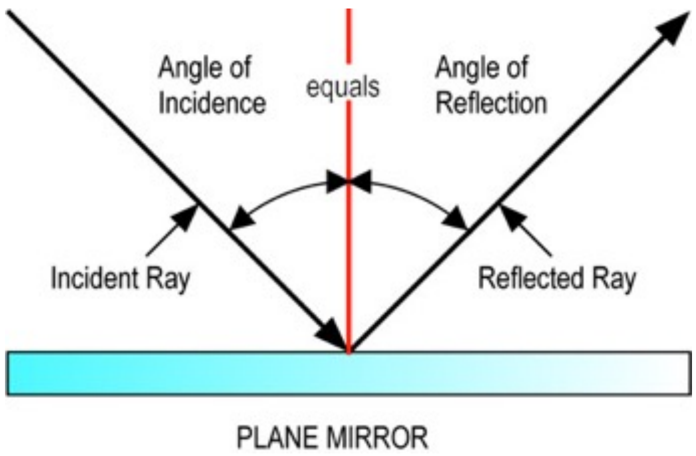
Parameterization – Applications

Mesh simplification:

- Approximate the geometry using few triangles

Idea:

- Decouple geometry from appearance

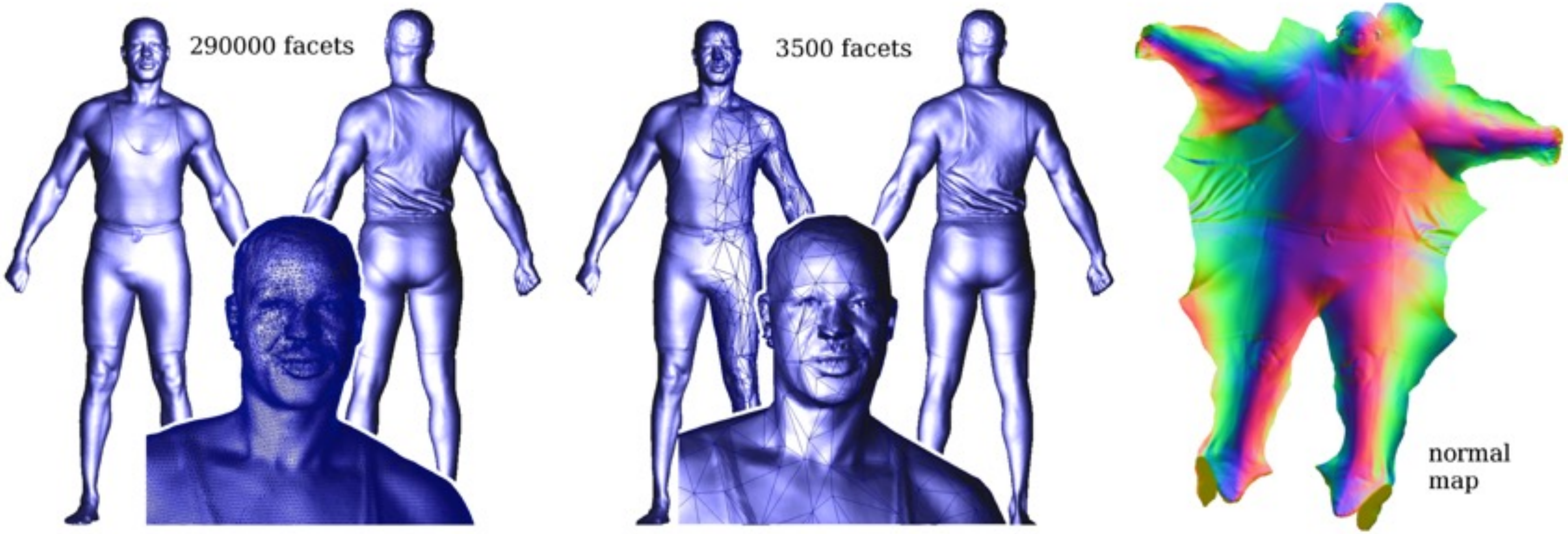


Observation: appearance (light reflection) depends on the geometry + normal directions.

Parameterization – Applications

Normal Mapping with parameterization:

- Store normal field as an RGB texture.

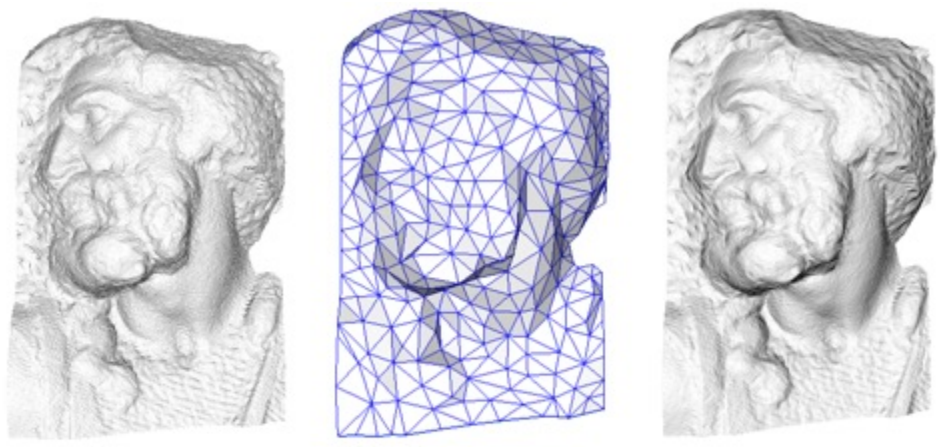


Parameterization – Applications

Normal Mapping

Idea:

- Decouple geometry from appearance
- Encode a normal field inside each triangle



original mesh
4M triangles

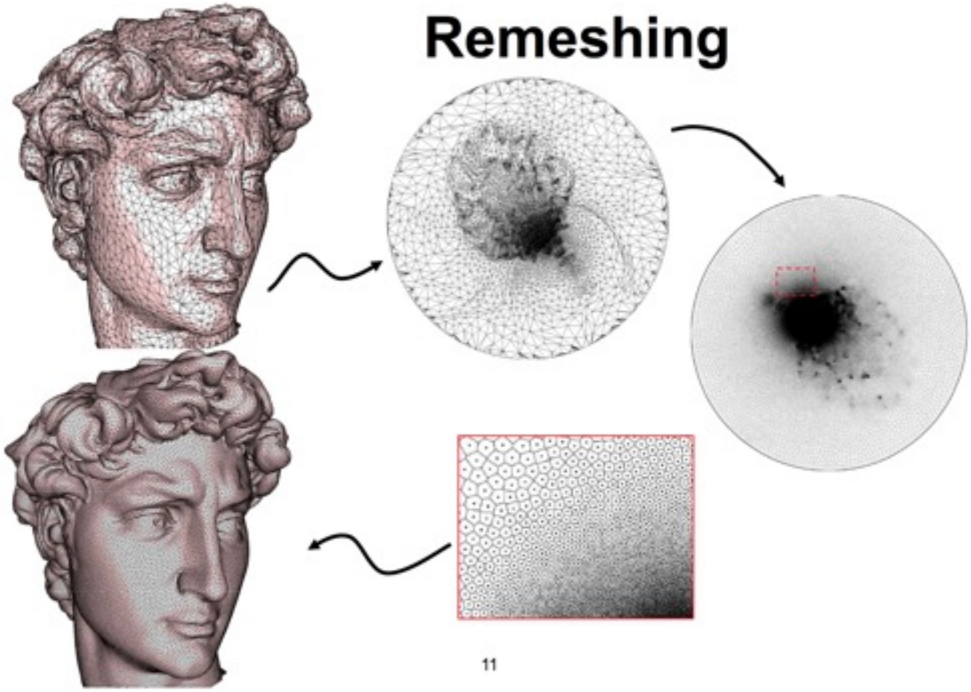
simplified mesh
500 triangles

simplified mesh
and normal mapping
500 triangles

Cohen et al., '98
Cignoni et al. '98

Parameterization – Applications

General Idea: Things become easier in a canonical domain (e.g. on a plane).



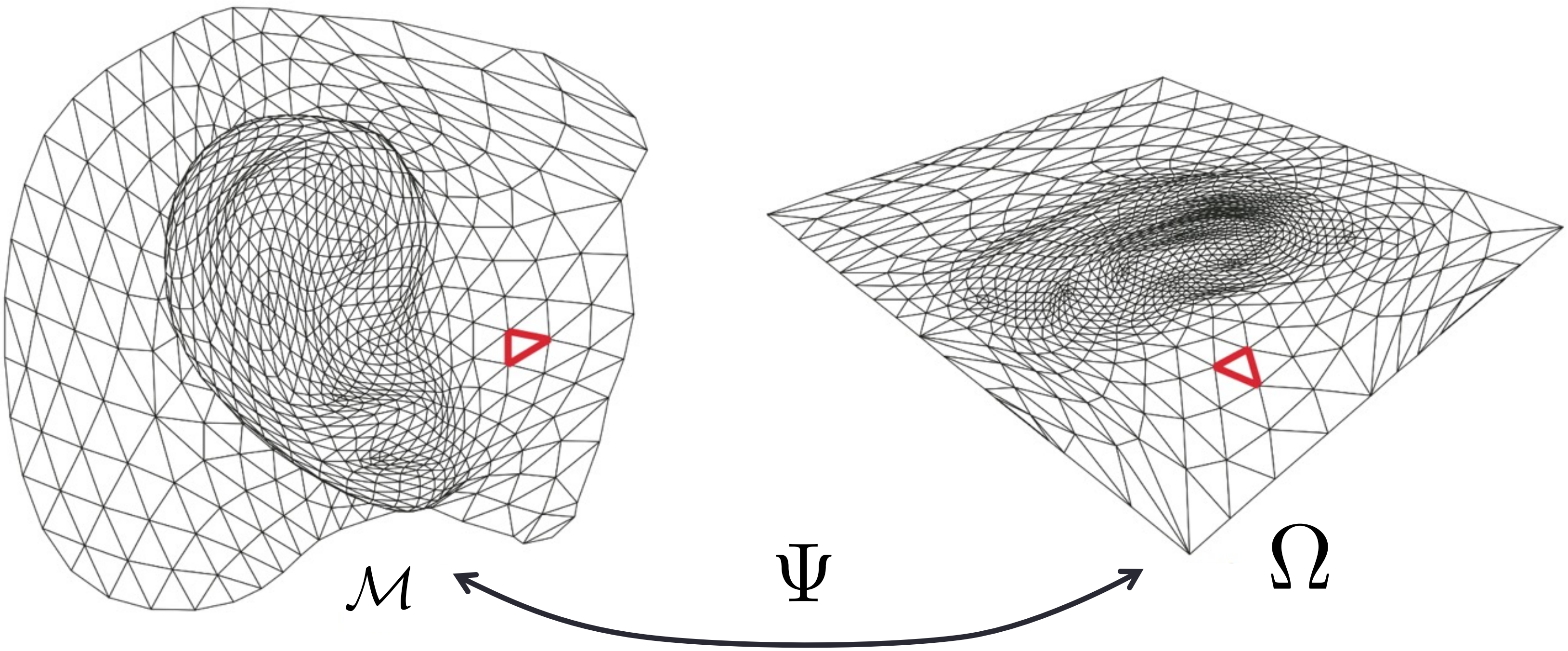
Other Applications:

- Surface Fitting
- Editing
- Mesh Completion
- Mesh Interpolation
- Morphing and Transfer
- Shape Matching
- Visualization
- Feature Learning

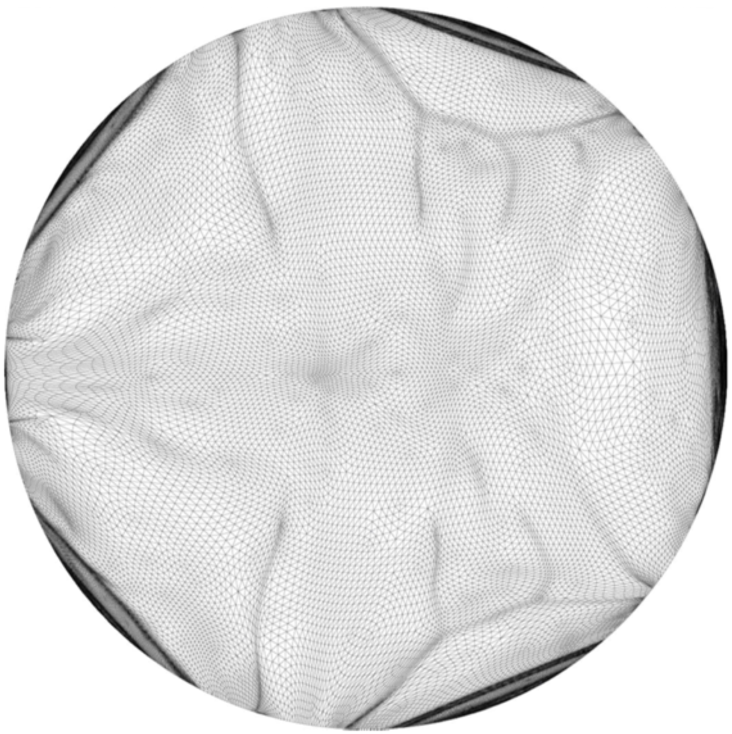
…

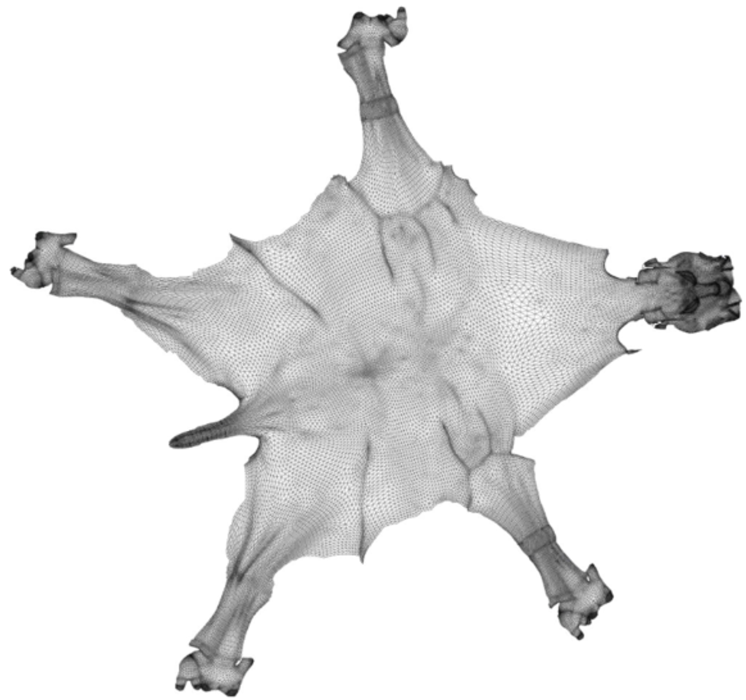
Parameterization Problem

Given a surface (mesh) $\mathcal{M}$ in R^3 and a planar domain Ω:
Find a bijective map $\Psi : \Omega \longleftrightarrow \mathcal{M}$.



Parameterization – Many Possibilities





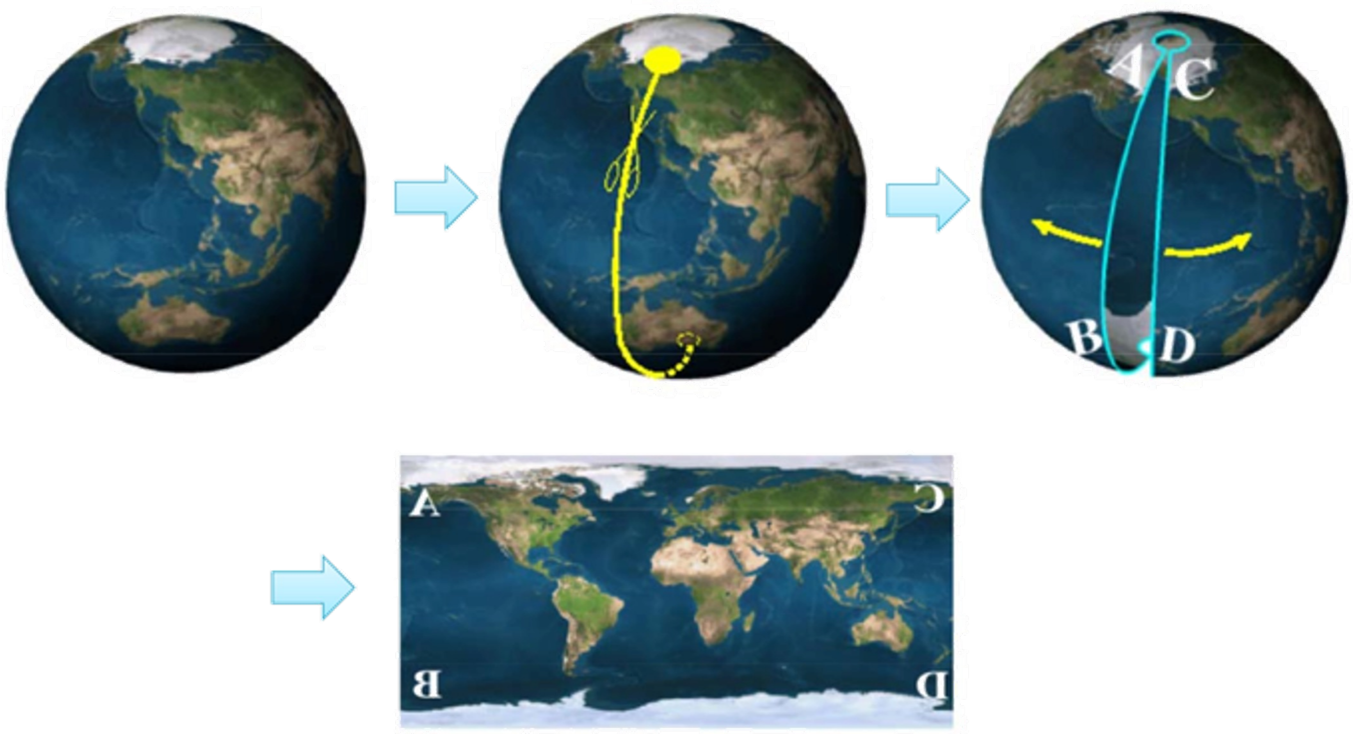
We need to quantify the distortion induced by the map

Image by Mirela Ben-Chen

Parameterization onto the plane

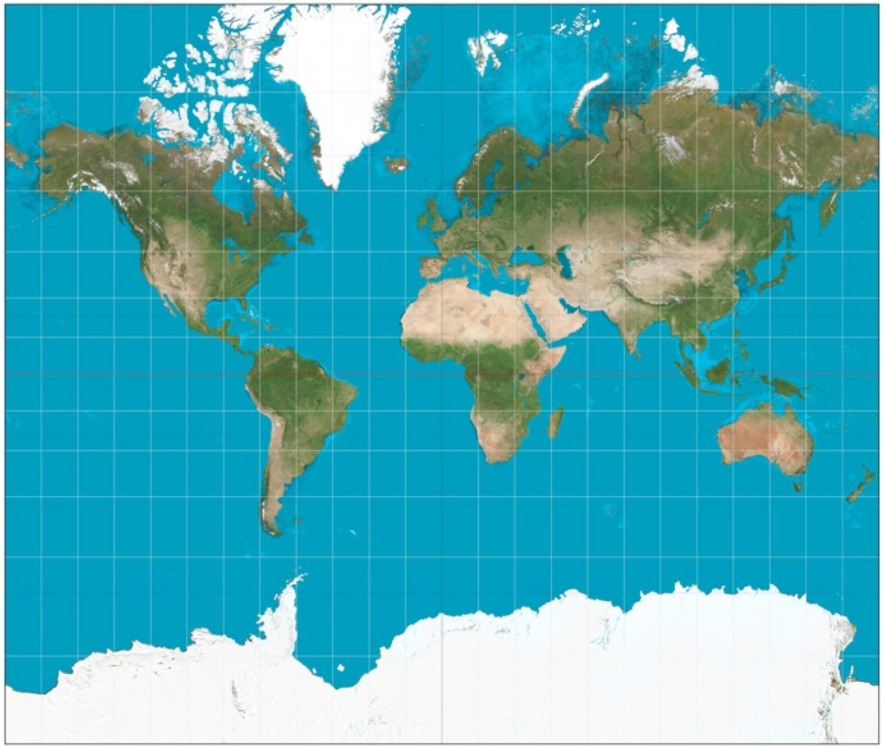
Recall a related problem.

Mapping the Earth: find a parameterization of a 3d object onto a plane.



Mapping the earth

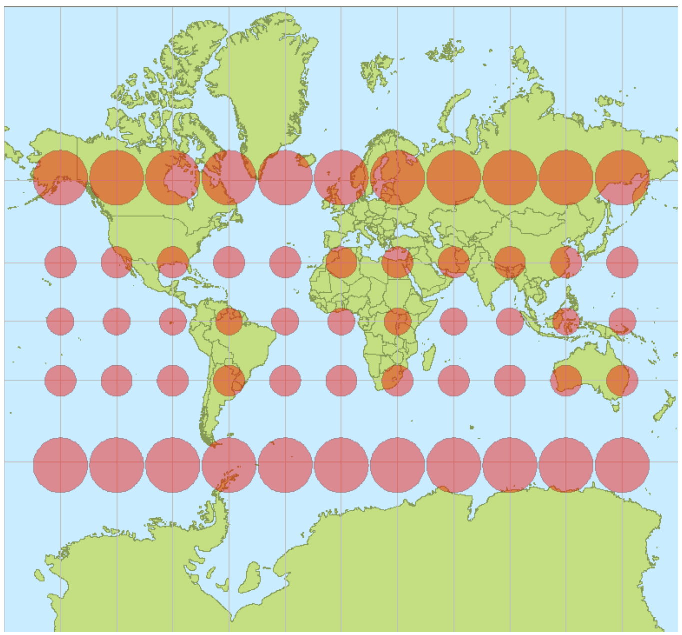
Mercator: meridians and latitudes are mapped to straight lines



Gerardus Mercator (1569)

Mapping the earth

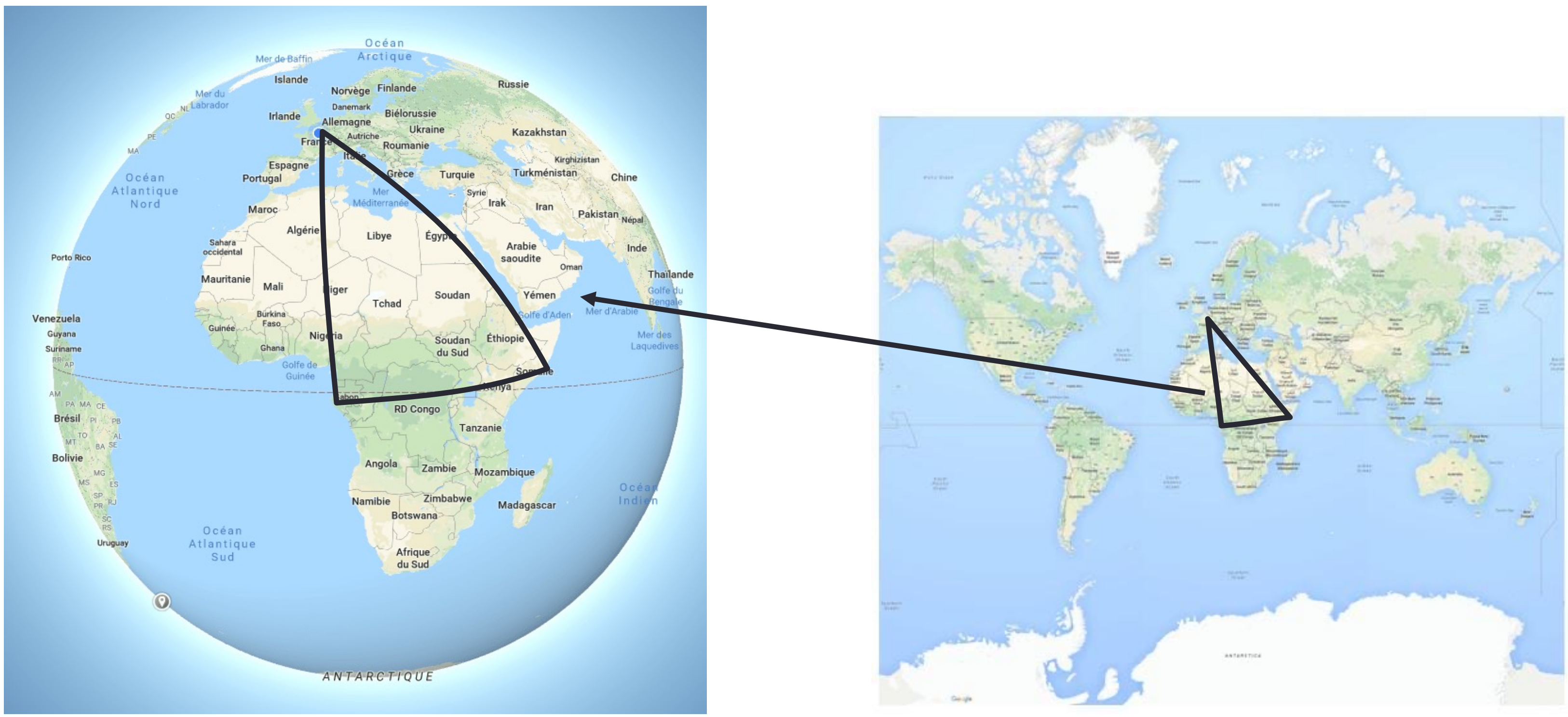
Mercator: undefined at poles, **distorts areas**



Gerardus Mercator (1569)

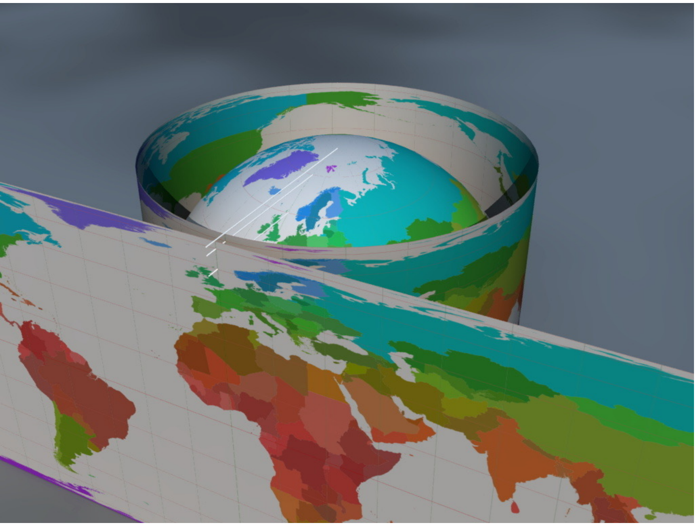
Mapping the earth

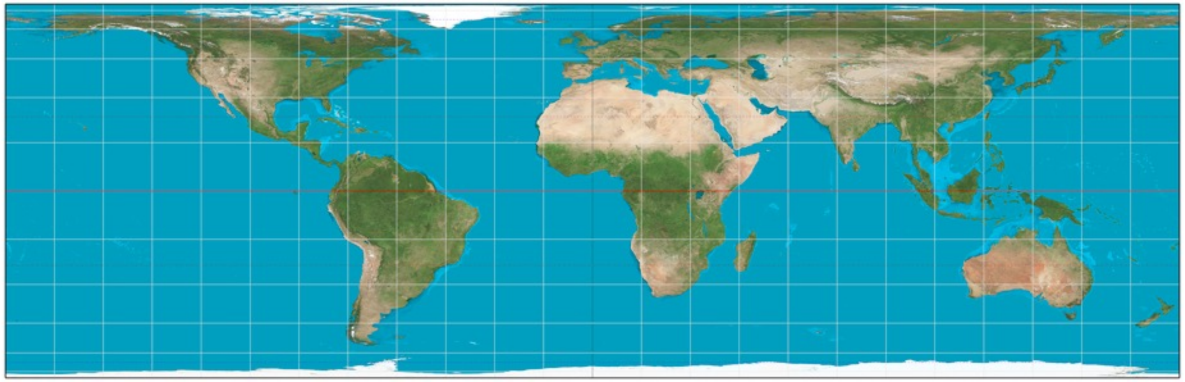
Mercator: undefined at poles, distorts areas, **preserves angles**

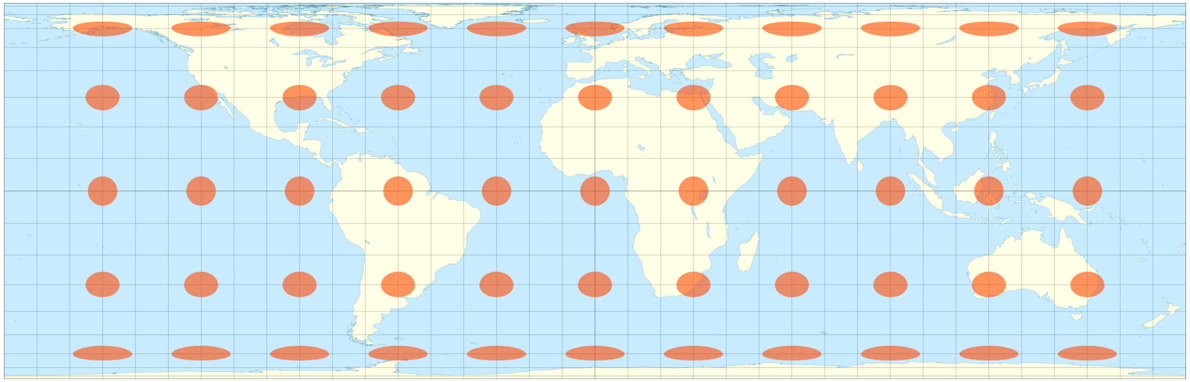


Mapping the earth

Lambert cylindrical projection: distorts angles, preserves areas



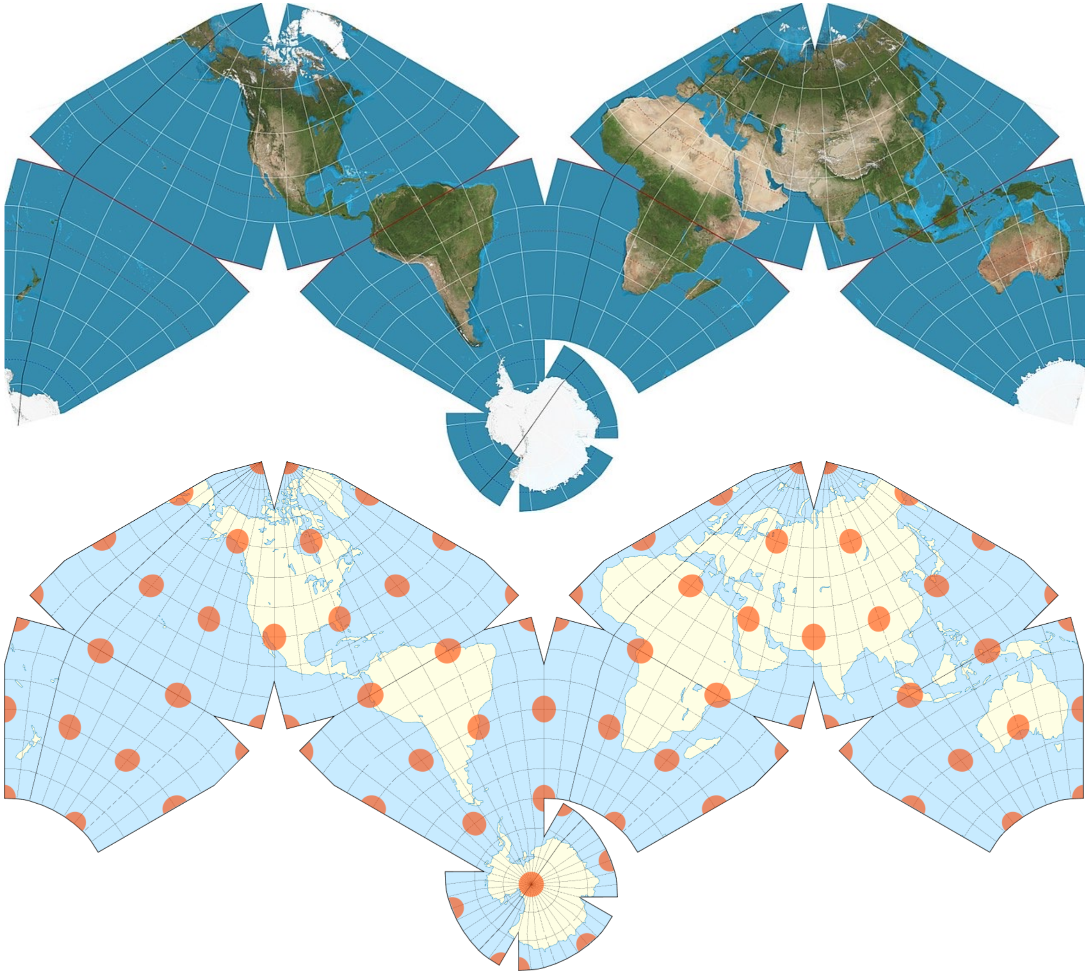




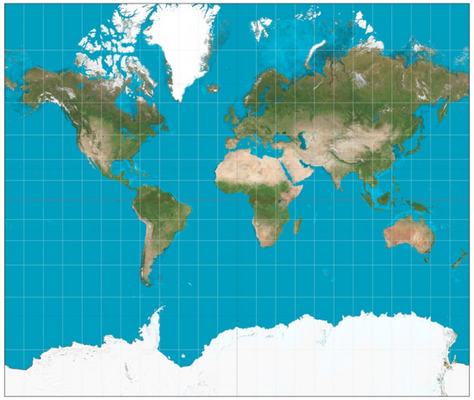
Johann Heinrich Lambert (1772)

Mapping the earth

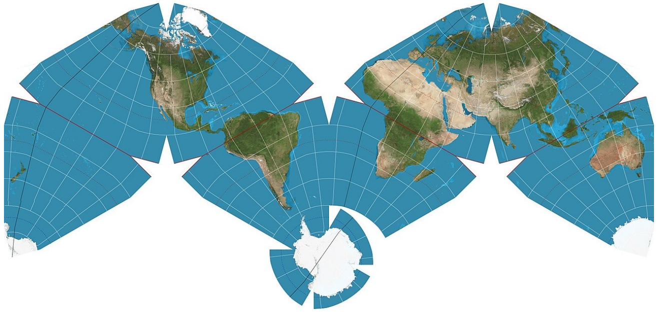
Cahill–Keyes polyhedral projection: compromise



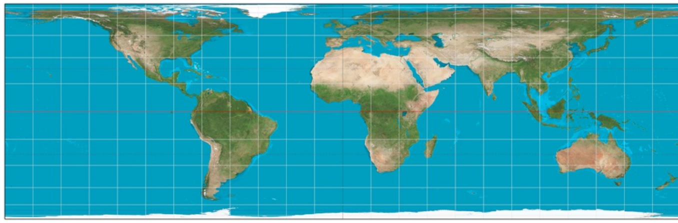
Different kinds of parameterization



Mercator
conformal



Cahill-Keyes



Lambert
Preserves area

Different kinds of Parameterization

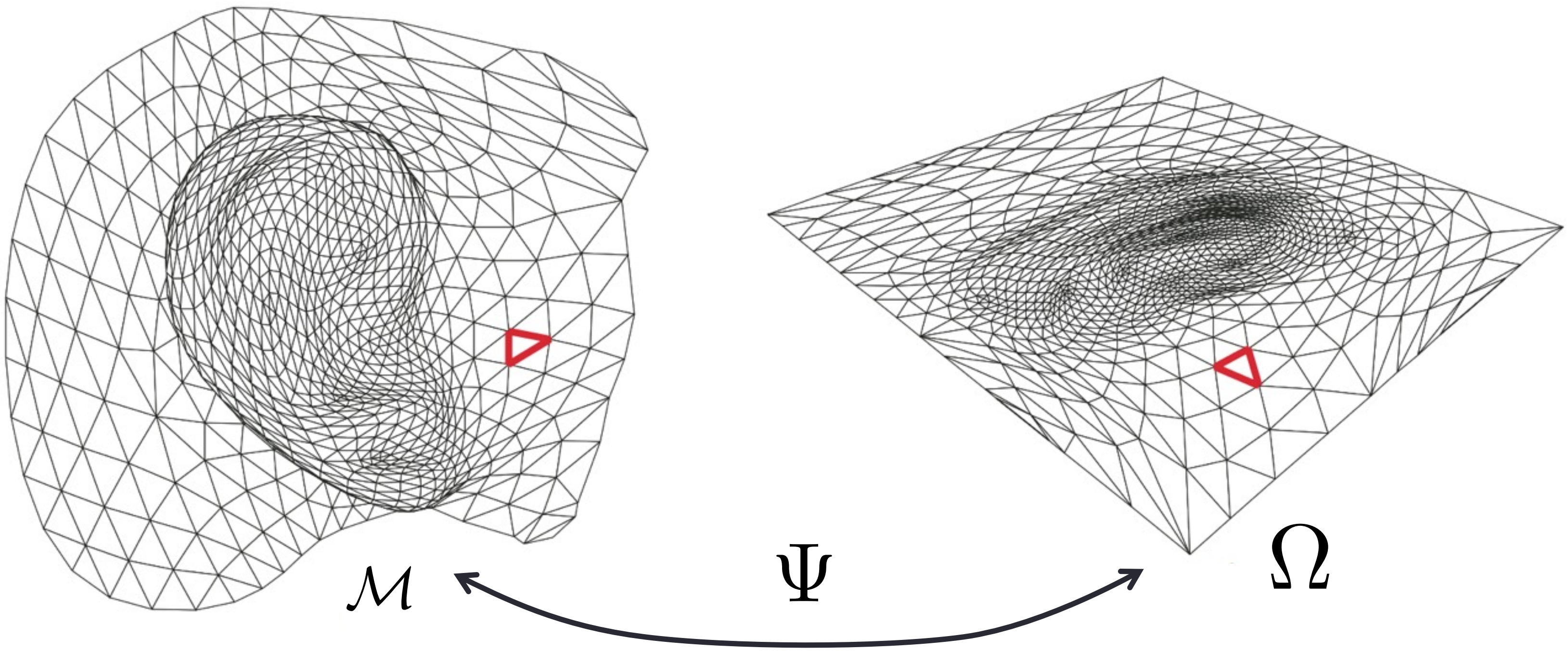
Various notions of distortion:

1. Equiareal: preserving areas
2. Conformal: preserving angles of intersections
3. Isometric: preserving geodesic distances

Theorem: Isometric = Conformal + Equiareal

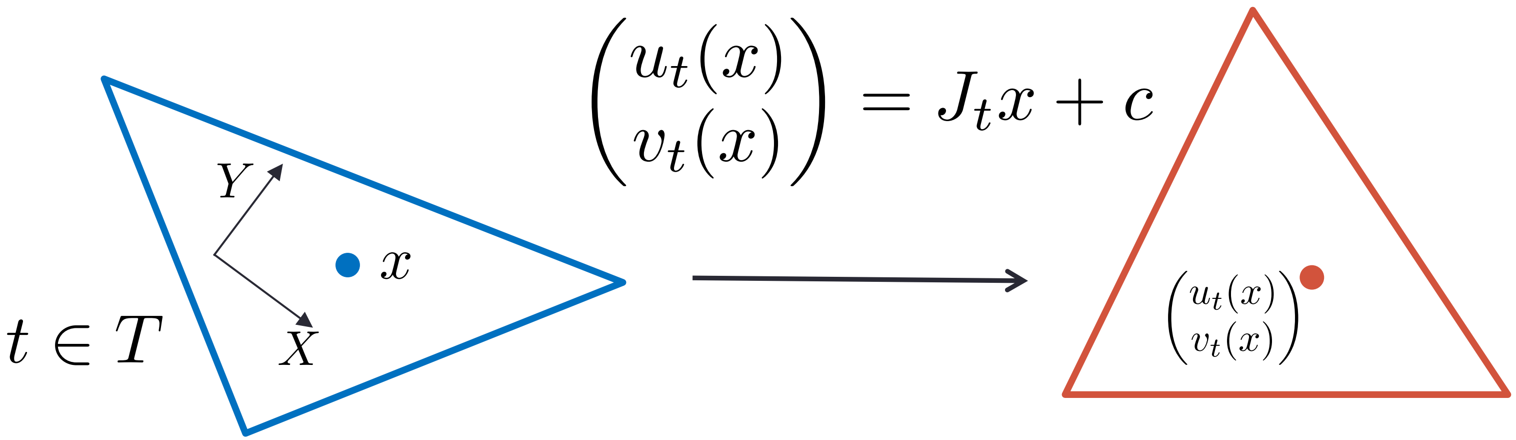
Parameterization Problem

Given a surface (mesh) $\mathcal{M}$ in R^3 and a planar domain Ω :
Find a bijective map $\Psi : \Omega \longleftrightarrow \mathcal{M}$.



Distortion of a Triangle

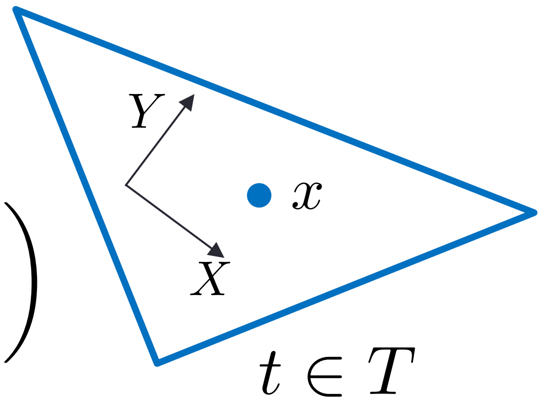
$$\begin{pmatrix} u_t(x) \\ v_t(x) \end{pmatrix} = J_t x + c$$



J_t : Jacobian of the transformation

Distortion energy: $E(f) := \sum_{t \in F} \mathrm{distortion}(J_t)$

Distortion Minimization



J_t : Jacobian of the transformation $x \mapsto \begin{pmatrix} u_t(x) \\ v_t(x) \end{pmatrix}$

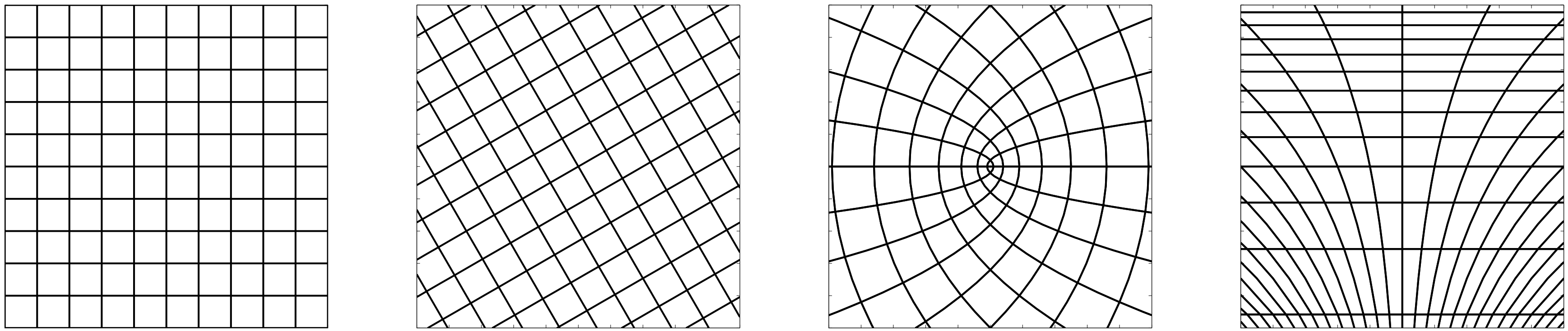
$$J_t = \begin{pmatrix} \frac{\partial u}{\partial X} & \frac{\partial v}{\partial X} \\ \frac{\partial u}{\partial Y} & \frac{\partial v}{\partial Y} \end{pmatrix} = \begin{pmatrix} \nabla u & \nabla v \end{pmatrix}$$

1. Isometric mapping: $J_t^\top J_t = I$
2. Conformal mapping: $\nabla v = n \times \nabla u$
3. Equiareal mapping: $\det J_t = 1$

Distortion energy can be non-linear and difficult to optimize for.

Distortion Minimization

1. Isometric mapping: $J_t^\top J_t = I$ (local rotation)
2. Conformal mapping: $\nabla v = n \times \nabla u$ (local rotation + scaling)
3. Equiareal mapping: $\det J_t = 1$ (same local area)

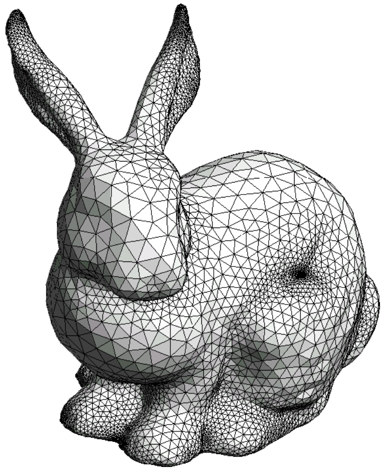


Initial grid | Isometric | Conformal | Equiareal

Distortion energy can be non-linear and difficult to optimize for.

Formalizing Parametrization

How do you solve this problem numerically:



Define a measure of distortion: $E(f) := \sum_{t \in F} \text{distortion}(J_t)$

Define a parametrization as the minimum of energy:

$$(u_{\text{opt}}, v_{\text{opt}}) = \underset{f=(u,v)}{\arg\min}\, E(f)$$

given boundary conditions

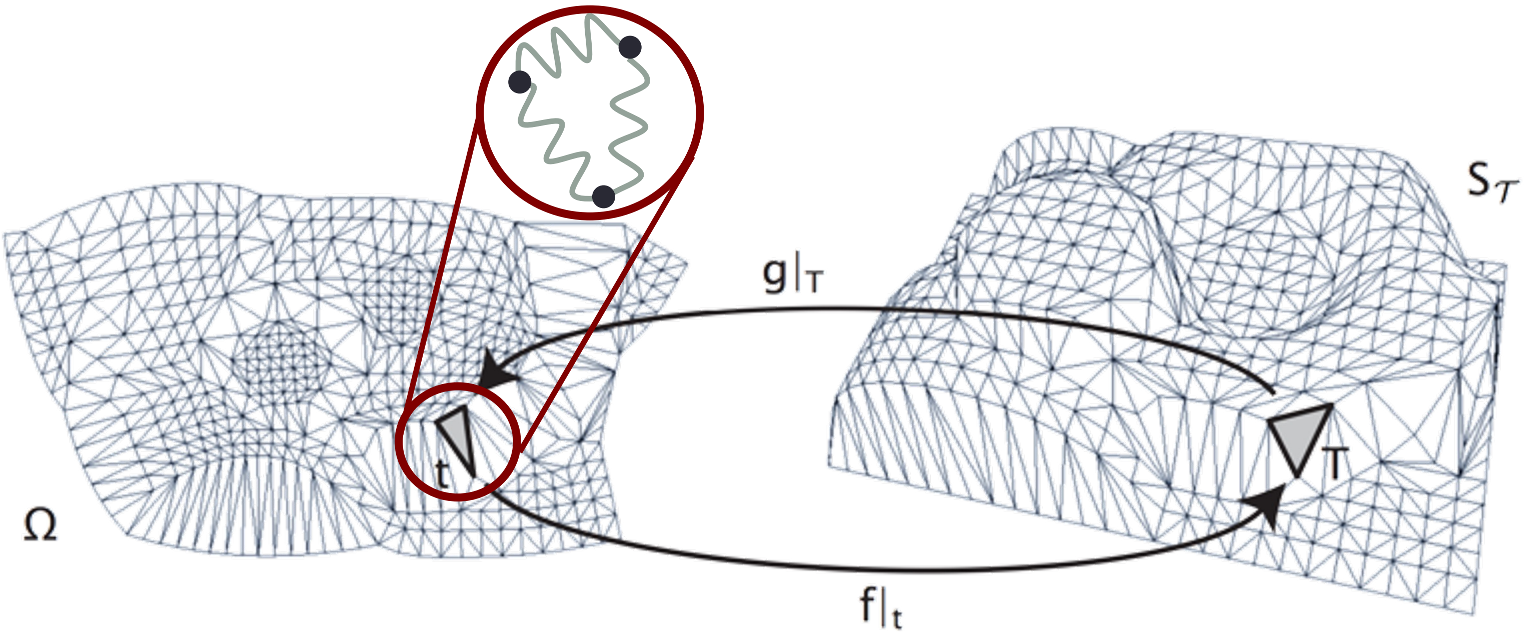
Parametrization with Fixed Boundary

- Can we compute a parametrization by solving a linear system
- Assume we know exactly where the boundary must go

Spring Model for Parameterization

Given a mesh (T, P) in 3D find a bijective mapping $g(\mathbf{p}_i) = \mathbf{u}_i$
given constraints: $g(\mathbf{b}_j) = \mathbf{u}_j$ for some $\{\mathbf{b}_j\}$

Model: imagine a **spring** at each edge of the mesh.
If the boundary is fixed, let the interior points find an **equilibrium**.

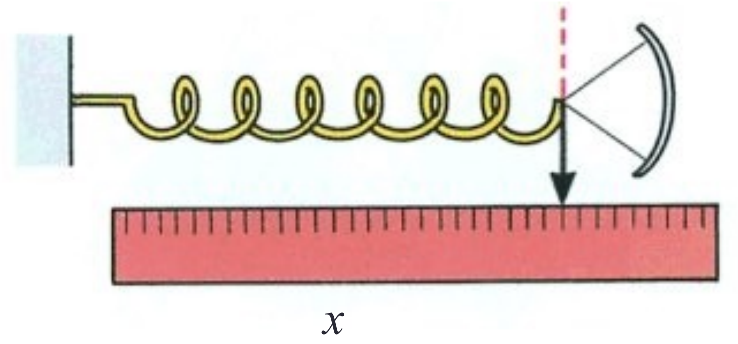


Spring Model for Parameterization

Recall: potential energy of a spring stretched by distance x:

$$E(x) = \frac{1}{2}kx^2$$

k: spring constant.



Spring Model for Parameterization

Given an embedding (parameterization) of a mesh, the potential energy of the whole system:

$$E = \sum_e \frac{1}{2} D_e \|\mathbf{u}_{e1} - \mathbf{u}_{e2}\|^2$$

$$= \frac{1}{2} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} \frac{1}{2} D_{ij} \|\mathbf{u}_i - \mathbf{u}_j\|^2$$

, $\mathcal{N}_i$ set of vertices adjacent to i

Where $D_e = D_{ij}$ is the spring constant of edge e between i and j

Goal: find the coordinates $\{\mathbf{u}_i\}$ that would minimize E.

Note: the boundary vertices prevent the degenerate solution.

Parameterization with Barycentric Coordinates

Finding the optimum of:

$$E = \frac{1}{2} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} \frac{1}{2} D_{ij} \|\mathbf{u}_i - \mathbf{u}_j\|^2$$

$$\frac{\partial E}{\partial \mathbf{u}_i} = 0 \Rightarrow \quad \sum_{j \in \mathcal{N}_i} D_{ij}(\mathbf{u}_i - \mathbf{u}_j) = 0$$

$$\Rightarrow \quad \mathbf{u}_i = \sum_{j \in \mathcal{N}_i} \lambda_{ij} \mathbf{u}_j, \text{ where } \lambda_{ij} = \frac{D_{ij}}{\sum_{j \in \mathcal{N}_i} D_{ij}}$$

I.e. each point $\mathbf{u}_i$ must be an **convex combination** of its neighbors.

Hence: barycentric coordinates.

Parameterization with Barycentric Coordinates

To find a minimizer of E in practice:

1. Fix the boundary points $\mathbf{b}_i, i \in \mathcal{B}$
2. Form linear equations

$$\begin{aligned} \mathbf{u}_i &= \mathbf{b}_i, & \text{if } i \in \mathcal{B} \\ \mathbf{u}_i - \sum_{j \in \mathcal{N}_i} \lambda_{ij} \mathbf{u}_j &= 0, & \text{if } i \notin \mathcal{B} \end{aligned}$$

3. Assemble into *two* linear systems (one for each coordinate):

$$LU = \bar{U}, \quad LV = \bar{V} \qquad L_{ij} = \begin{cases} 1 & \text{if } i = j \\ -\lambda_{ij} & \text{if } j \in \mathcal{N}_i, \ i \notin \mathcal{B} \\ 0 & \text{otherwise} \end{cases}$$

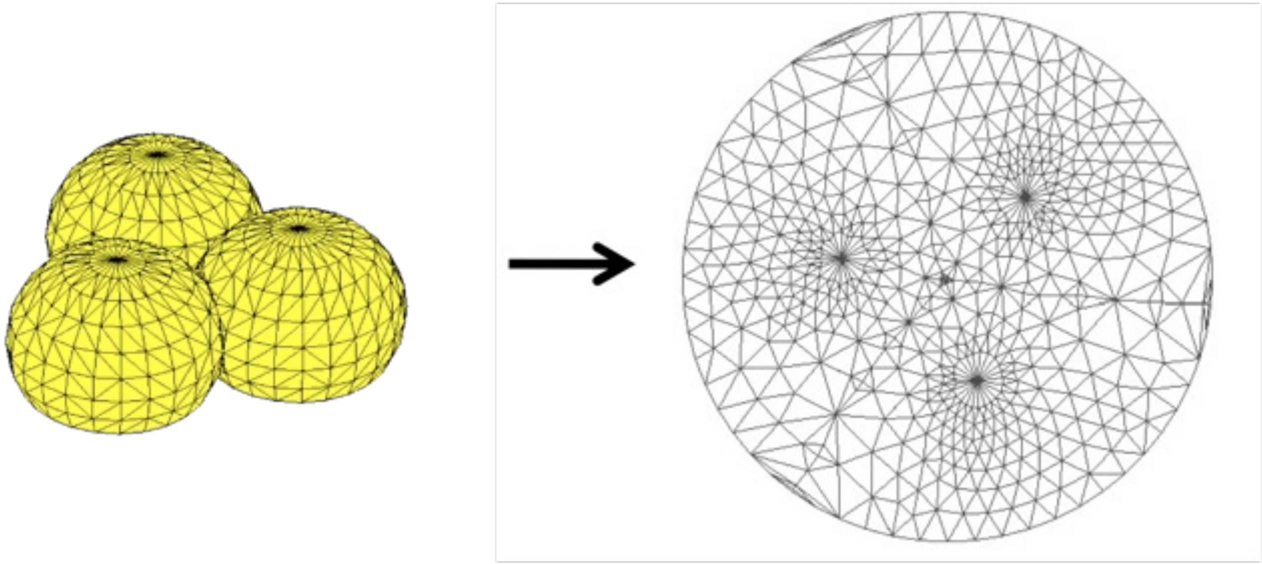
4. Solution of the linear system gives the coordinates: $\mathbf{u}_i = (u_i, v_i)$
   Note: system is very sparse, can solve efficiently.

Parameterization with Barycentric Coordinates

Does this work?

Tutte's spring-embedding theorem:

Every **barycentric** drawing of a 3-connected planar graph (triangle mesh) is a valid embedding if its boundary is **convex**.



Laplace Operator

Laplace operator in Euclidean space:

Given a function $f : \mathbb{R}^n \to \mathbb{R}^n$

$$\nabla \cdot (\nabla f) = \Delta f = \sum_{i}^{n} \frac{\partial^2}{\partial x_i^2} f$$

$$\Delta f = Lf$$

Laplacian Matrix

Our system of equations (forgetting about boundary):

$$\mathbf{u}_i = \sum_{j \in \mathcal{N}_i} \lambda_{ij} \mathbf{u}_j , \ \text{ where } \lambda_{ij} = \frac{D_{ij}}{\sum_{j \in \mathcal{N}_i} D_{ij}}$$

$$LU = 0 \qquad L_{ij} = \begin{cases} 1 & \text{if } i = j \\ -\lambda_{ij} & \text{if } j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases}$$

L is not symmetric

Alternatively, if we write it as:

$$\mathbf{u}_i \sum_{j \in \mathcal{N}_i} D_{ij} = \sum_{j \in \mathcal{N}_i} D_{ij} \mathbf{u}_j$$

We get:

$$LU = 0 \qquad L_{ij} = \begin{cases} \sum_{k \in \mathcal{N}_i} D_{ij} & \text{if } i = j \\ -D_{ij} & \text{if } j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases}$$

L is symmetric

Parameterization with Barycentric Coordinates

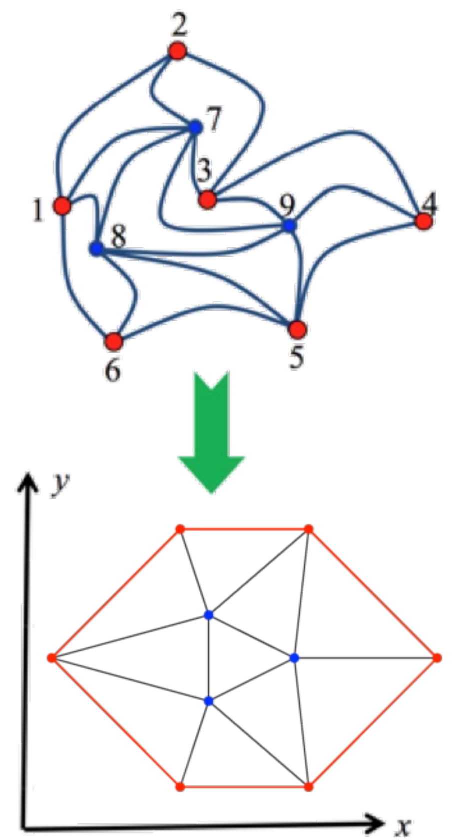
Example:

Uniform weights:

$$D_{ij} = 1$$

Laplacian Matrix

$$W = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & -5 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & -5 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & -5 \end{pmatrix} \quad b_x = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 3 \\ 2 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad b_y = \begin{pmatrix} 2 \\ 3 \\ 3 \\ 2 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$



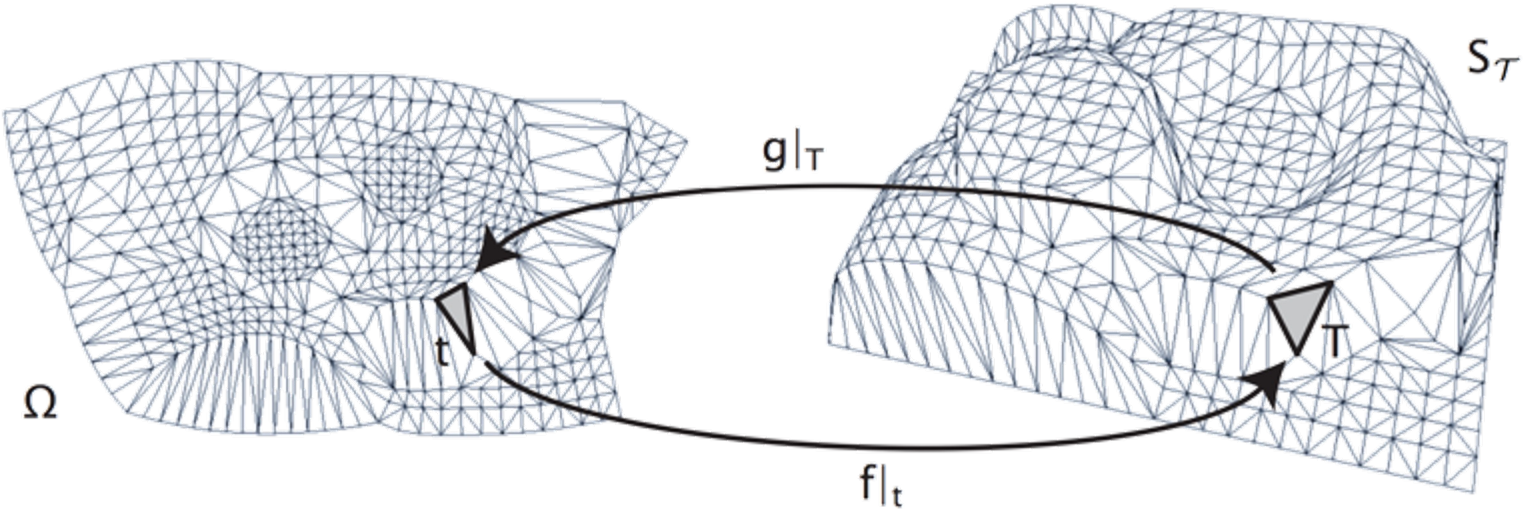
Parameterization with Barycentric Coordinates

Linear Reproduction:

- If the mesh is already **planar** we want to recover the original coordinates.

Problem:

- Uniform weights do not achieve linear reproduction
- Same for weights proportional to distances.



Parameterization with Barycentric Coordinates

Linear Reproduction:

- If the mesh is already **planar** we want to recover the original coordinates.

Problem:

- Uniform weights do not achieve linear reproduction
- Same for weights proportional to distances.

Solution:

- If the weights are **barycentric** with respect to **original points**:

$$\mathbf{p}_i = \sum_{j \in \mathcal{N}_i} \lambda_{ij} \mathbf{p}_j , \quad \sum_{j \in \mathcal{N}_i} \lambda_{ij} = 1$$

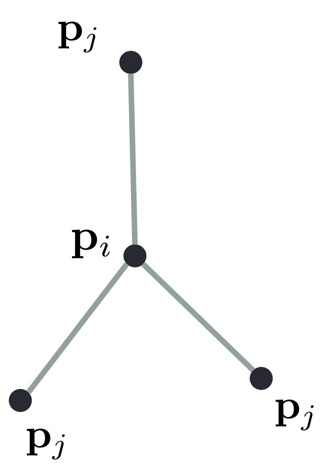
The resulting system will recover the planar coordinates.

Parameterization with Barycentric Coordinates

Solution:

- Barycentric coordinates with respect to original points:

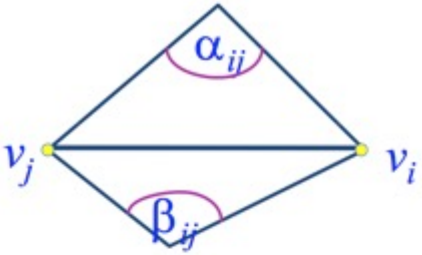
$$\mathbf{p}_i = \sum_{j \in \mathcal{N}_i} \lambda_{ij} \mathbf{p}_j, \quad \sum_{j \in \mathcal{N}_i} \lambda_{ij} = 1$$



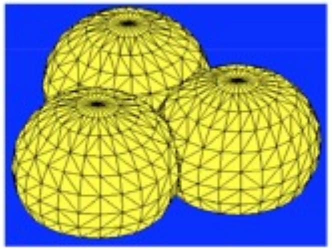
- If a point $\mathbf{p}_i$ has 3 neighbors, then the barycentric coordinates are **unique.**
- For more than 3 neighbors, many possible choices exist.

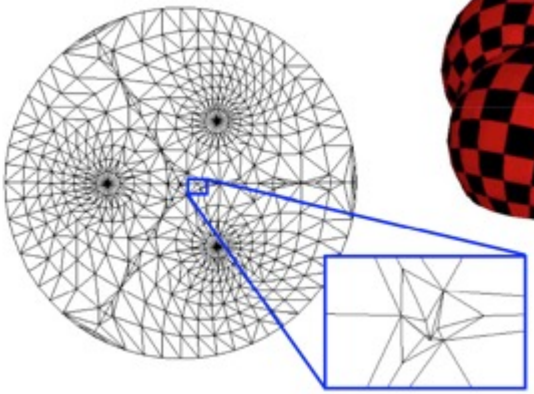
Barycentric (cotangent) weights

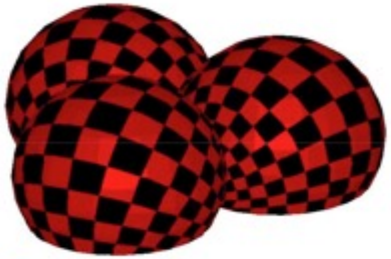
$$D_{ij} = \frac{\cot(\alpha_{ij}) + \cot(\beta_{ij})}{2}$$



- Weights can be negative – not always valid
- Weights depend only on angles - close to conformal
- 2D reproducible



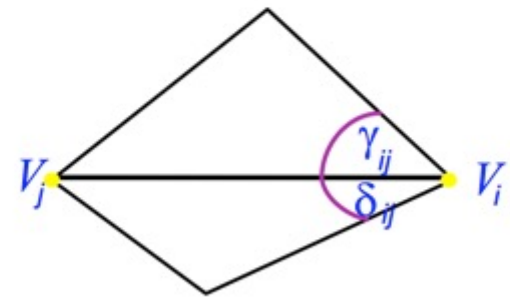




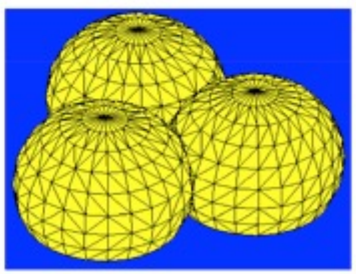
Mesh Parameterization Methods and Their Applications, 2006

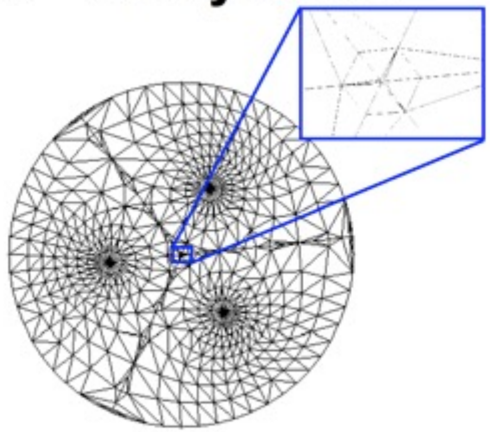
Barycentric (mean value) weights

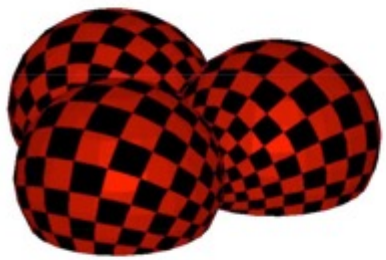
$$D_{ij} = \frac{\tan(\gamma_{ij}/2) + \tan(\delta_{ij}/2)}{2\,||V_i - V_j||}$$



- Result visually similar to harmonic
- No negative weights – **always** valid
- 2D reproducible







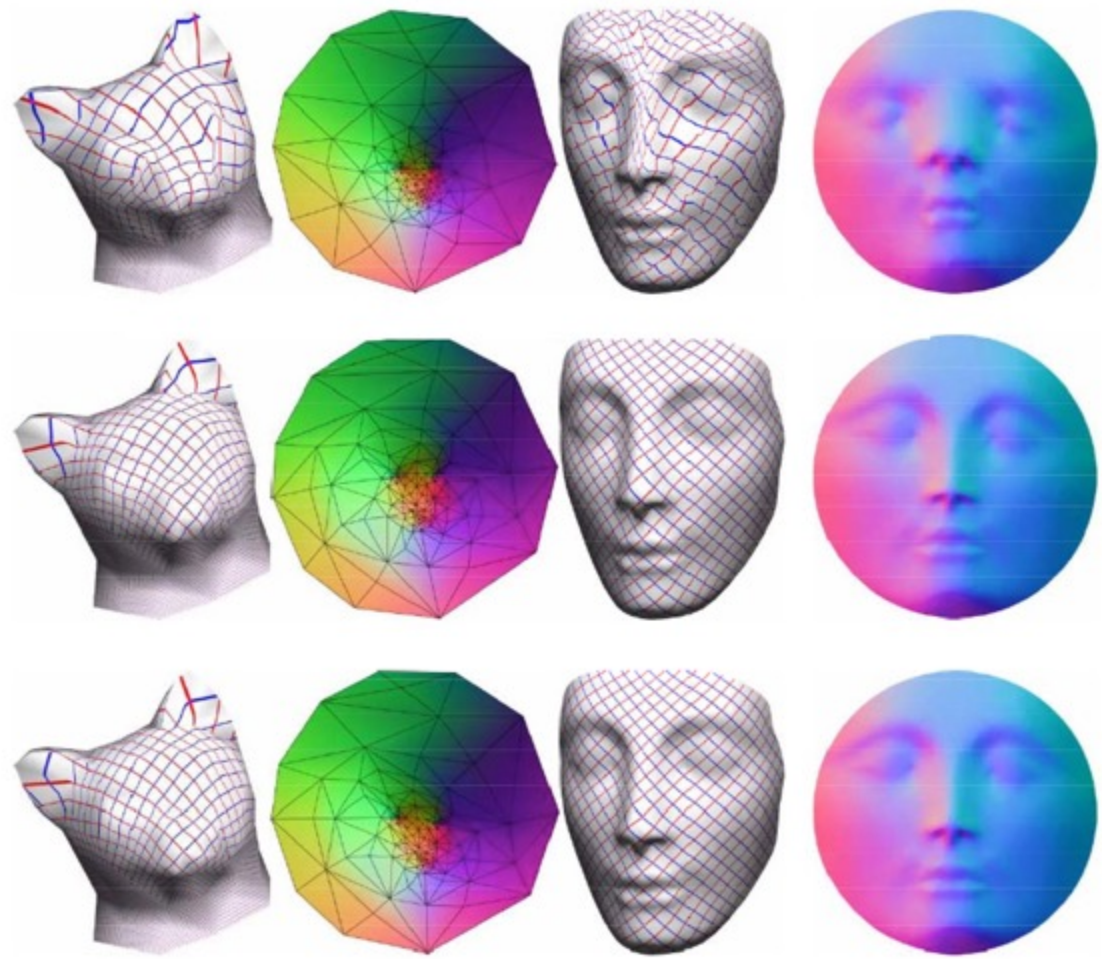
Mesh Parameterization Methods and Their Applications, 2006

Barycentric Coordinates

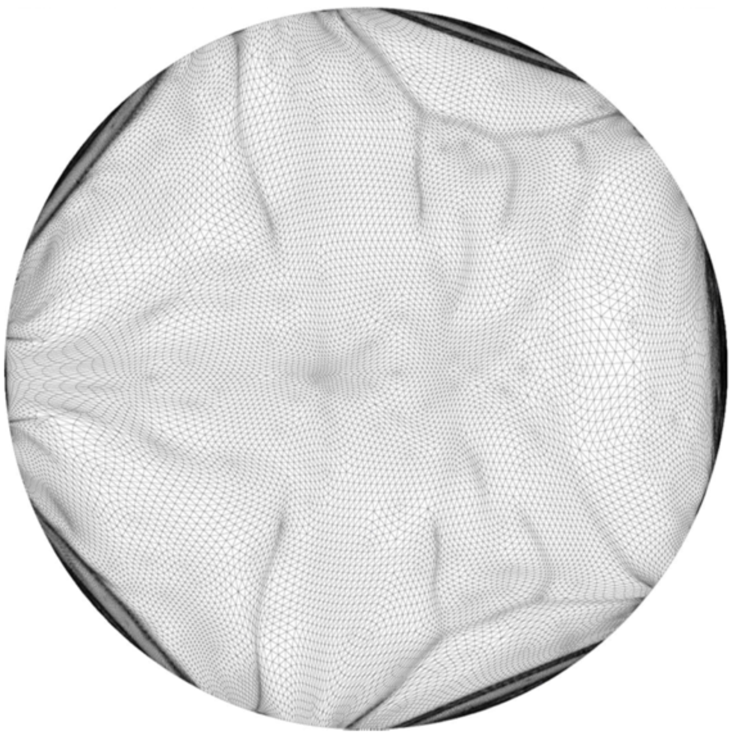
uniform

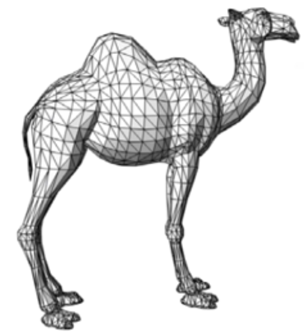
harmonic

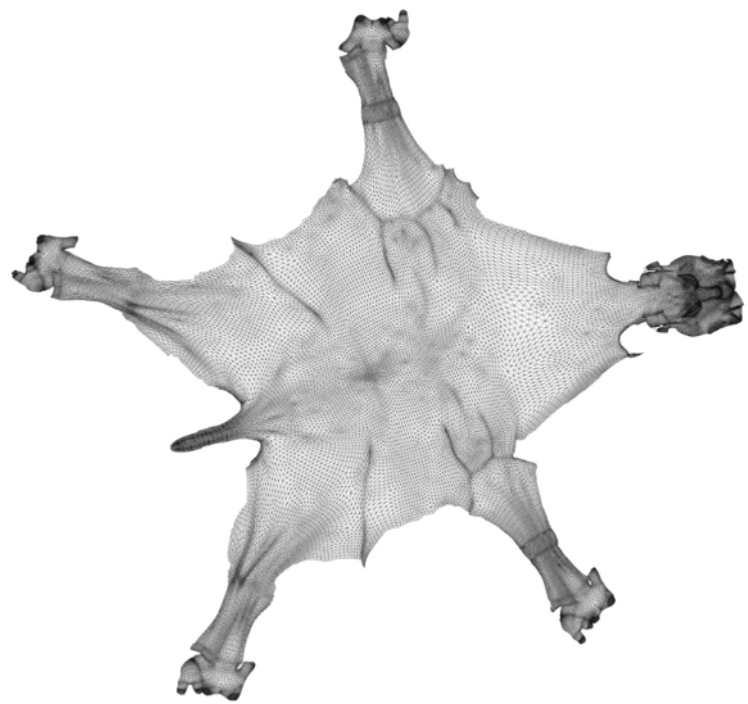
mean-value



Fixed vs Free boundary



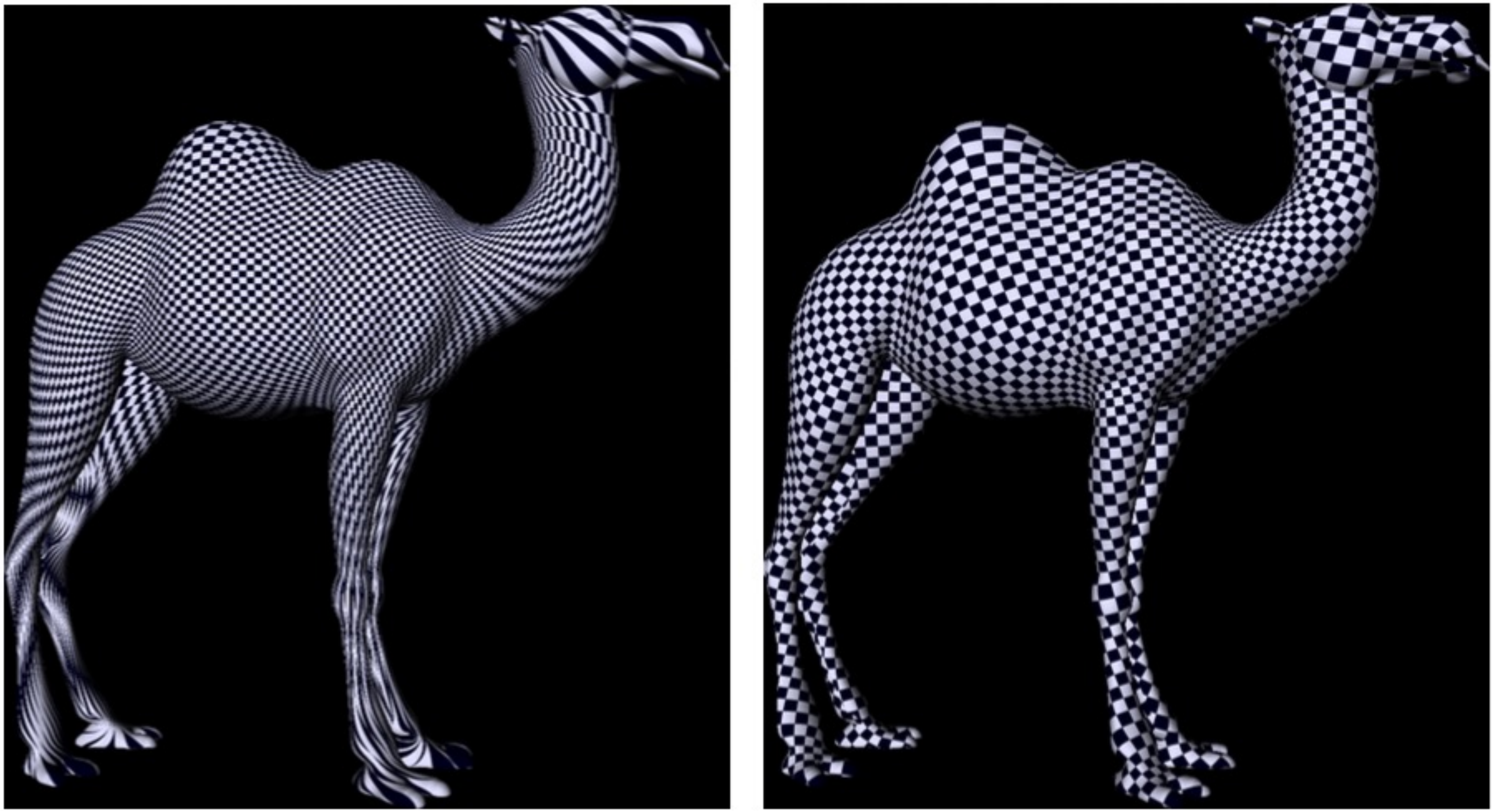




Tutte embedding

images by Mirela Ben-Chen

Fixed vs Free boundary



images by Mirela Ben-Chen

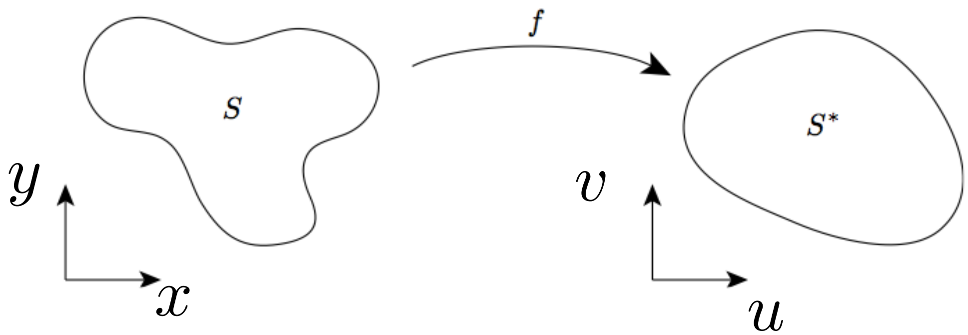
Conformal Mappings

J_t : Jacobian of the transformation $x \mapsto \begin{pmatrix} u_t(x) \\ v_t(x) \end{pmatrix}$

Conformal mapping: $\nabla v = n \times \nabla u$

Riemann Mapping Theorem:

Any surface topologically equivalent to a disk, **can be** conformally mapped to a unit disk.



Conformal Mappings

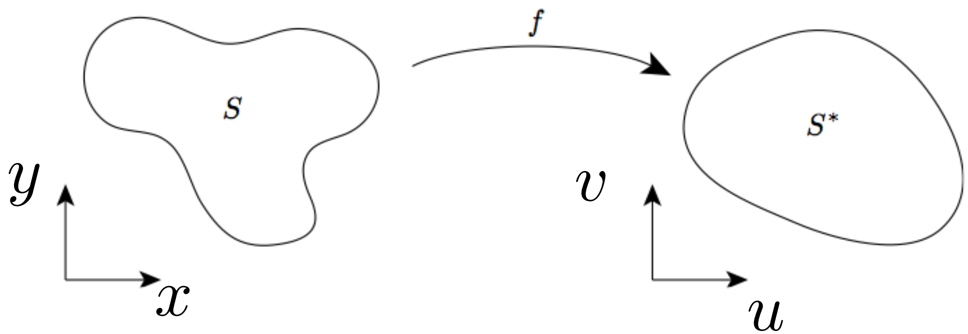
J_t : Jacobian of the transformation $x \mapsto \begin{pmatrix} u_t(x) \\ v_t(x) \end{pmatrix}$

Conformal mapping: $\nabla v = n \times \nabla u$

Riemann Mapping Theorem:

Any surface topologically equivalent to a disk, **can be** conformally mapped to a unit disk.

$$\begin{aligned} \Delta v &= \mathrm{div} \nabla v \\ &= \mathrm{div}(n \times \nabla u) \\ &= \mathrm{curl} \nabla u \\ &= 0 \end{aligned}$$



$$\Delta u = 0$$

Conformal Mappings

J_t : Jacobian of the transformation $x \mapsto \begin{pmatrix} u_t(x) \\ v_t(x) \end{pmatrix}$

Conformal mapping: $\nabla v = n \times \nabla u$

Riemann Mapping Theorem:

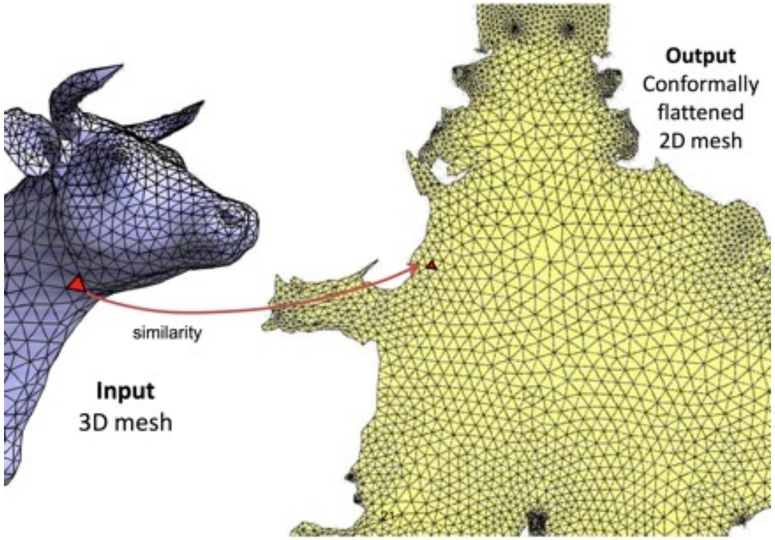
Any surface topologically equivalent to a disk, **can be** conformally mapped to a unit disk.

If a map $S \rightarrow$ (u,v) is conformal then both u and v are harmonic:

$$\Delta_S u = 0 \;\; \Delta_S v = 0$$

Δ_S : Laplacian on S.

Like Tutte embeddings: each point must be an **convex combination** of its neighbors.



Conformal Free Boundary Method

J_t : Jacobian of the transformation $x \mapsto \begin{pmatrix} u_t(x) \\ v_t(x) \end{pmatrix}$

Conformal mapping: $\nabla v = n \times \nabla u$

Solve in a least-squares sense:

$$E_C(u,v) = \sum_{t \in T} A_t \|\nabla v - n \times \nabla u\|_t^2$$, A_t area of triangle t

$$= u^\top W u + v^\top W v - \sum_{ij \text{ at bnd}} (u_i v_j - u_j v_i)$$

W Cotangent matrix

Spectral Conformal Parametrization, 2008

Conformal Free Boundary Method

$$\min_{u,v} E_C(u,v) = \begin{pmatrix} u \\ v \end{pmatrix}^\top \begin{pmatrix} W & M \\ M^\top & W \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$ Subject to: $$\|u\|^2 + \|v\|^2 = 1$$

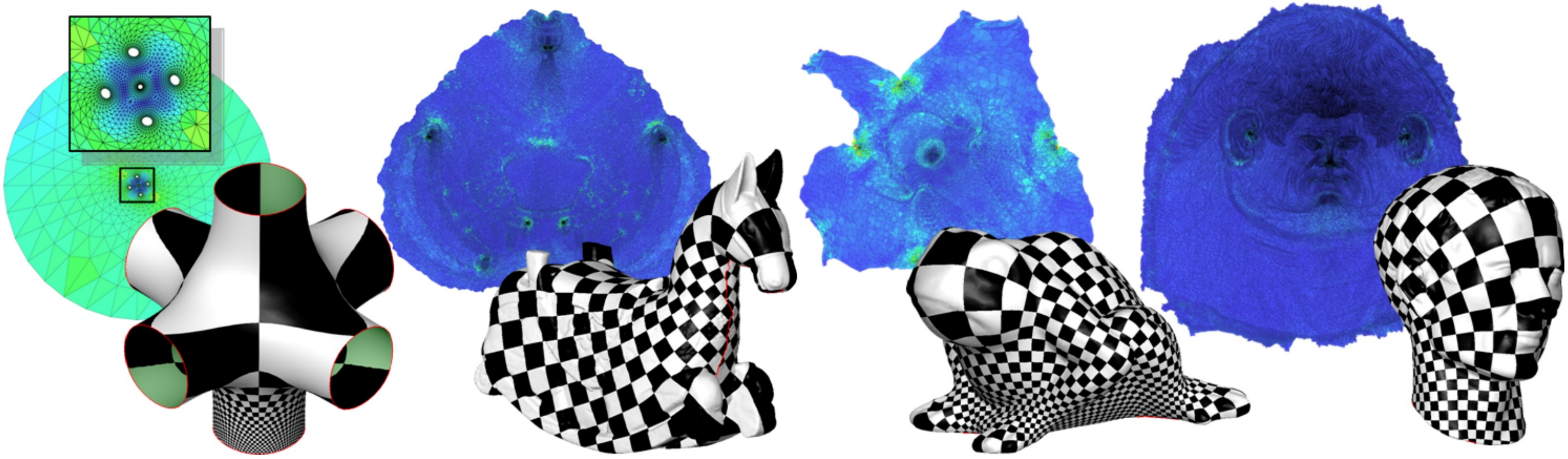
Equivalent to the eigenvalue problem:

$$\begin{pmatrix} W & M \\ M^\top & W \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} A & 0 \\ 0 & A \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$ with A the area matrix

Optimal solution: eigenfunction associated to the **third** smallest eigenvalue

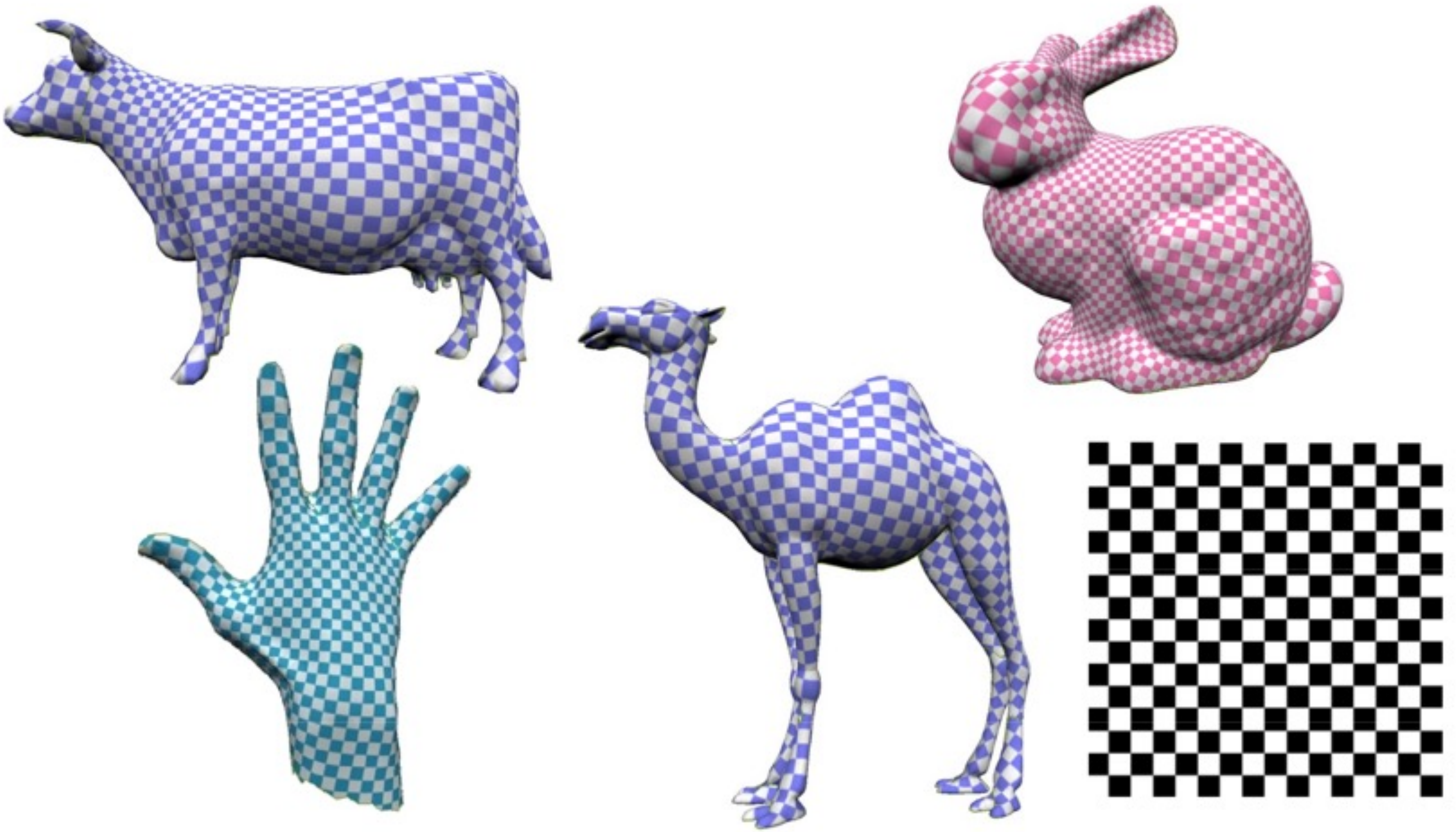
The first eigenfunctions are constant

Conformal Free Boundary Method

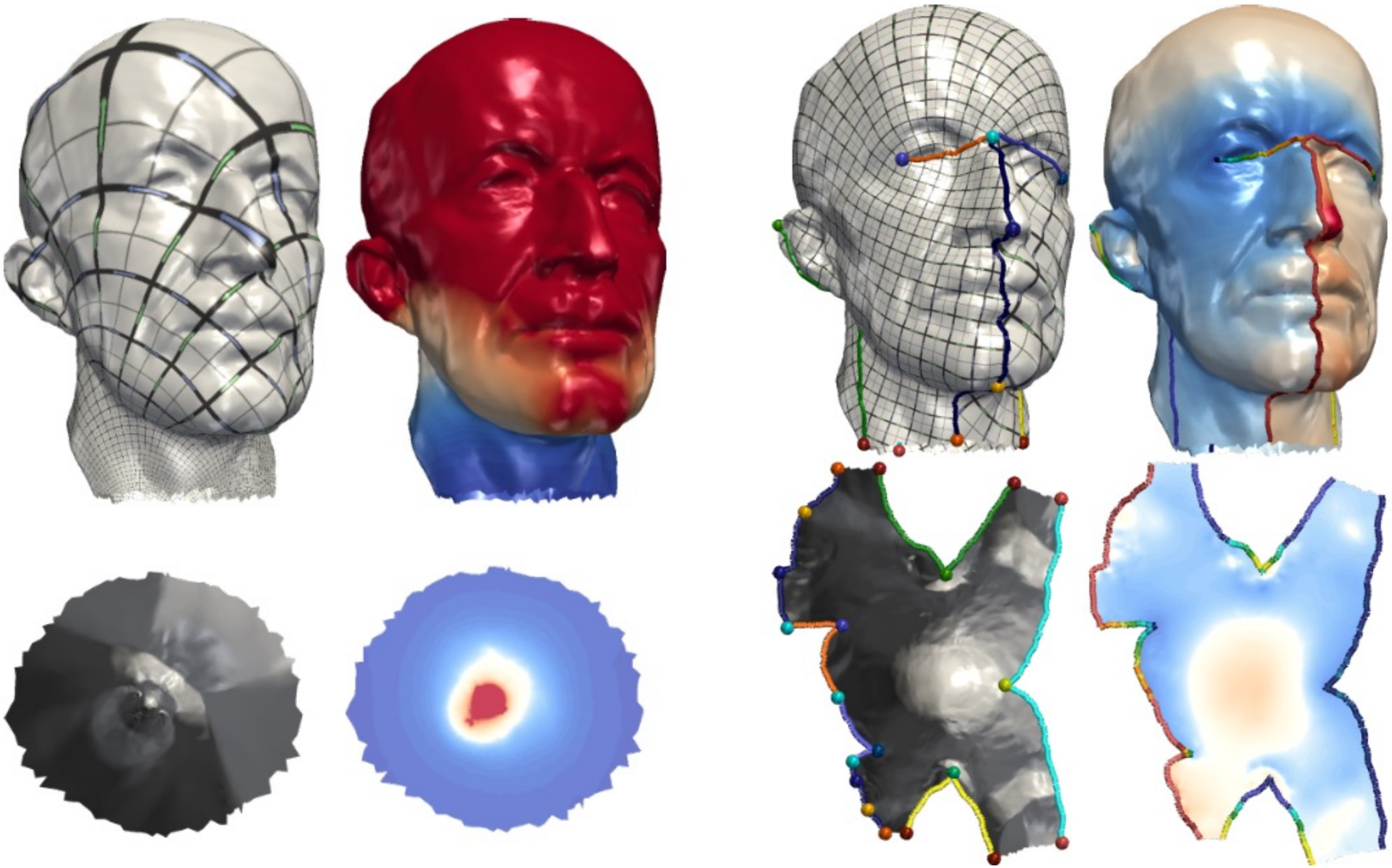


Spectral Conformal Parametrization, 2008

Conformal Mappings



Reducing Distortion - More Cuts



Going Further: Curvature Prescription

- **Parametrization:** find a deformation to a surface with zero Gaussian curvature
- Gather Gaussian curvature into “cone points”
  - Target curvature is not 0 everywhere
- Cuts must pass through cone points
- Position of the cuts has no impact on the distortion

Going Further: Curvature Prescription

Adding more and more cone singularities…

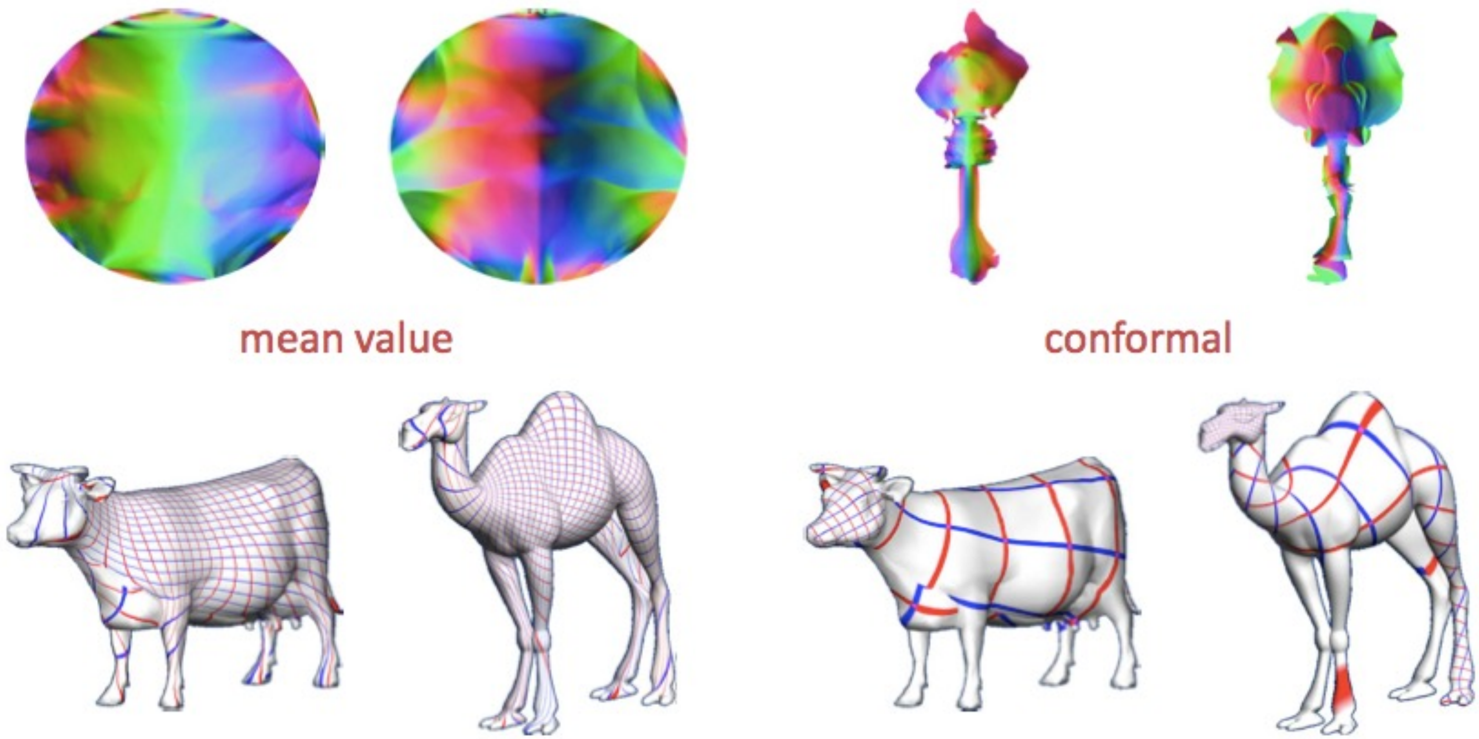
(Texture courtesy NASA Earth Observatory), from Soliman et al. 2018

Free boundary methods …

- Solve for the (u,v) coordinates
  - MIPS [Hormann et al., 2000]
  - Stretch optimization [Sander et al., 2001]
  - LSCM (conformal, linear) [Levy et al., 2002]
  - DCP (conformal, linear) [Desbrun et al., 2002]
- Solve for the angles of the map (conformal)
  - ABF [Sheffer et al., 2001], ABF++ [Sheffer et al., 2004]
  - LinABF (linear) [Zayer et al., 2007]
- Solve for the edge lengths of the map by prescribing curvature
  - Circle patterns [Kharevych et al., 2006]
  - CPMS (linear) [Ben-Chen et al., 2008]
  - CETM [Springborn et al., 2008]
- Balance area/conformality
  - ARAP [Liu et al., 2008]
- More…

Some results

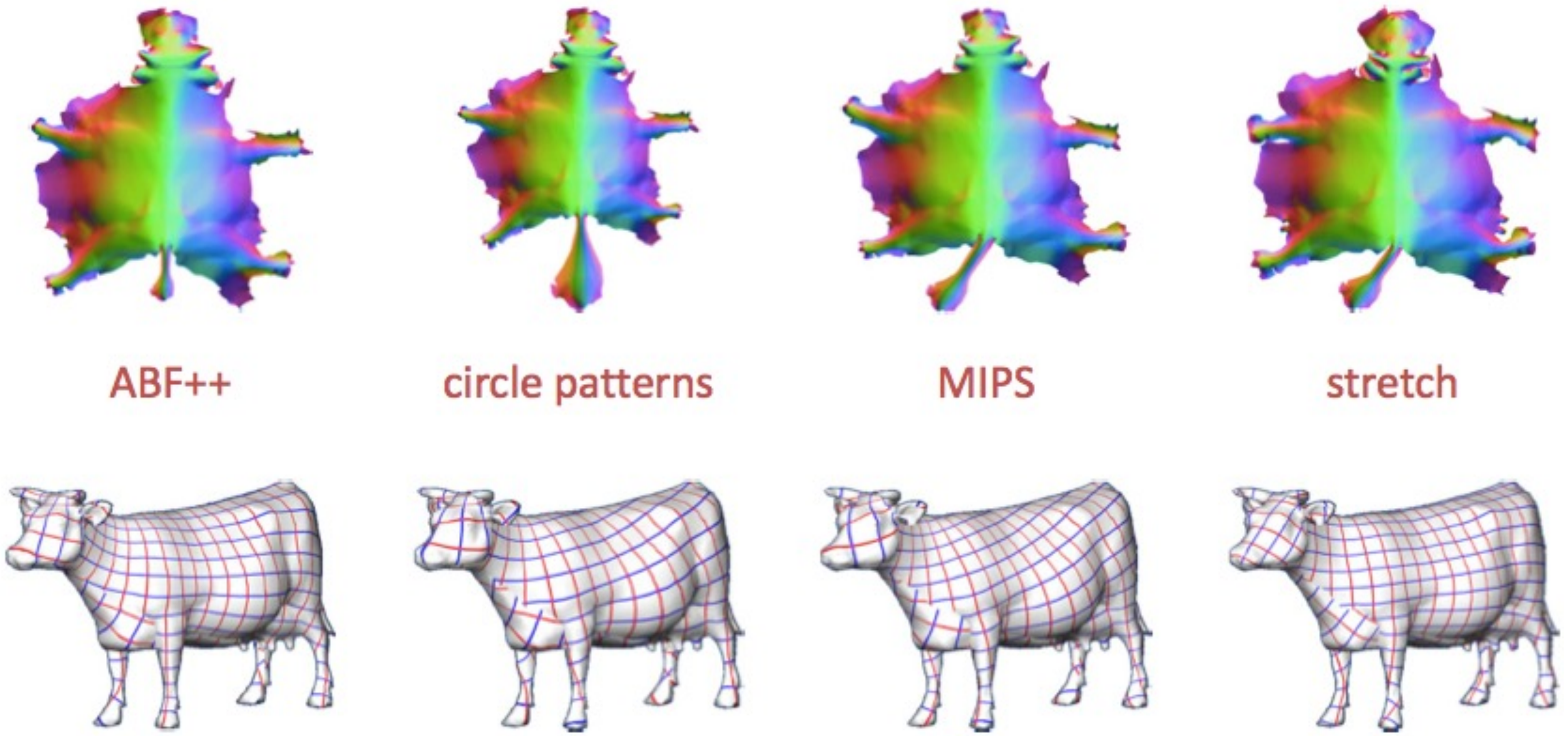
Linear Methods:



Purely linear methods can cause a very significant distortion.

Some results

Non-linear Methods:



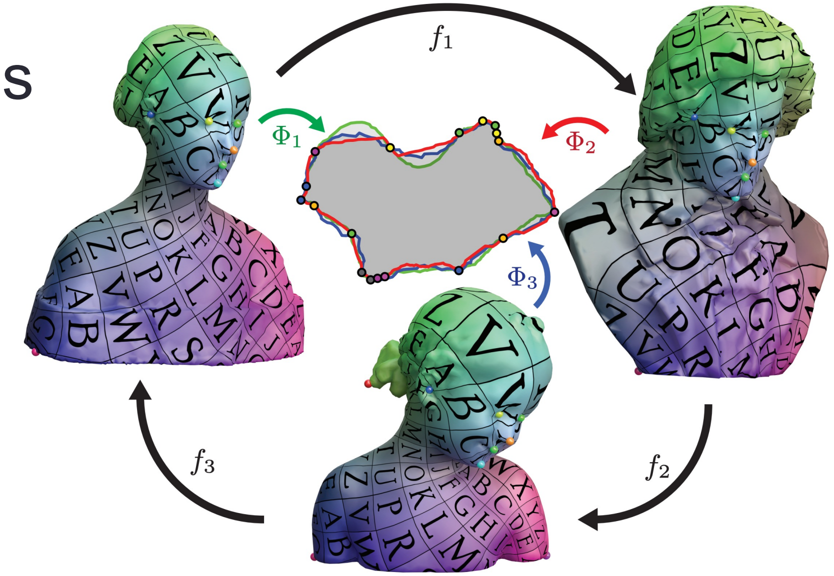
Conclusions

Surface parameterization:

- No perfect mapping method
- A very large number of techniques exists
- Conformal model:
  - Nice theoretical properties
  - Leads to a simple (linear) system of equations
  - Closely related to the Poisson equation and Laplacian operator
- More general methods
  - Can get smaller distortion using non-linear optimization
  - Very difficult to guarantee bijectivity in general

Cross Parameterization for Continuous **Maps**

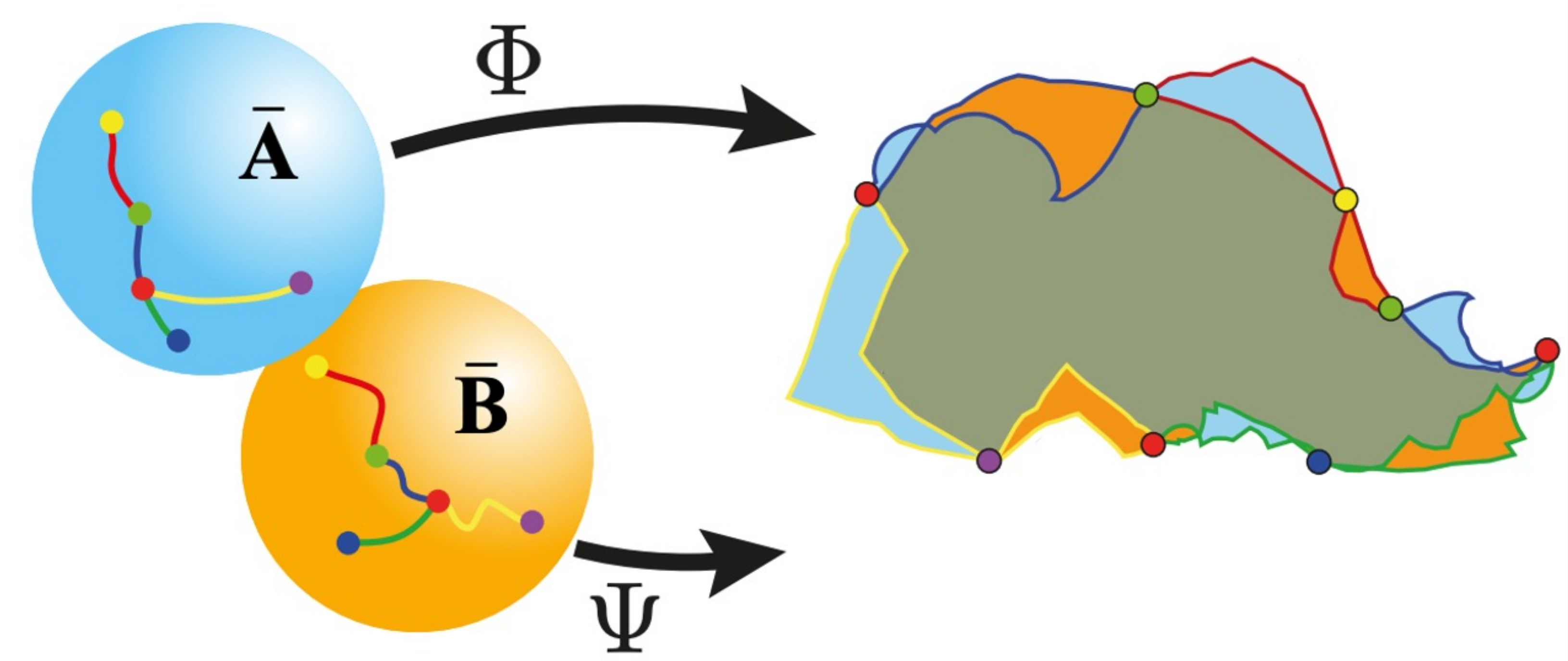
- Topological obstruction to the computation of maps
- Triangle mesh parametrization
  - Tutte embedding
  - Conformal mappings
  - And more…
- Computing correspondences



Parametrization for Correspondences

Input: a set constrained points and cut graph

Where to cut so that the parametrization of A and B are the same?



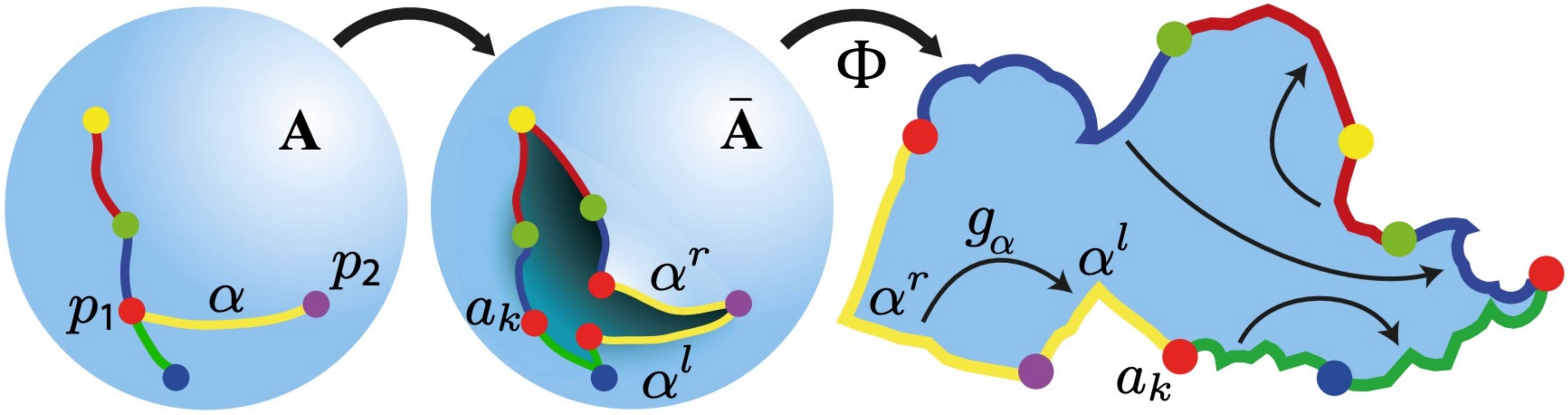
Aigerman et al. 2015

Parametrization for Correspondences

Seamless parametrization: set constrained points and cut graph, add global linear constraints on duplicated edges

Affine transition functions:

$$\begin{aligned} g: \quad \mathbb{R}^2 &\to \mathbb{R}^2 \\ x &\mapsto \begin{pmatrix} a & b \\ -b & a \end{pmatrix} x + \tau \end{aligned}$$



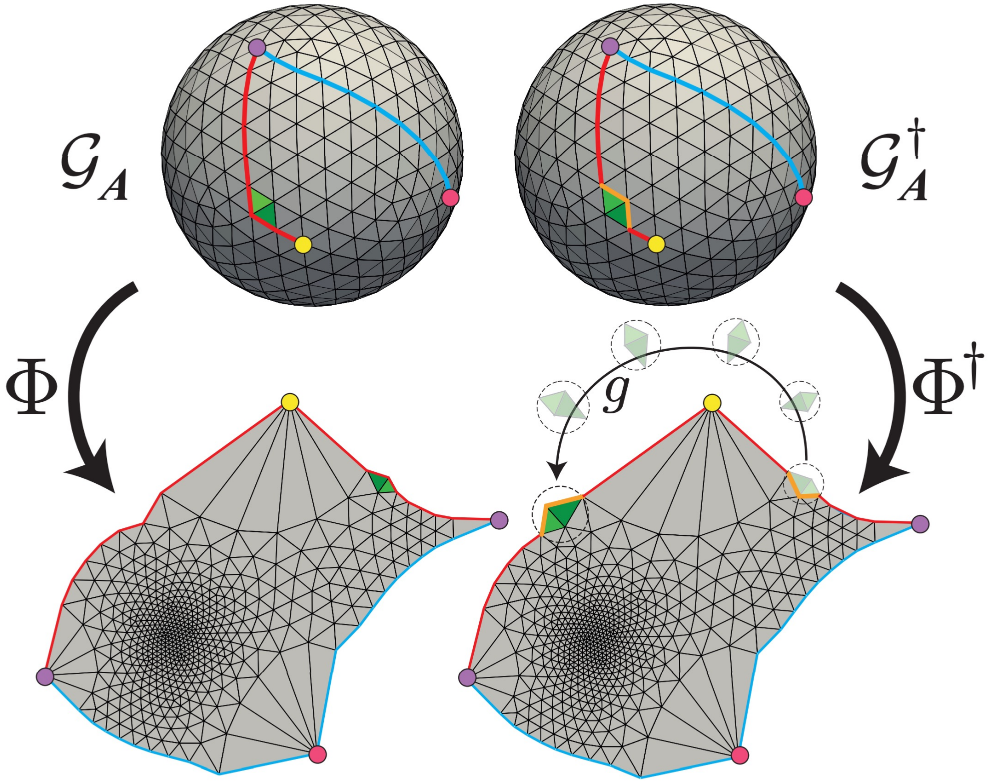
Aigerman et al. 2015

Parametrization for Correspondences

Seamless parametrization: set constrained points and cut graph, add global linear constraints on duplicated edges

Affine transition functions:

$$g: \mathbb{R}^2 \to \mathbb{R}^2$$
$$x \mapsto \begin{pmatrix} a & b \\ -b & a \end{pmatrix} x + \tau$$



Cuts become « invisible »

Aigerman et al. 2015

Parametrization for Correspondences

In practice:

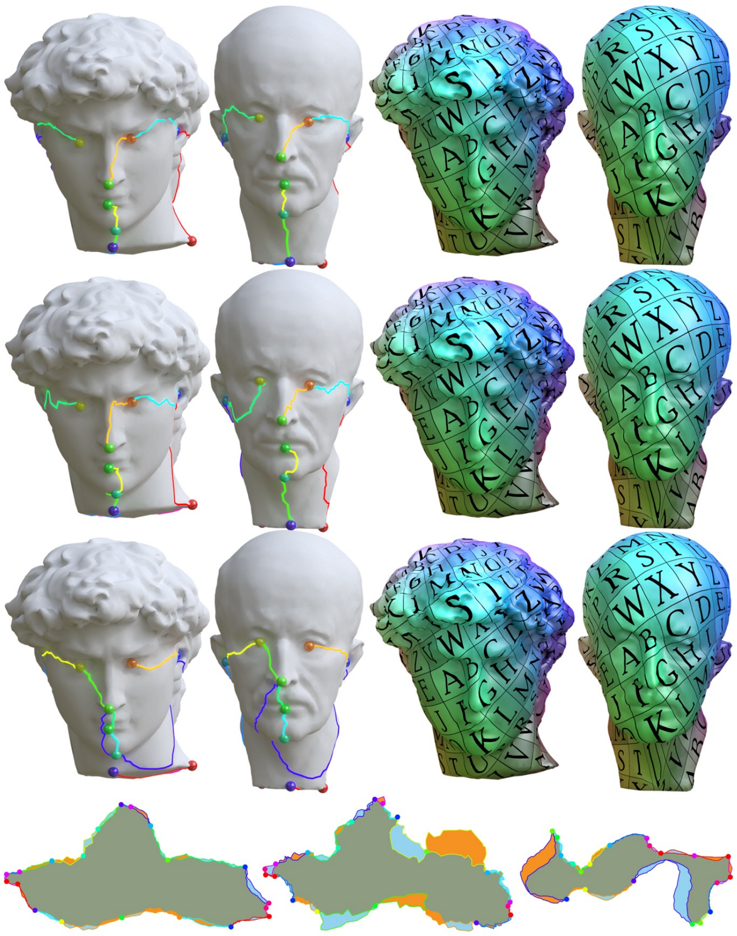
1. Find the duplicated boundary vertices $i_r, i_l \in \mathcal{B}$
2. Fix the constrained points $\mathbf{b}_i, i \in \mathcal{C}$
3. Solve the linear system of equations:

$$
\begin{aligned}
&\mathbf{u}_i = \mathbf{b}_i, && \text{if } i \in \mathcal{C} \\
&\mathbf{u}_{i_r} - \mathbf{u}_{j_r} = g_\alpha(\mathbf{u}_{i_l} - \mathbf{u}_{j_l}), && \text{if } i, j \in \mathcal{B} \\
&\mathbf{u}_i - \sum_{j \in \mathcal{N}_i} \lambda_{ij} \mathbf{u}_j = 0, && \text{if } i \notin \mathcal{B}
\end{aligned}
$$

4. Solution of the linear system gives the coordinates: $\mathbf{u}_i = (u_i, v_i)$

Parametrization for Correspondences

Same cut graphs, same mapping.

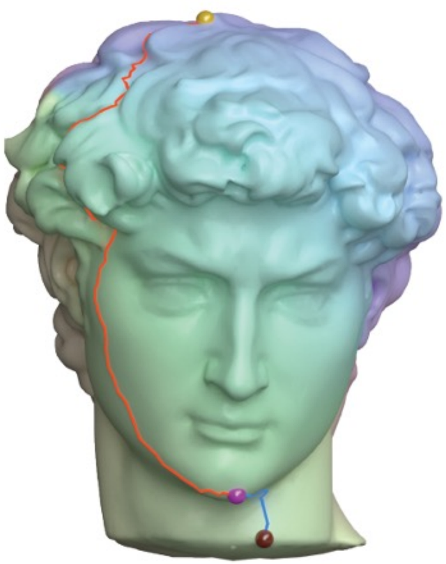


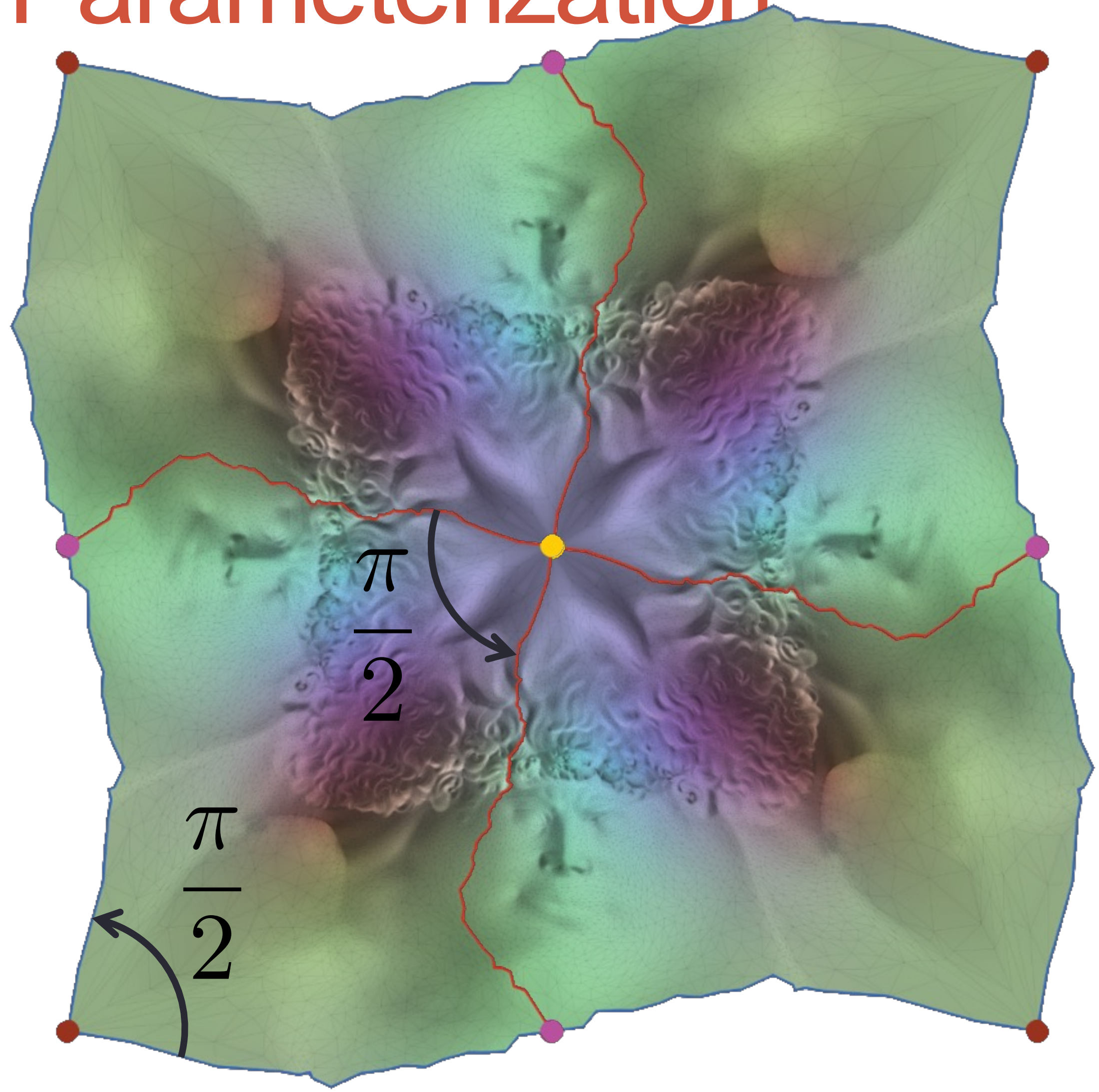
Aigerman et al. 2015

Parametrization domain lack structure!

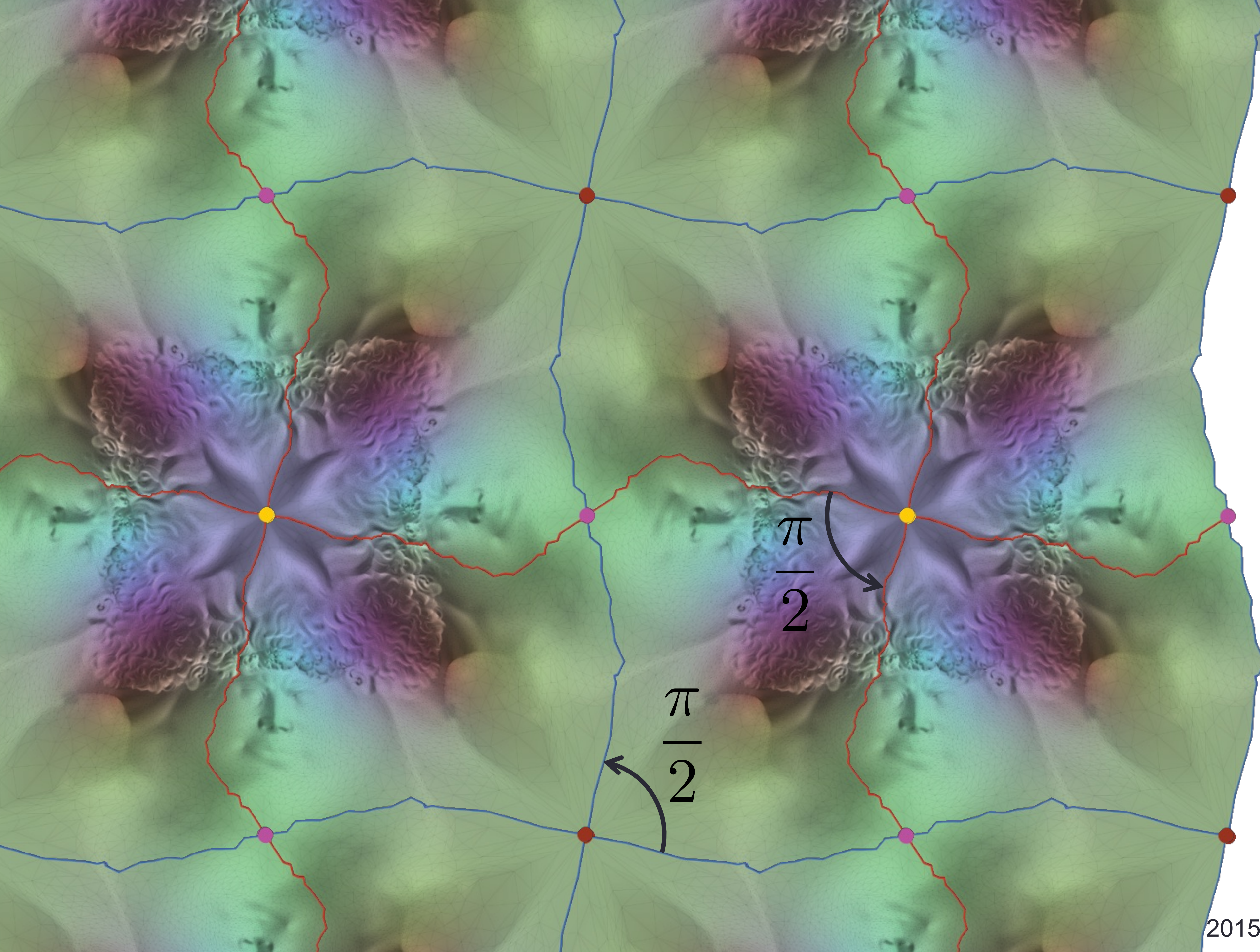
Toric Seamless Parameterization

- Three point cuts
- Rotation constraints on cuts
- Spring distortion
- Tiles the entire space





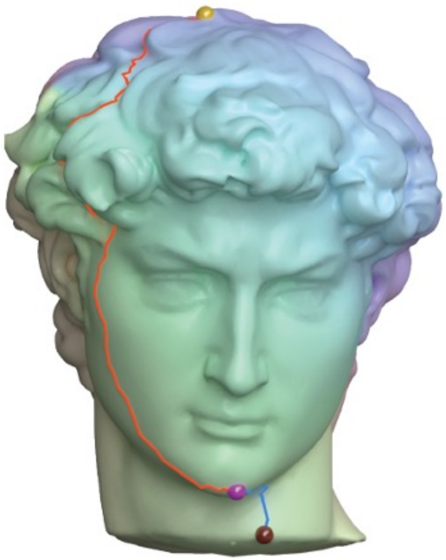
Orbifold Tutte Embeddings, Aigerman et al. 2015

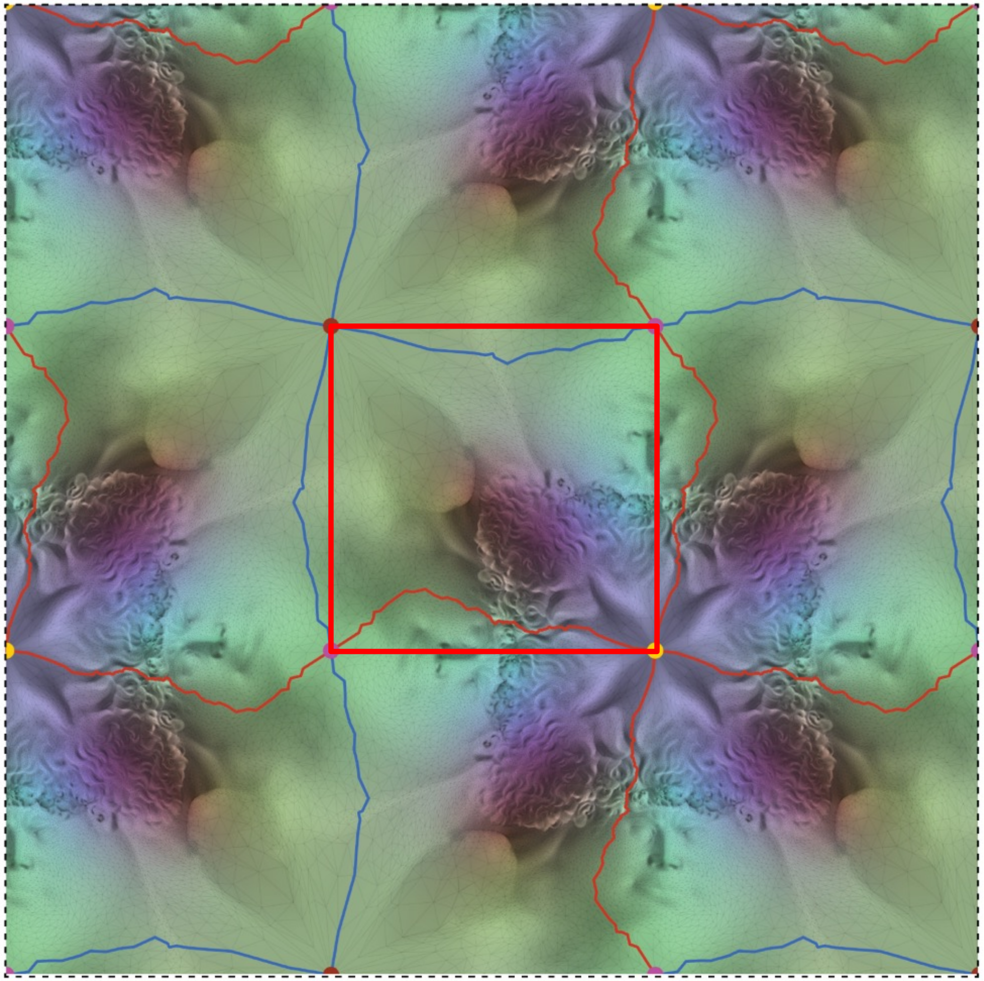


2015

Toric Parameterization

- Three point cuts
- Rotation constraints on cuts
- Spring or LSCM distortion
- Tiles the entire space



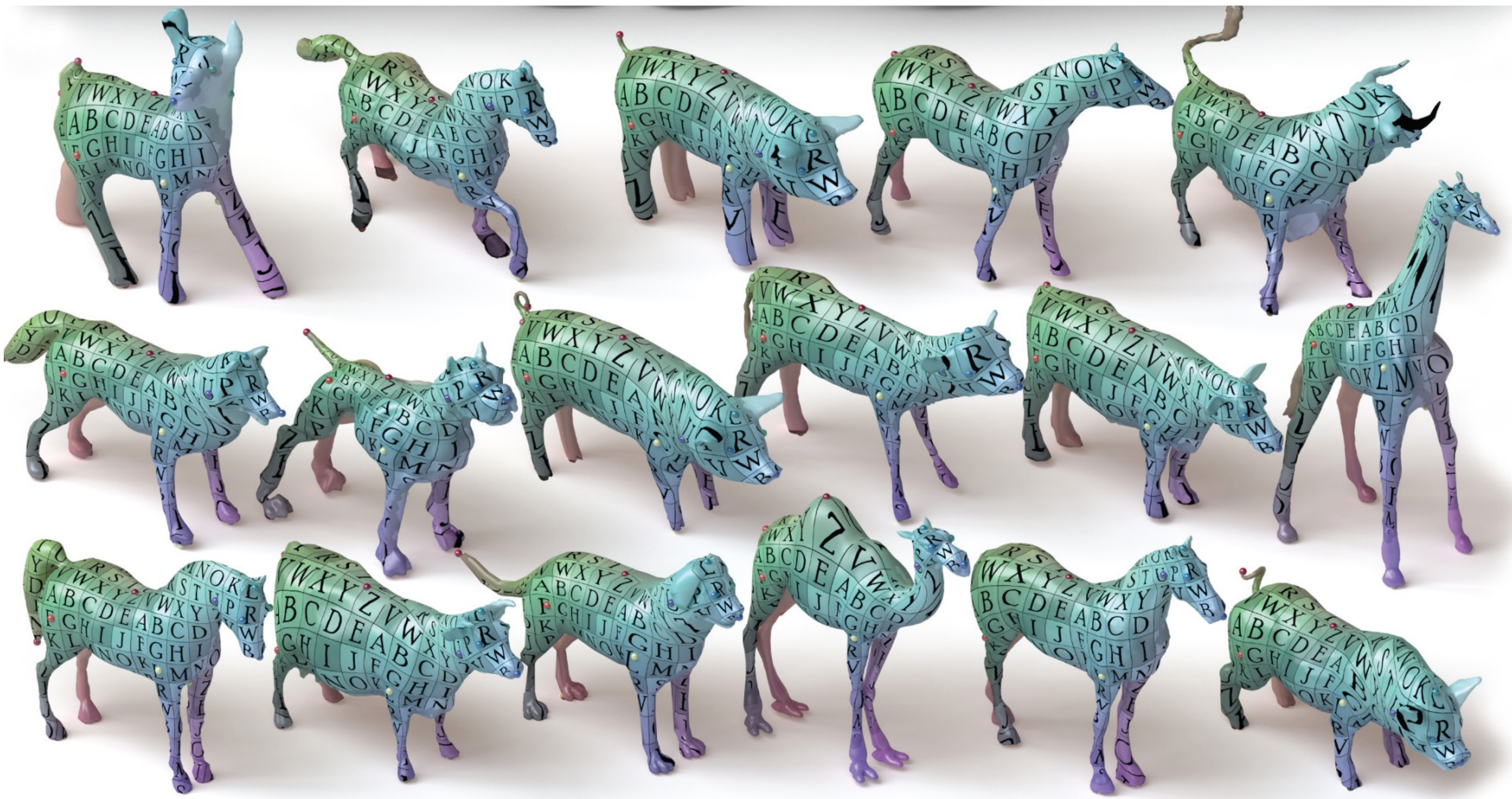


Cuts are invisible!

Orbifold Tutte Embeddings, Aigerman et al. 2015

Toric Parameterization

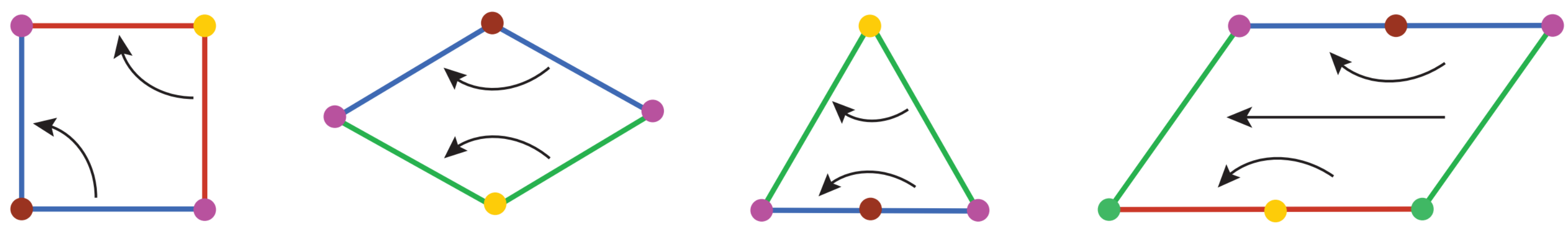
- Compute continuous maps between surfaces from few constraints



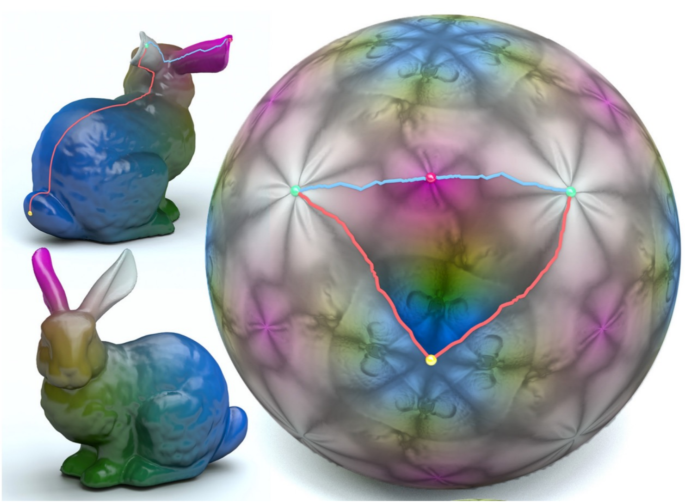
Orbifold Tutte Embeddings, Aigerman et al. 2017

Toric Parameterization

- Different space tiling
- Different parametric spaces



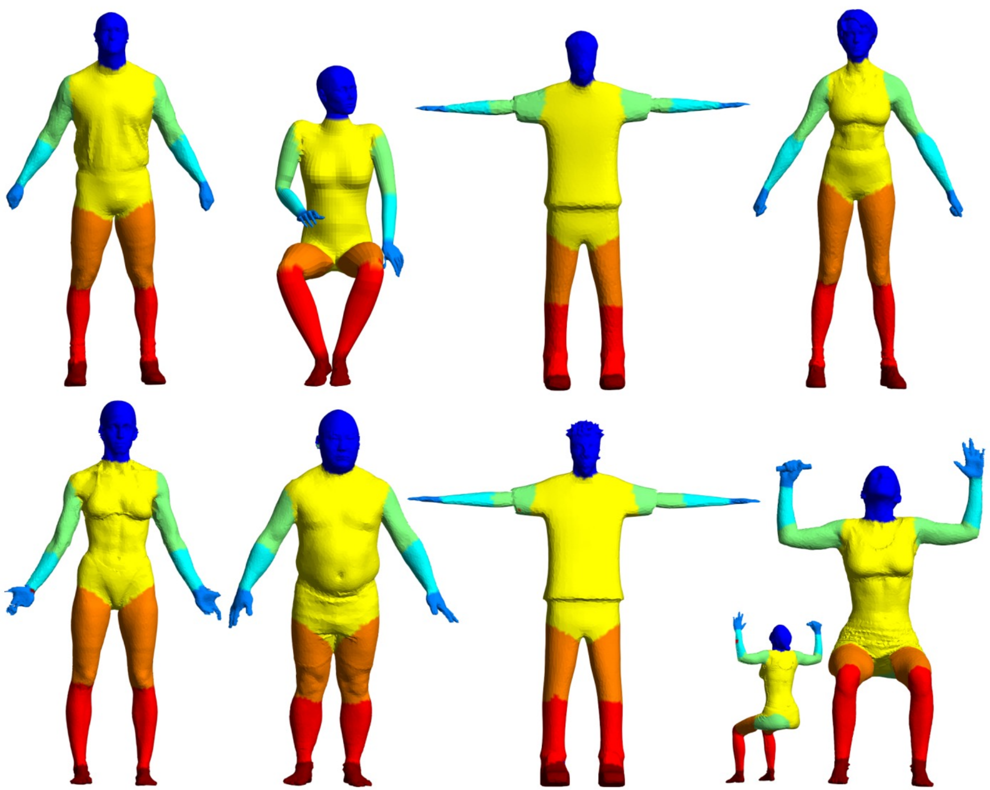
$\left\{\frac{\pi}{2}, \pi, \frac{\pi}{2}\right\}$ $\left\{\frac{2\pi}{3}, \frac{2\pi}{3}, \frac{2\pi}{3}\right\}$ $\left\{\pi, \frac{2\pi}{3}, \frac{2\pi}{6}\right\}$ $\left\{\pi, \pi, \pi, \pi\right\}$



Orbifold Tutte Embeddings, Aigerman et al. 2015
Spherical Orbifold Tutte Embeddings, Aigerman et al. 2017

Toric Parameterization

- Use image CNN on parametrization for segmentation



Convolutional Neural Networks on Surfaces via Seamless Toric Covers, Maron et al.

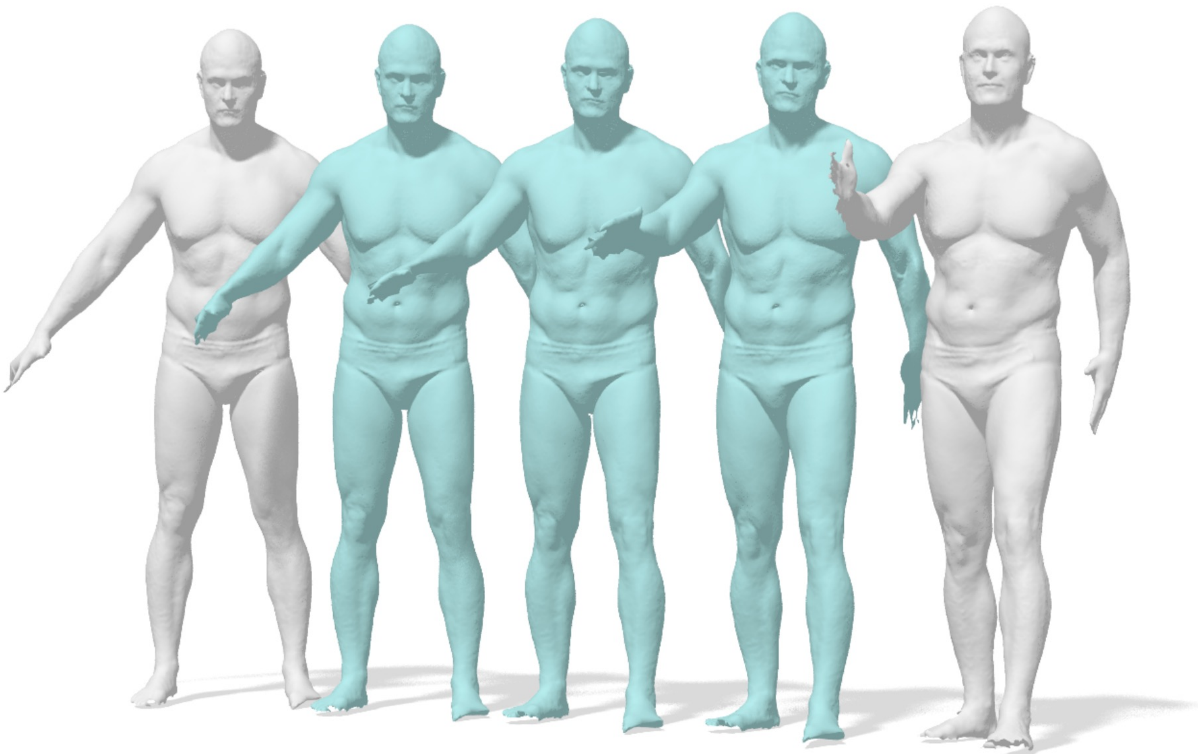
Conclusion

Toric parametrization

- Compute bijective maps from a small set of landmarks
- Very efficients
- Little control over the distortion

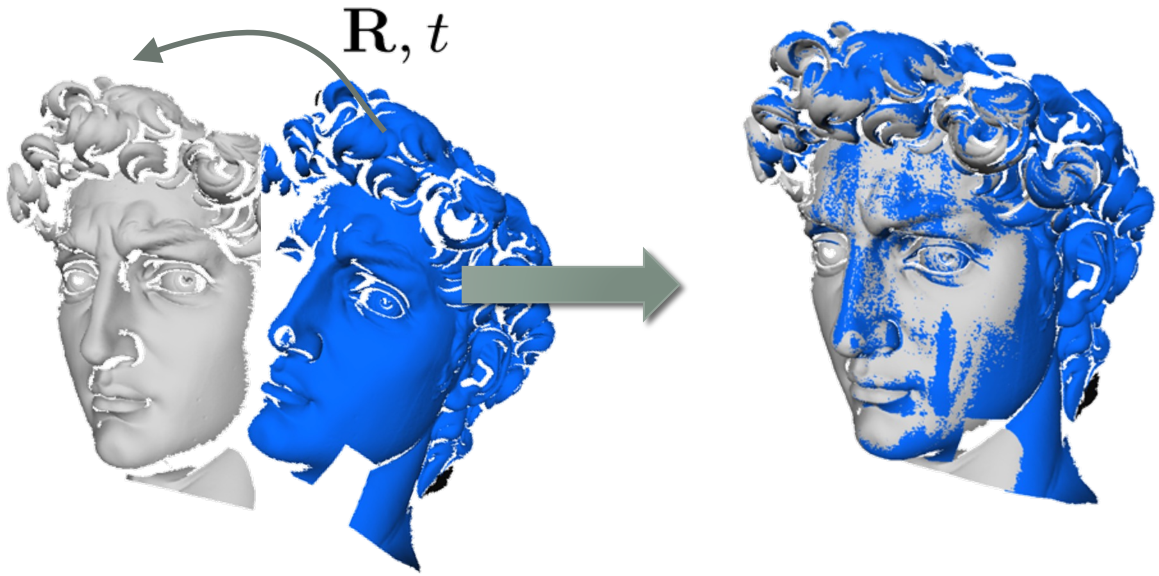
Surface Non-Rigid Alignment

- Compute a deformation for aligning shapes



Divergence-Free Shape Interpolation and Correspondence

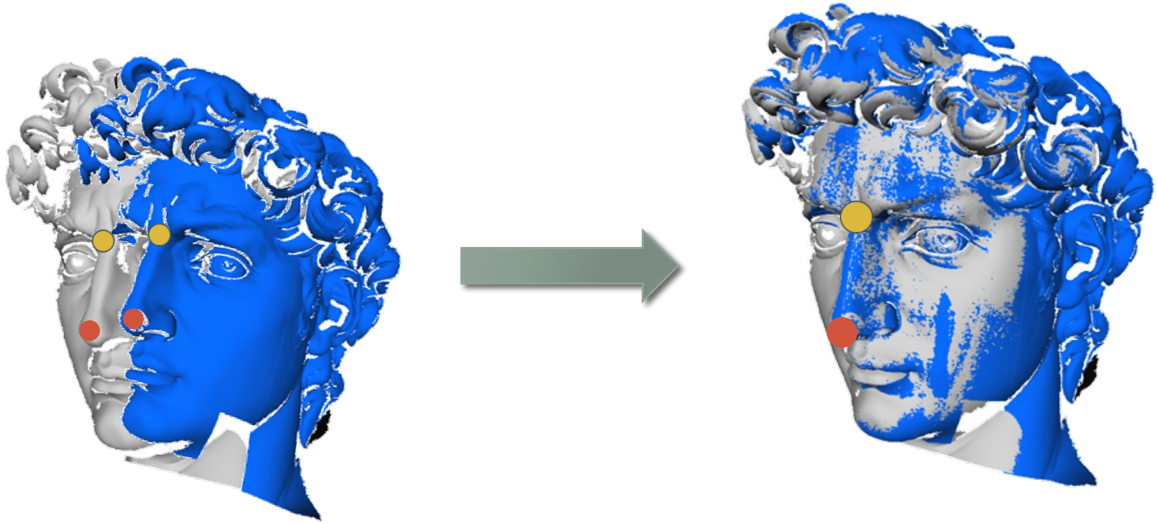
Simpler problem: **Rigid** Alignment



- Given a pair of shapes, find the optimal *Rigid Alignment* between them.
- The unknowns are the rotation/translation parameters of the source onto the target shape.

Simpler problem: **Rigid** Alignment

- What does it mean for an alignment to be **good**?



Intuition: want corresponding points to be close after transformation.

Problems
1. We don’t know what points correspond.
2. We don’t know the optimal alignment.

Iterative Closest Point (ICP)

- Approach: iterate between finding correspondences and finding the transformation:

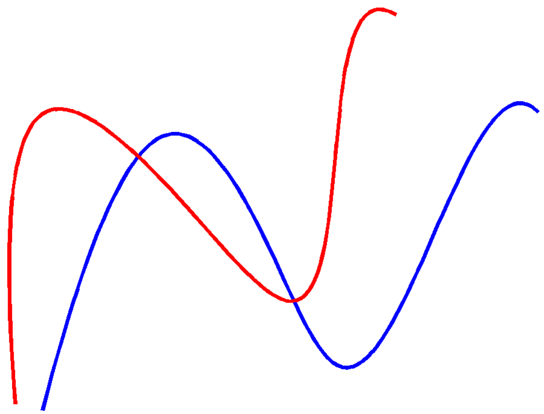
Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing:

$$\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$$

Iterative Closest Point (ICP)

- Approach: iterate between finding correspondences and finding the transformation:

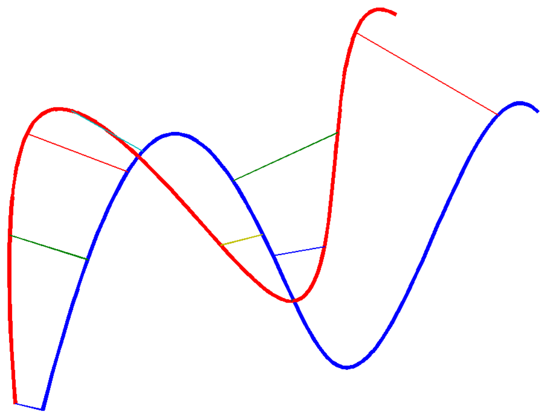


Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing: $\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$

Iterative Closest Point

- Approach: iterate between finding correspondences and finding the transformation:

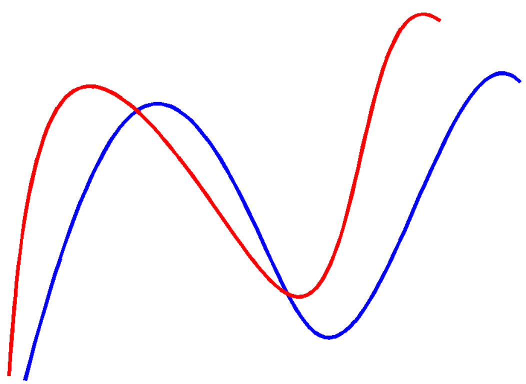


Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing: $\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$

Iterative Closest Point

- Approach: iterate between finding correspondences and finding the transformation:

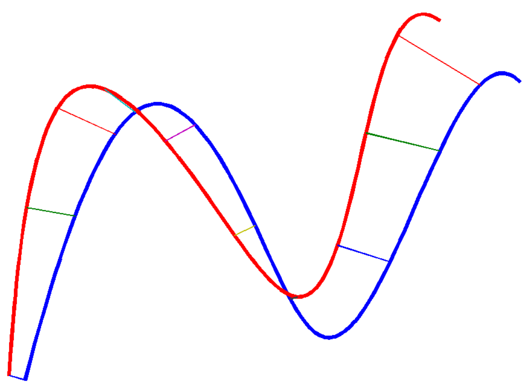


Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing: $\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$

Iterative Closest Point

- Approach: iterate between finding correspondences and finding the transformation:

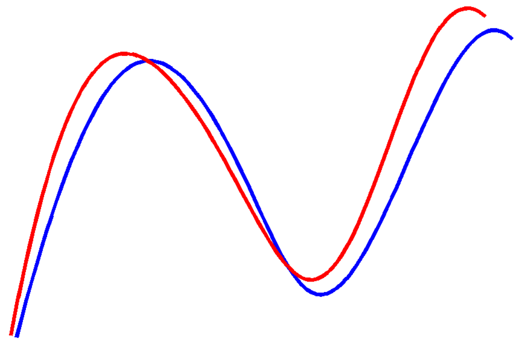


Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing: $\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$

Iterative Closest Point

- Approach: iterate between finding correspondences and finding the transformation:

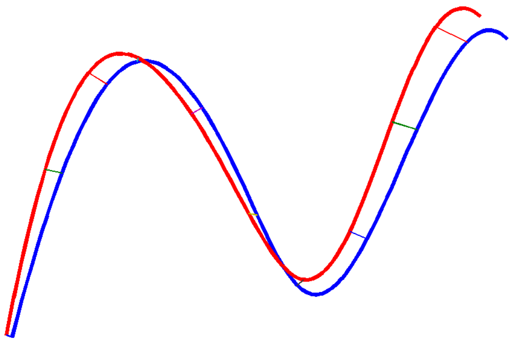


Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing: $\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$

Iterative Closest Point

- Approach: iterate between finding correspondences and finding the transformation:

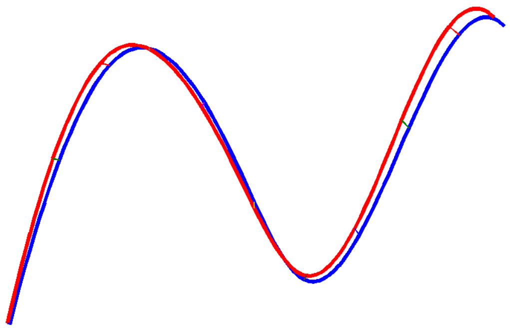


Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing: $\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$

Iterative Closest Point

- Approach: iterate between finding correspondences and finding the transformation:

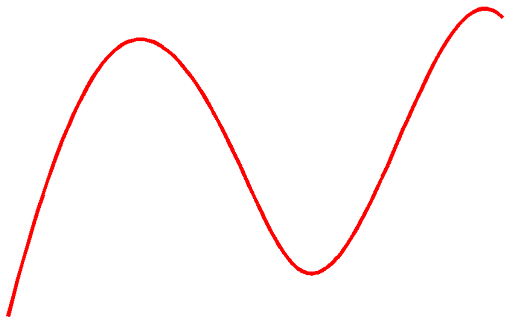


Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing: $\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$

Iterative Closest Point

- Approach: iterate between finding correspondences and finding the transformation:



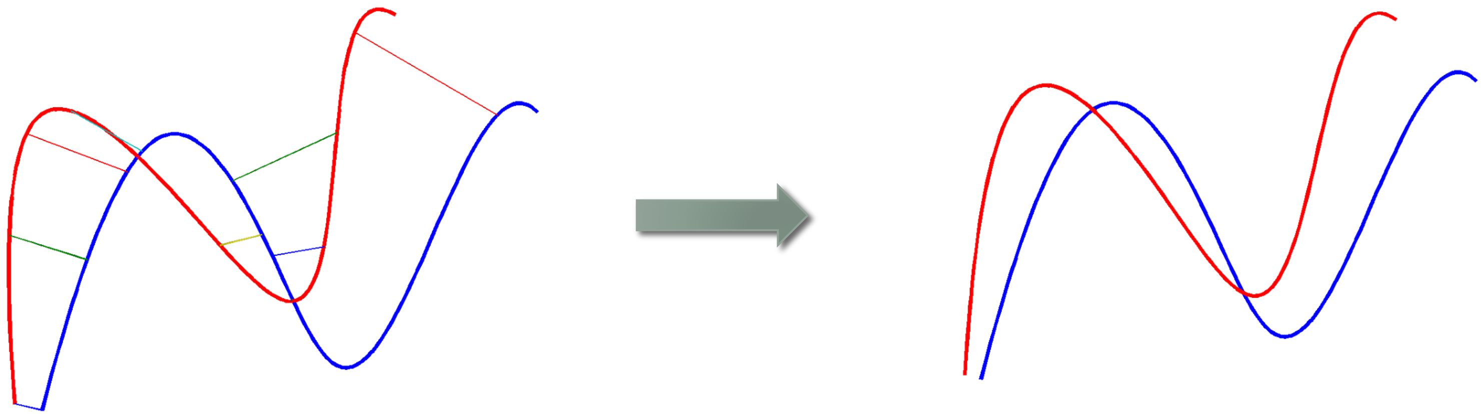
Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing: $\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$

Iterative Closest Point

- Requires two main computations:

1. Computing nearest neighbors.
2. Computing the optimal transformation



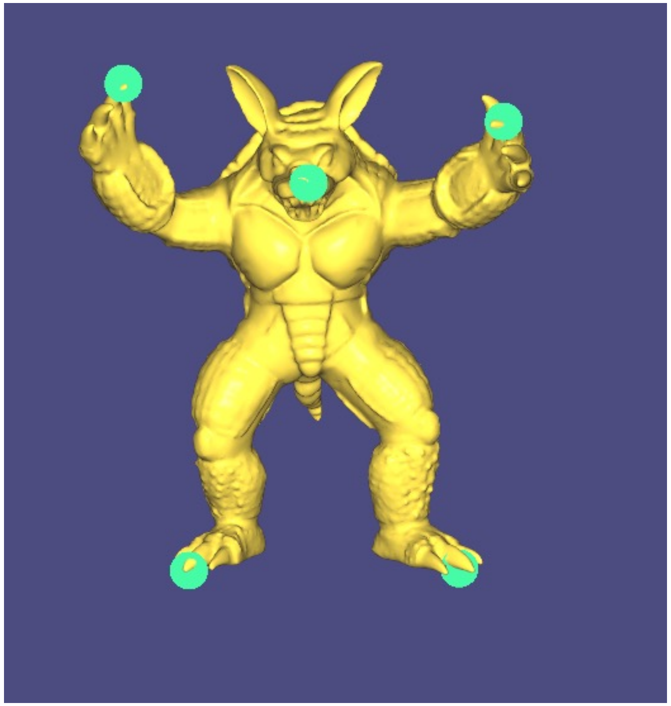
Non-Rigid Alignment Problem

- Compute a deformation for aligning shapes
  - Non-rigid deformation!

https://erkaman.github.io/posts/sorkine2007.html

As-Rigid-As Possible Deformation

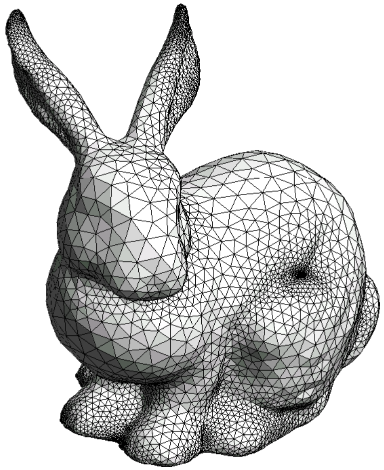
- Compute a deformation for aligning shapes



https://erkaman.github.io/posts/sorkine2007.html

Formalizing Deformation

How do you solve this problem numerically:



$$\xrightarrow{f} \quad \mathbb{R}^3$$

Define a measure of distortion: $E(f) := \sum_{t \in F} \text{distortion}(J_t)$

Define the **deformation** as the minimum of energy:

$$(u_{\text{opt}}, v_{\text{opt}}, w_{\text{opt}}) = \underset{f=(u,v,w)}{\arg\min}\, E(f)$$

Possibly given **handle** conditions

As-Rigid-As Possible Deformation

- Deformation of vertex neighborhood $\mathcal{N}_i$ limited to a rotation R_i:

$$\mathbf{u}_i - \mathbf{u}_j = R_i(\mathbf{x}_i - \mathbf{x}_j)$$

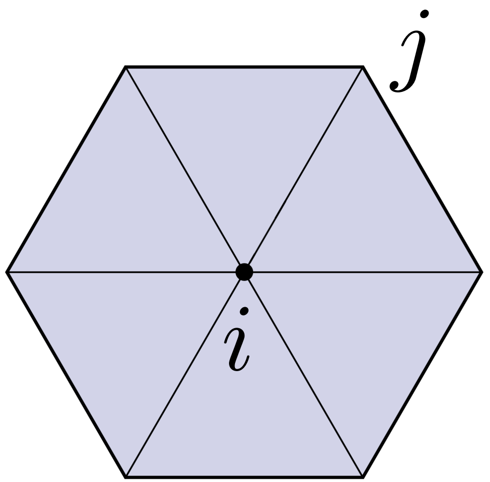
- Energy for edges incident to i:

$$\sum_{j \in \mathcal{N}_i} w_{ij} \|\mathbf{u}_i - \mathbf{u}_j - R_i(\mathbf{x}_i - \mathbf{x}_j)\|^2$$

- Energy on all vertices:

$$E(\mathbf{u}, R) = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} \|\mathbf{u}_i - \mathbf{u}_j - R_i(\mathbf{x}_i - \mathbf{x}_j)\|^2$$

$$w_{ij} = \frac{1}{2}(\cot \alpha_{ij} + \cot \beta_{ij})$$



As-Rigid-As Possible Deformation

- Solving the optimization problem:

$$\min_{\mathbf{u},R} E(\mathbf{u},R) = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} \|\mathbf{u}_i - \mathbf{u}_j - R_i(\mathbf{x}_i - \mathbf{x}_j)\|^2$$

- Iterates between:
  - Minimization for $\mathbf{u}$ (solve three linear systems of size nxn)
  - Minimization for each rotation R_i (compute SVD at each vertex)

As-Rigid-As Possible Deformation

Finding the optimum for the variable $\mathbf{u}$:

$$E(\mathbf{u}, R) = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} \|\mathbf{u}_i - \mathbf{u}_j - R_i(\mathbf{x}_i - \mathbf{x}_j)\|^2$$

$$\frac{\partial E}{\partial \mathbf{u}_i} = 0 \Rightarrow \sum_{j \in \mathcal{N}_i} w_{ij}\left((\mathbf{u}_i - \mathbf{u}_j) - \frac{1}{2}(R_i + R_j)(\mathbf{x}_i - \mathbf{x}_j)\right) = 0$$

$$\Rightarrow (W\mathbf{u})_i = \sum_{j \in \mathcal{N}_i} \frac{w_{ij}}{2}(R_i + R_j)(\mathbf{x}_i - \mathbf{x}_j))$$

W Cotangent matrix

As-Rigid-As Possible Deformation

Finding the optimum for the variable R_i:

$$E(\mathbf{u}, R) = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} \|\mathbf{u}_i - \mathbf{u}_j - R_i(\mathbf{x}_i - \mathbf{x}_j)\|^2$$

Under the constraint: $R_i^\top R_i = I, \quad \det R_i = 1$

Equivalent to solving:

$$\max_{R_i} \langle R_i, B_i \rangle_F, \text{ with } B_i = \sum_{j \in \mathcal{N}_i} w_{ij} (\mathbf{u}_i - \mathbf{u}_j)(\mathbf{x}_i - \mathbf{x}_j)^\top$$

Orthogonal Procrustes problem: there exists a closed form solution using the SVD

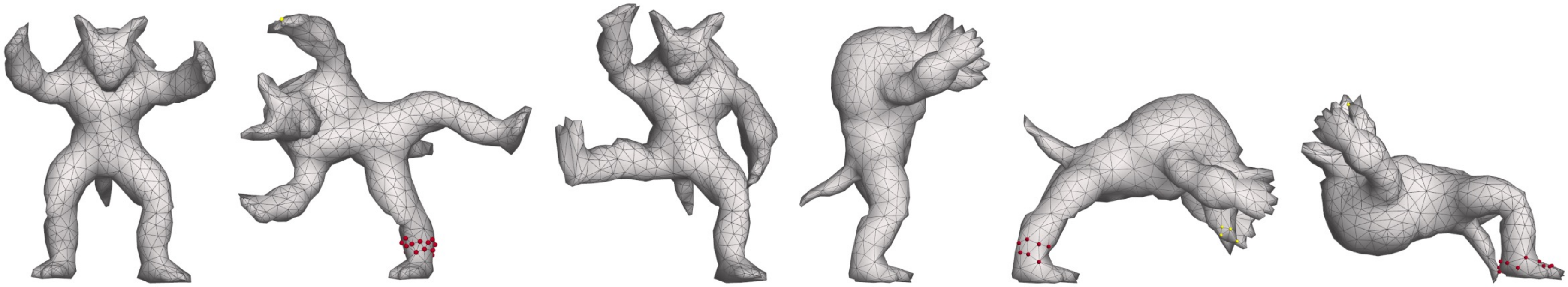
$B_i = U \Sigma V^\top$ with U, V orthogonal matrix $\quad R_i = U \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(UV^\top) \end{pmatrix} V^\top$

As-Rigid-As Possible Deformation

- Solving the optimization problem:

$$\min_{\mathbf{u},R} E(\mathbf{u},R) = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} \|\mathbf{u}_i - \mathbf{u}_j - R_i(\mathbf{x}_i - \mathbf{x}_j)\|^2$$

- Iterates between:
  - Minimization for (solve three linear systems of size nxn)
  - Minimization for each rotation (compute SVD at each vertex)

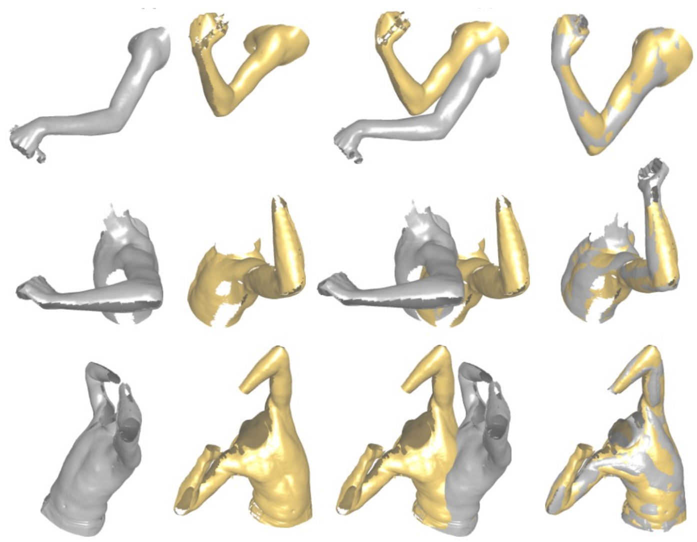


As-Rigid-As-Possible Surface Modeling

Non-Rigid Surface Alignment

Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find the deformation of X minimizing the distance: $E(\mathbf{u}, R) = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} \|\mathbf{u}_i - \mathbf{u}_j - R_i(y_i - y_j)\|^2$



Non-Rigid Registration Under Isometric Deformations

Non-Rigid Surface Alignment

- Pros:
  - Need to find a **good** deformation model
  - Strong regularization of the deformation (volume preservation, basis of deformation)
  - Lots of research and good theoretical understanding

- Cons:
  - Approximative matching (no continuity, no bijectivity)
  - Computationally slow
  - Often limited to small deformations