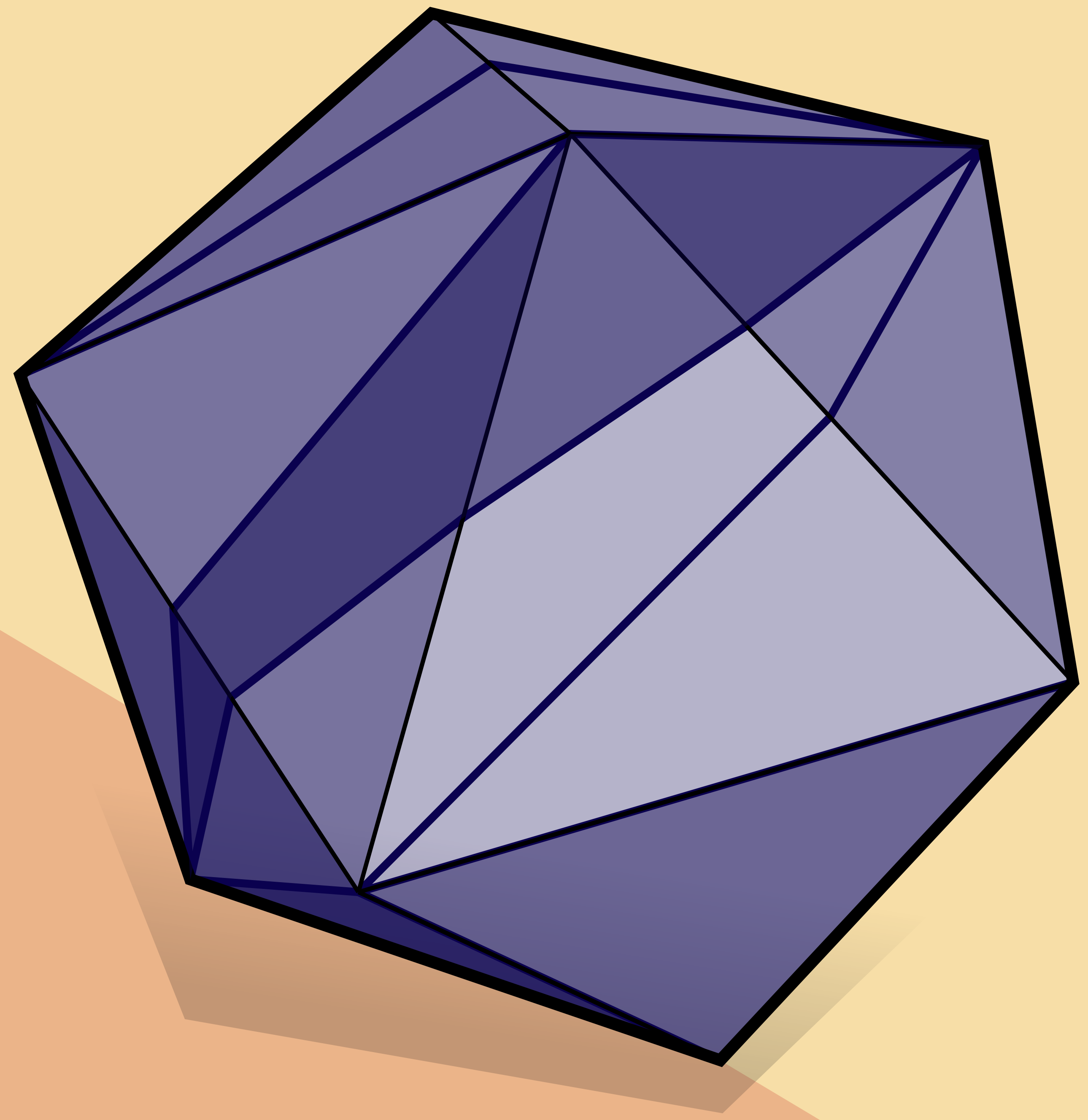
Dynamic Intrinsic Geometry Processing

Mark Gillespie, Carnegie Mellon University

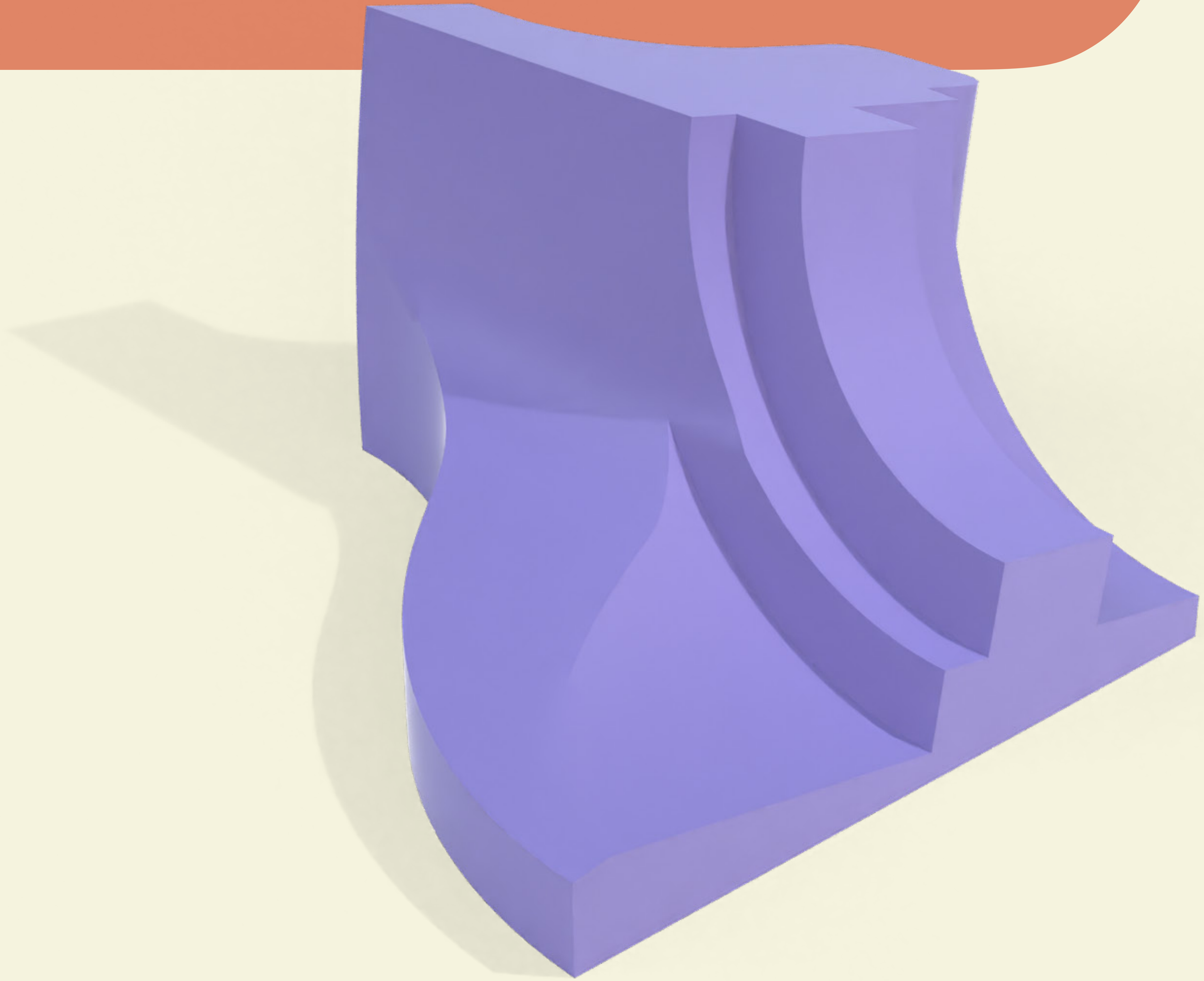


Committee:
Keenan Crane (Chair)
James McCann
Ioannis Gkioulekas
Boris Springborn

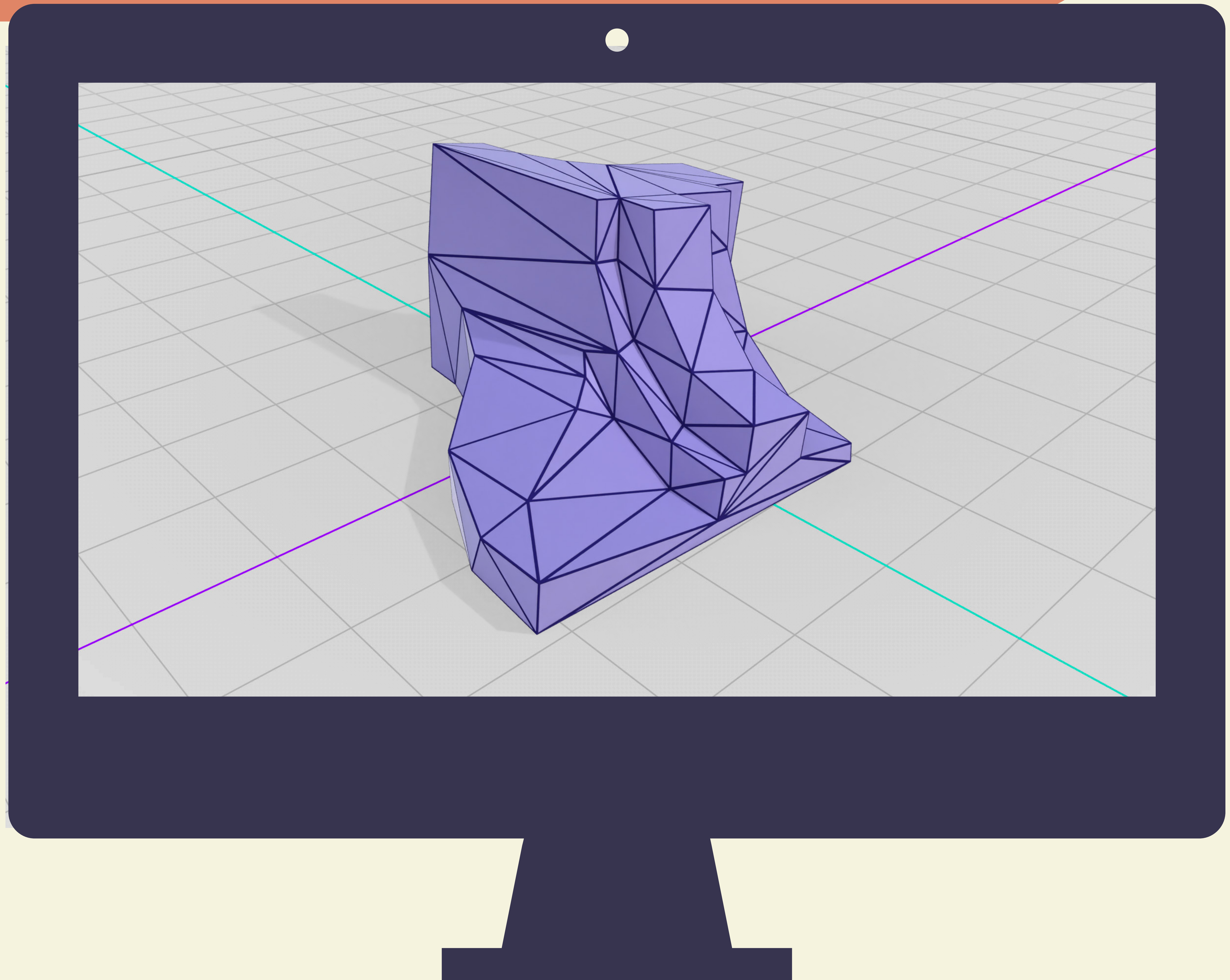
Thesis Proposal

Monday, 11 December 2023

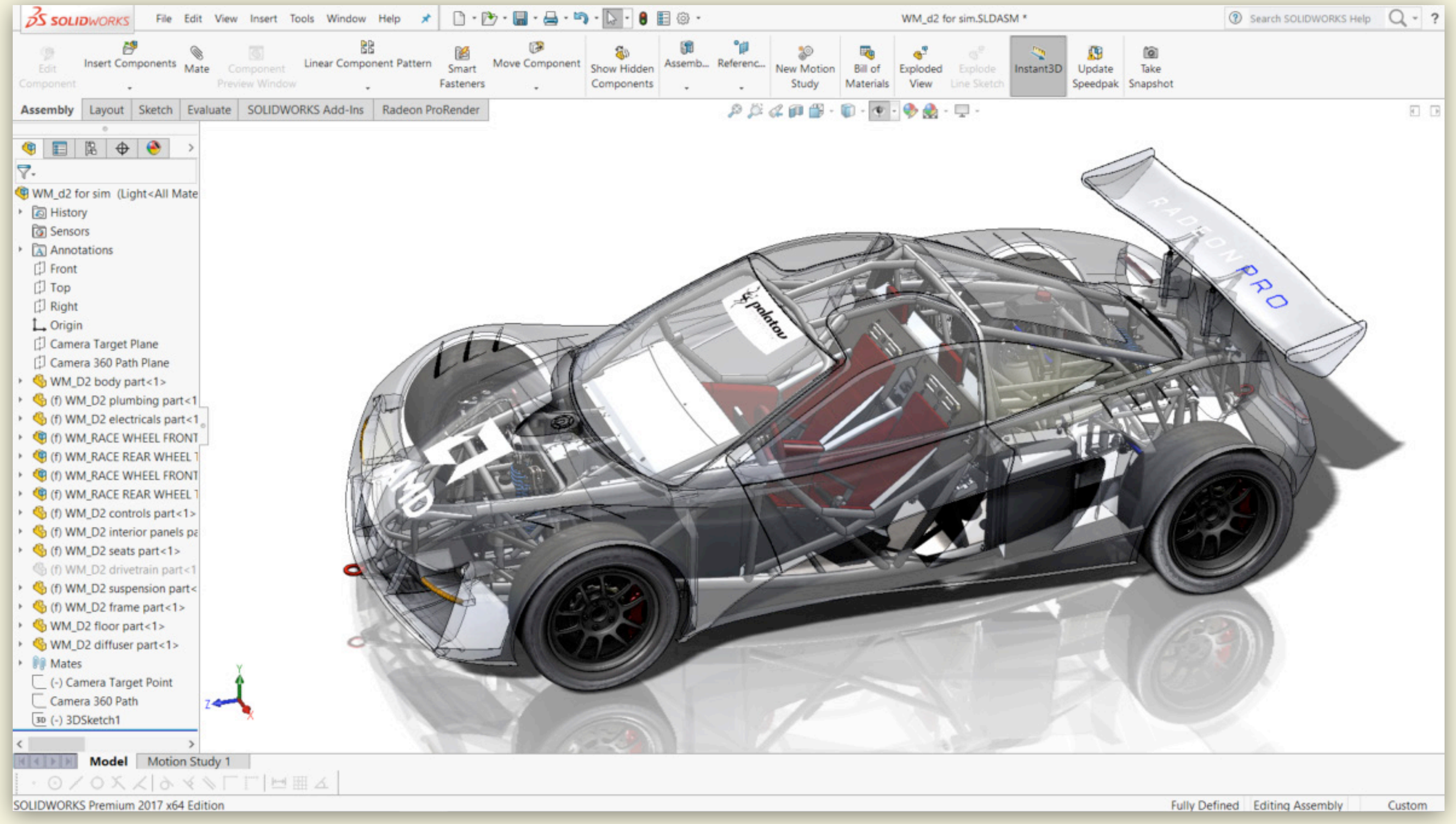
My field: Geometry Processing

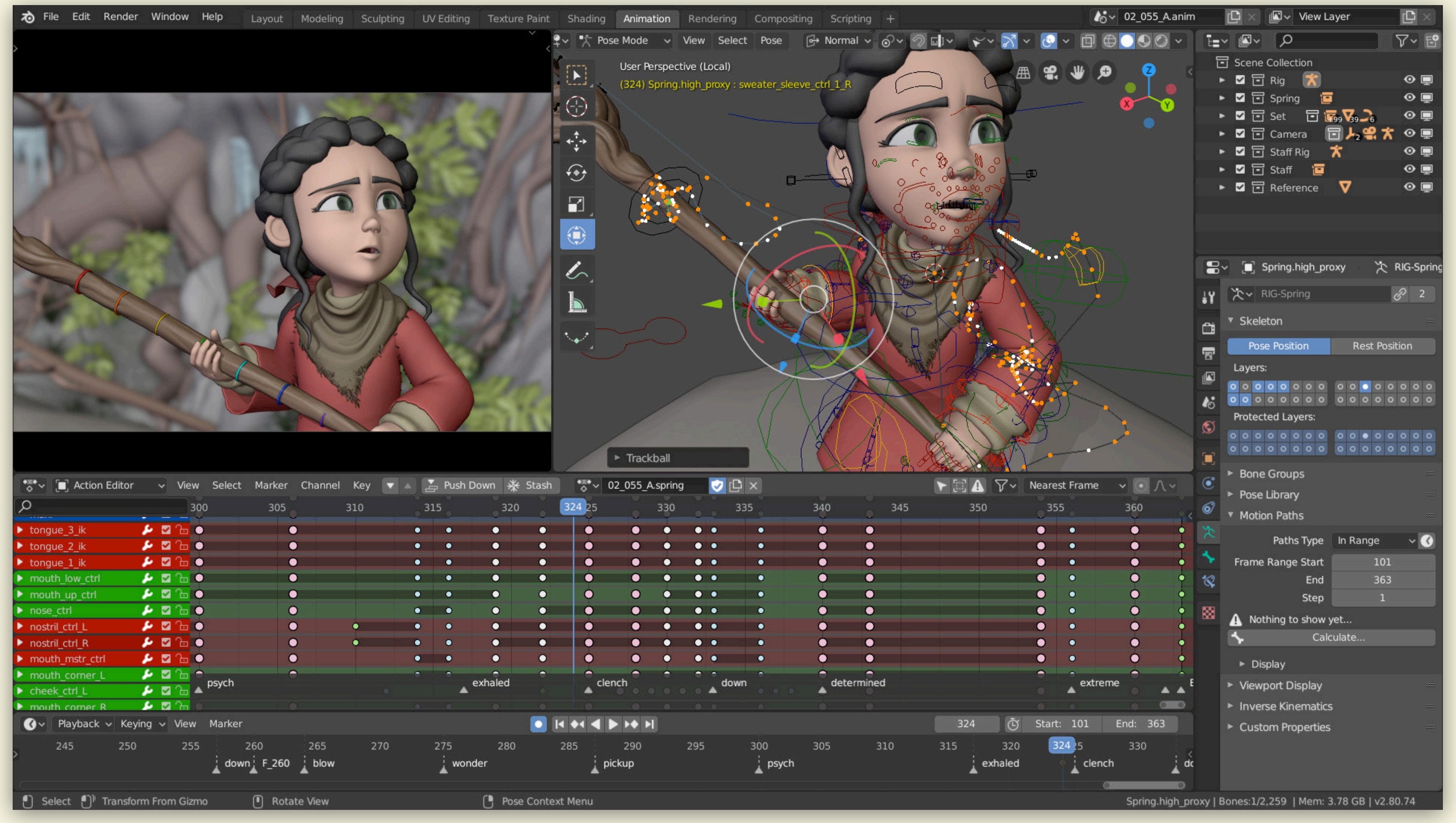


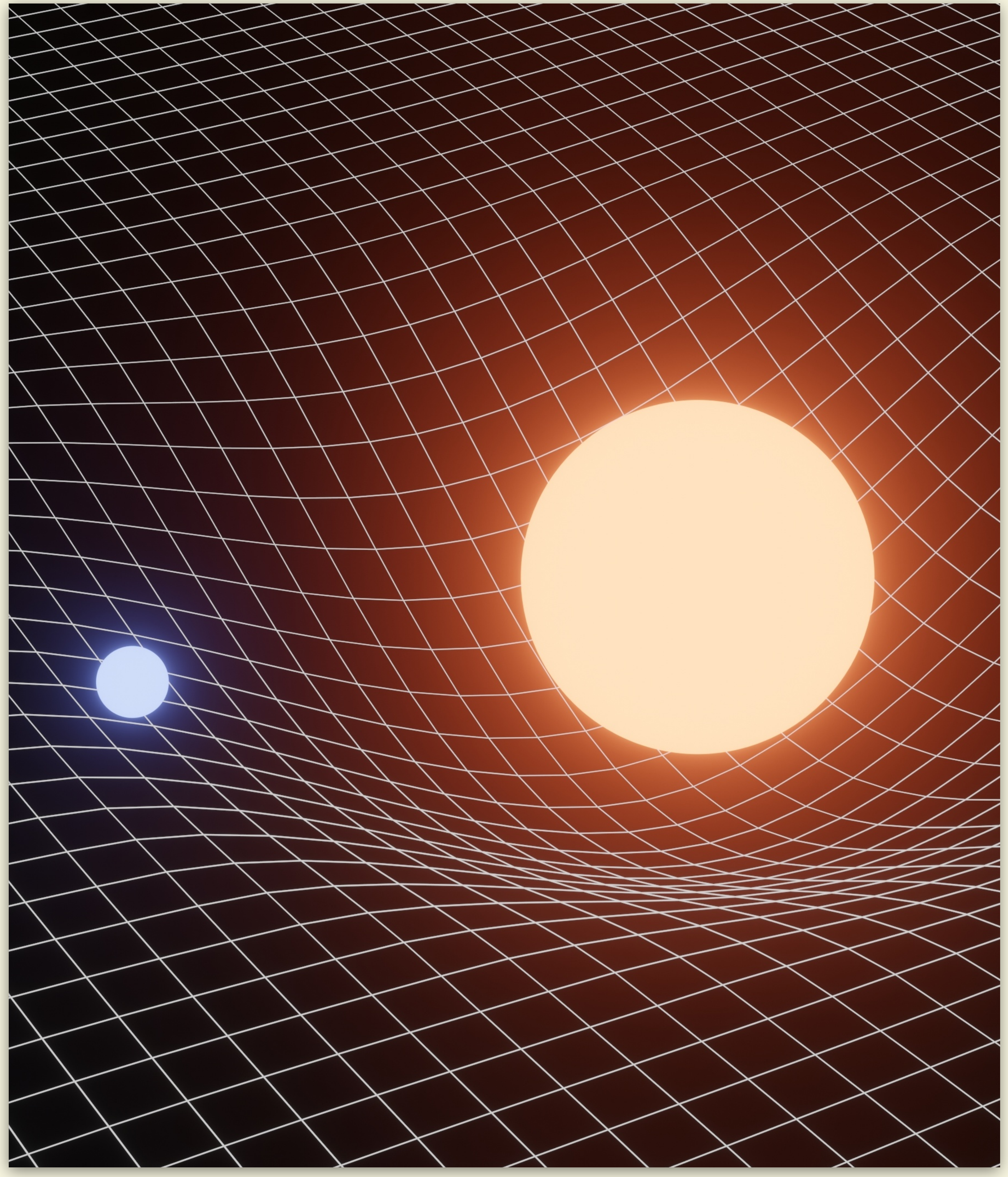
My field: Geometry Processing

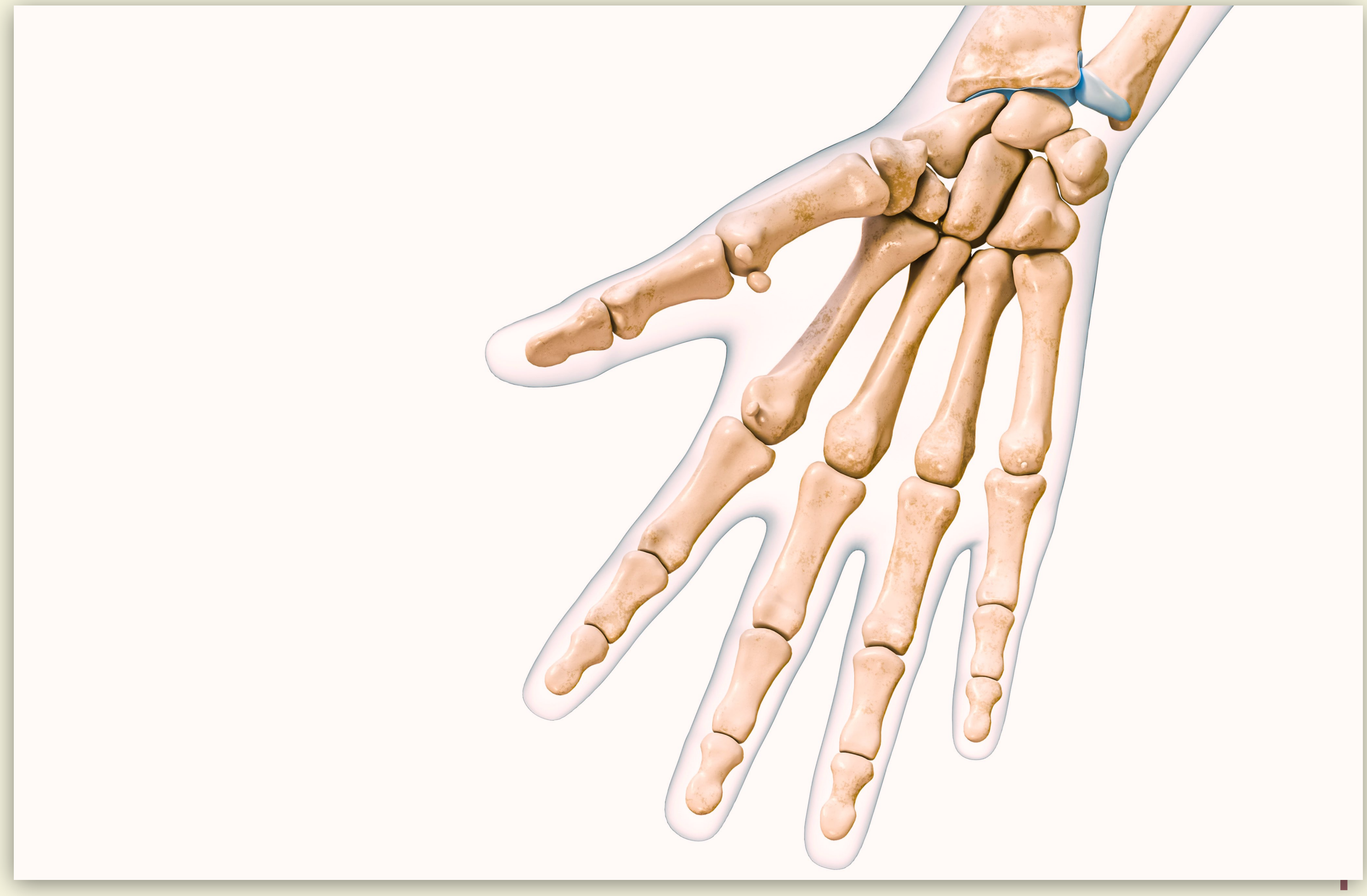


Geometric data is all around us

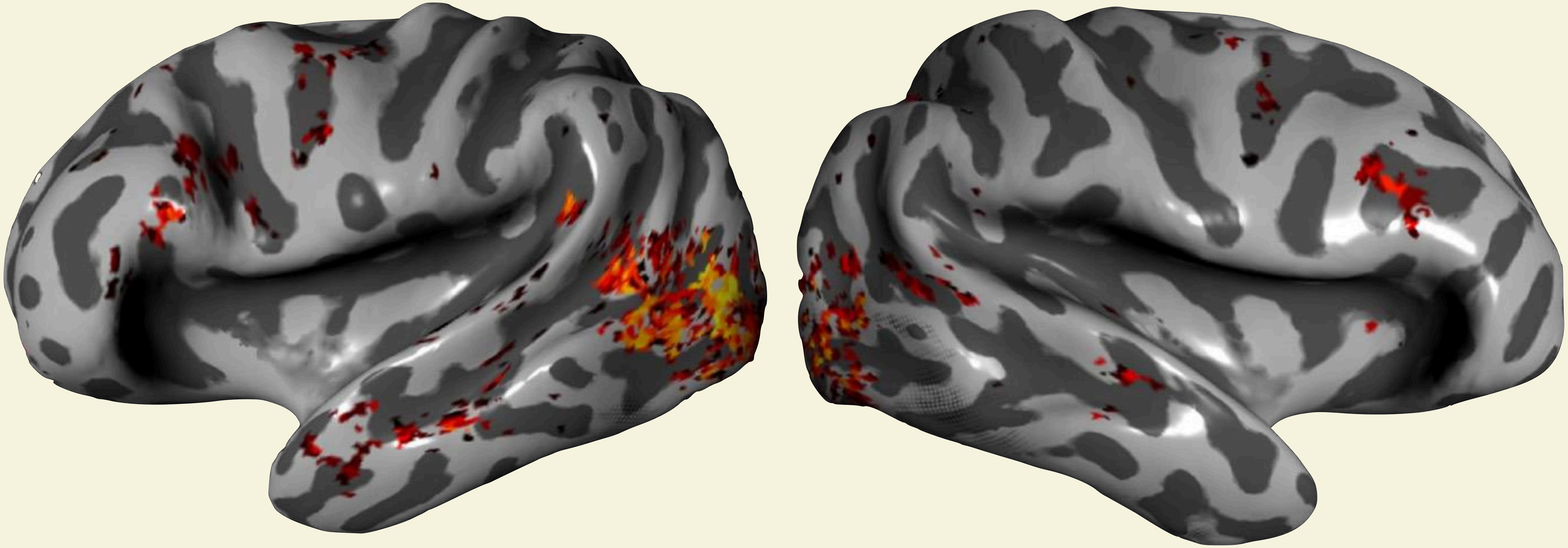






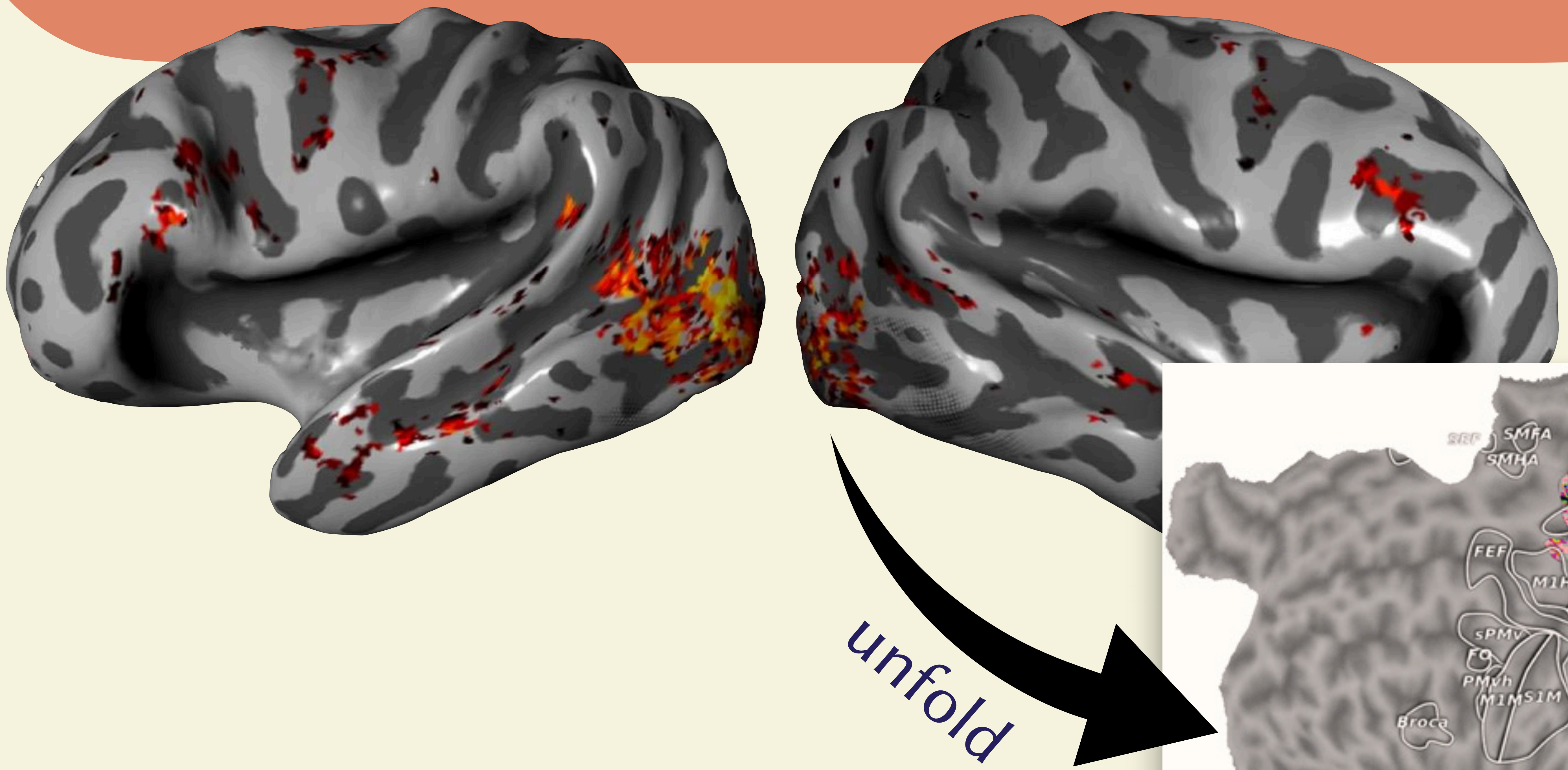


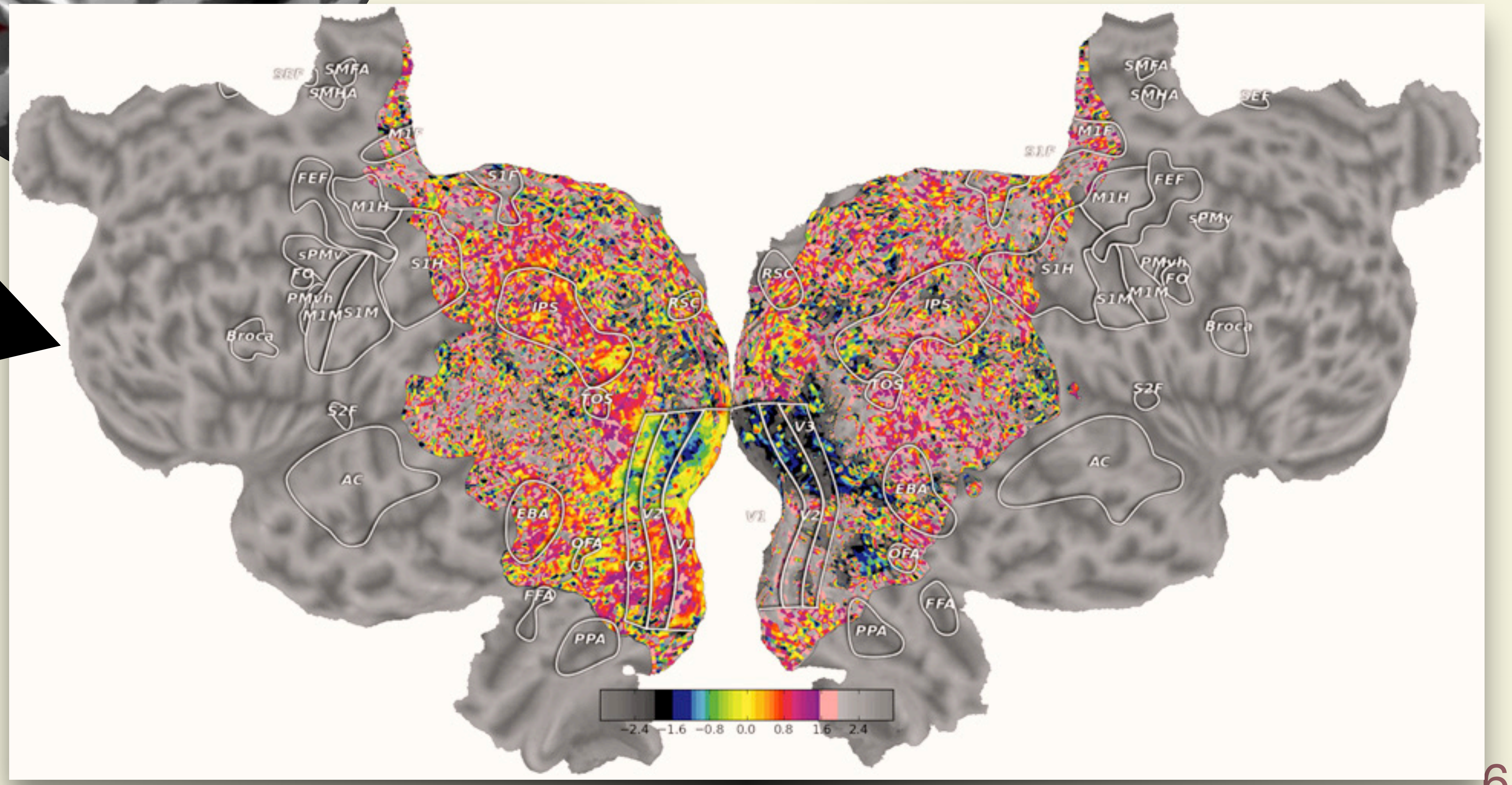
Computation is essential



[Gao, Huth, Lescroart & Gallant 2015]

Computation is essential



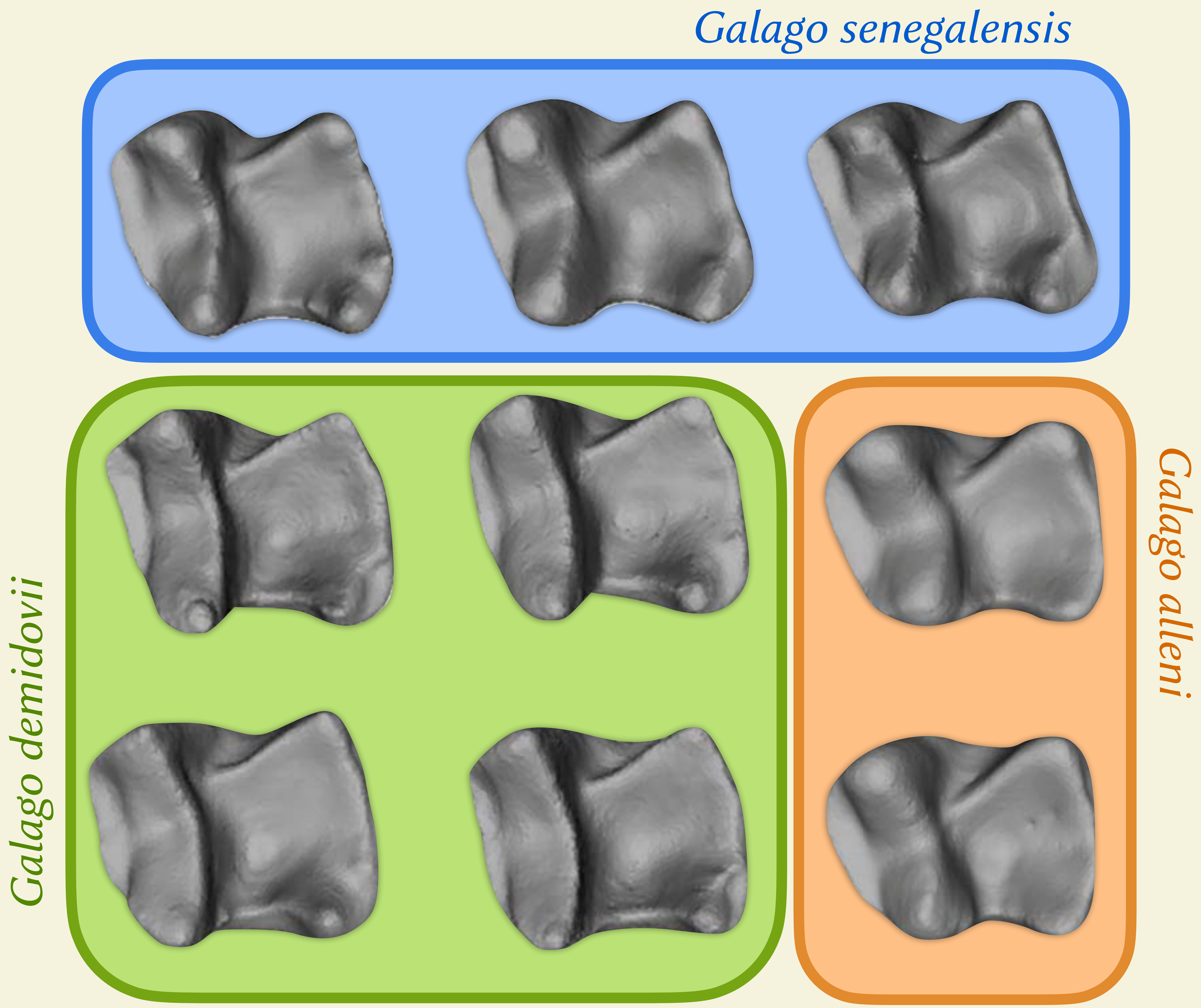


[Gao, Huth, Lescroart & Gallant 2015]

Computation is essential

[Boyer, Lipman,
St. Clair, et al. 2011]

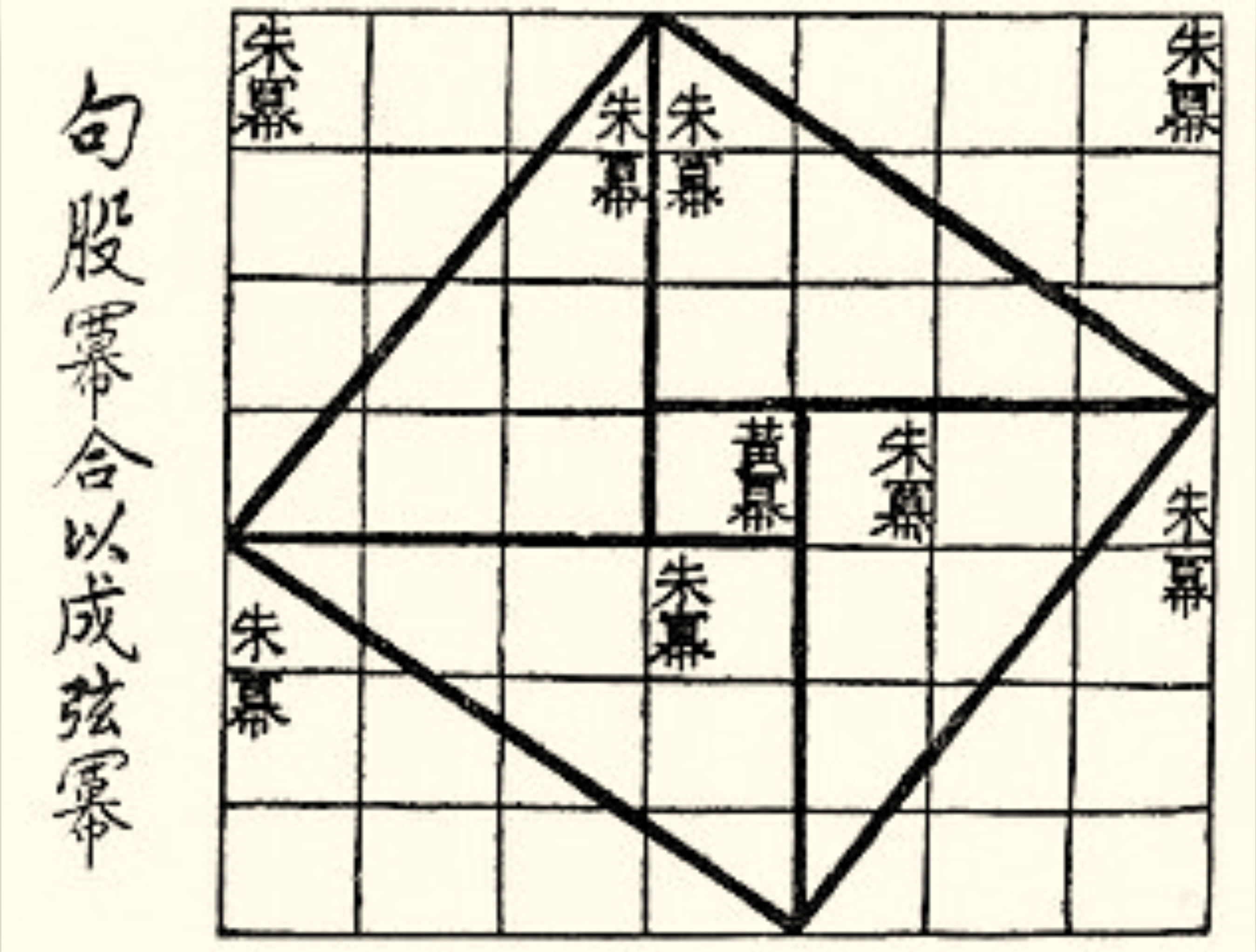
Computation is essential



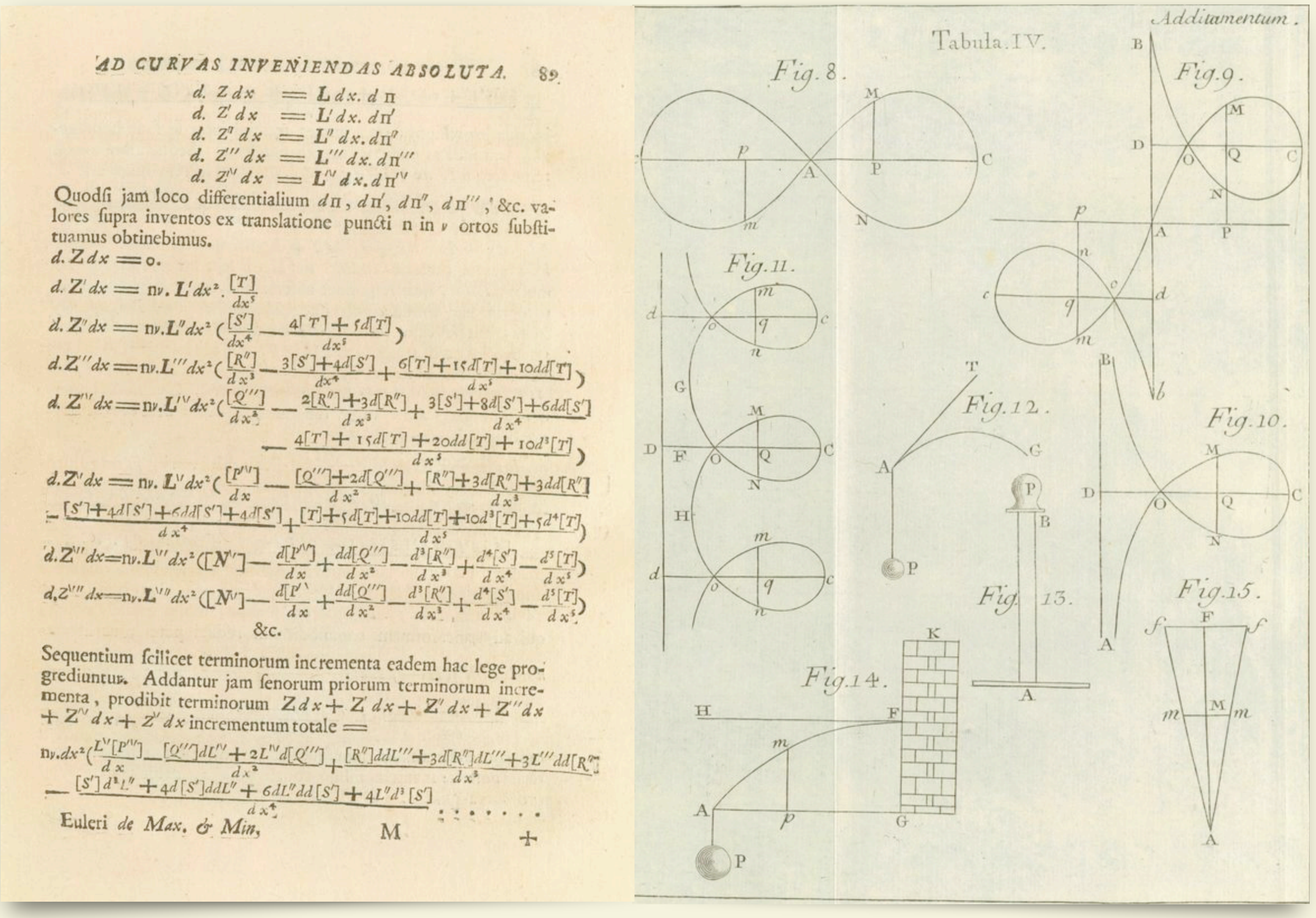
[Boyer, Lipman,
St. Clair, et al. 2011]

Key tool: Math

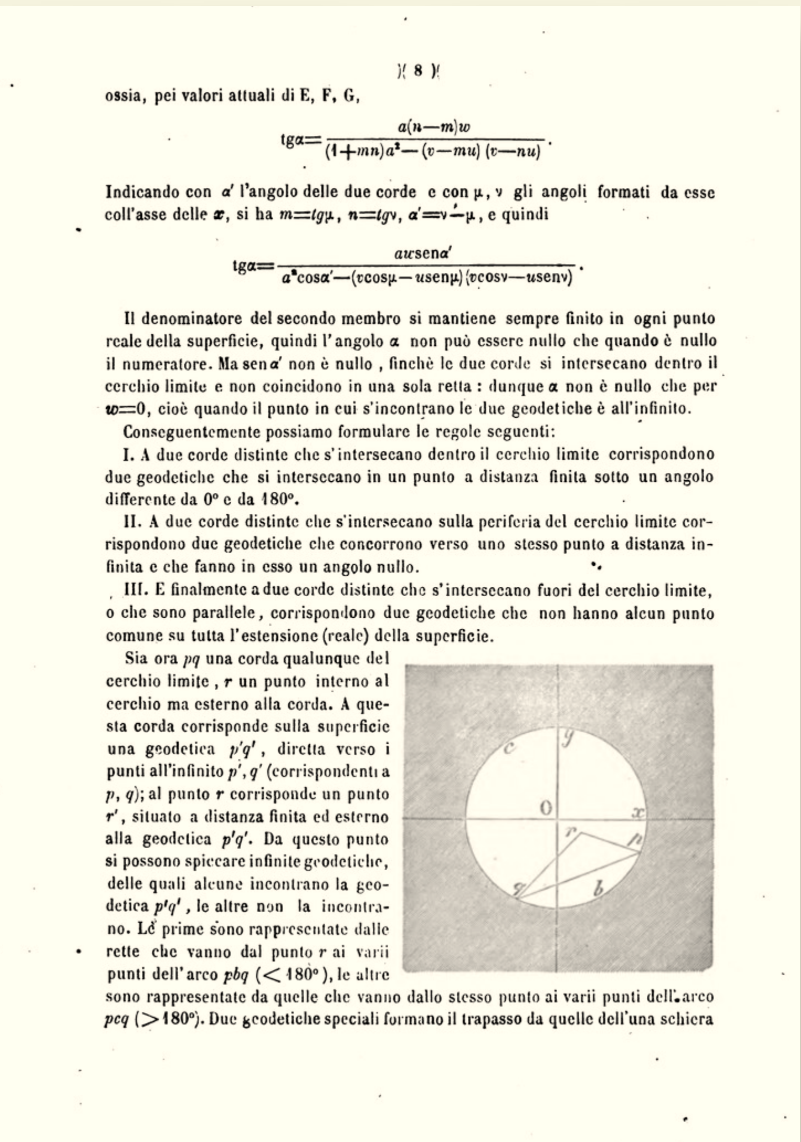
[Babylonian table, c.1800 BCE]



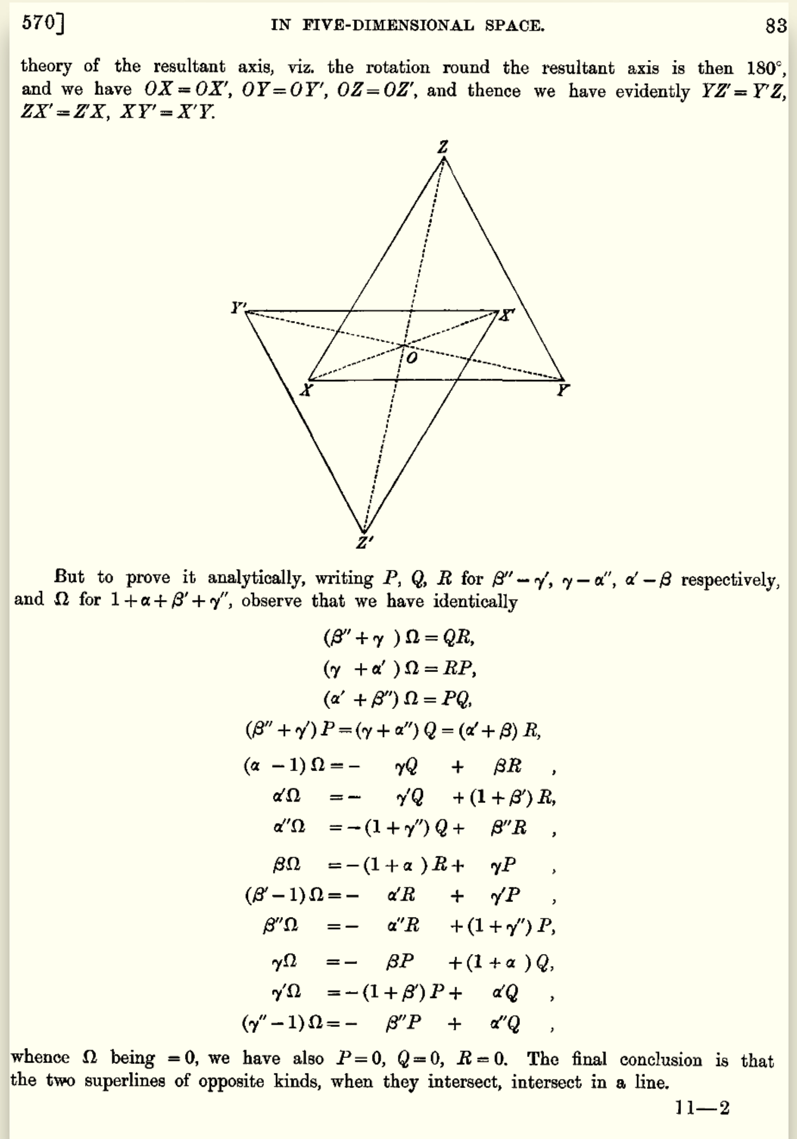
[Zhou, c. 200]



[Euler, 1744]



[Beltrami, 1868]

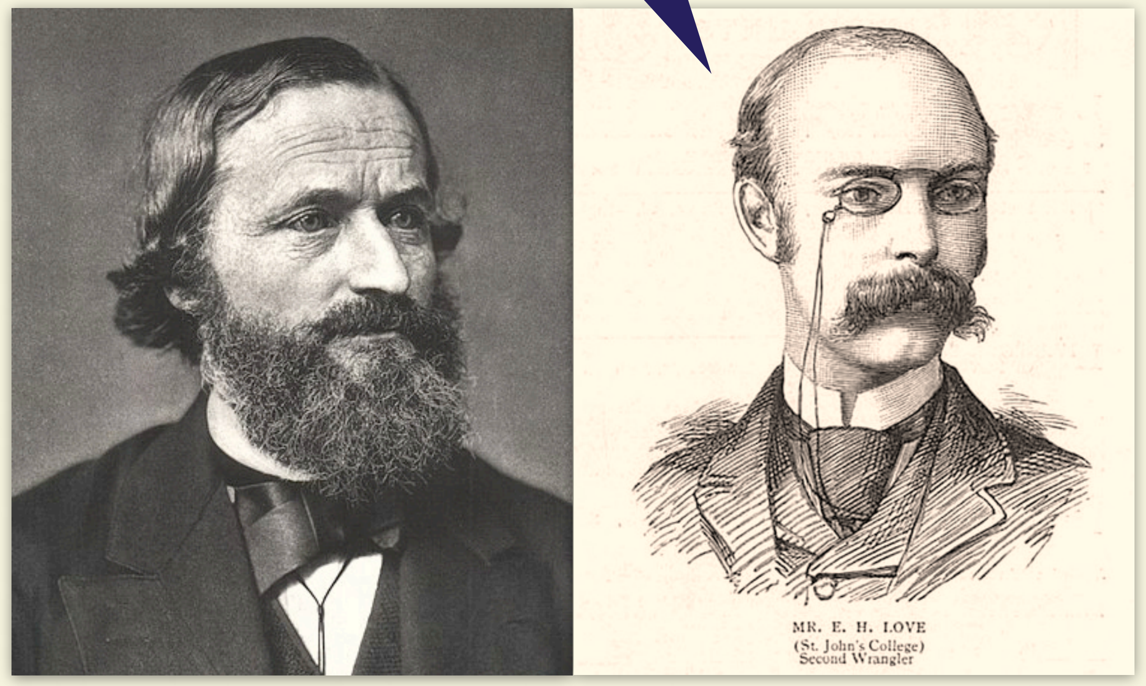


[Cayley, 1896]

Working with 3D shapes is hard

Goal: predict sound by finding *vibrational modes*

build *bilaplacian* matrix and find eigenvectors

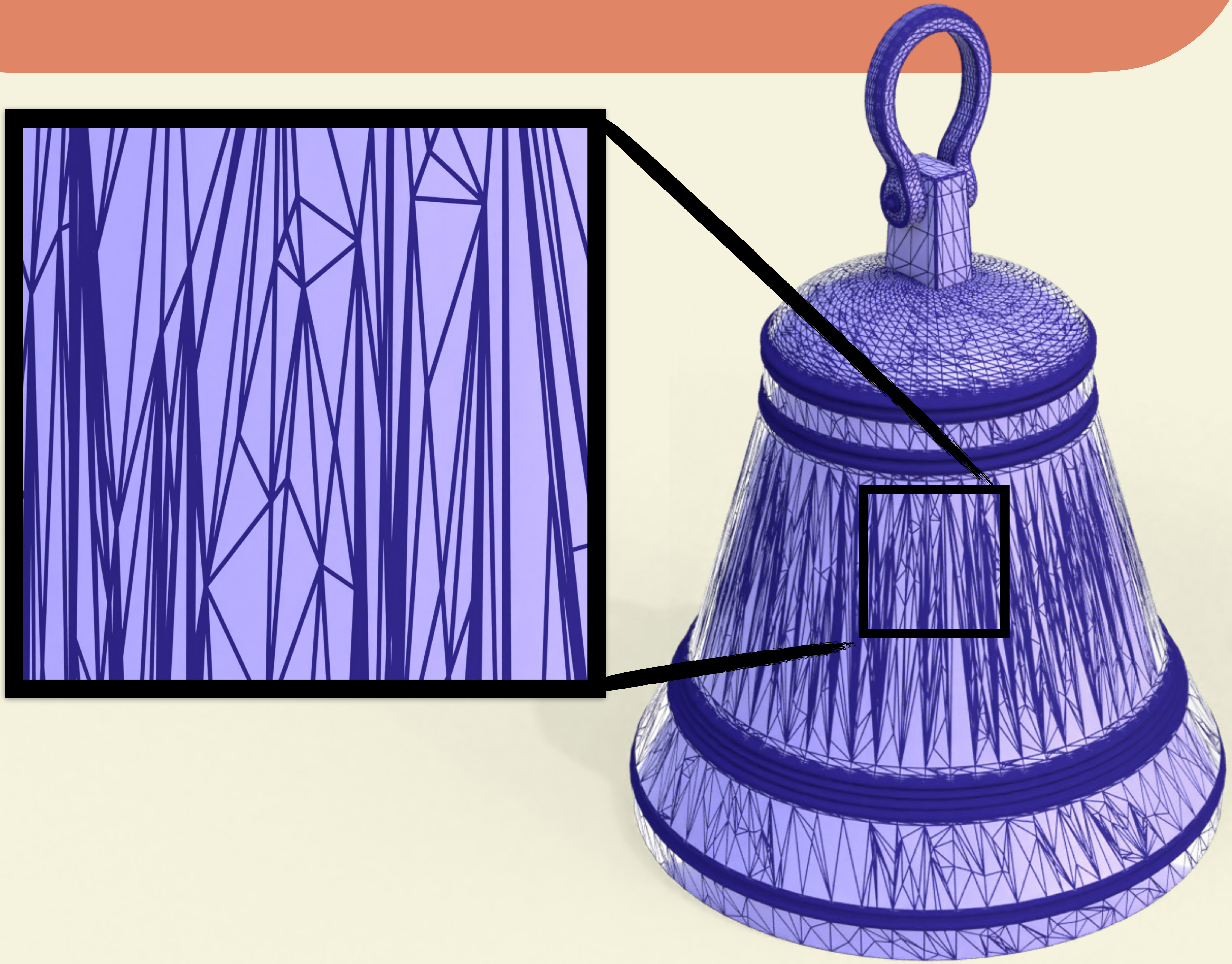


Working with 3D shapes is hard

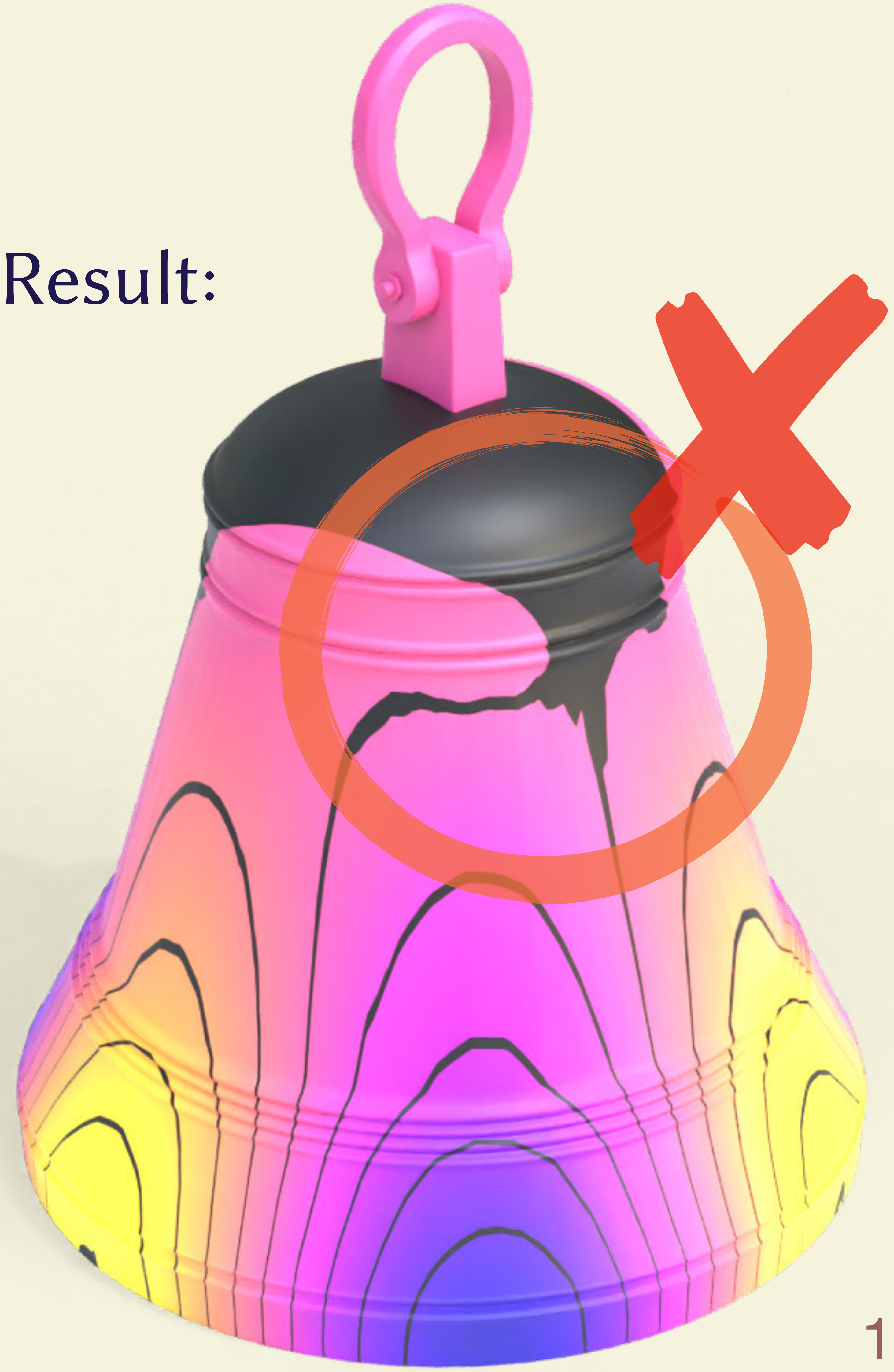
Goal: predict sound by finding *vibrational modes*

Result:

Problem: triangle quality

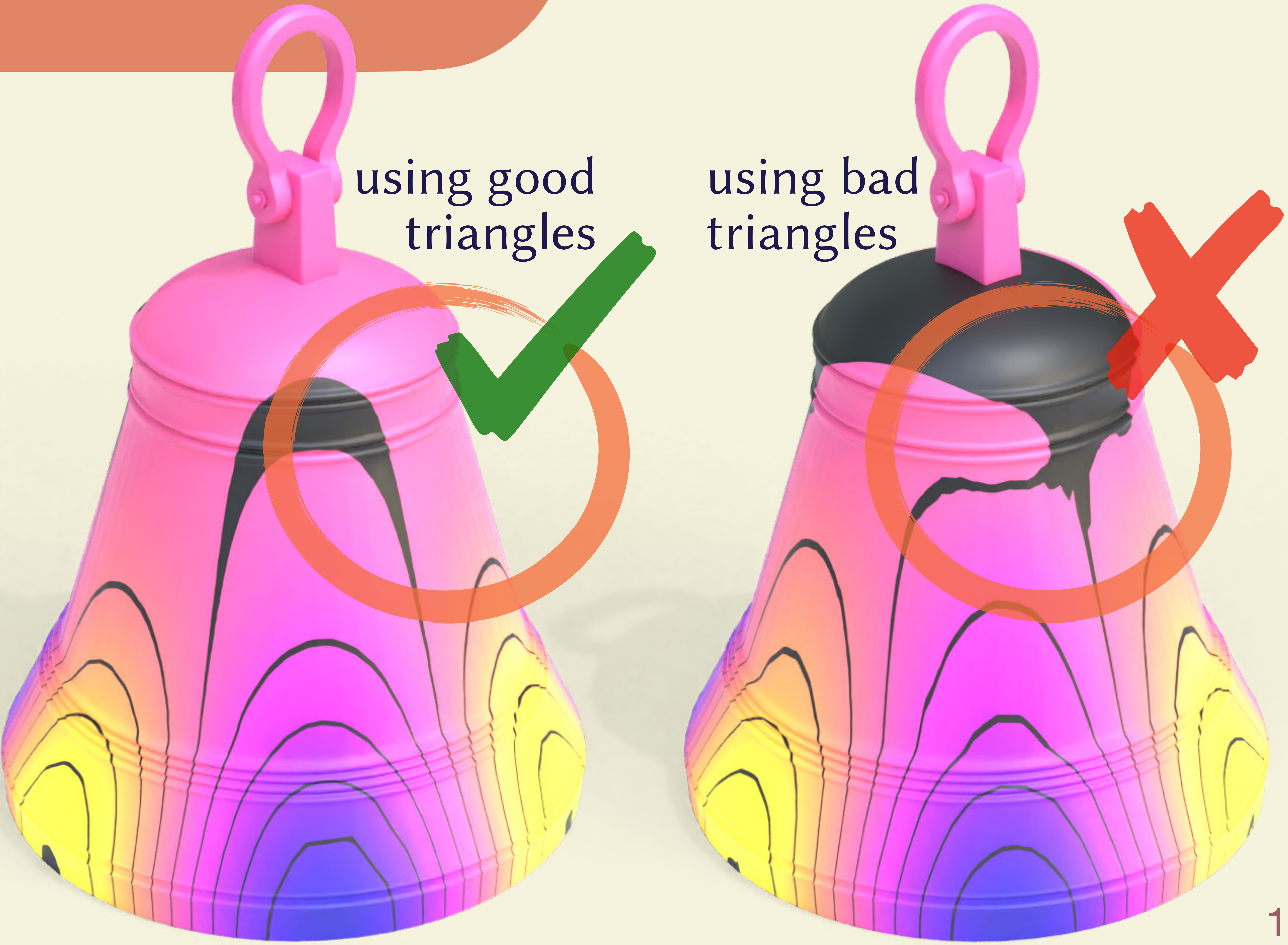


Result:



Problem: triangle quality

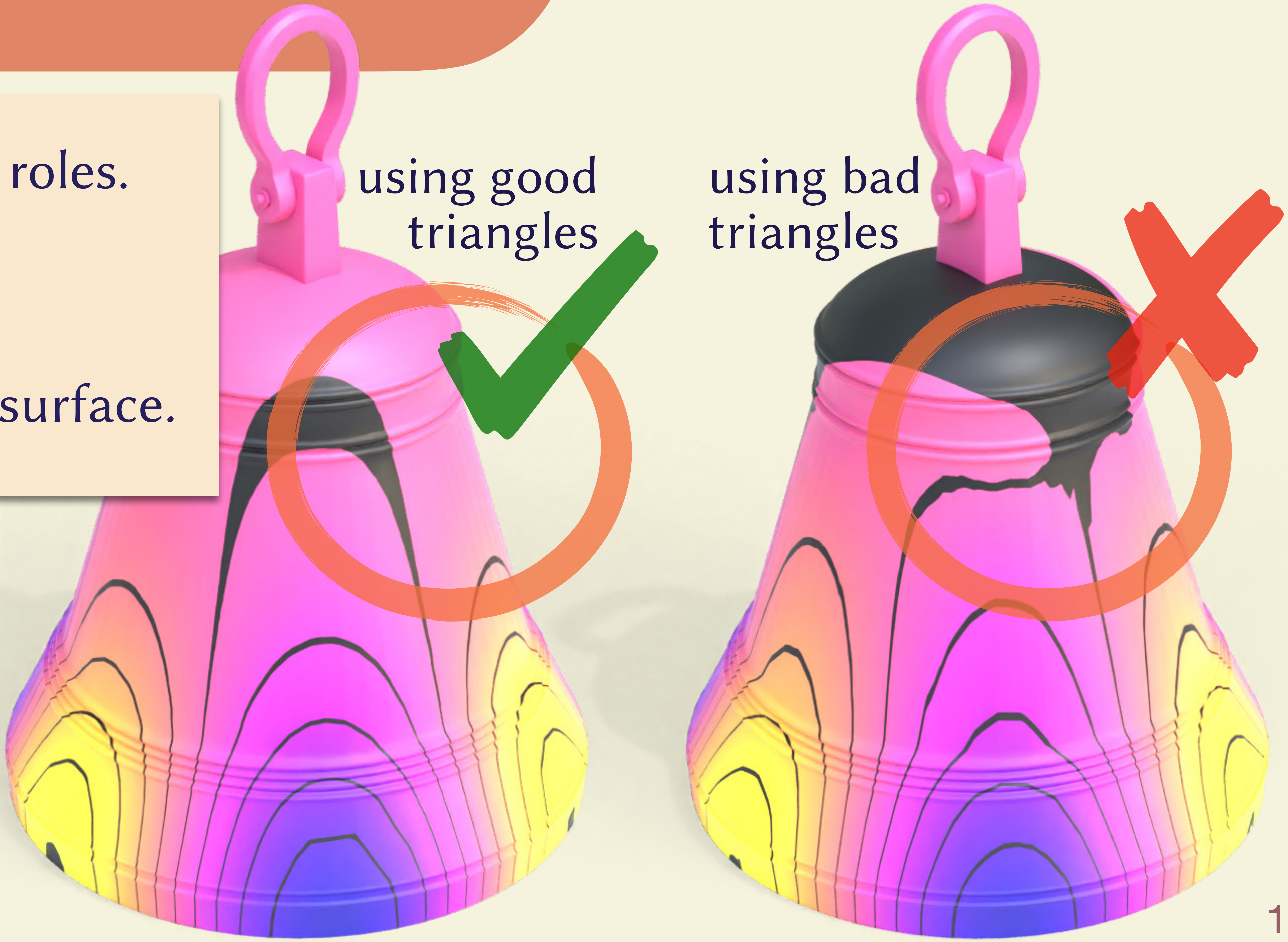
- *Same* number of vertices
  - Not a resolution issue
- *Same* geometry
  - Not an approximation issue



Problem: triangle quality

Problem. Our triangles play two roles. They encode both:

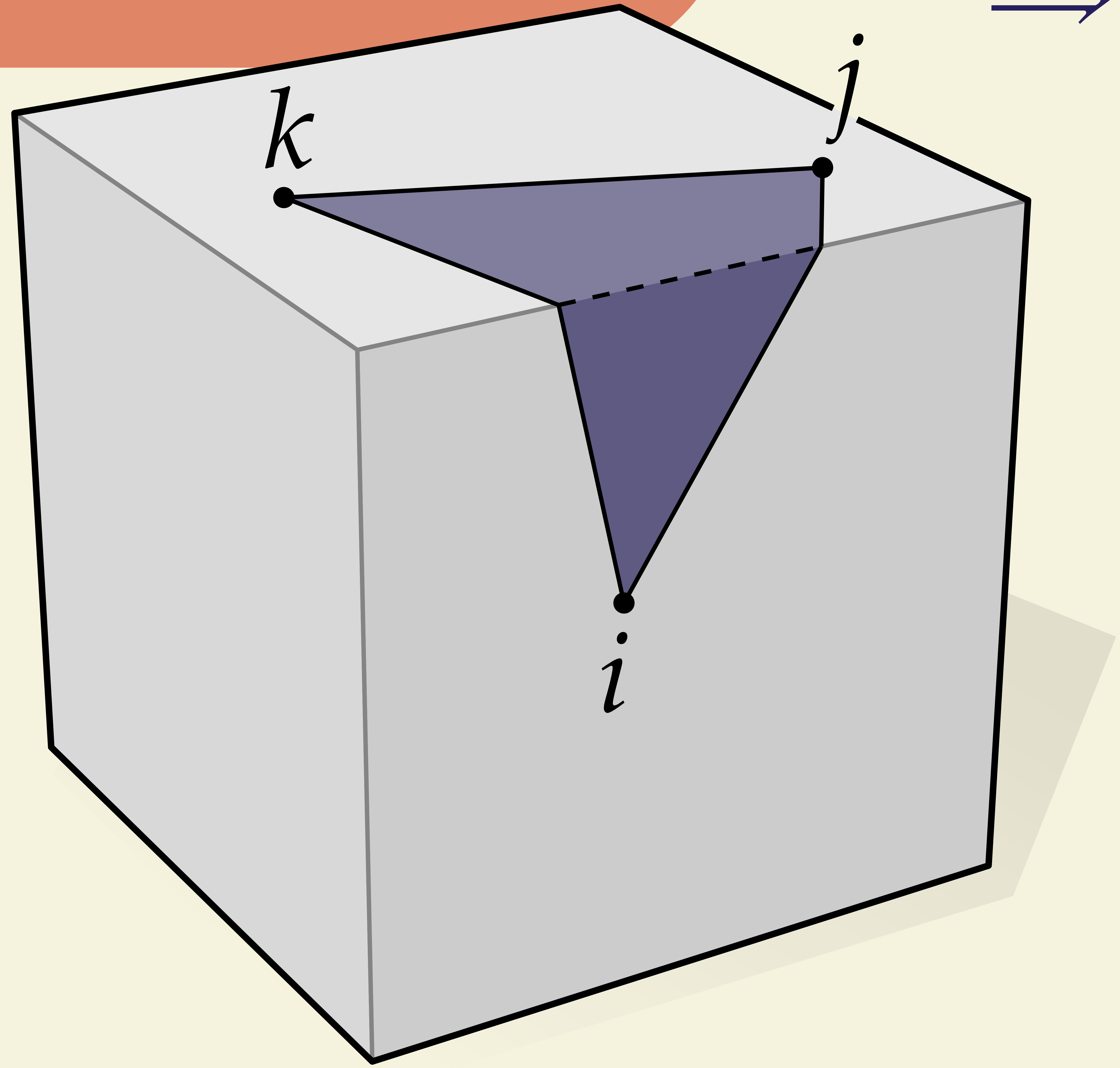
1. the *geometry* of a surface
2. a *space of functions* on that surface.



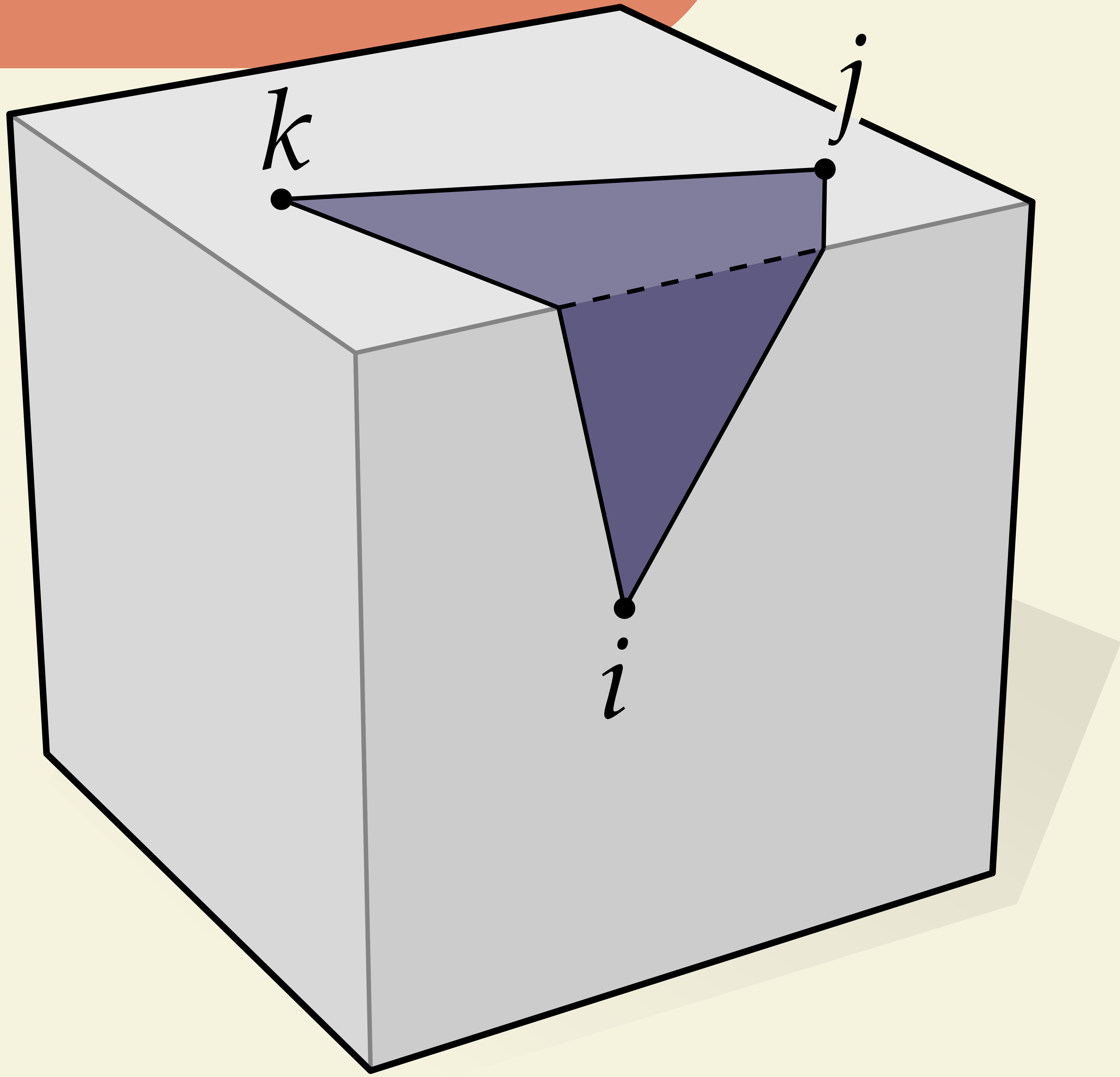
Intrinsic triangles

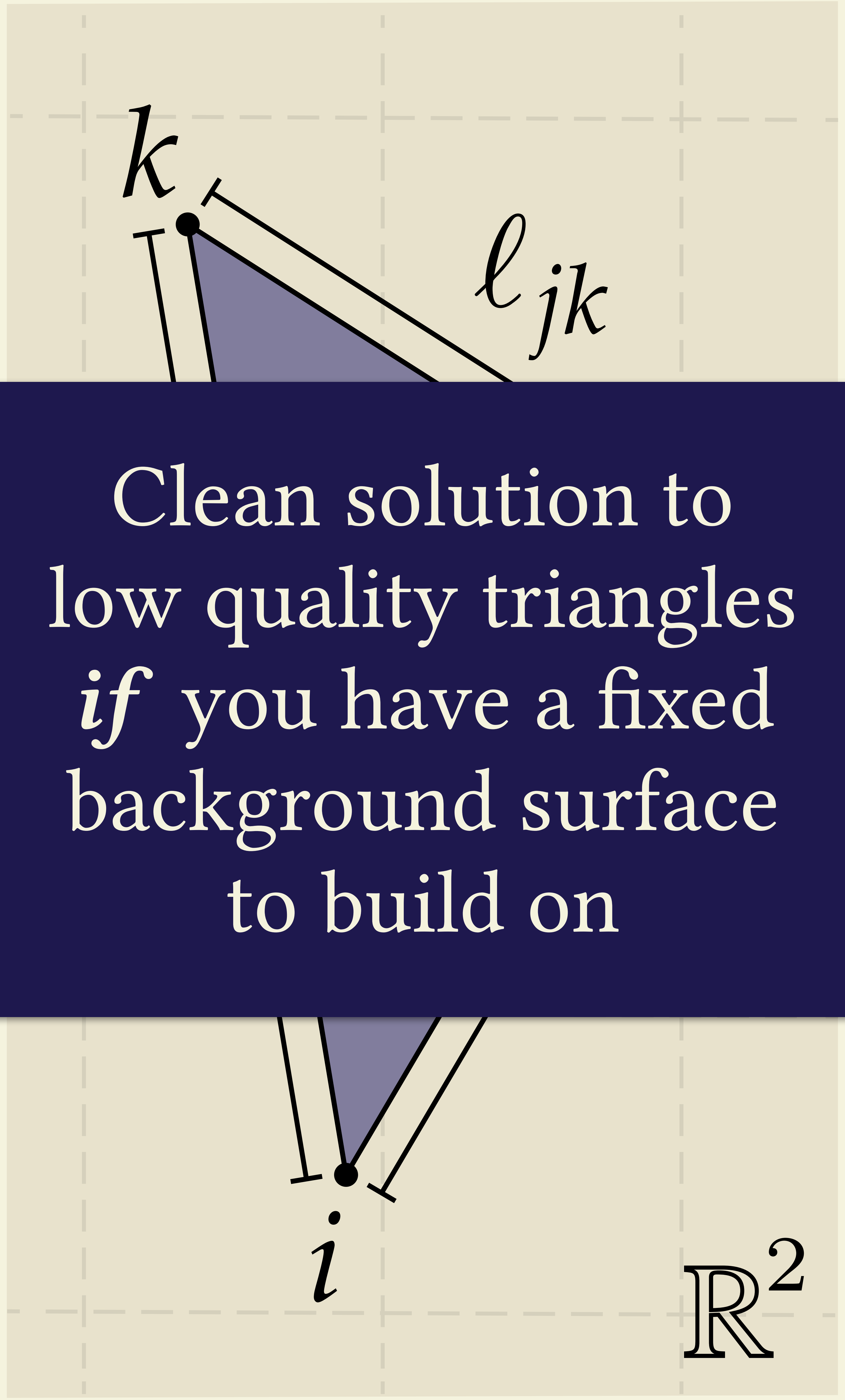
broadening our idea of what a triangle is

$\implies$ flexibility to build models out of good triangles



Intrinsic triangles





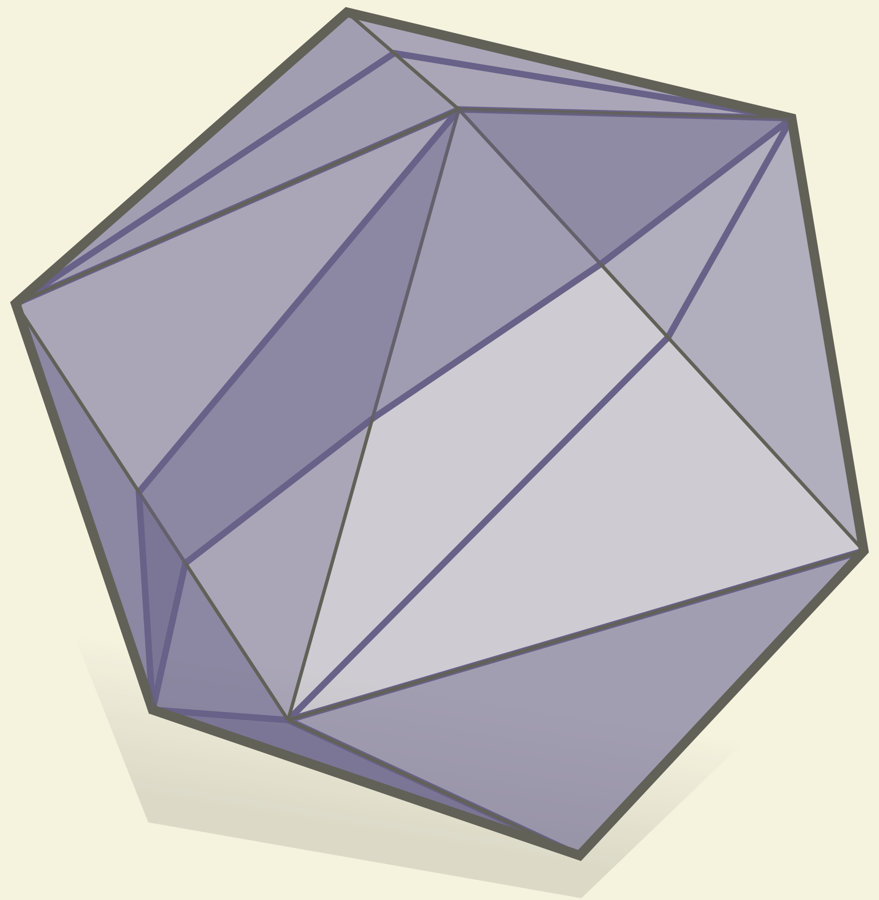
Clean solution to low quality triangles *if* you have a fixed background surface to build on

What if there is no fixed background surface?

What if our geometry changes over time?

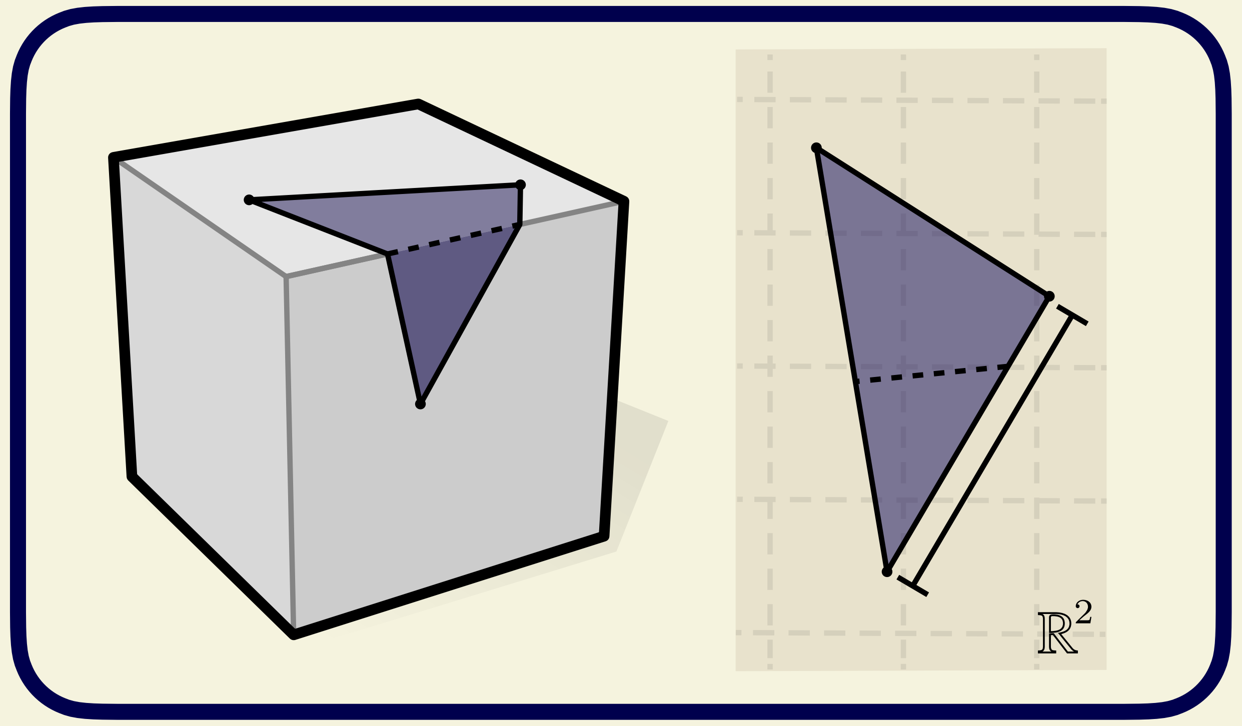
Dynamic intrinsic geometry processing

In my thesis,
I present data structures & algorithms
for using intrinsic triangulations
on time-evolving surfaces



Outline

I. BACKGROUND

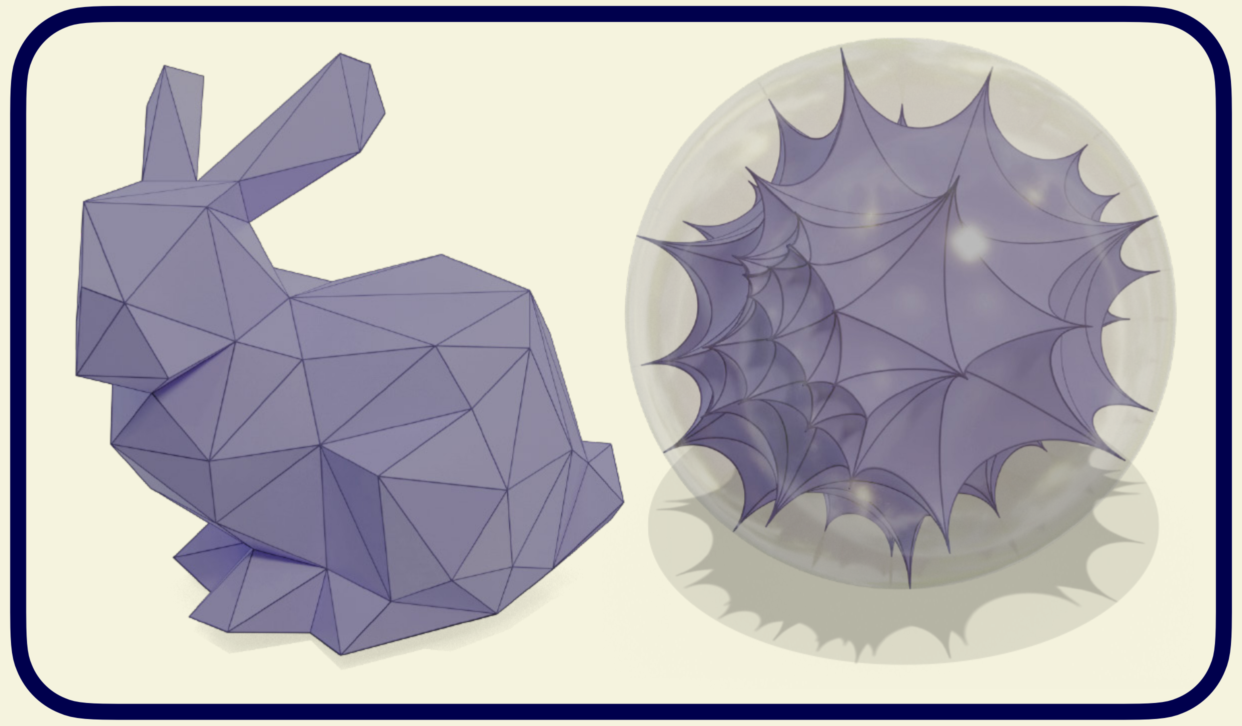


II. SIMPLIFICATION

[Liu, **Gillespie**, Chislett, Sharp, Jacobson & Crane. 2023. Surface Simplification using Intrinsic Error Metrics. *ACM TOG*]

Track intrinsic triangulation while *simplifying* a surface

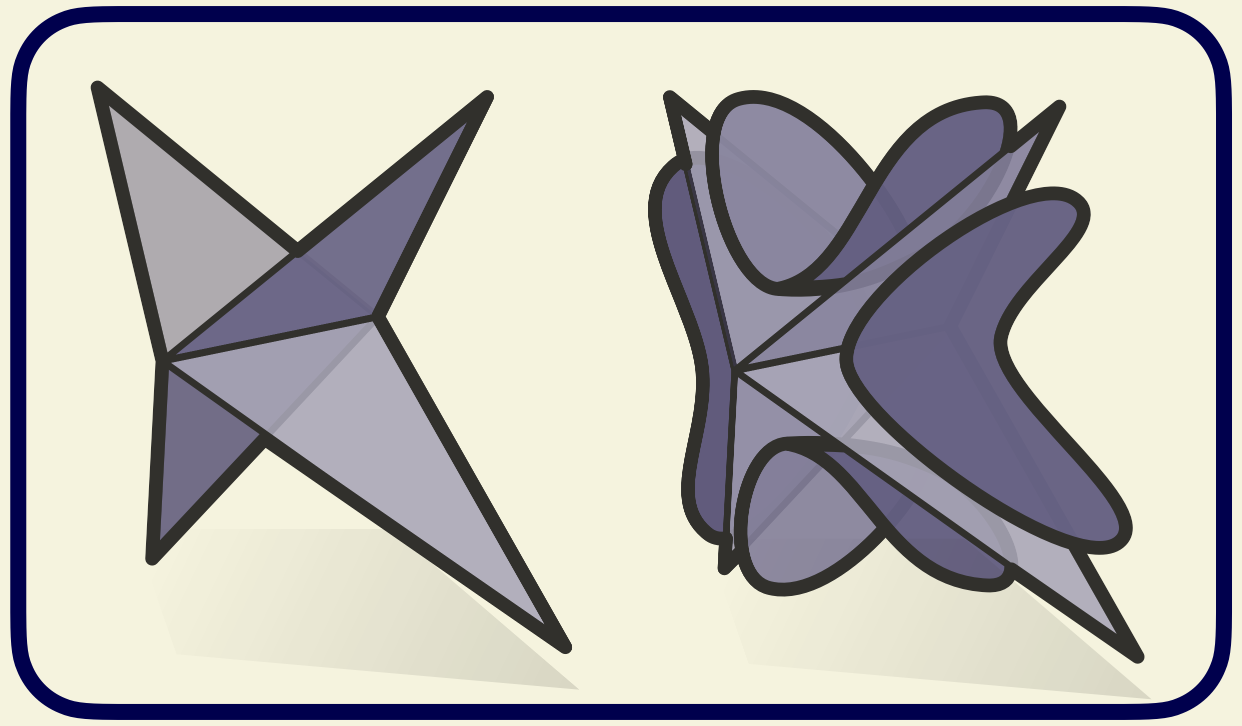
III. PARAMETERIZATION



[**Gillespie**, Springborn, & Crane. 2021. Discrete conformal equivalence of polyhedral surfaces. *ACM TOG*]

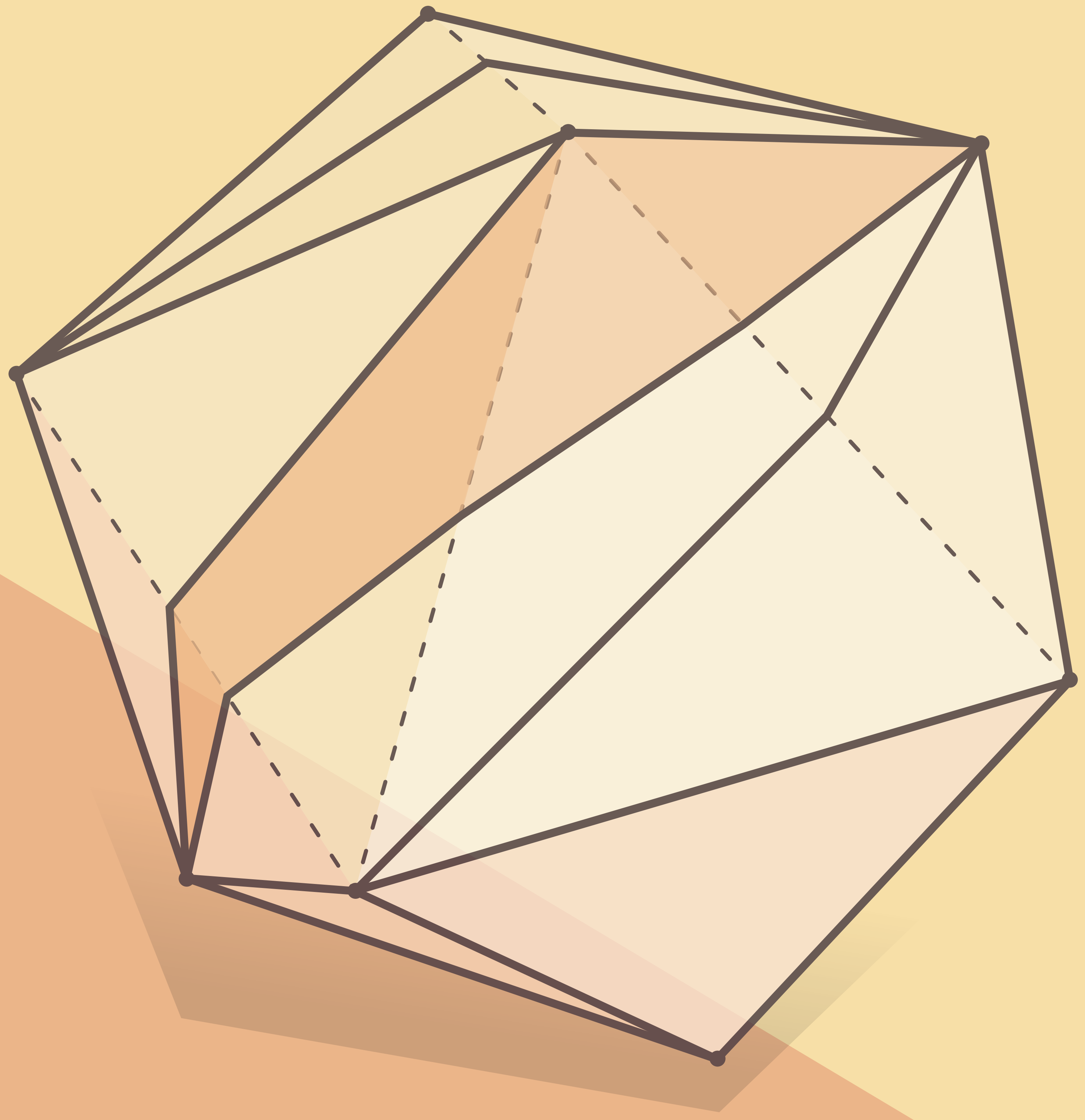
Track intrinsic triangulation while *flattening* a surface

IV. PROPOSED WORK

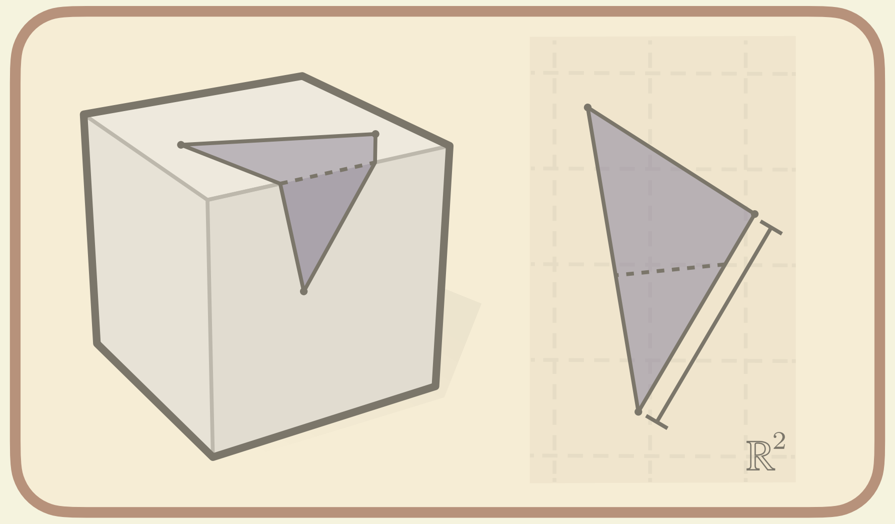


Track intrinsic triangulation on *more general* surfaces

I. Background

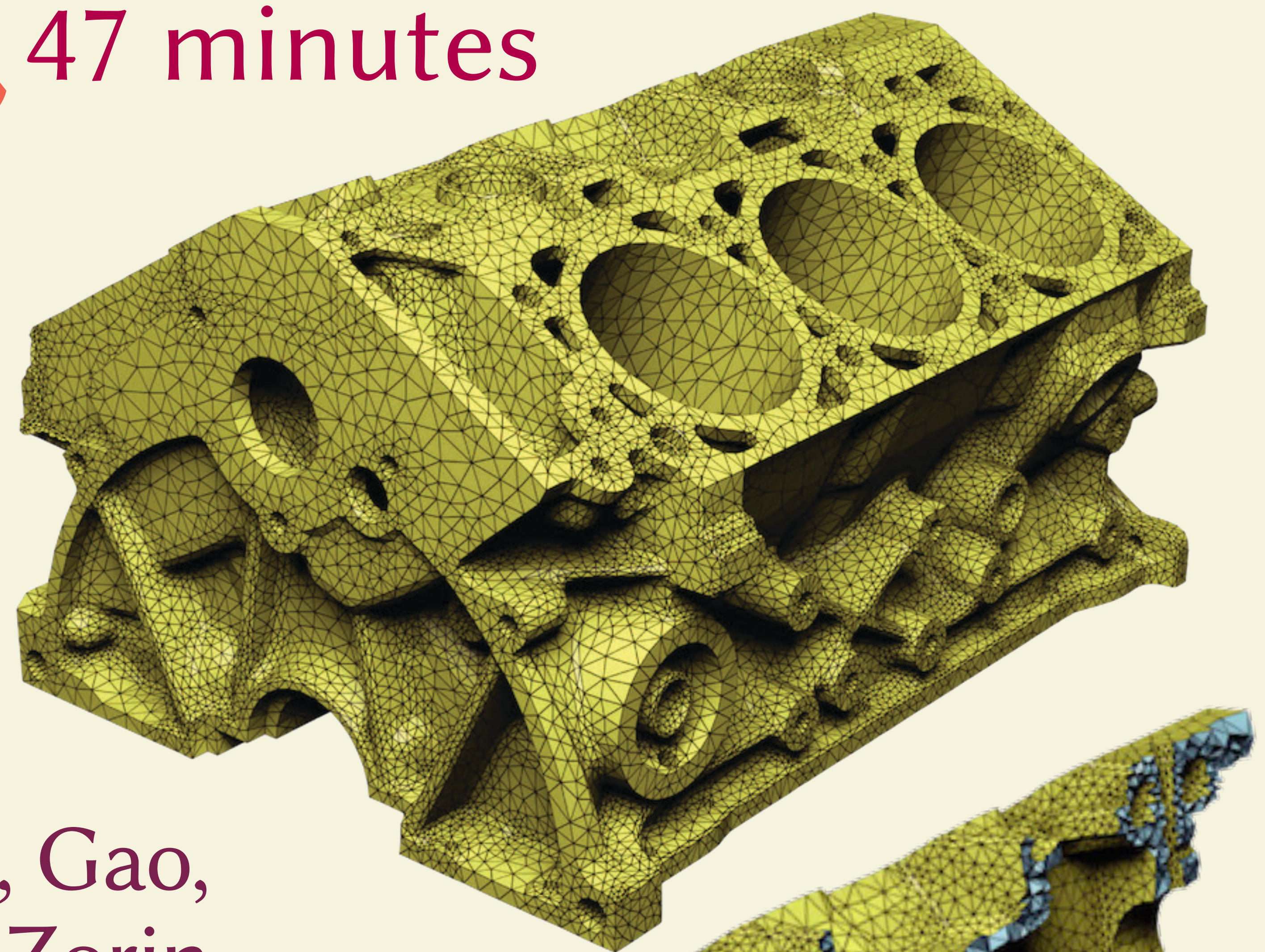


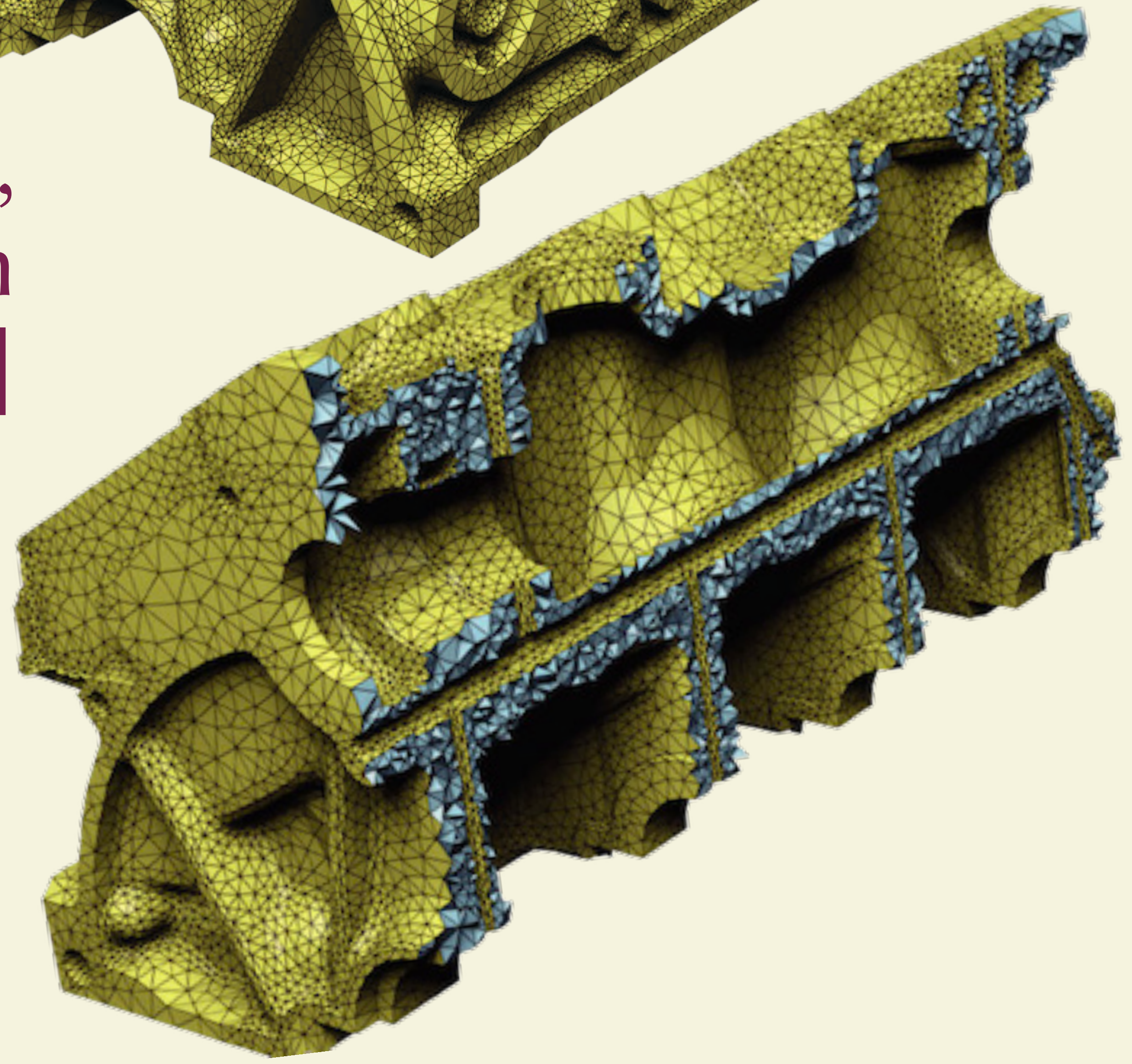
Status quo: remeshing



- State-of-the-art is robust but slow
  - Volumetric techniques

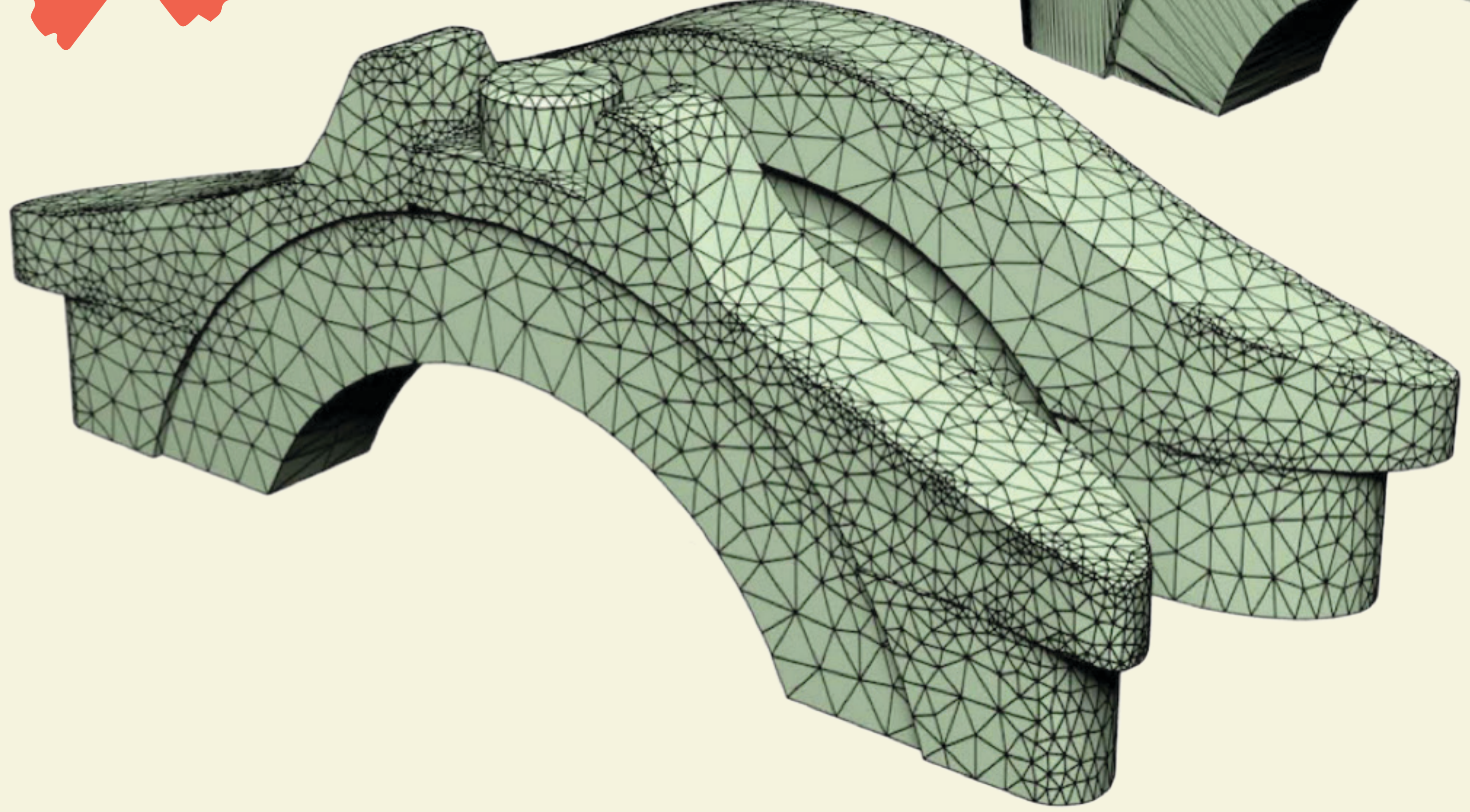
runtime:
47 minutes

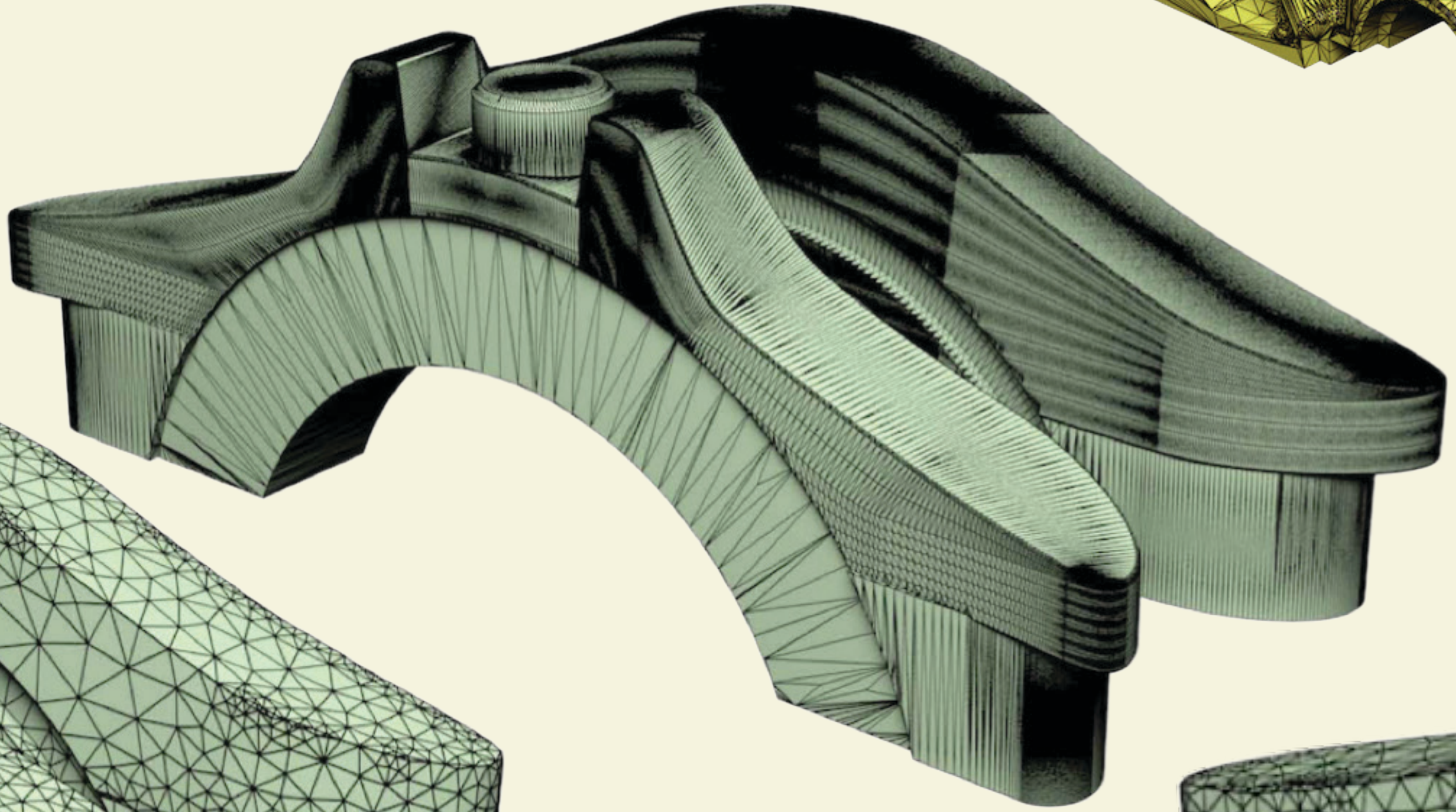


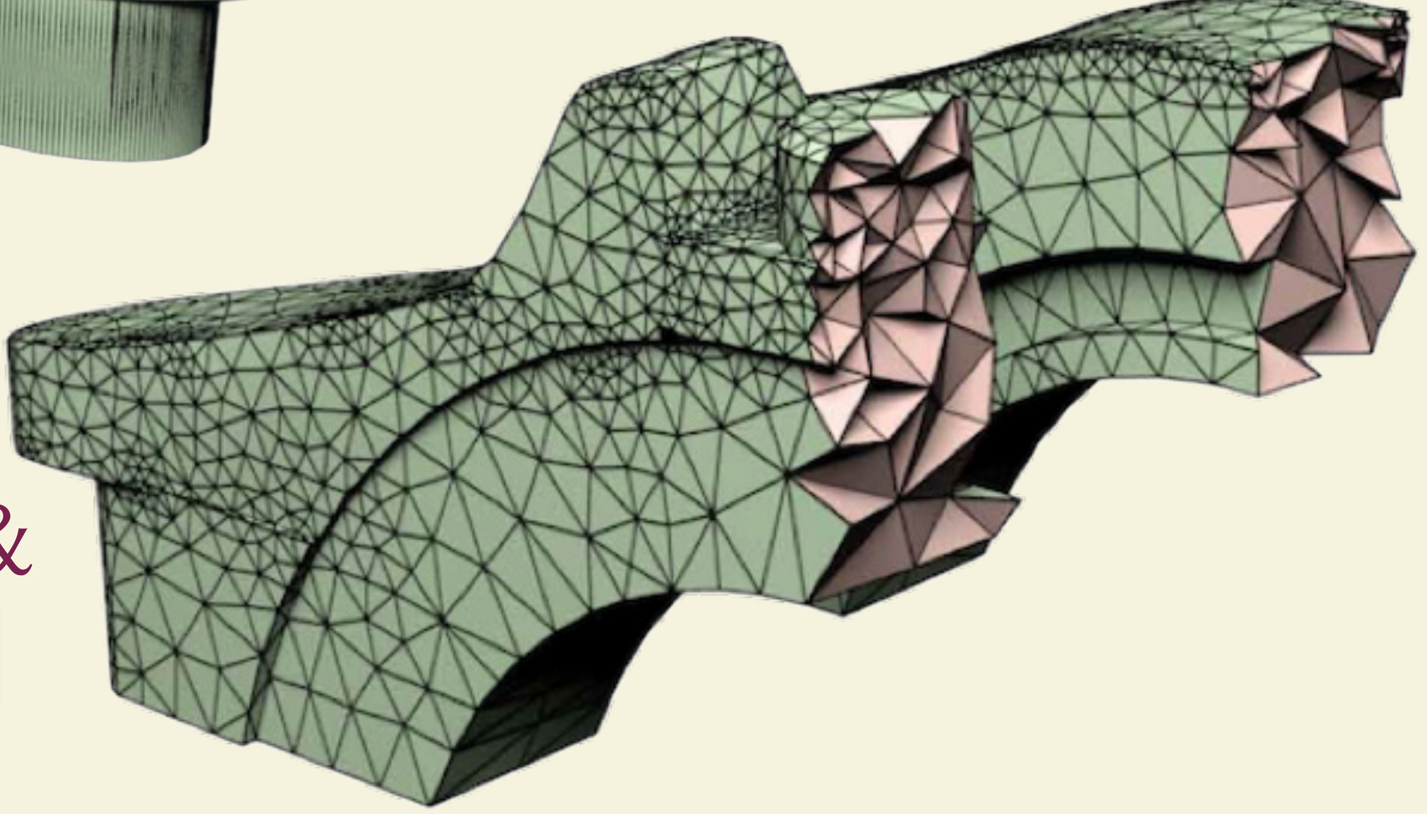


[Hu, Zhou, Gao, Jacobson, Zorin & Panozzo 2018]

runtime:
43 minutes

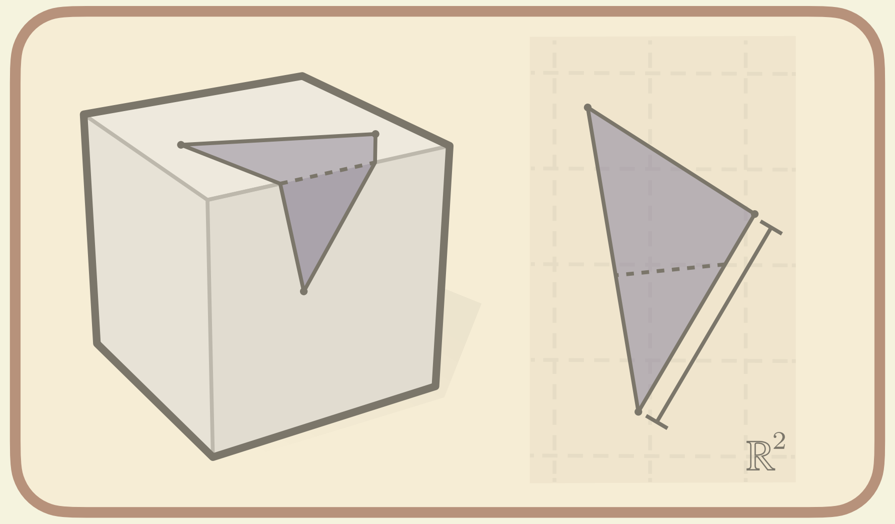


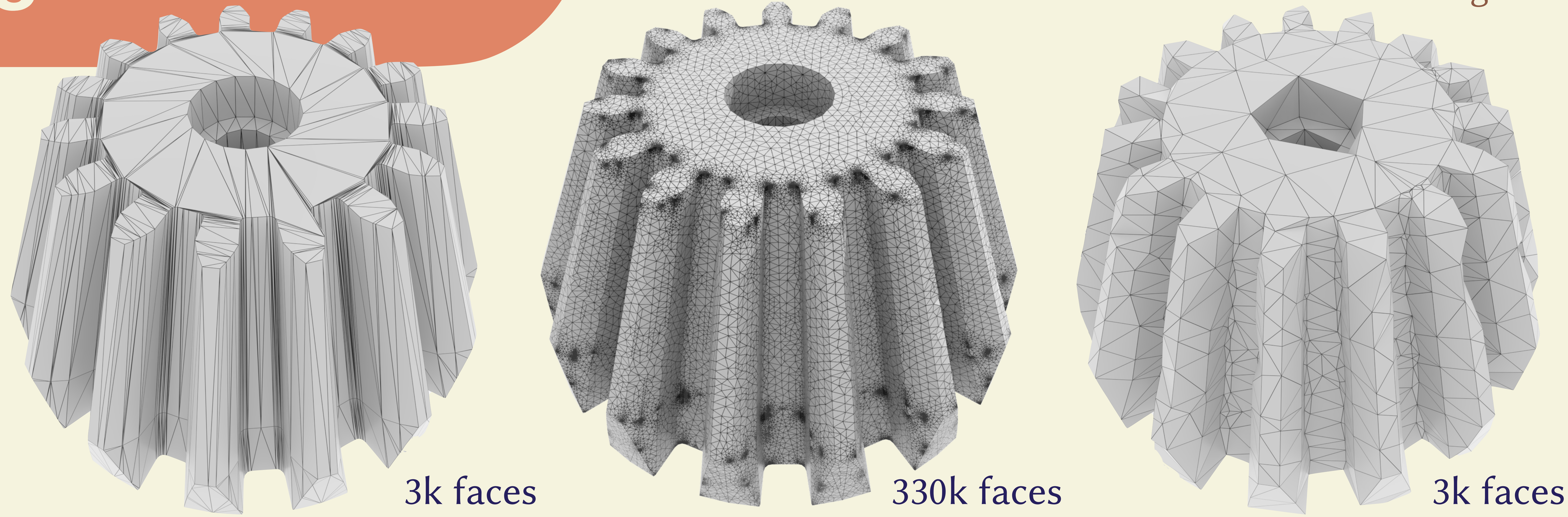




[Hu, Schneider, Wang, Zorin & Panozzo 2020]

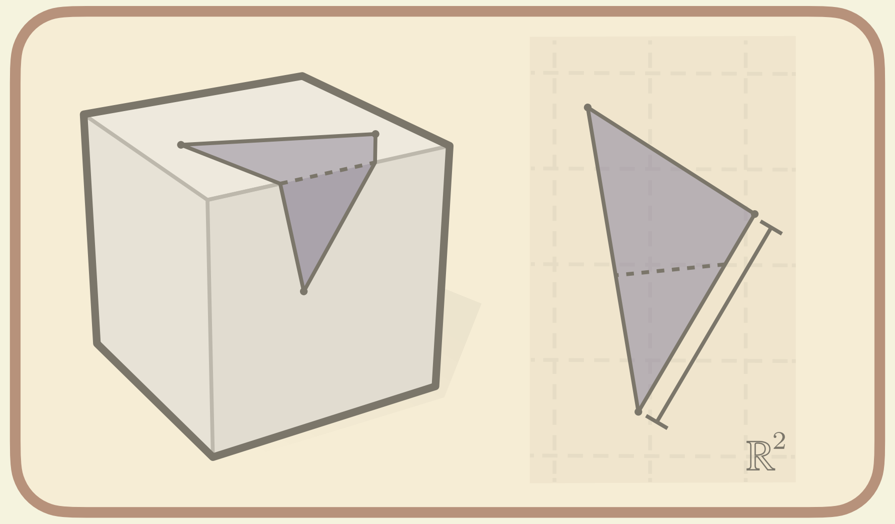
Trade offs of extrinsic remeshing

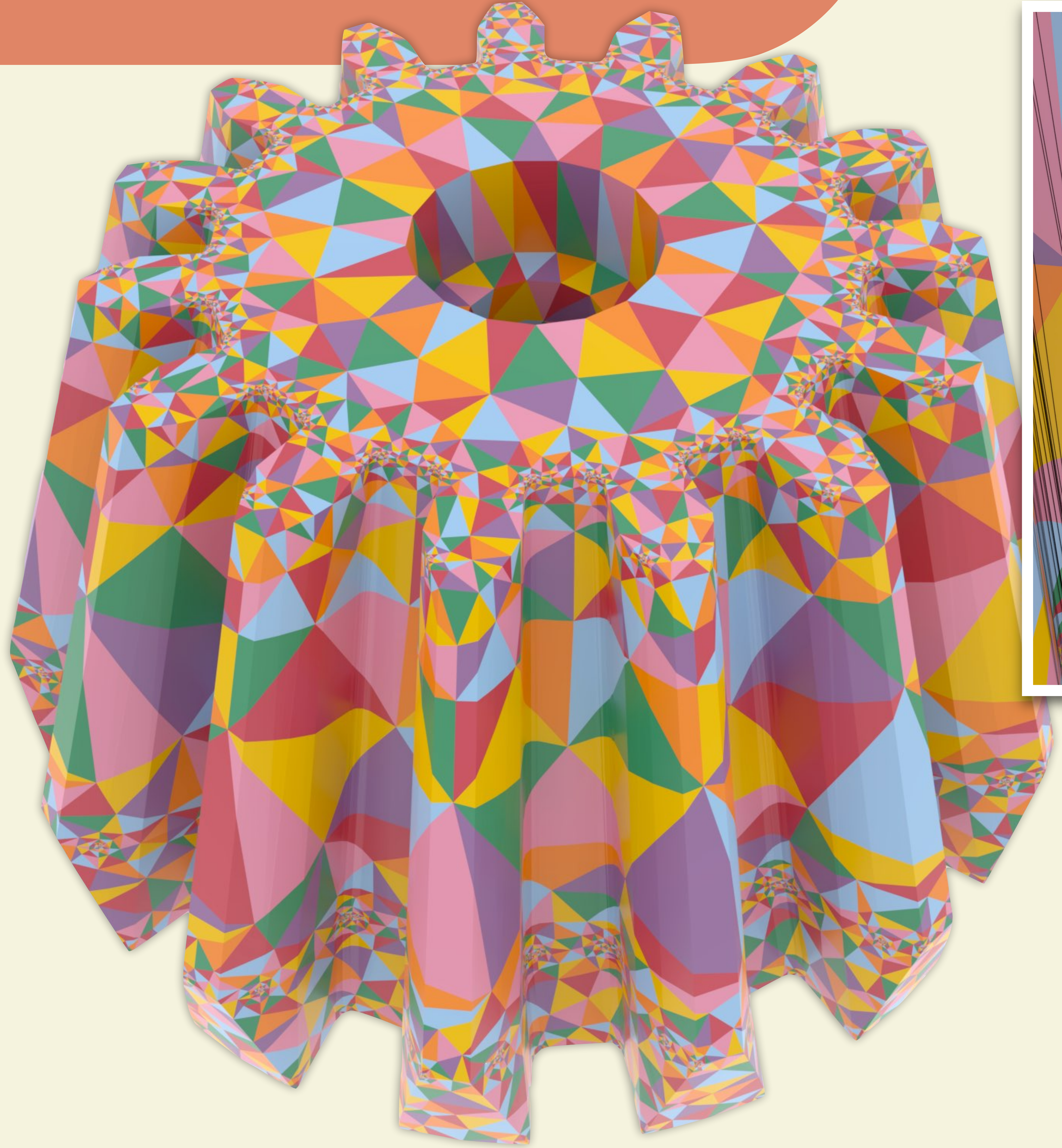


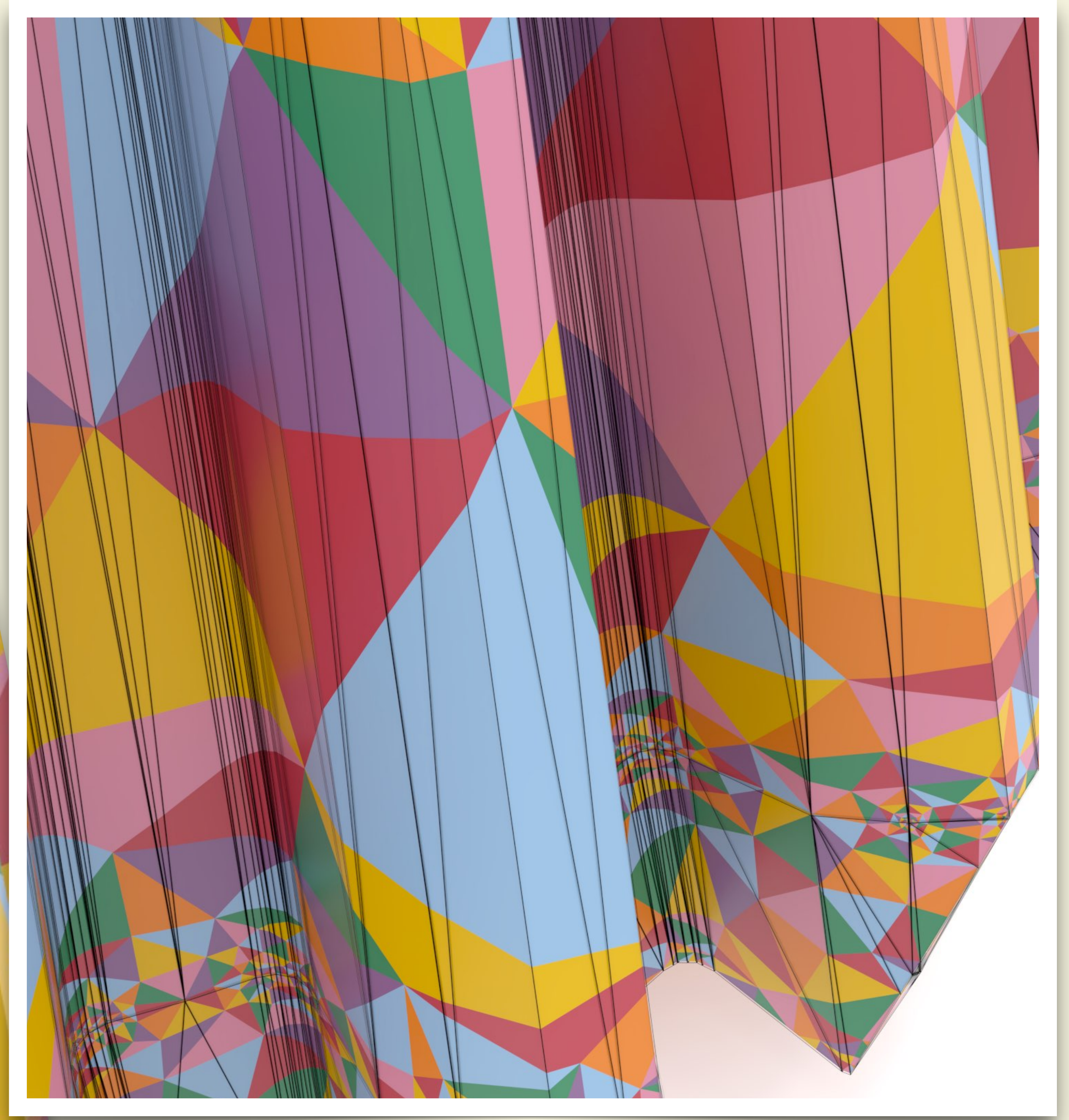


triangle quality	✘	✔	✔
mesh size	✔	✘	✔
geometric fidelity	✔	✔	✘

Intrinsic triangulations sidestep the trade off



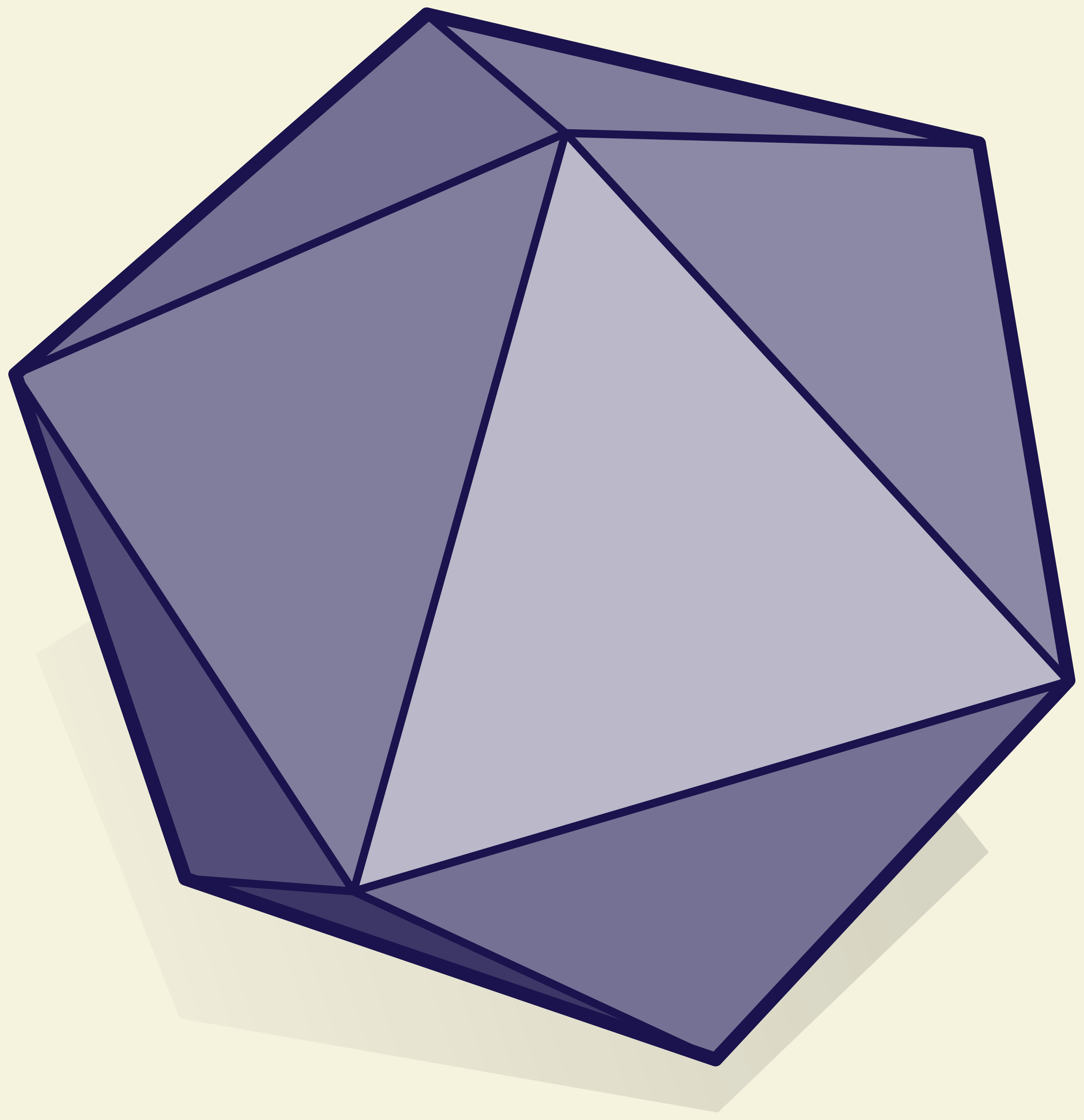




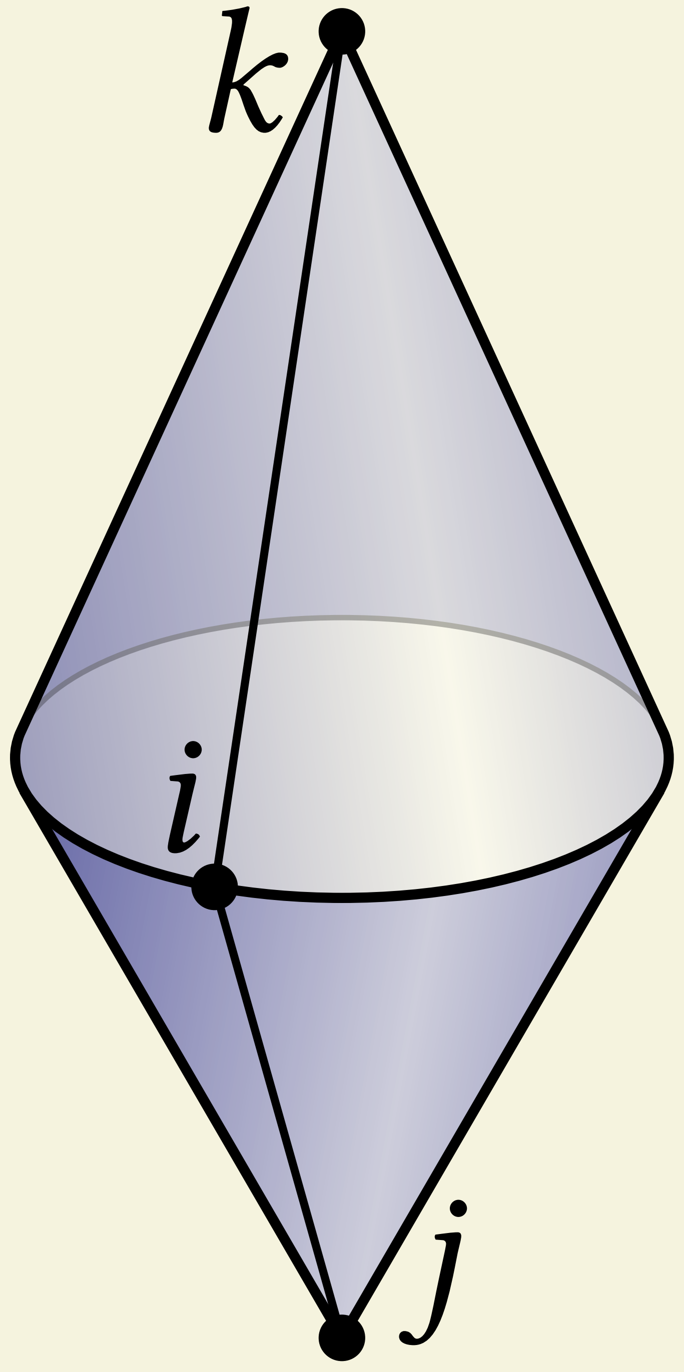
runs in seconds

Triangulations

> A *triangulation* is a collection of triangles glued together along their edges to form a surface



- Only combinatorial information
- May be *irregular* (*e.g.*, two edges of a face may be glued together)

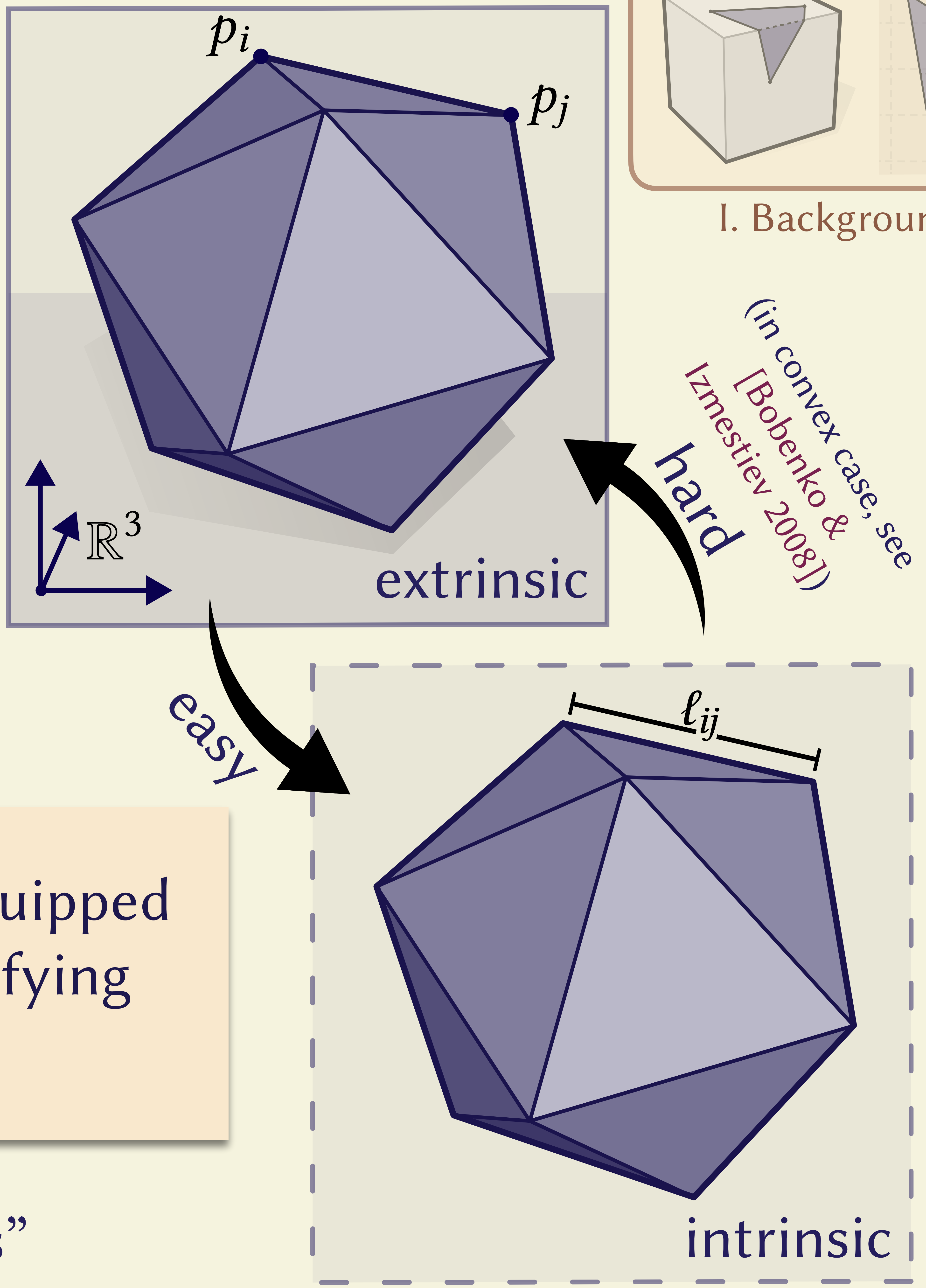


Extrinsic and intrinsic triangulations

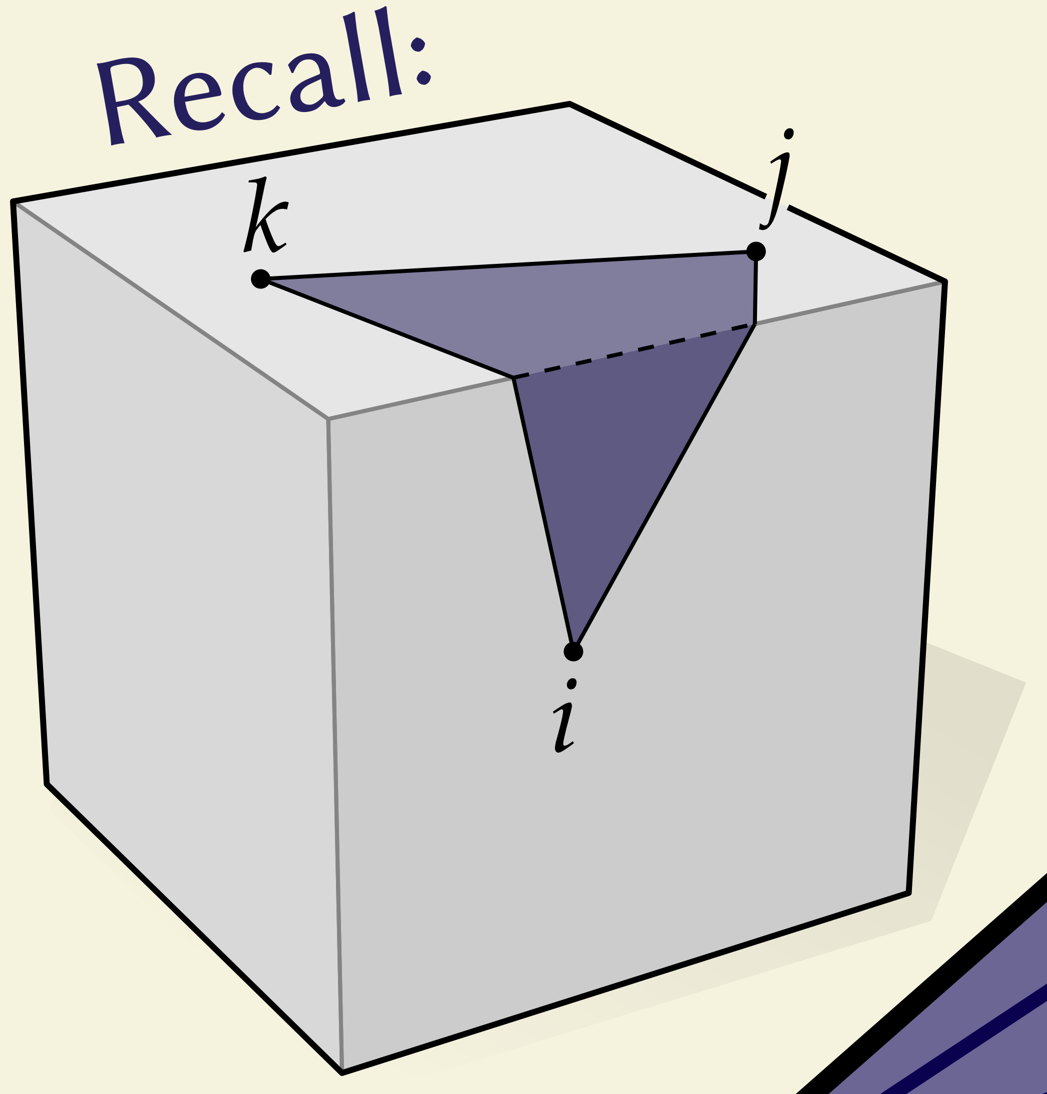
An *extrinsic triangulation* is a triangulation equipped with vertex positions $p : V \to \mathbb{R}^3$

An *intrinsic triangulation* is a triangulation equipped with positive edge lengths $\ell : E \to \mathbb{R}_{>0}$ satisfying the triangle inequality

I'll refer to both as "triangle meshes"

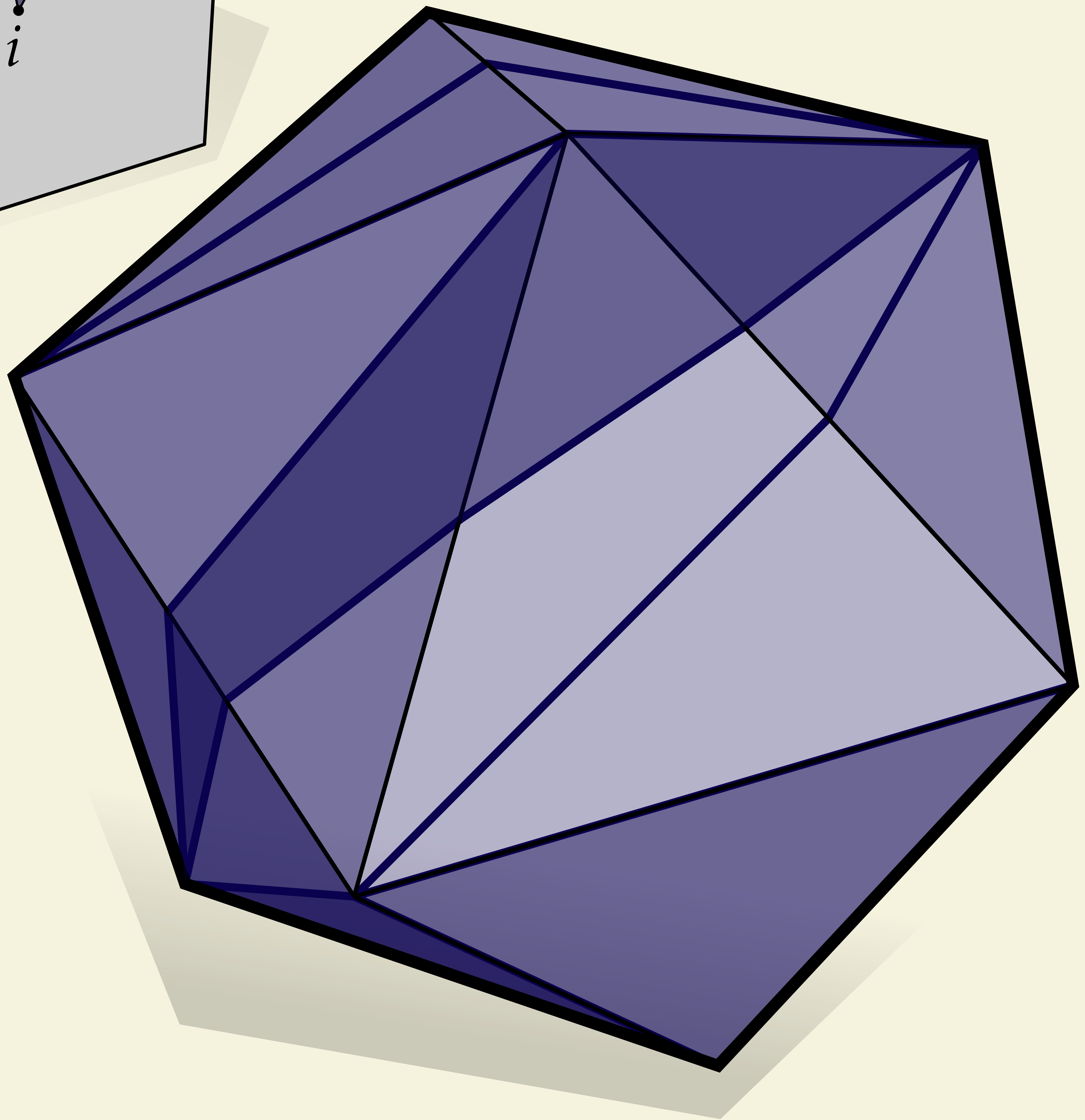


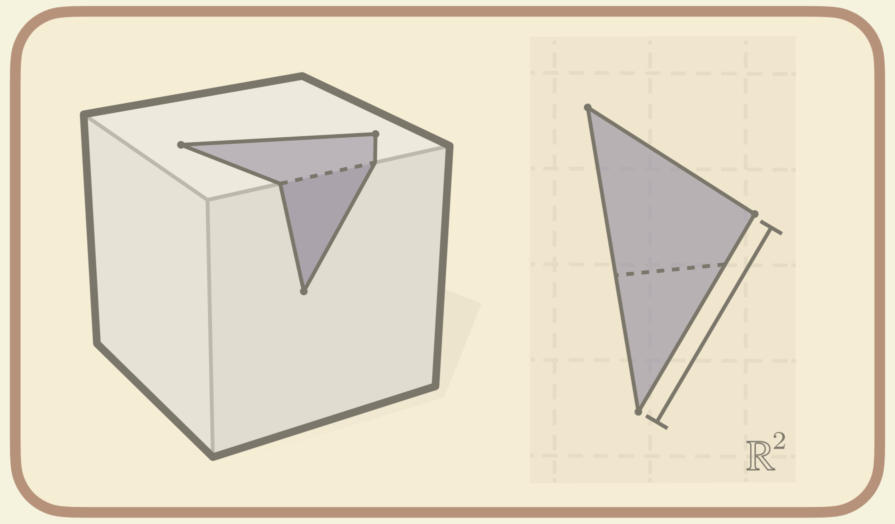
Correspondence



A *correspondence* between two triangulations is a function mapping one onto the other

- Traditional case: intrinsic triangulation sitting on top of an extrinsic triangulation
  - Exact same geometry

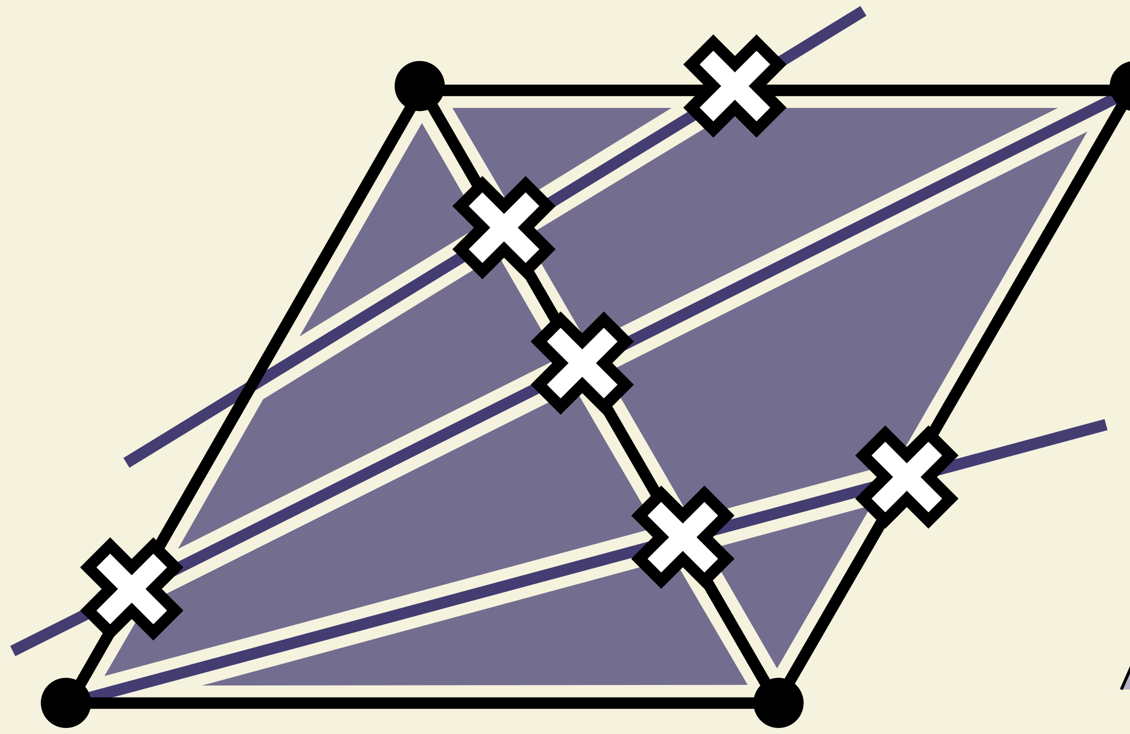




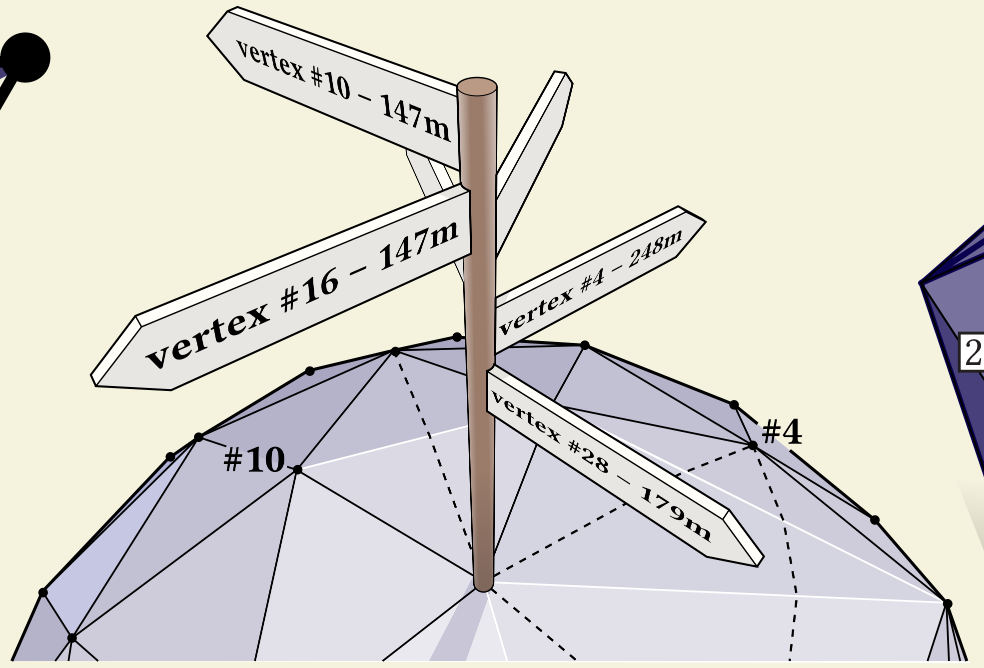
The challenge of dynamic intrinsic triangulations

- Tracking correspondence between meshes with different geometry

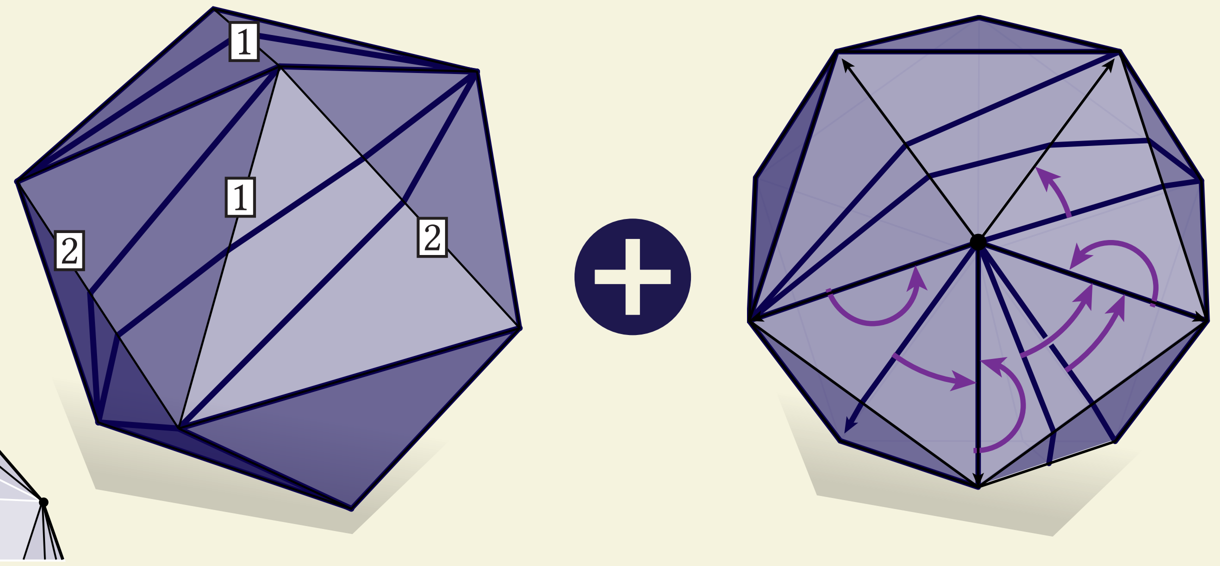
Correspondence with same geometry



[Fisher, Springborn, Bobenko & Schröder 2006]



[Sharp, Soliman & Crane 2019]

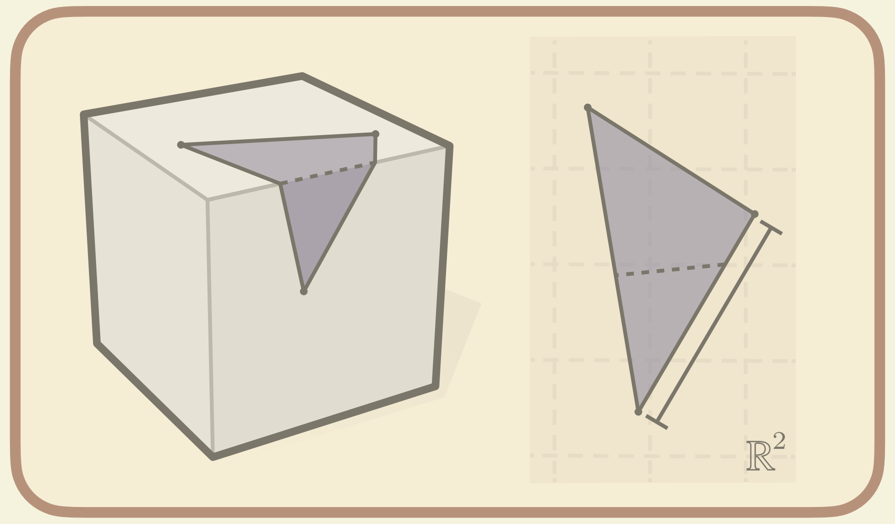


[Gillespie, Sharp & Crane 2021]

Correspondence with different geometry

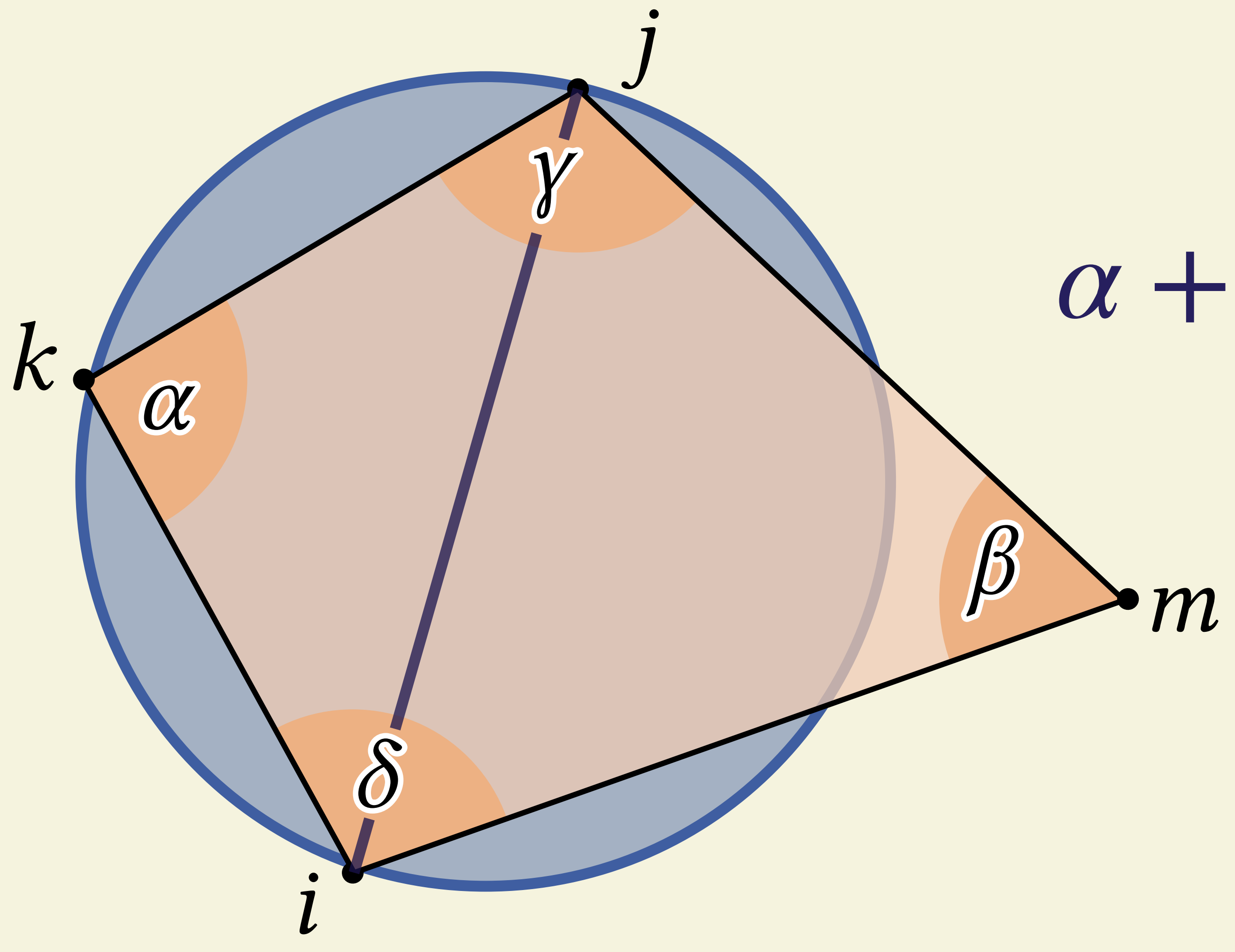
???

The space of intrinsic triangulations is large

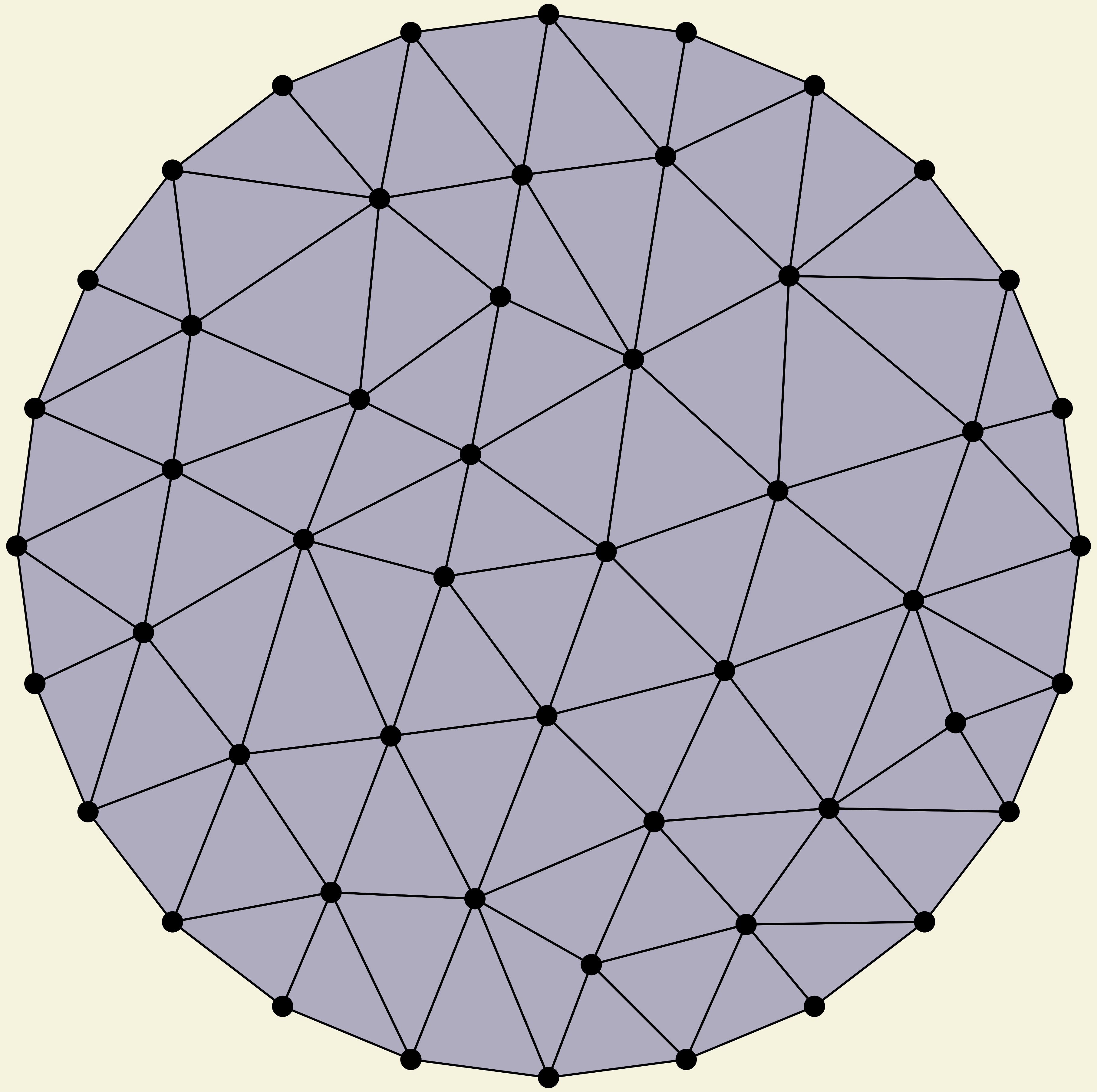


Delaunay triangulations

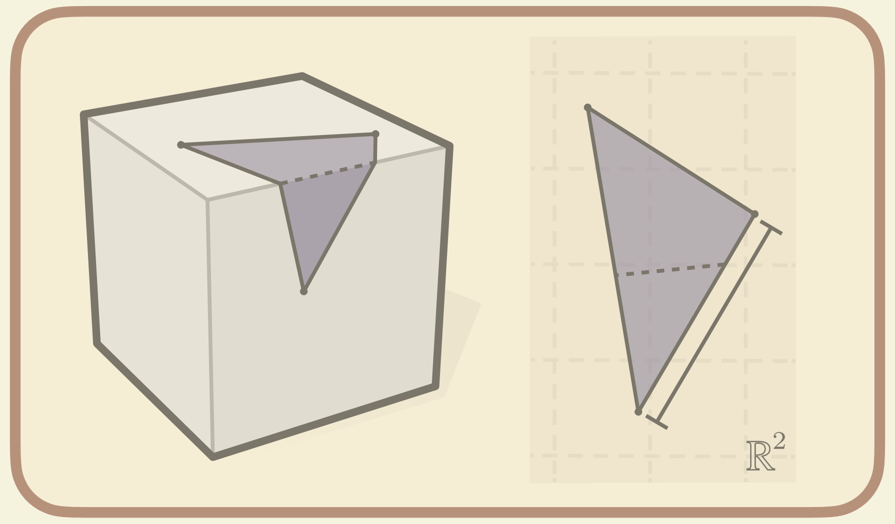
- Countless useful properties:
  - Essentially unique, maximize angles lexicographically, minimize spectrum lexicographically, smoothest interpolation, positive cotan weights...
- Characterized by *empty circumcircle condition*



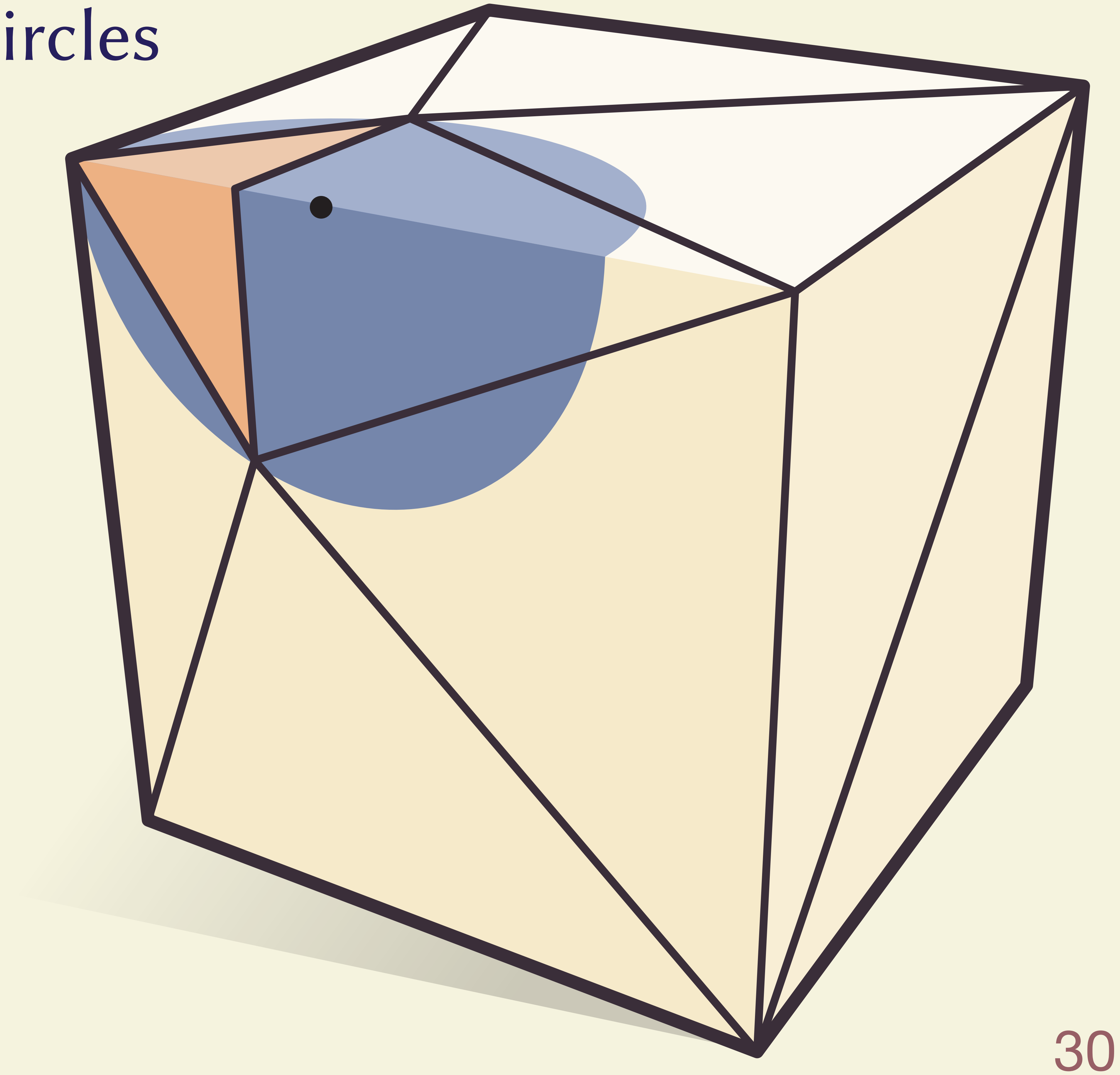
$$\alpha + \beta \leq \gamma + \delta$$

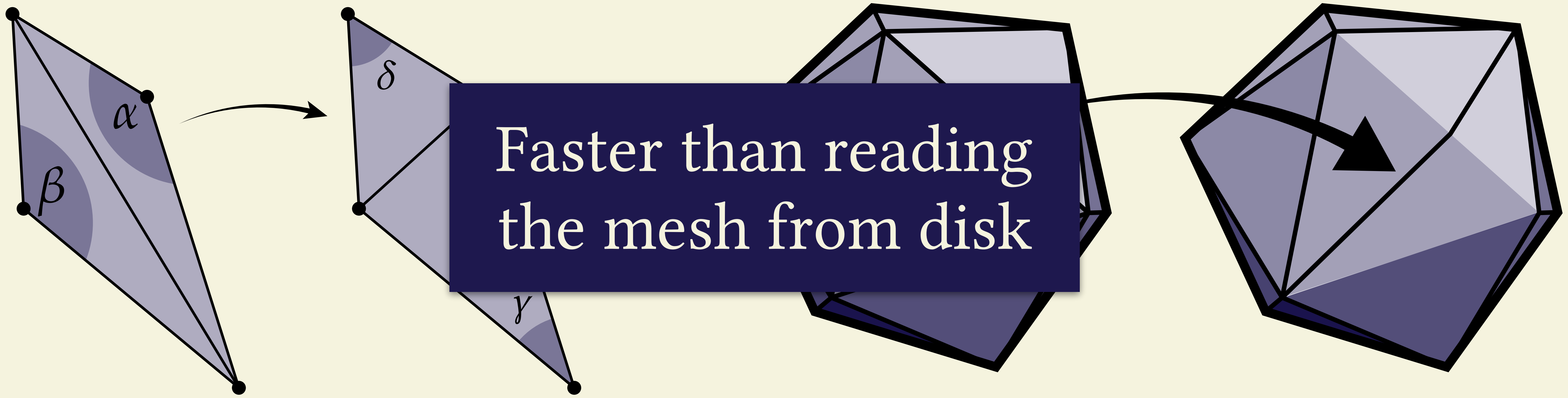


Intrinsic Delaunay triangulations



- [Indermitte, Liebling, Troyanov & Clemençon 2001, Bobenko & Springborn 2007]: empty intrinsic circumcircles
    - Maintain many nice properties. [Sharp, **Gillespie** & Crane 2021; §4.1.1]
- Compute by a simple algorithm:
    - Flip any non-Delaunay edge until none remain



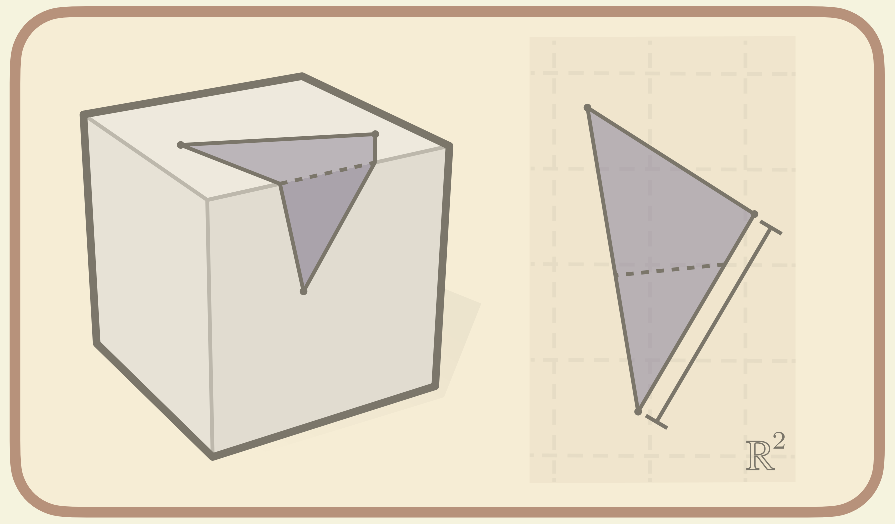


Intrinsic Delaunay triangulations provide good function spaces

original mesh

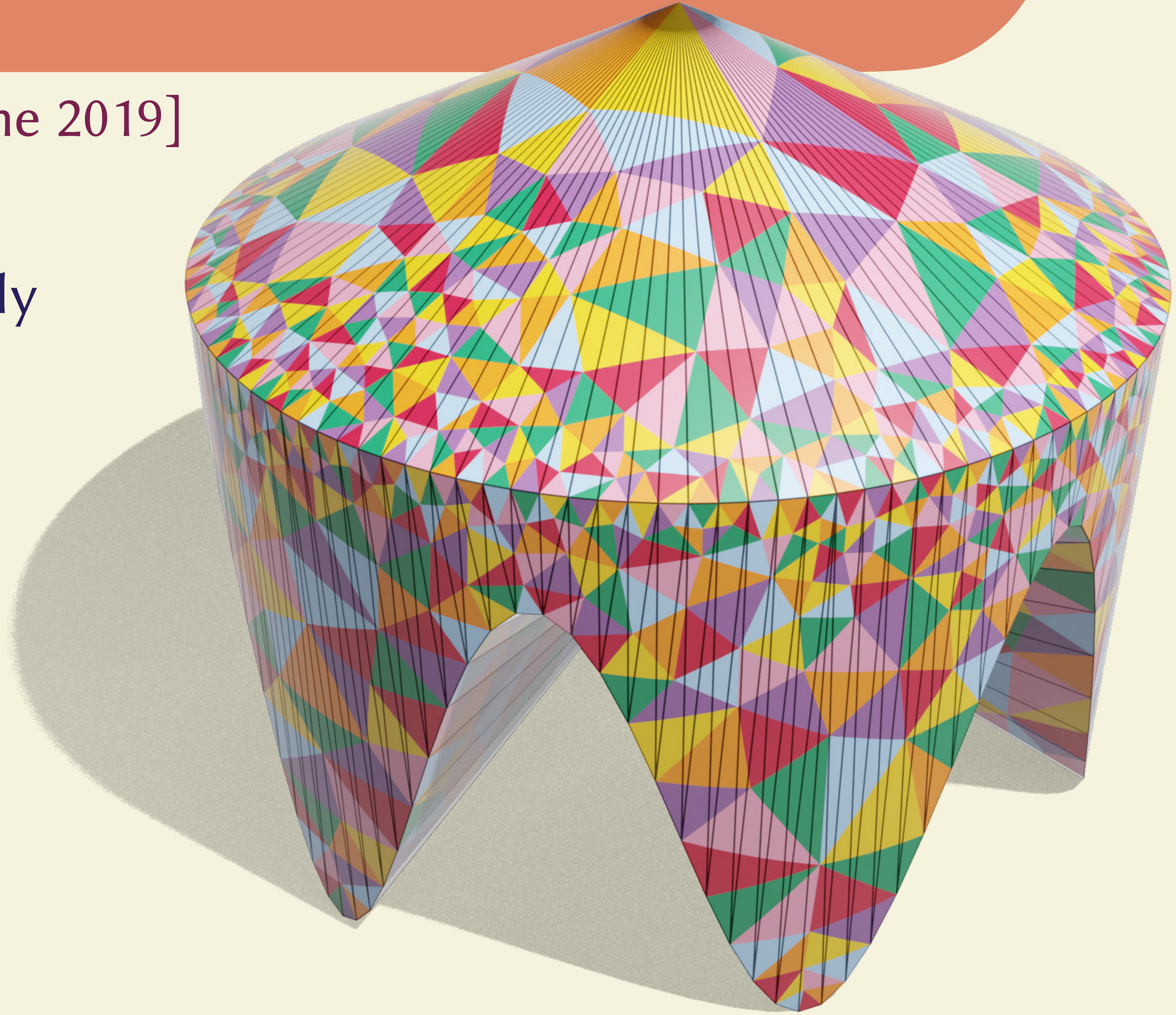
intrinsic Delaunay triangulation

Intrinsic Delaunay Refinement

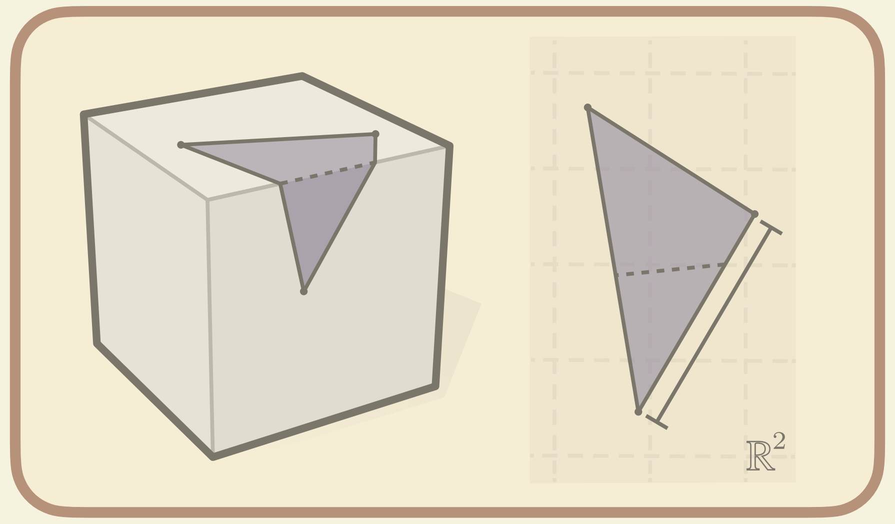


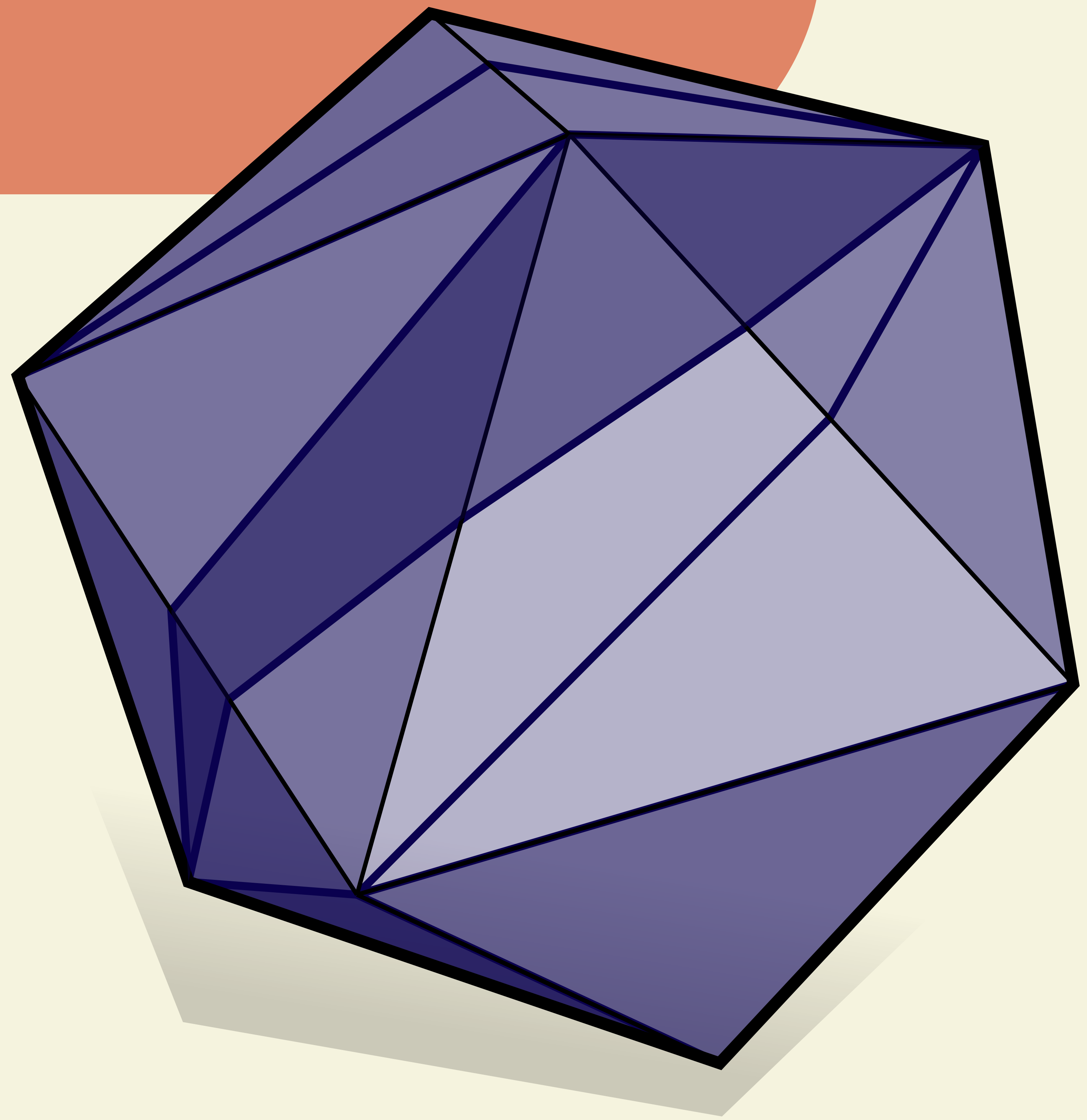
[Sharp, Soliman & Crane 2019]

Add vertices intrinsically
to improve quality



A brief history of intrinsic triangulations





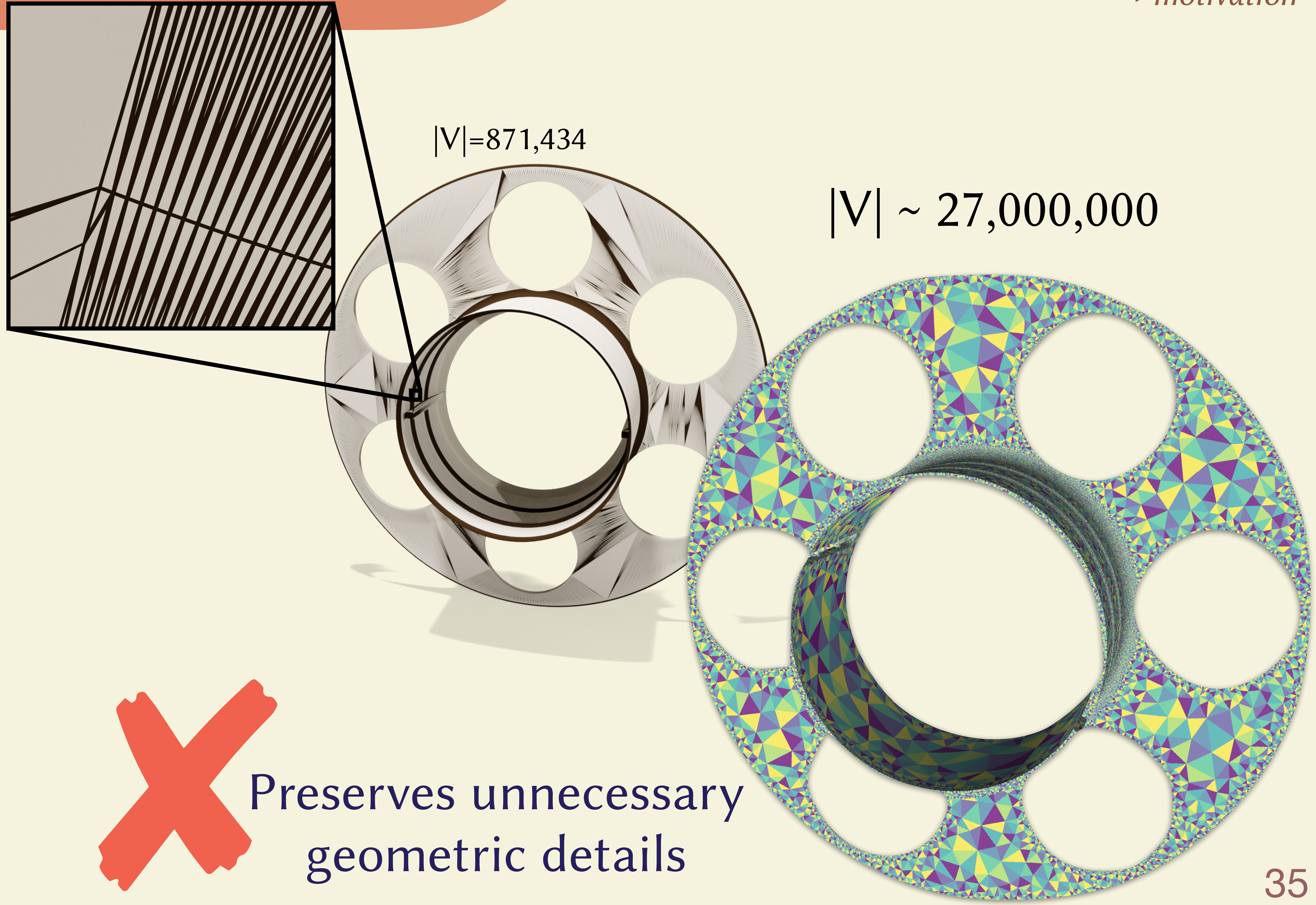
Foundations: [Alexandrov 1948; Regge 1961]

Geometry Processing: [Fisher, Springborn, Bobenko & Schröder 2006; Bobenko & Springborn 2007, Bobenko & Izmestiev 2008; Sun, Wu, Gu & Luo 2015; Sharp, Soliman & Crane 2019; Fumero, Möller & Rodolà 2020; Gillespie, Springborn & Crane 2021; Finnendahl, Schwartz & Alexa 2023]

II. Intrinsic Simplification

[Liu, **Gillespie**, Chislett, Sharp, Jacobson, & Crane. 2023. Surface Simplification using Intrinsic Error Metrics. *ACM Transactions on Graphics*]

Exact geometry preservation: a blessing and a curse



✓ Compute geometric quantities directly on the original surface

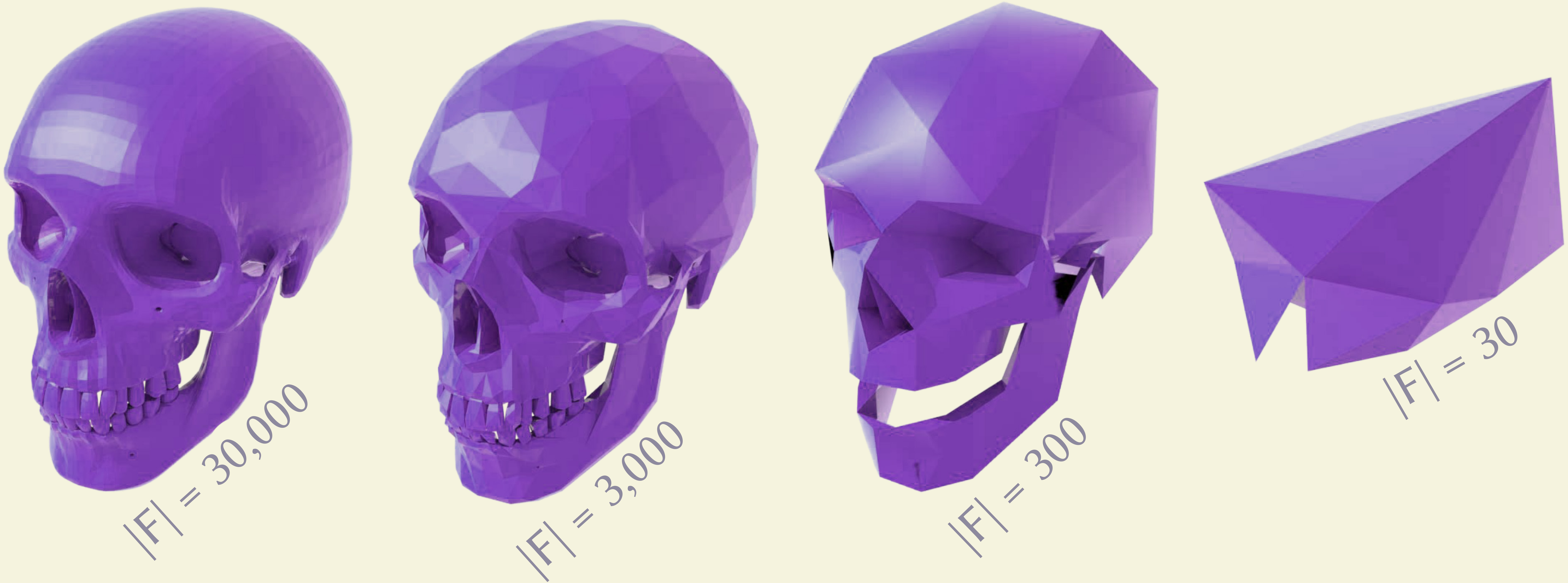
✗ Preserves unnecessary geometric details

Coarse meshes can be perfectly adequate

Coarse meshes can be perfectly adequate

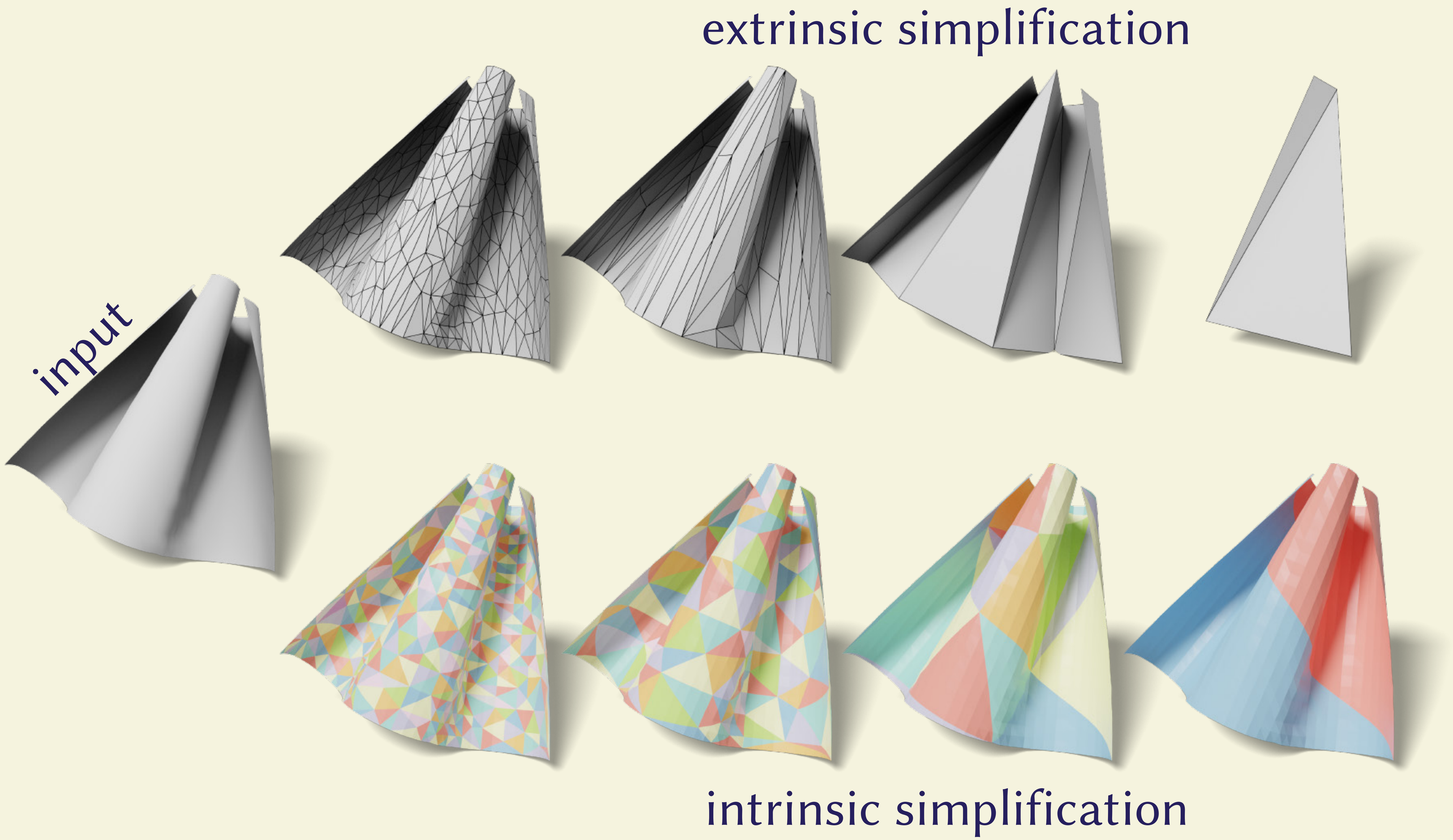
Traditional goal: *extrinsic* simplification

- Find a coarse mesh close in space to the original
  - Often designed to optimize for visual fidelity



Intrinsic problems benefit from intrinsic simplification

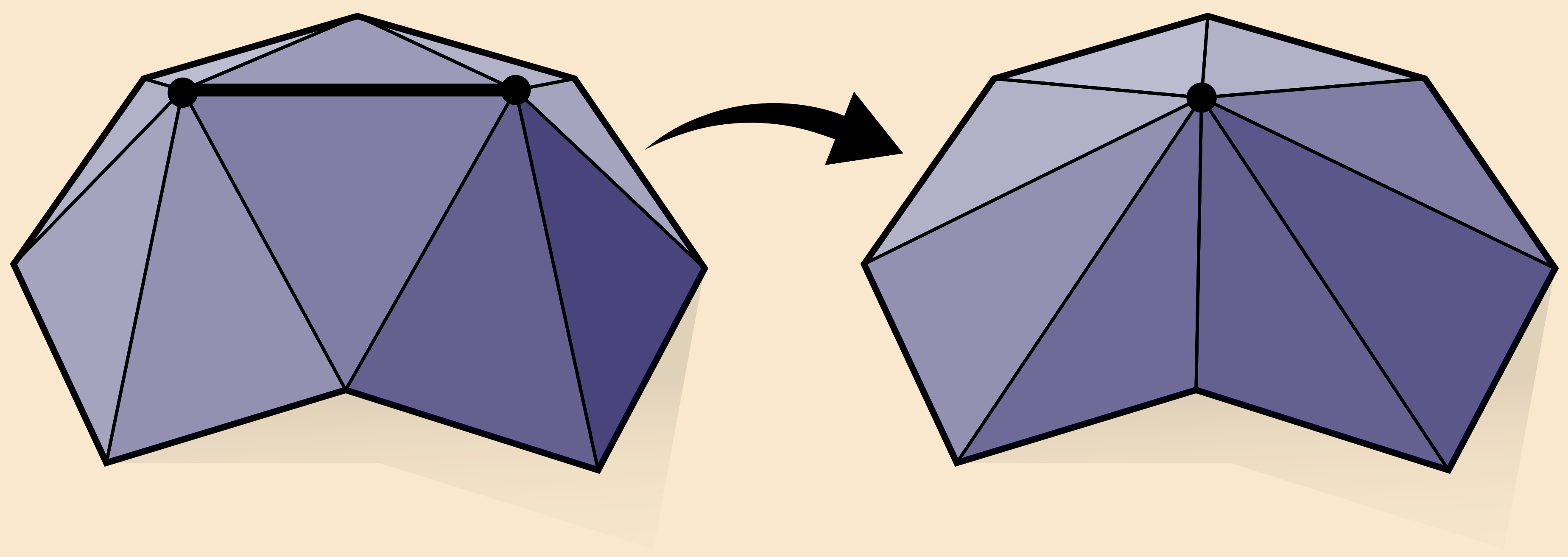
- Extrinsic methods preserve irrelevant extrinsic details
- Intrinsic approach opens up a larger space of triangulations
- Extreme example: near-developable surfaces



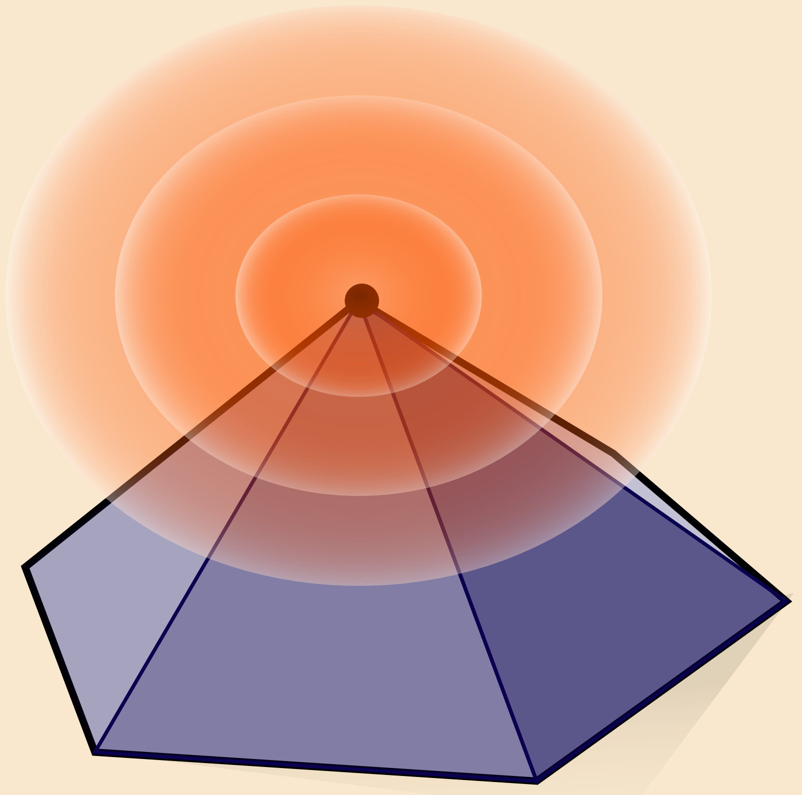
Inspiration: quadric error simplification

[Garland & Heckbert 1997]

1. Local simplification operation



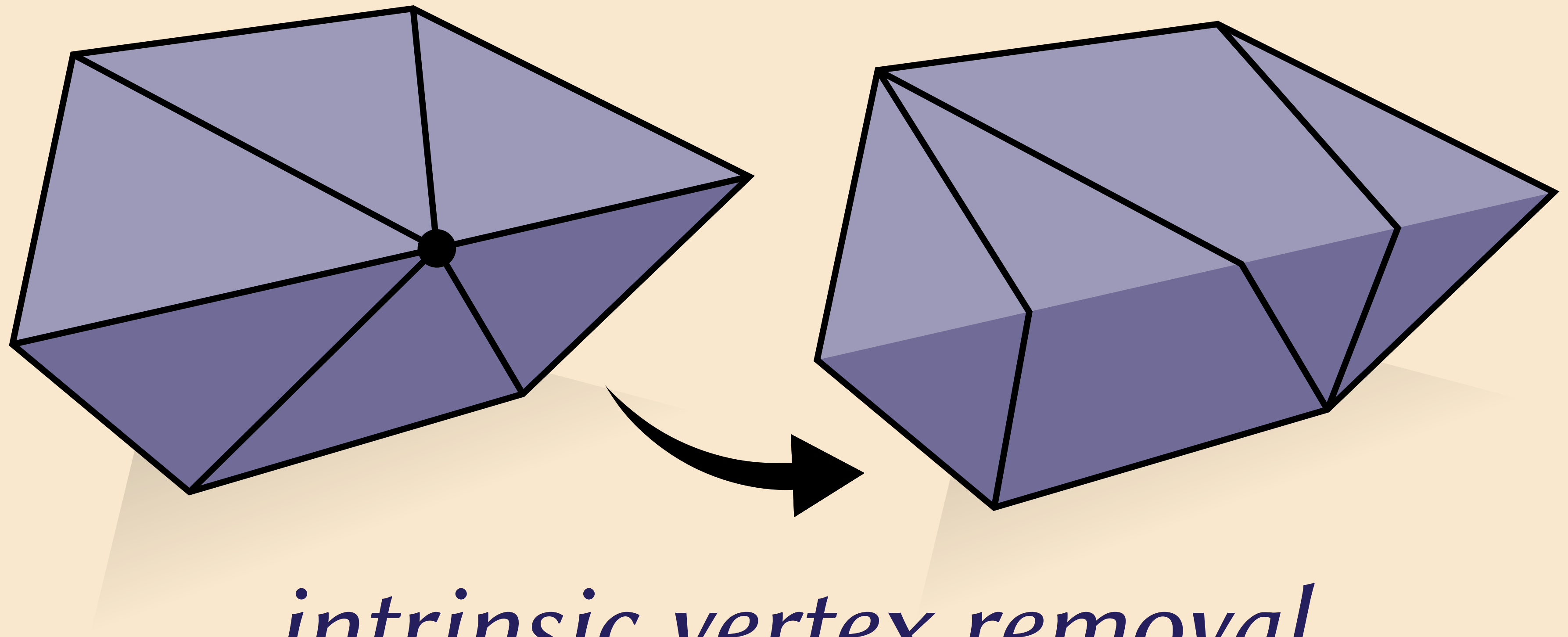
2. Accumulated distortion measurements



- Algorithm: repeatedly collapse cheapest edge
  - Efficient: all local operations
  - Accurate: accumulates error estimates

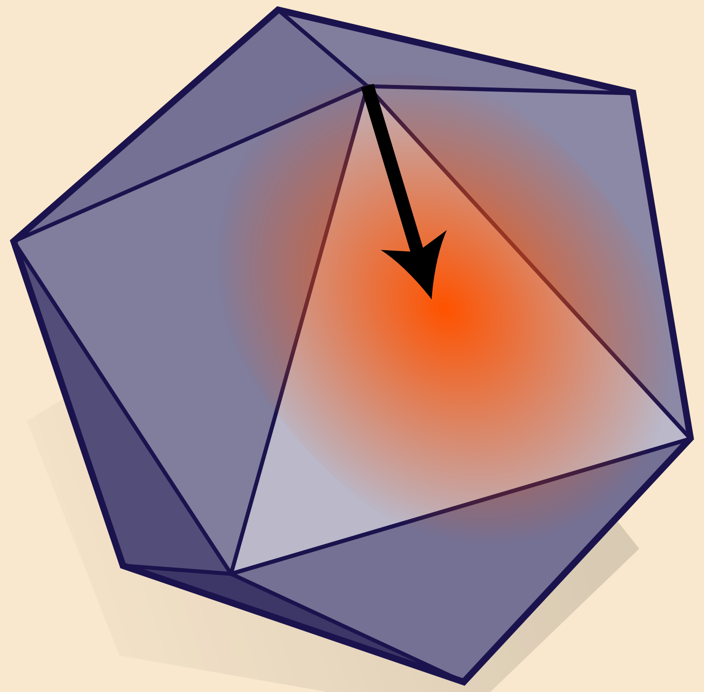
Intrinsic simplification

1. Local simplification operation



intrinsic vertex removal

2. Accumulated distortion measurements

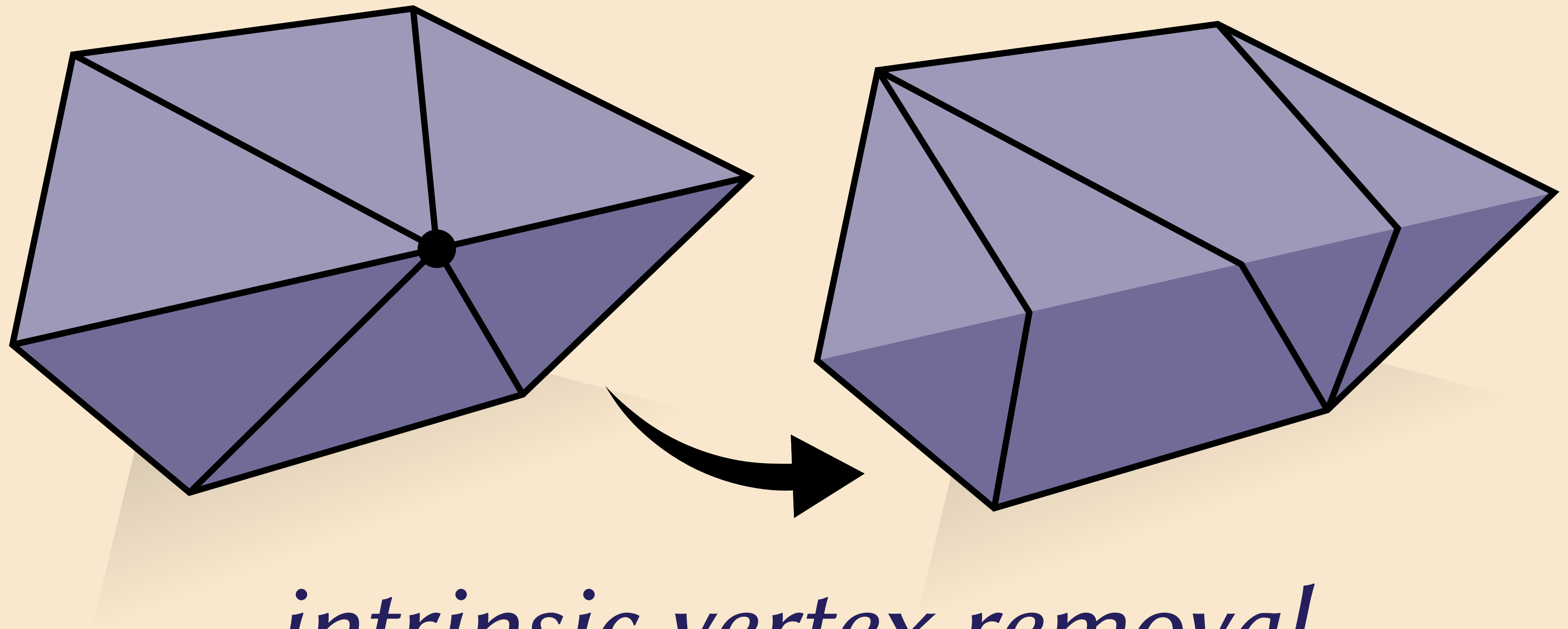


intrinsic curvature error

- Algorithm: repeatedly remove cheapest vertex

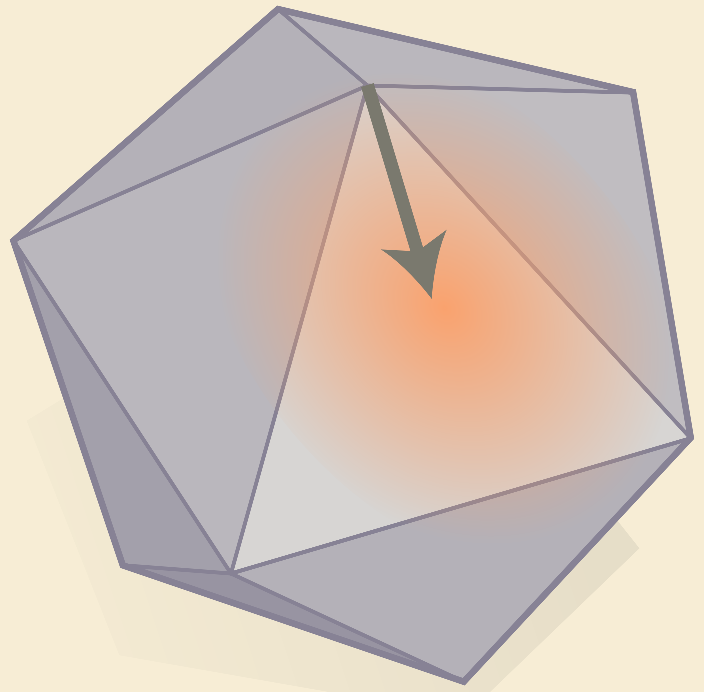
Intrinsic simplification

1. Local simplification operation



intrinsic vertex removal

2. Accumulated distortion measurements

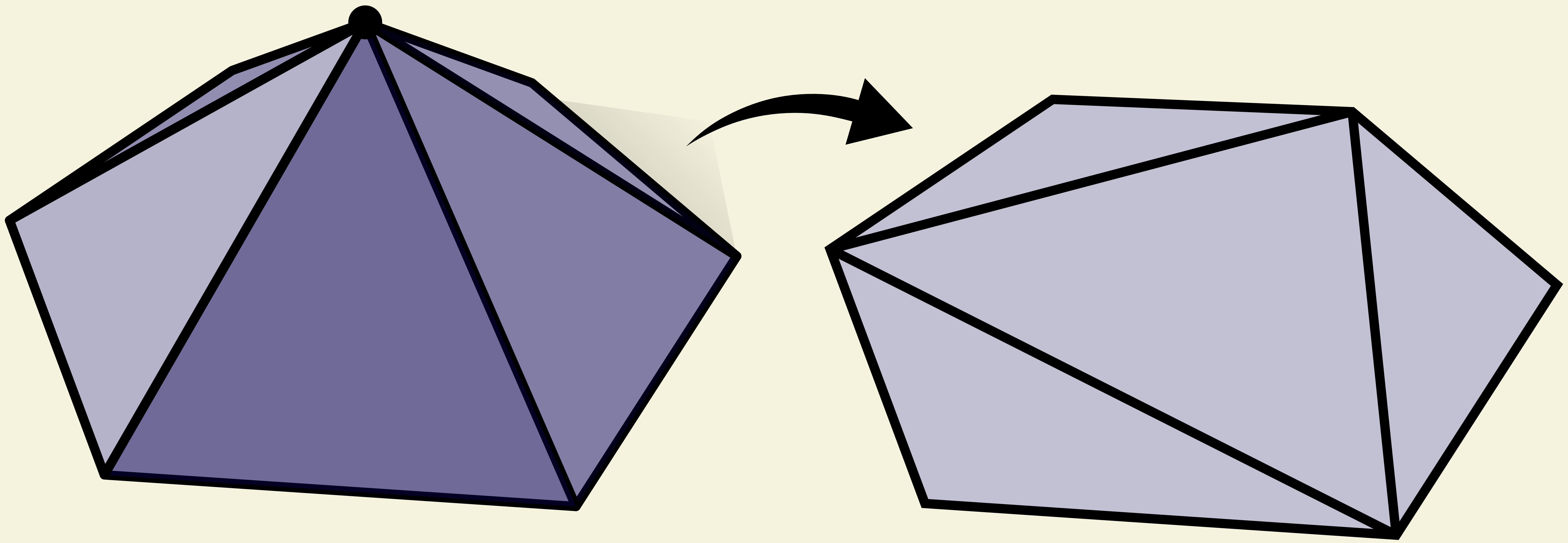


intrinsic curvature error

- Algorithm: repeatedly remove cheapest vertex

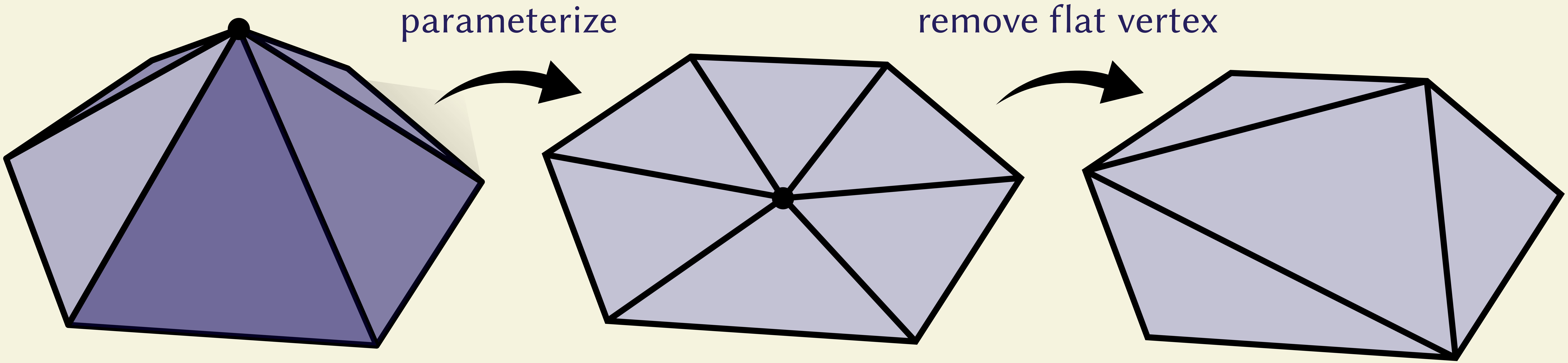
Intrinsic vertex removal

- Intrinsic view: replace curved vertex with flat patch

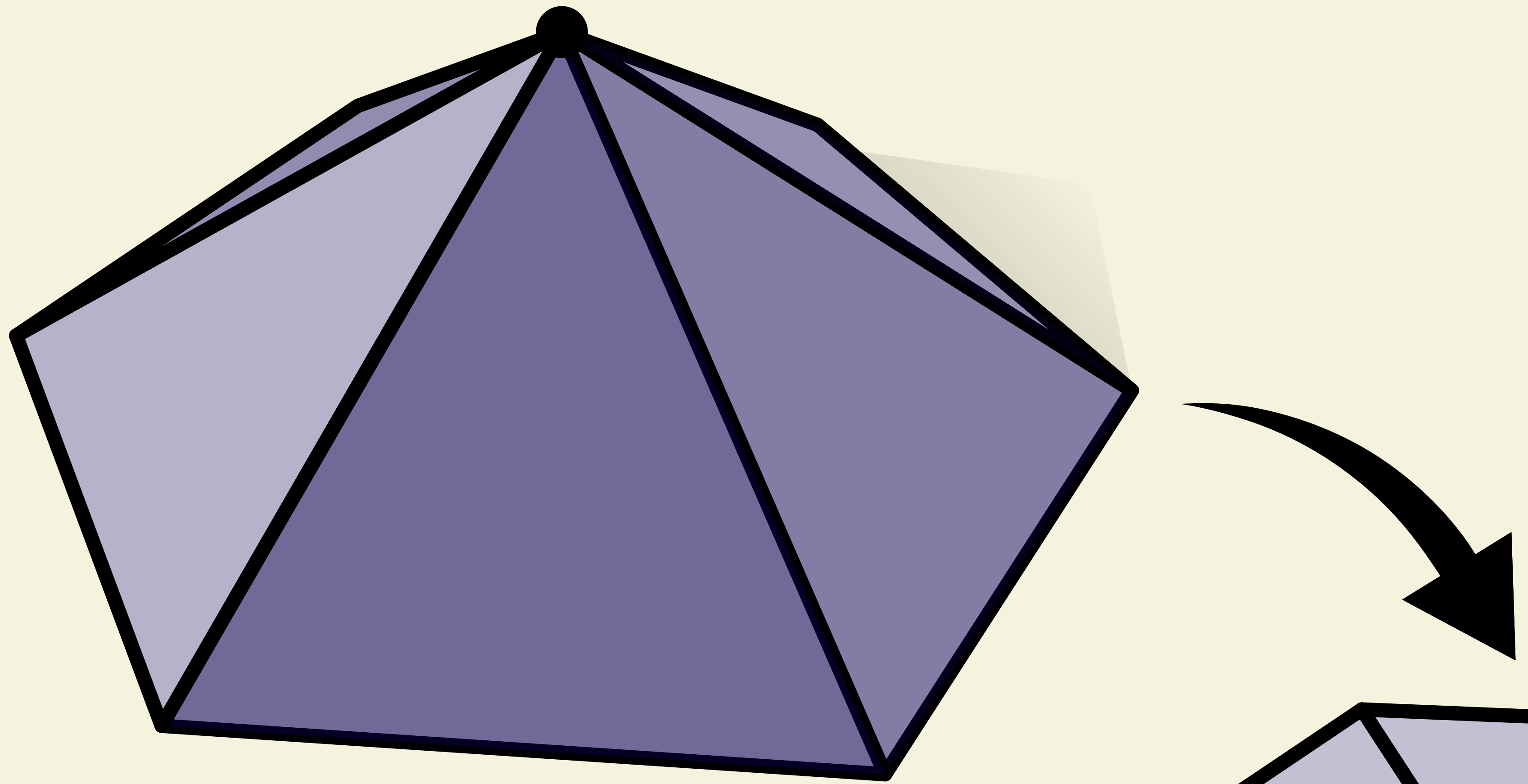


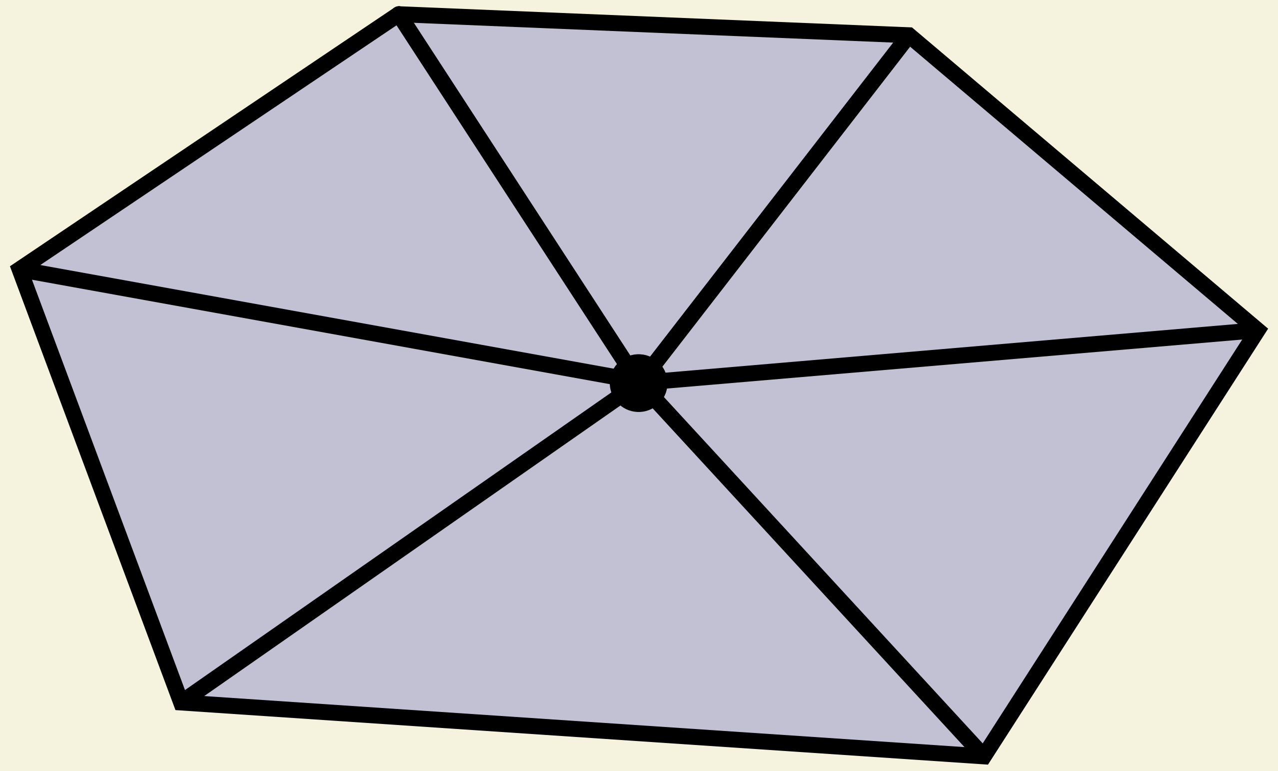
Intrinsic vertex removal

- Intrinsic view: replace curved vertex with flat patch



Vertex flattening





II. Intrinsic simplification
‣ *intrinsic vertex removal*

- Map neighboring triangles to plane such that:

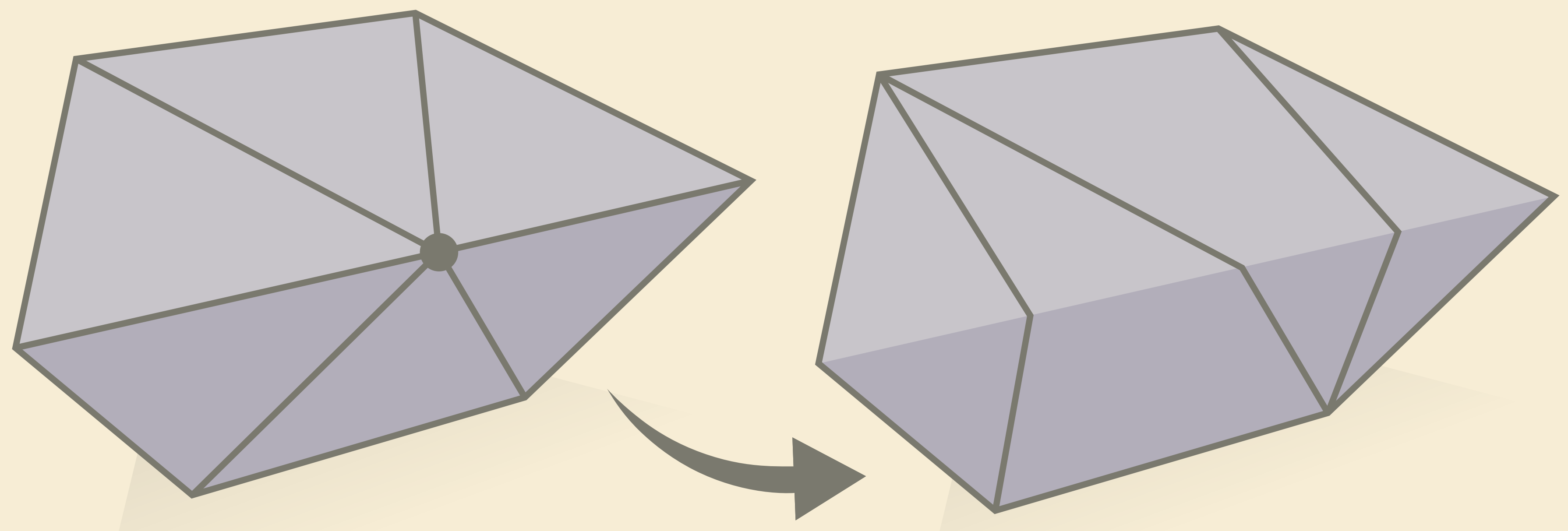
  (1) Distortion is low

  (2) Boundary edge lengths are preserved

- Discrete conformal parameterization [Springborn, Schröder & Pinkall 2008]
  - Constraint easy to impose
  - Efficient 1D optimization problem
- Flat vertex removal – also a standard operation

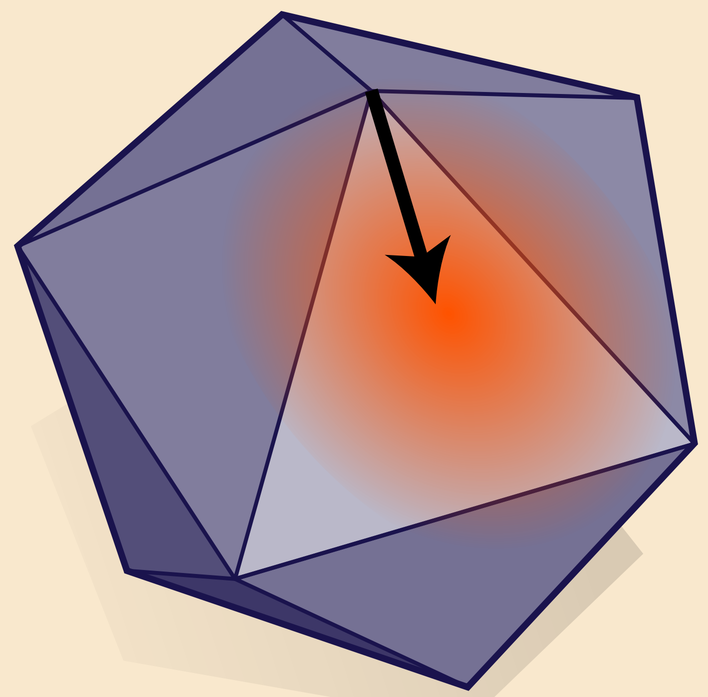
Intrinsic simplification

1. Local simplification operation



intrinsic vertex removal

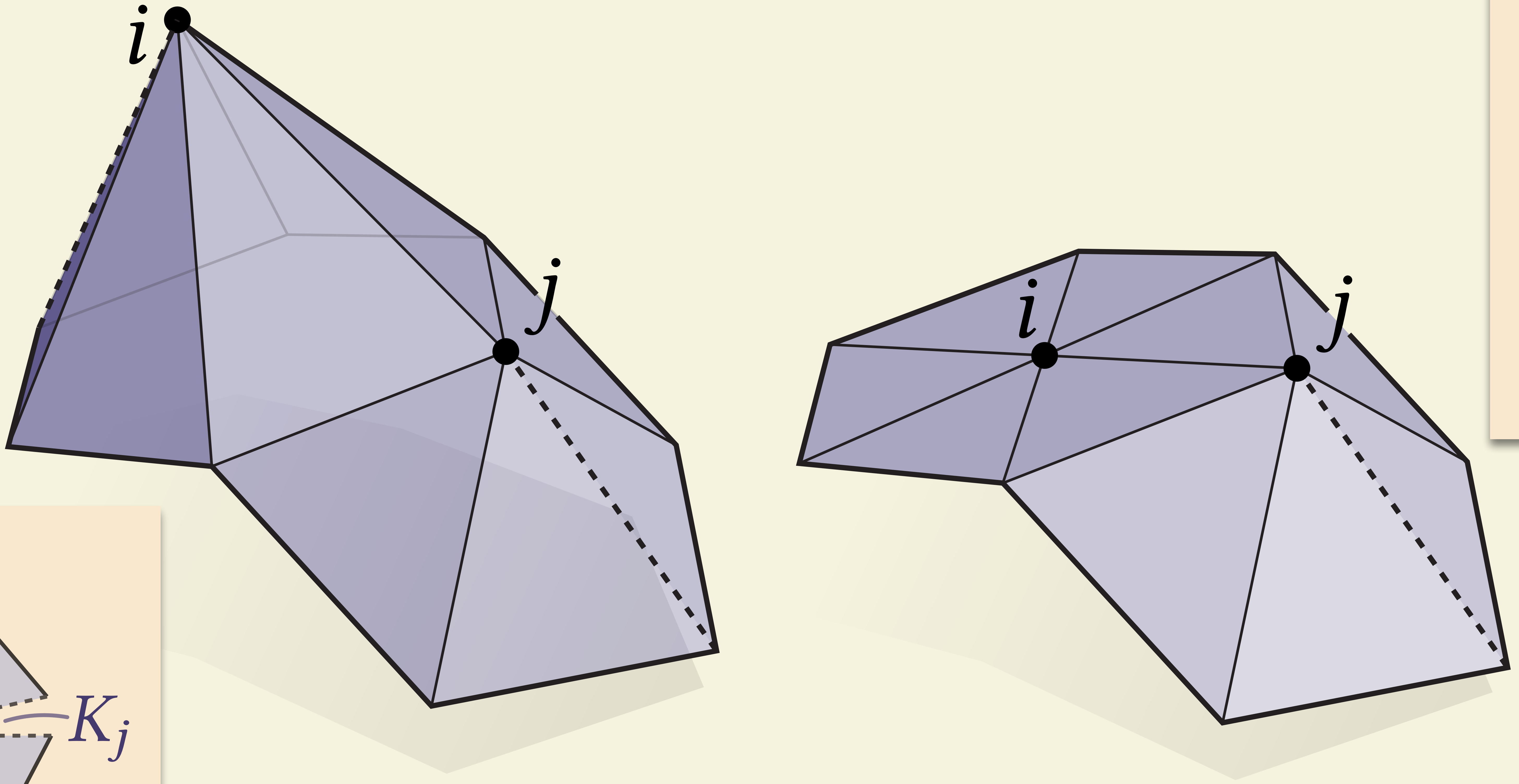
2. Accumulated distortion measurements

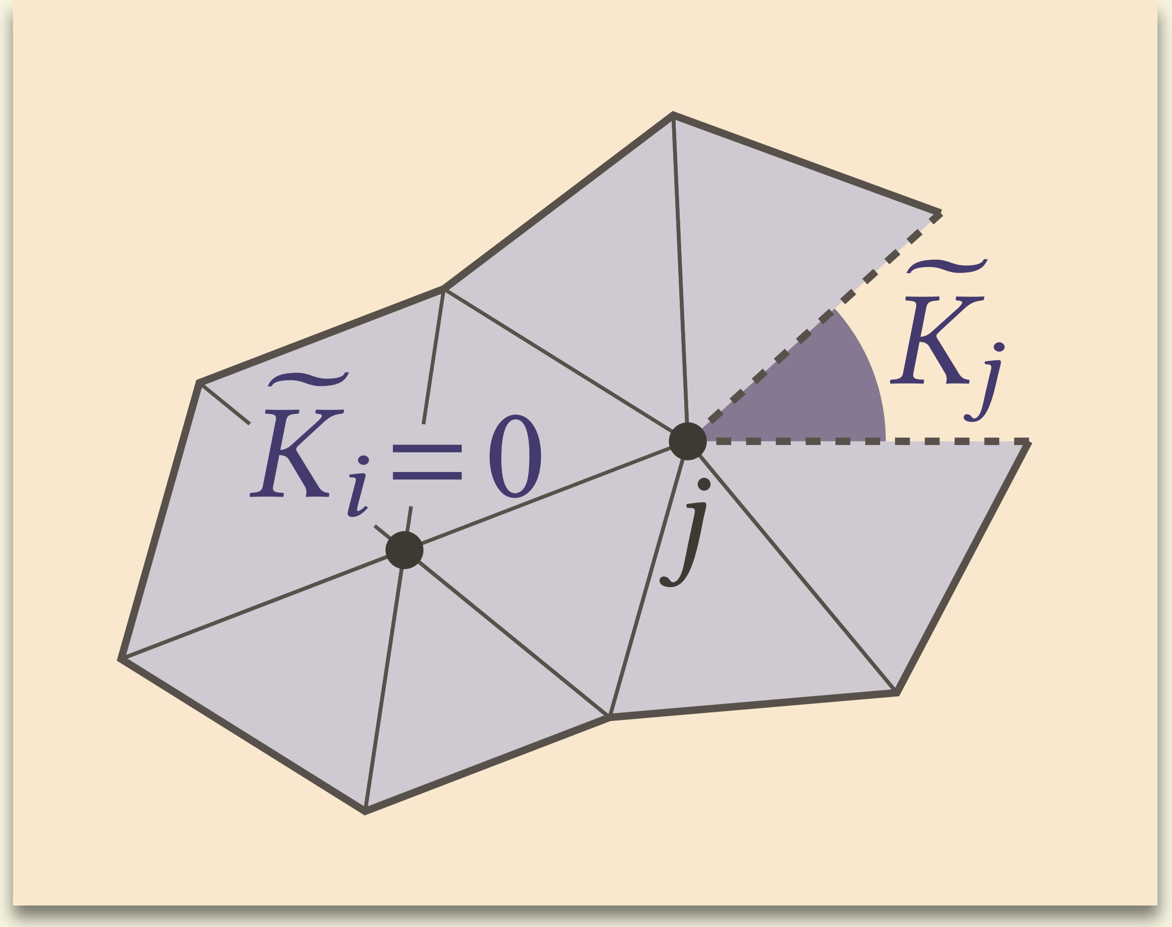


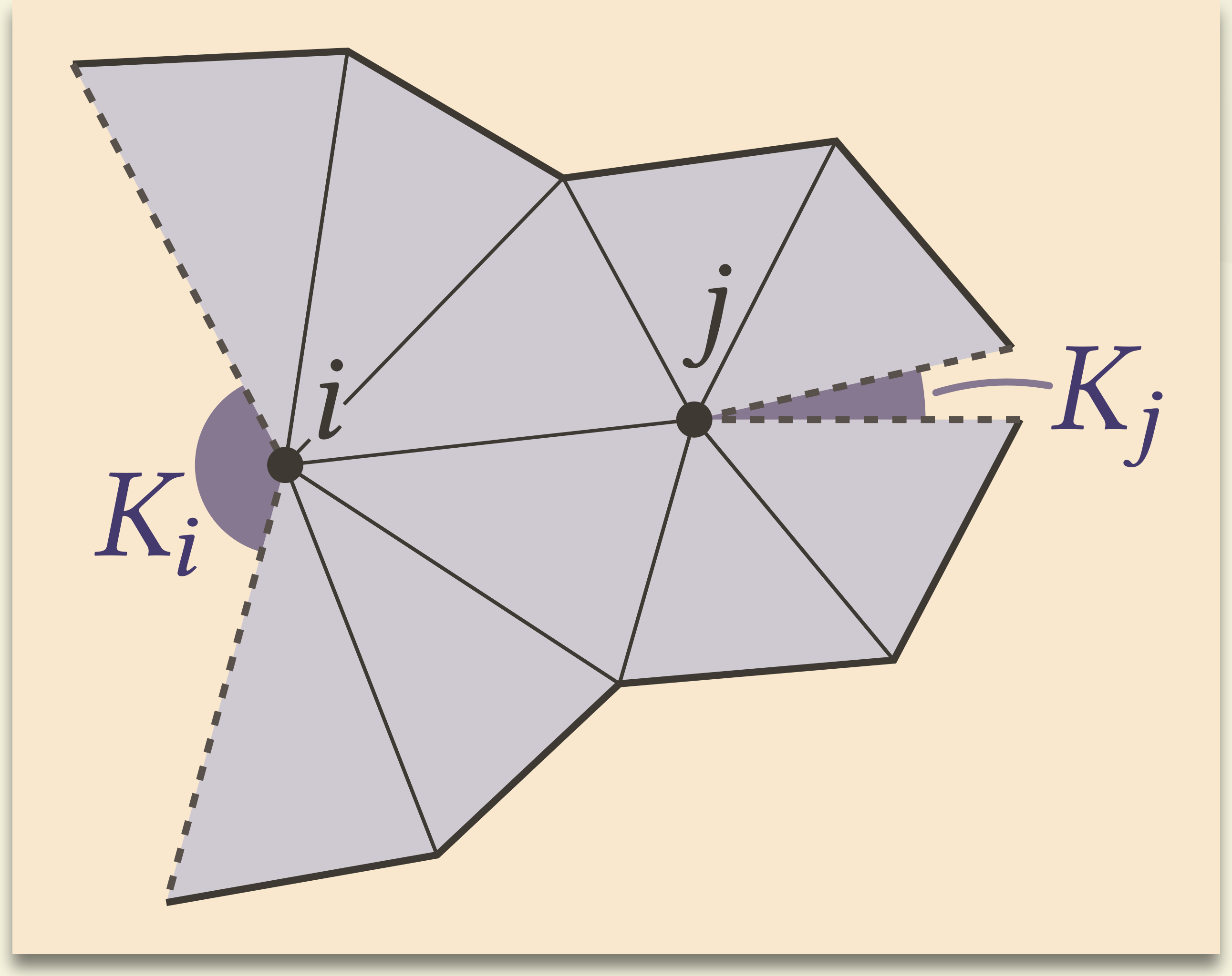
intrinsic curvature error

- Algorithm: repeatedly remove cheapest vertex

Distortion: curvature redistribution

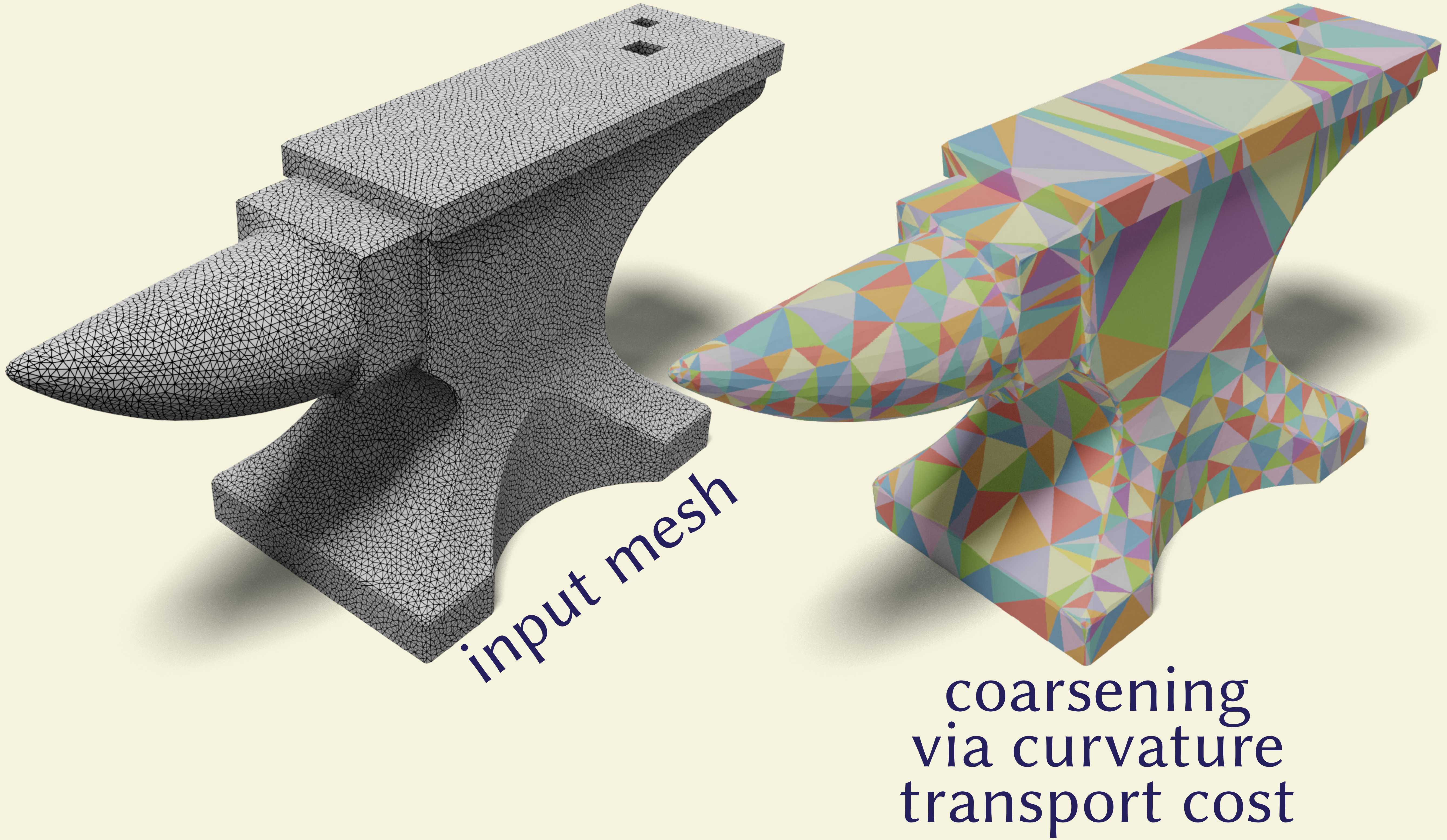






We approximate the *transport cost* of this curvature redistribution

Simplification with the curvature transport cost



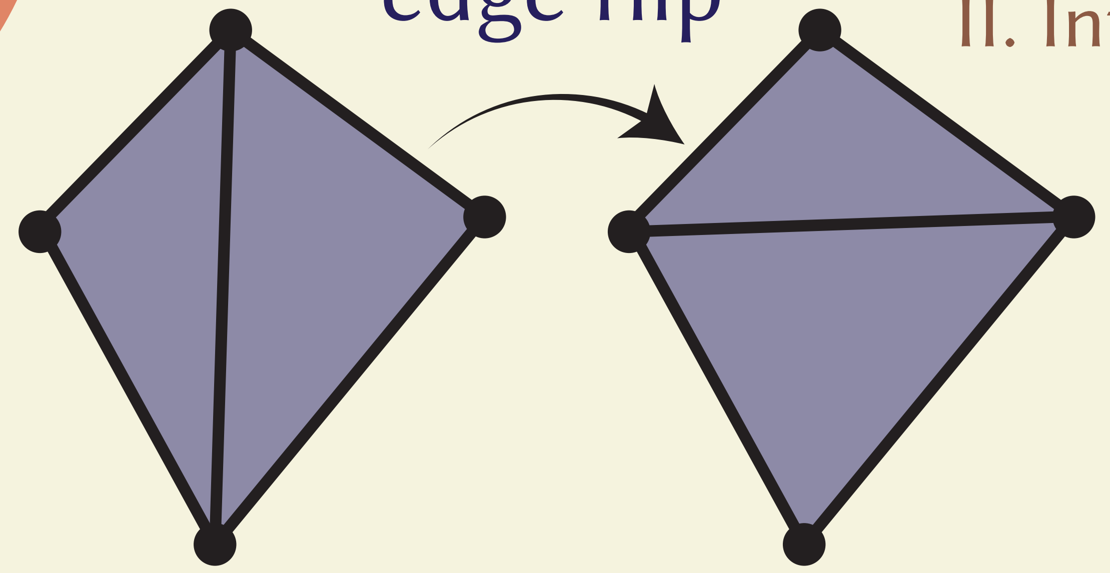
Other transport costs

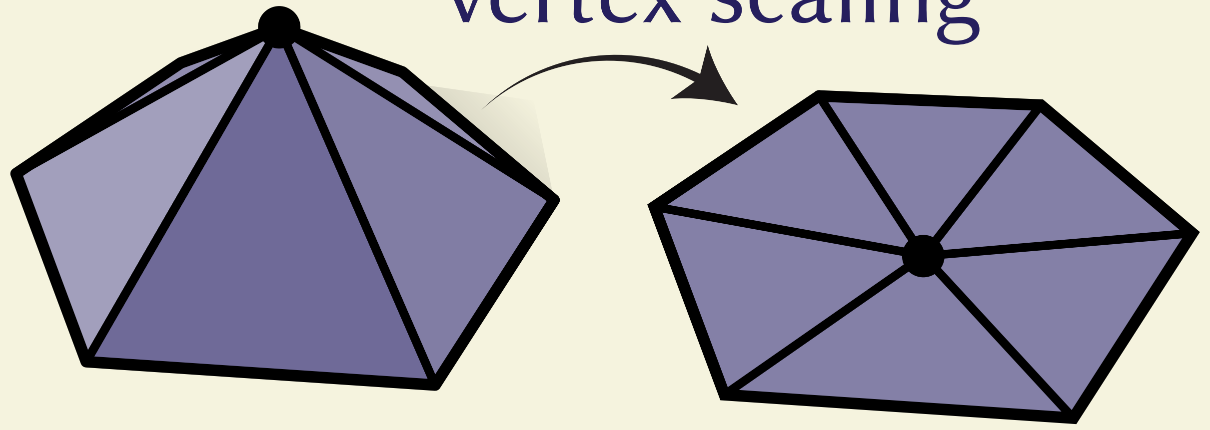
- Track transport cost of other data in same way
  - Can take weighted combinations of costs

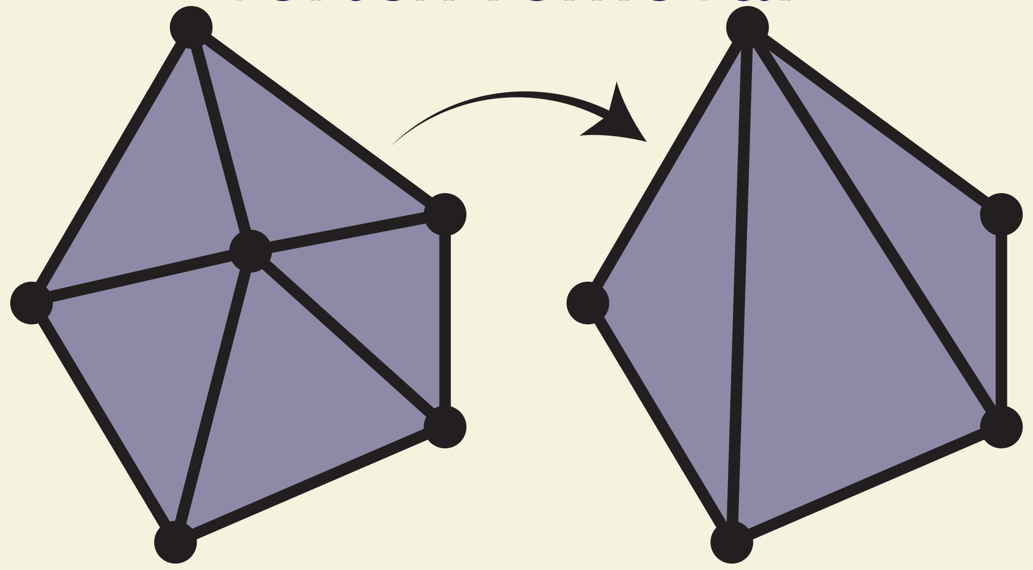
Surface correspondence

- Simplifying the mesh changes its geometry
    - Breaks existing data structures
- But, only uses a few local operations
    - Each is a simple mapping
- Encode correspondence via list of operations

1. Flip edge 1
2. Scale vertex 5
3. Remove vertex 5
4. Flip edge 8
5. Flip edge 12
6. Scale vertex 2

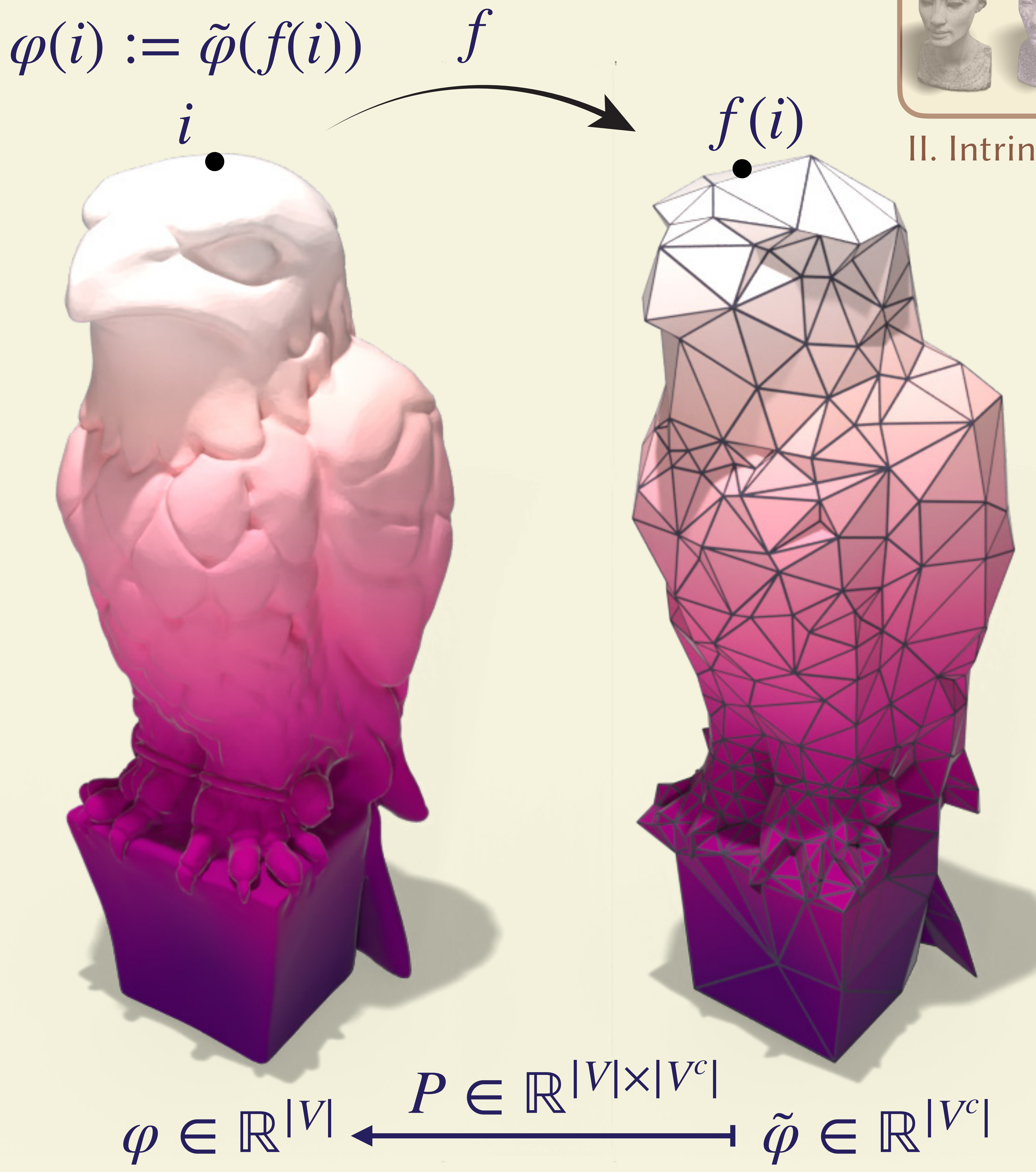






Prolongation

- Transfer piecewise-linear functions:
  - Just find values at vertices
  - Encode by a matrix

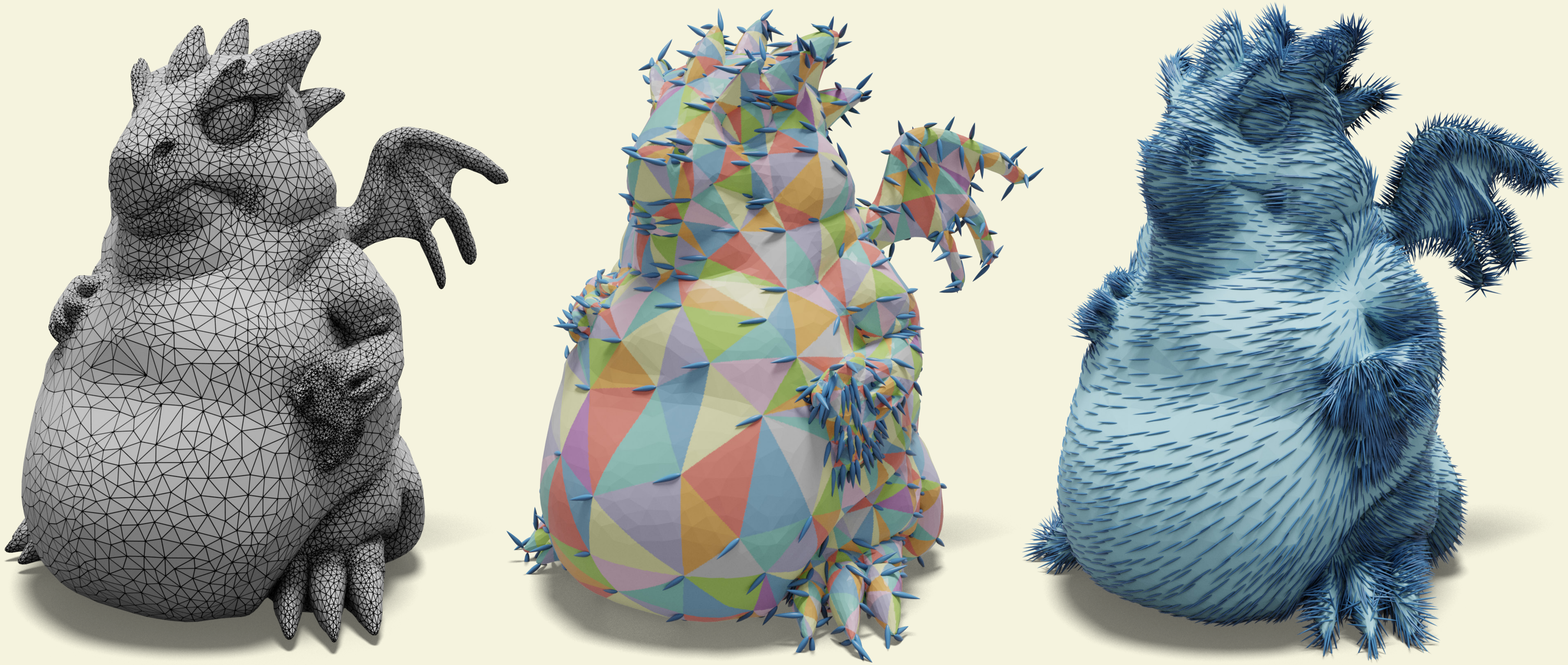


II. Intrinsic simplification
▸ *correspondence*

Pulling back vector fields

- Approximate differential of point mapping

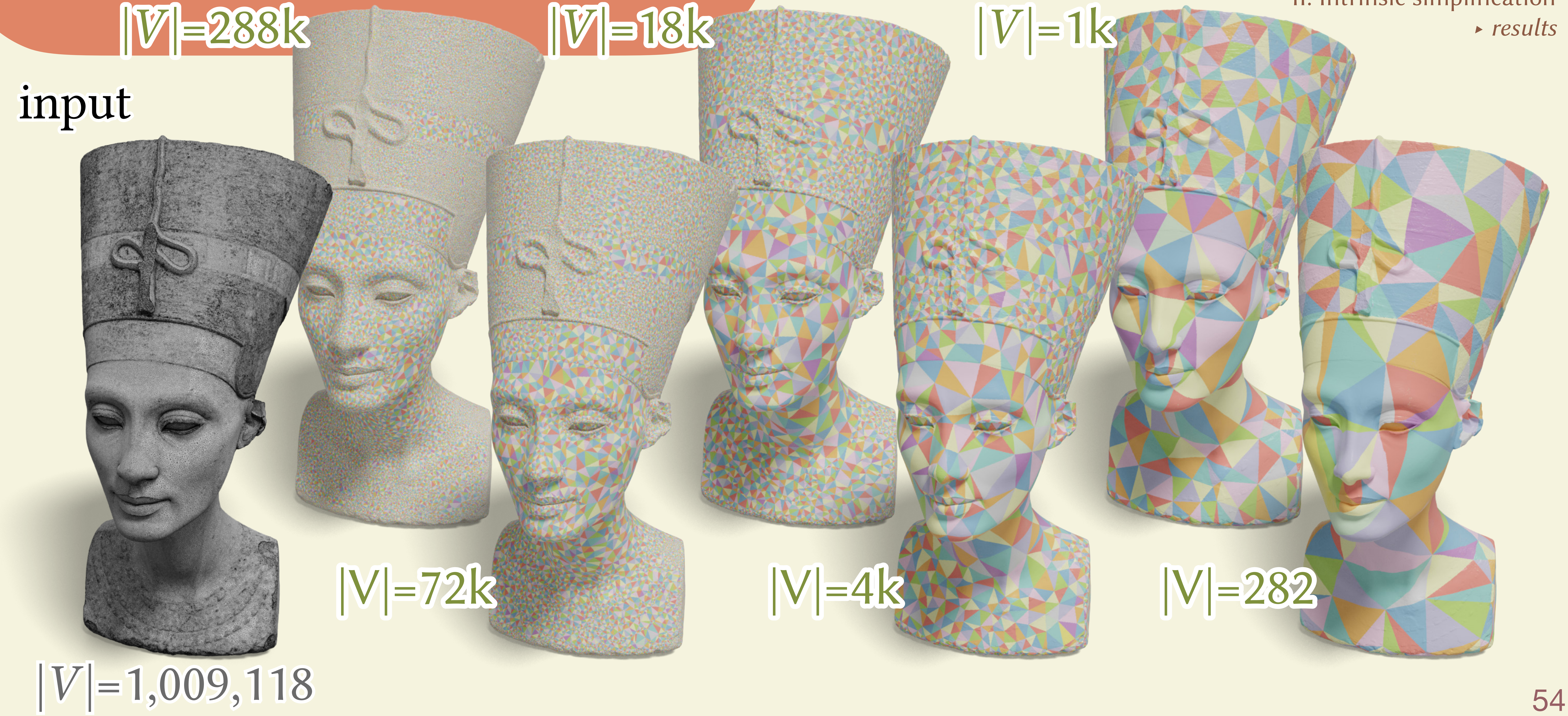
Encode by complex prolongation matrix



Results

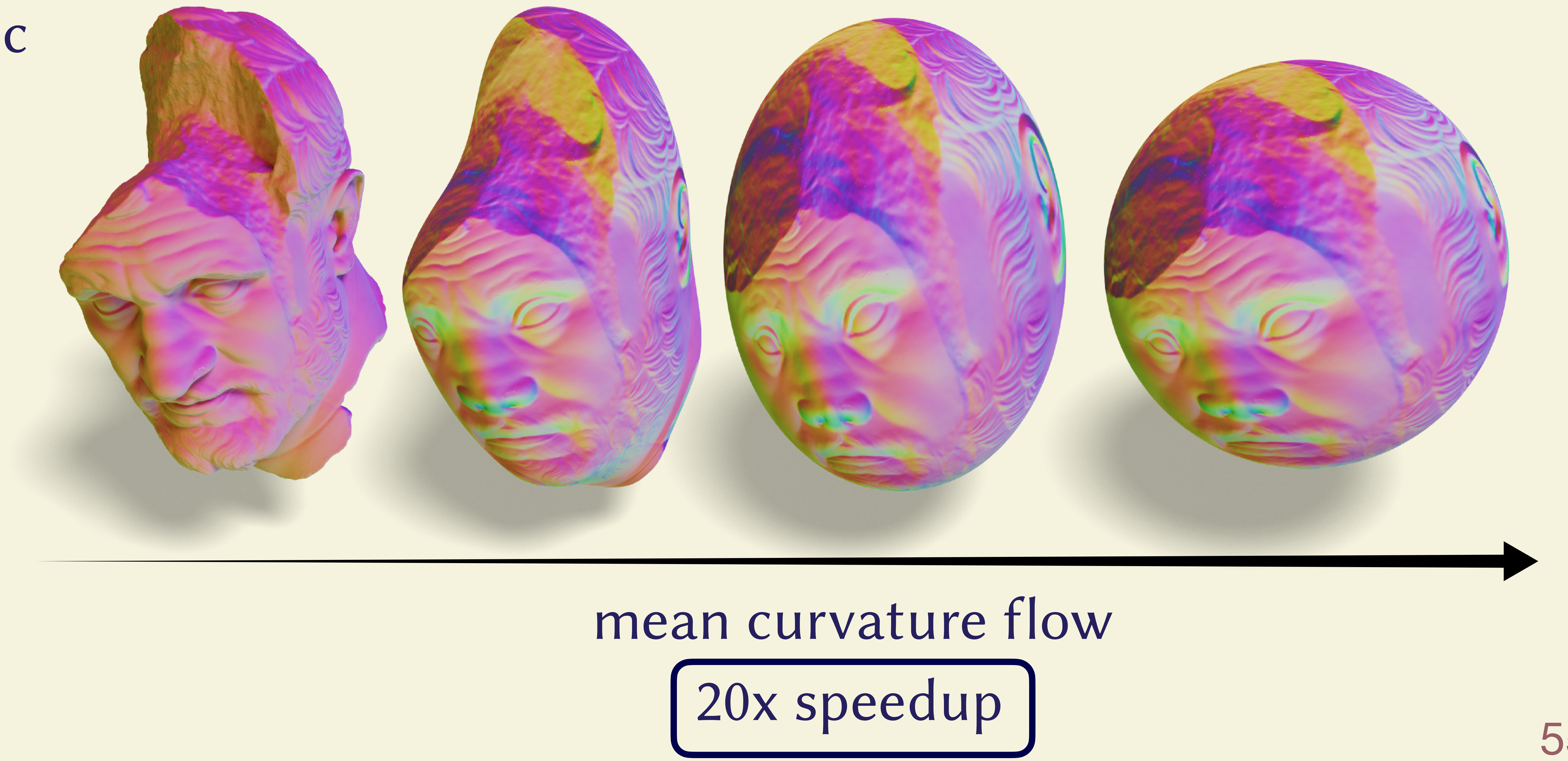
II. Intrinsic simplification

Surface hierarchies

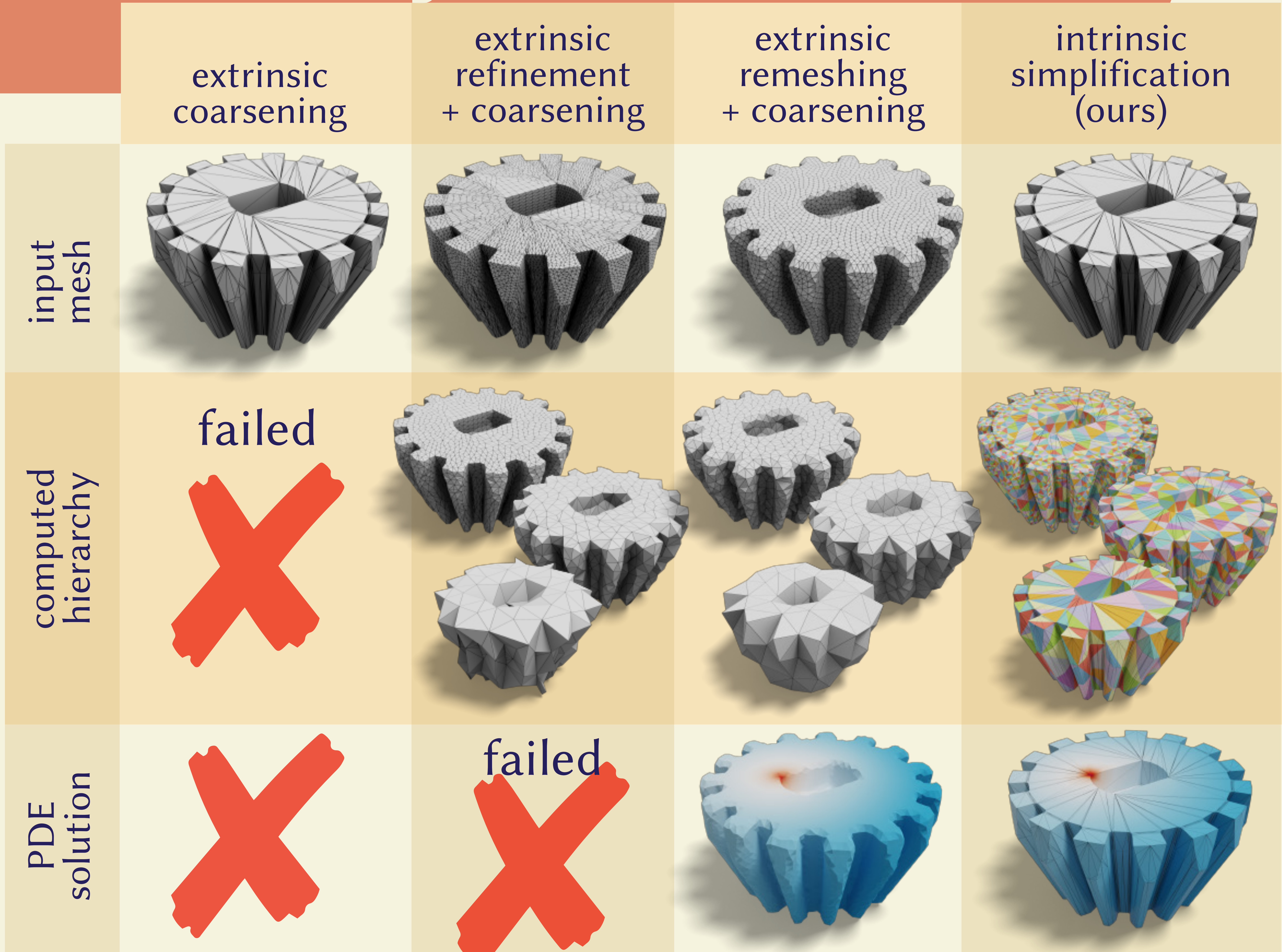


Hierarchies accelerate computation

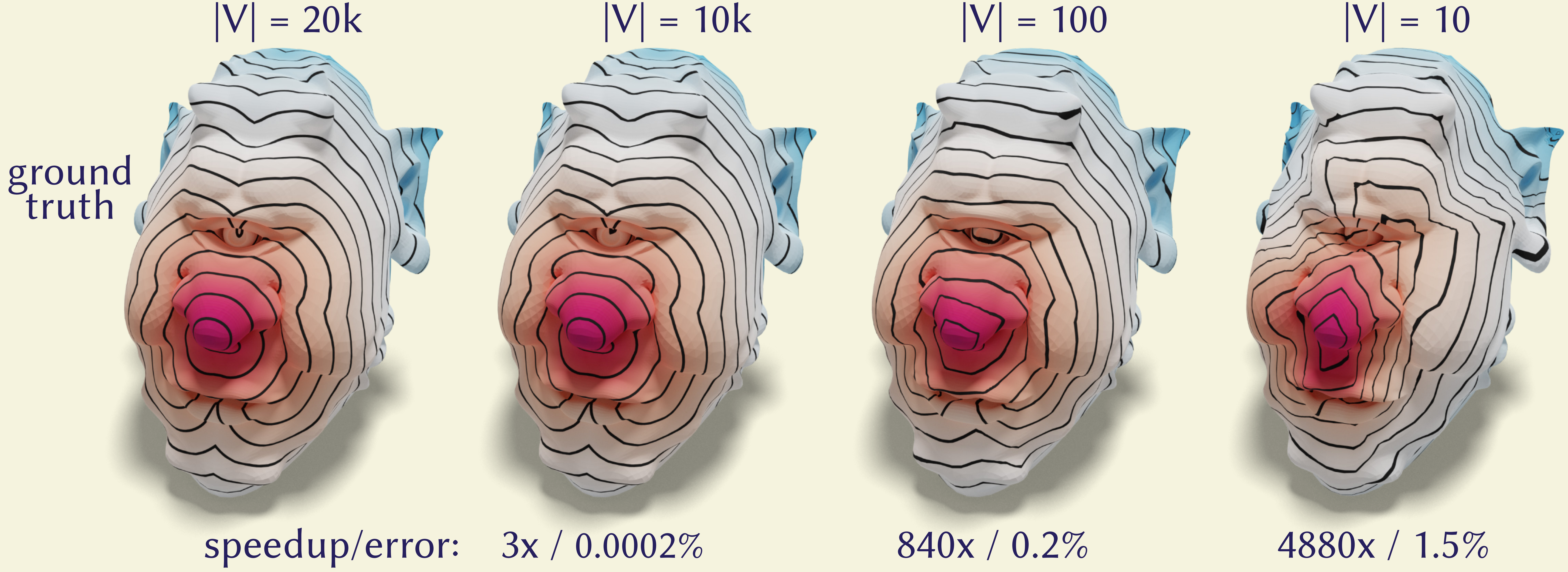
- Accelerate many geometric tasks
  - Even helps with extrinsic problems



Robust hierarchy construction



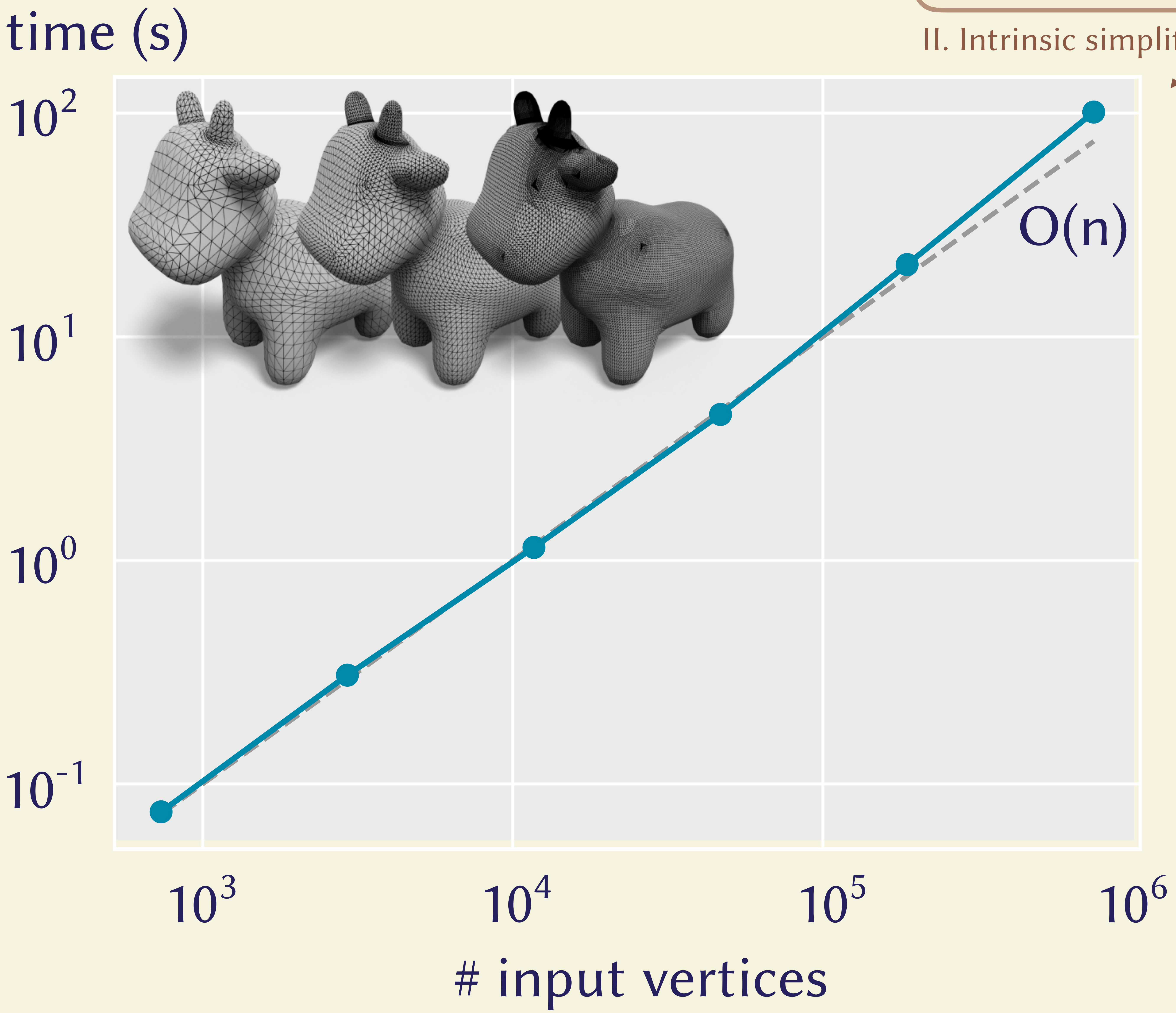
Speedup vs error in geodesic distance



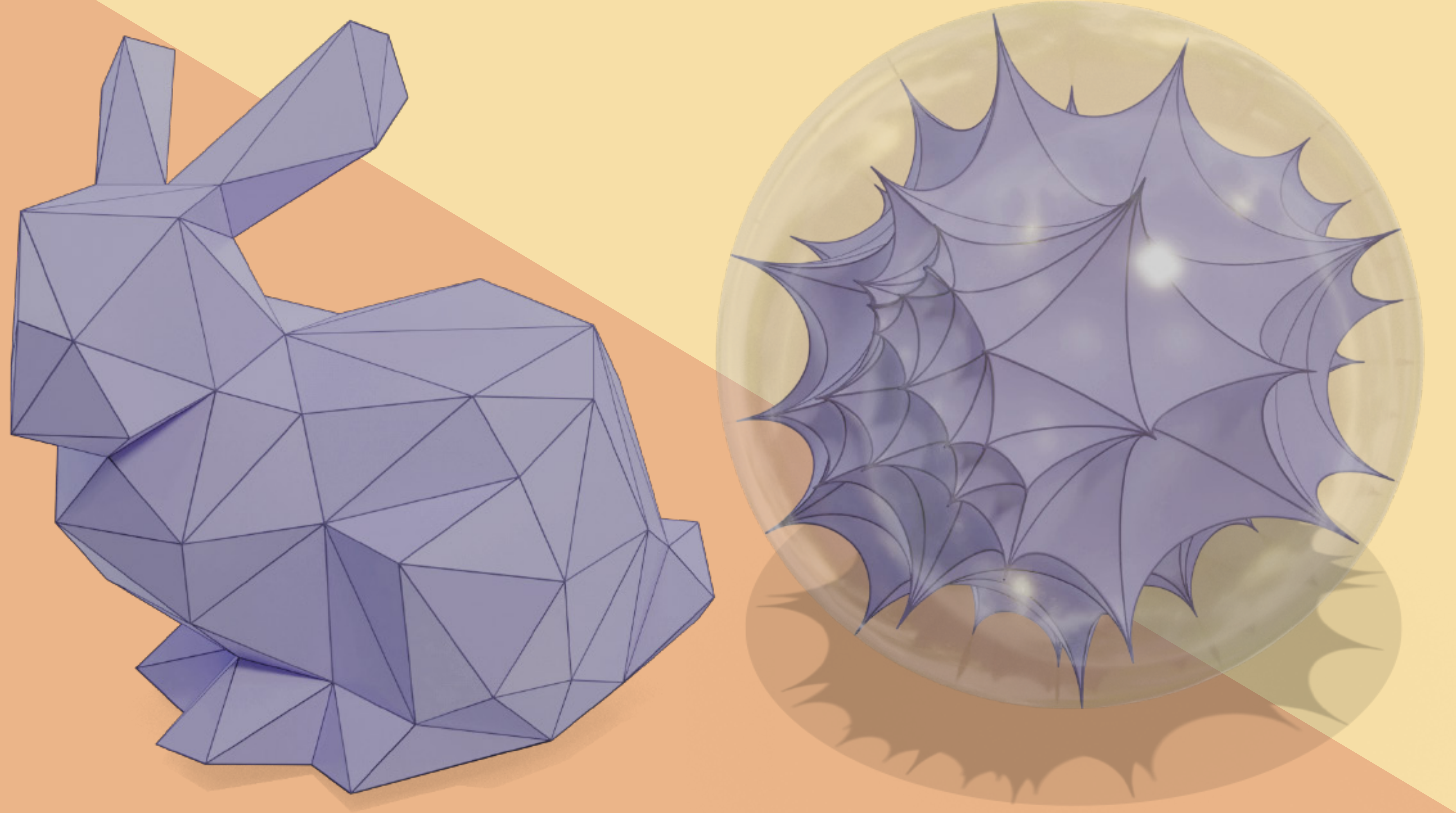
Performance

- Linear scaling
  - Constant work per vertex

Removes ~10,000 vertices per second



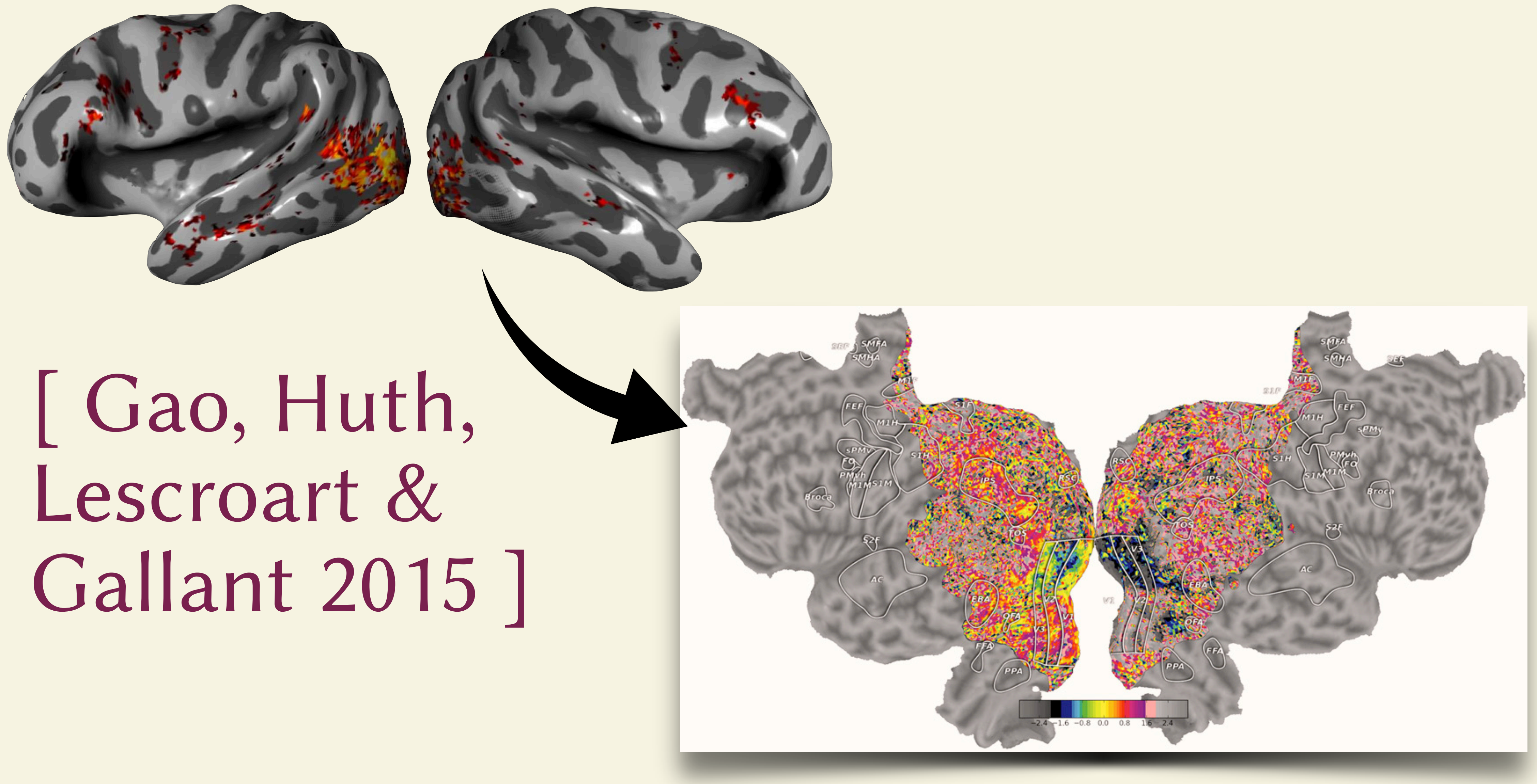
III. Surface Parameterization



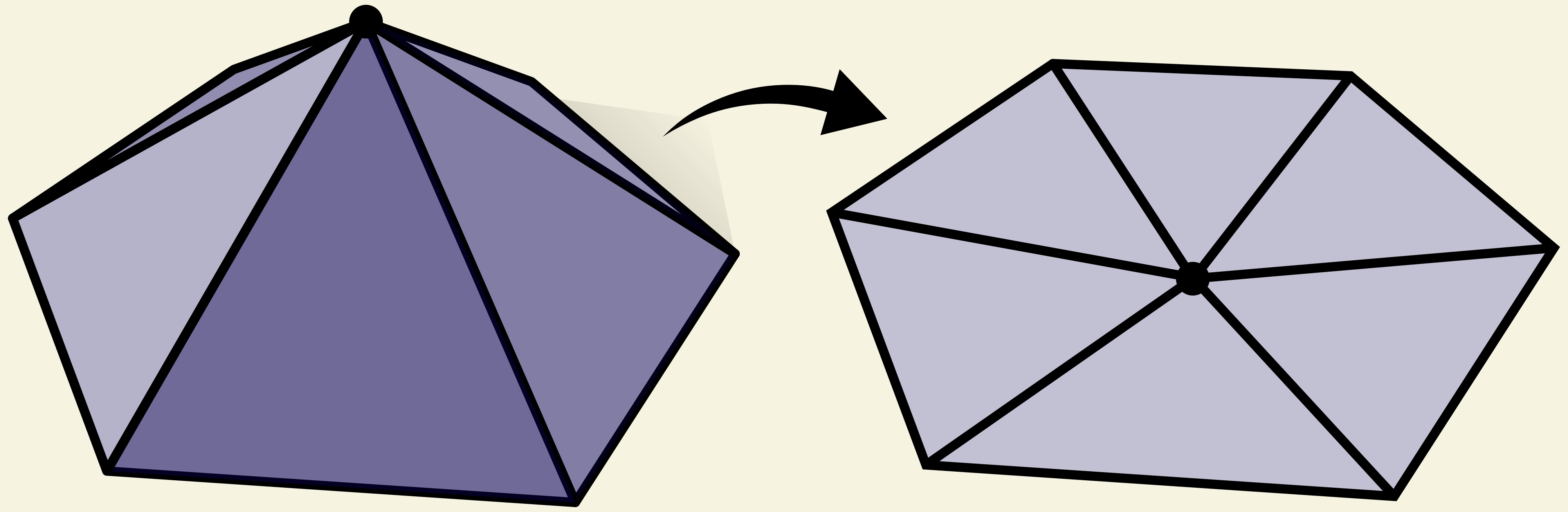
[**Gillespie**, Springborn, & Crane. 2021. Discrete conformal equivalence of polyhedral surfaces. *ACM Transactions on Graphics*]

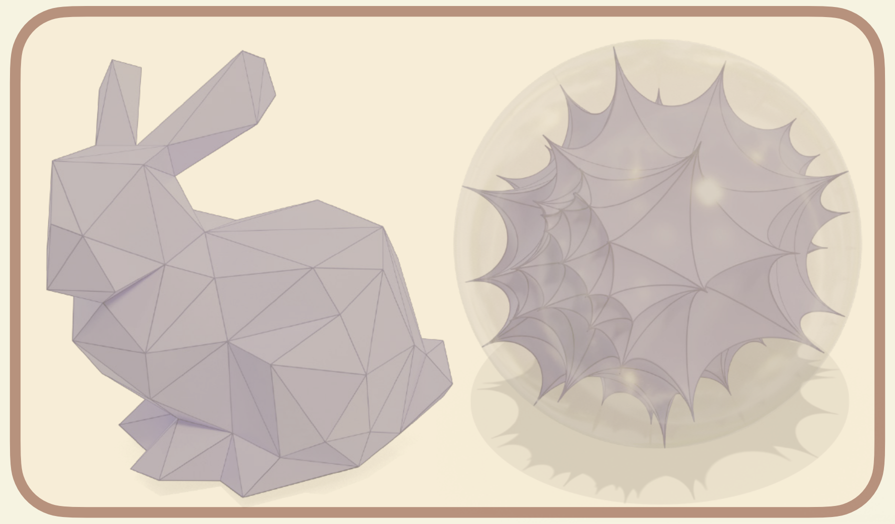
Parameterization

Mapping surfaces into the plane

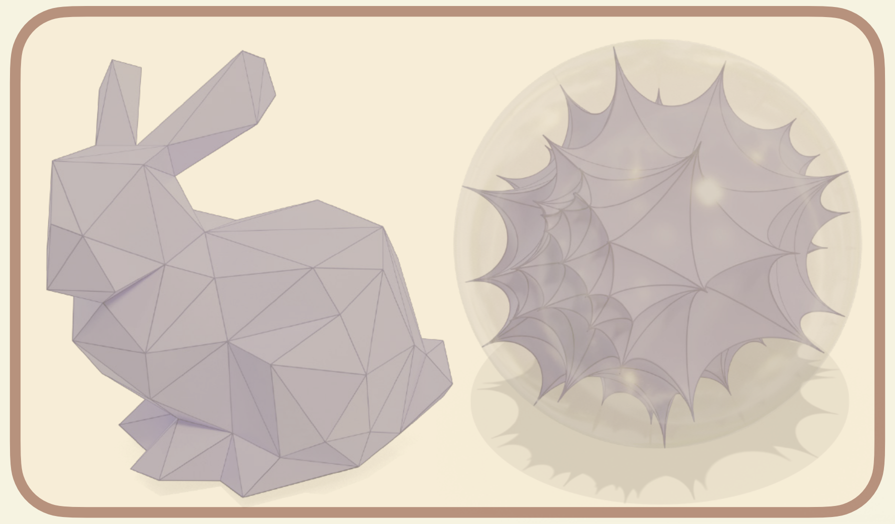


[Gao, Huth, Lescroart & Gallant 2015]



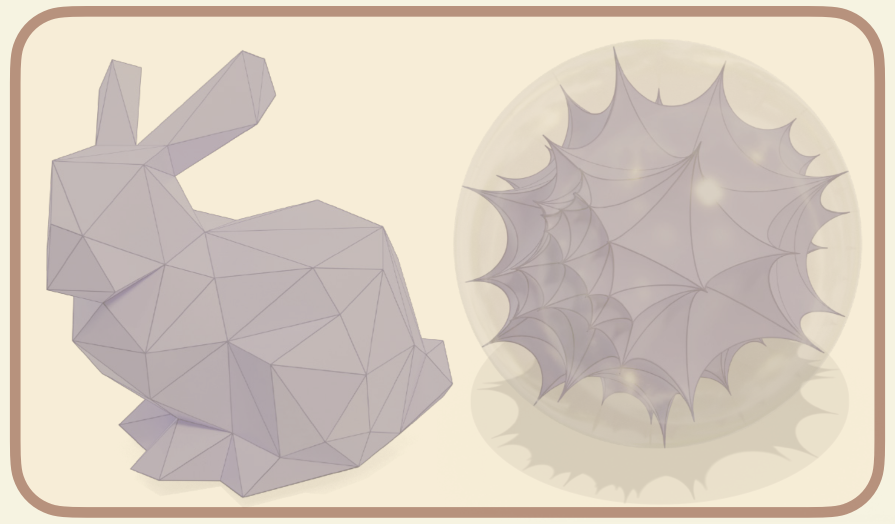


Texture mapping



[Timen 2012]

The uniformization theorem
[Poincare 1907; Koebe 1907; Troyanov 1991]



Any surface is conformally equivalent to a surface of constant curvature.

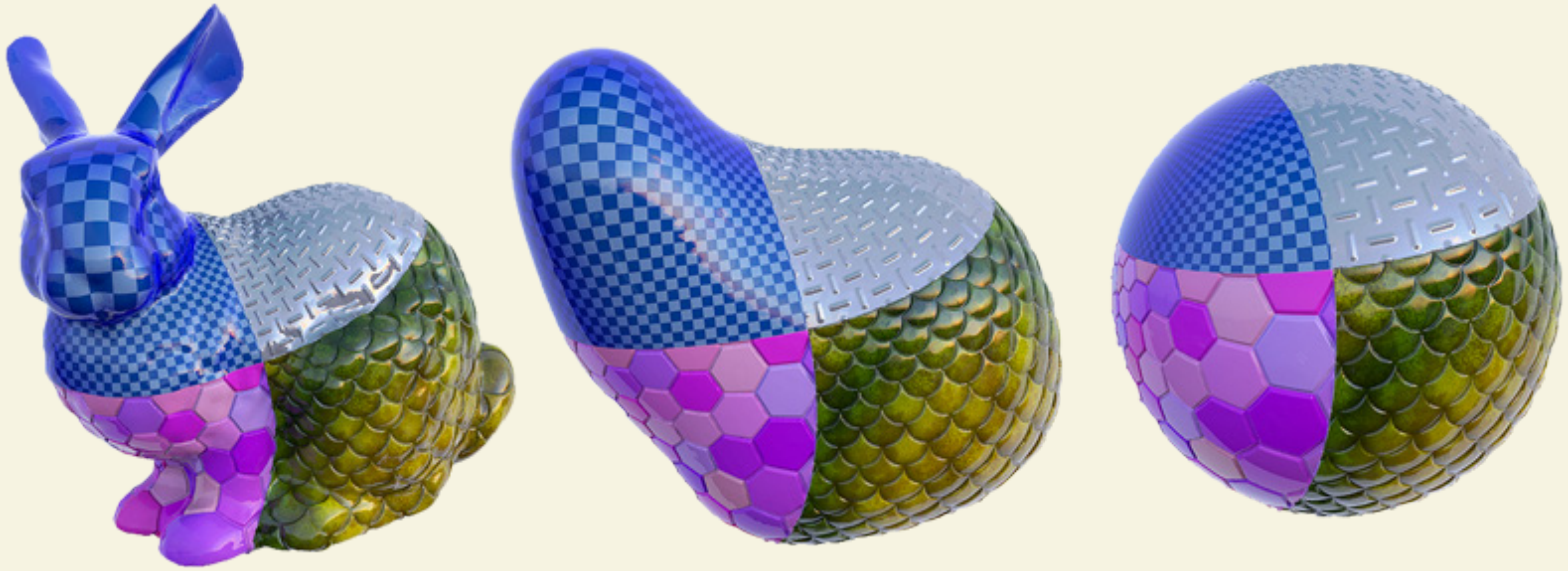
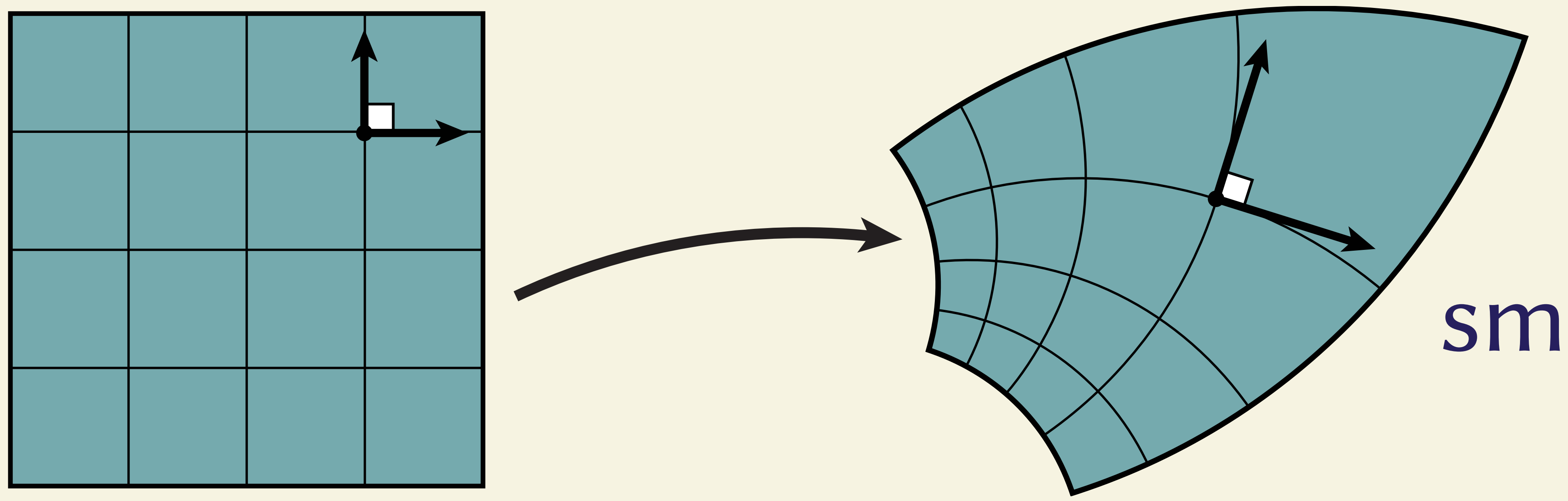


Image: [Crane, Pinkall & Schröder 2013]

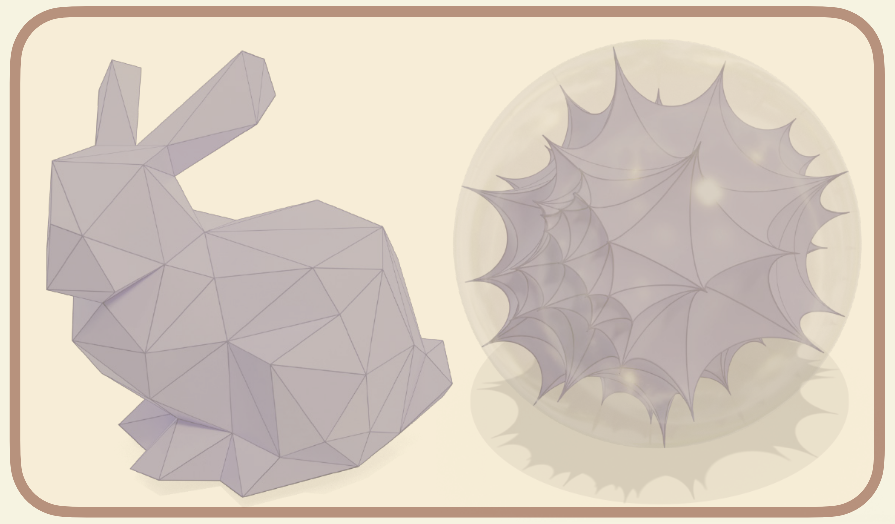


conformal map = angle-preserving

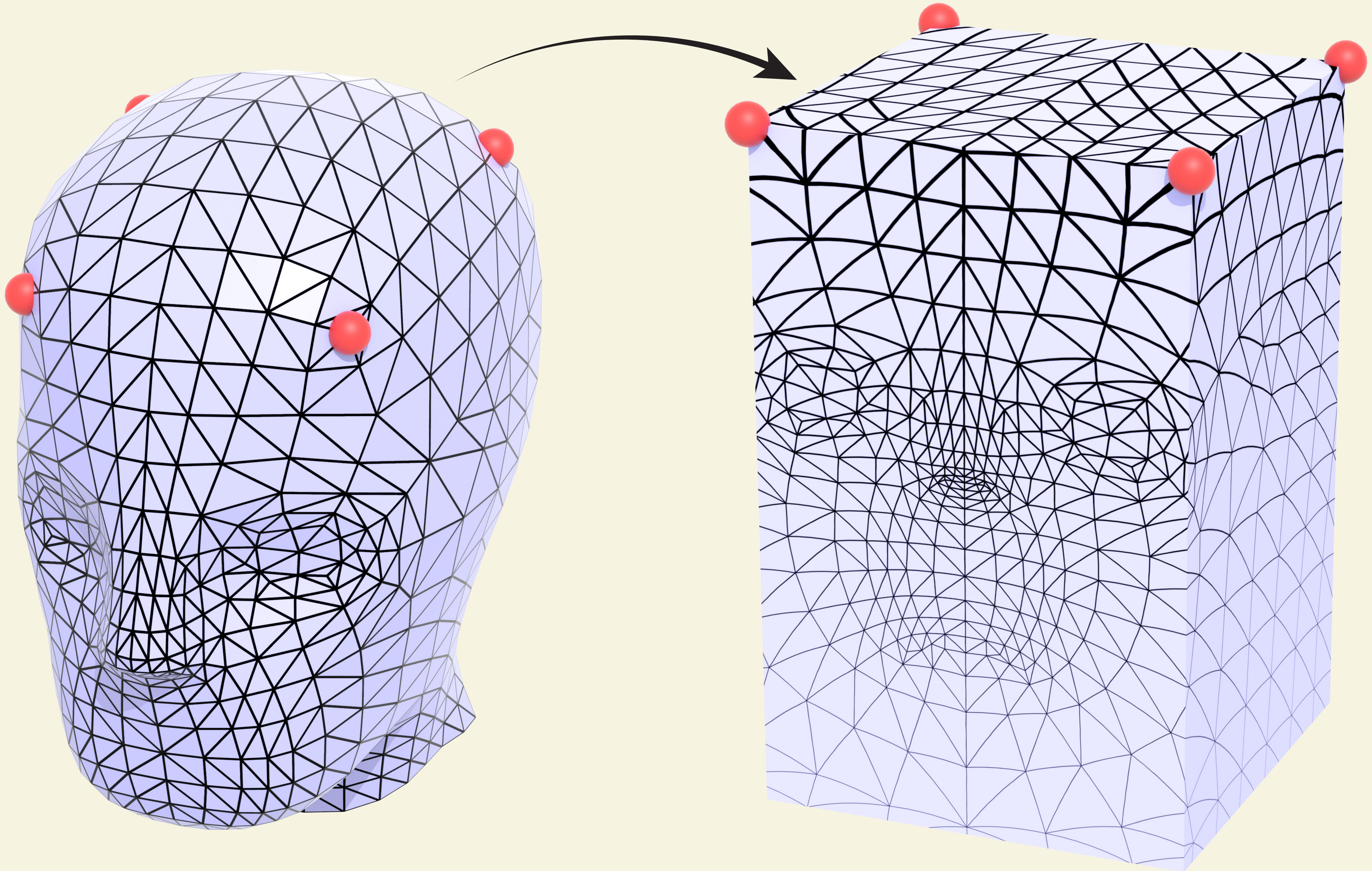
smooth maps with helpful properties

The discrete uniformization theorem

[Gu, Luo, Sun & Wu 2018; Springborn 2019]



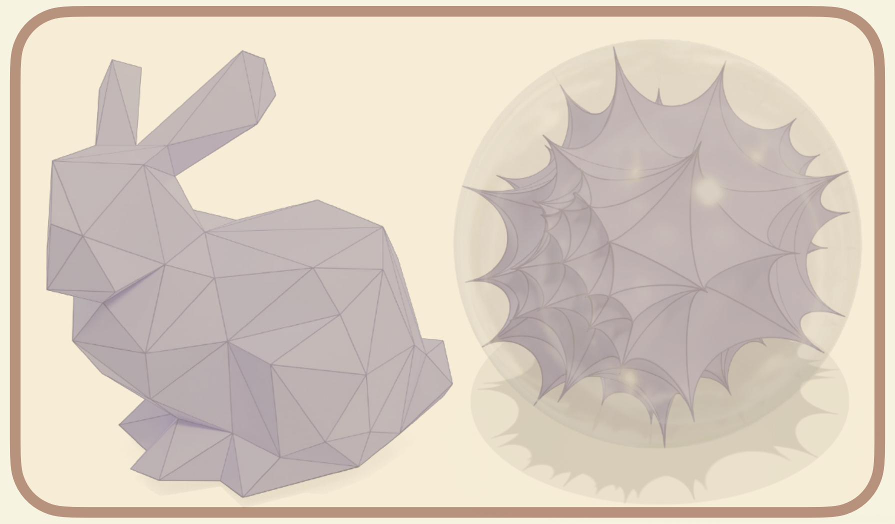
Any valid† vertex curvatures can be realized by some discrete conformal map.



†*i.e.* $\leq 2\pi$ and satisfying Gauss-Bonnet

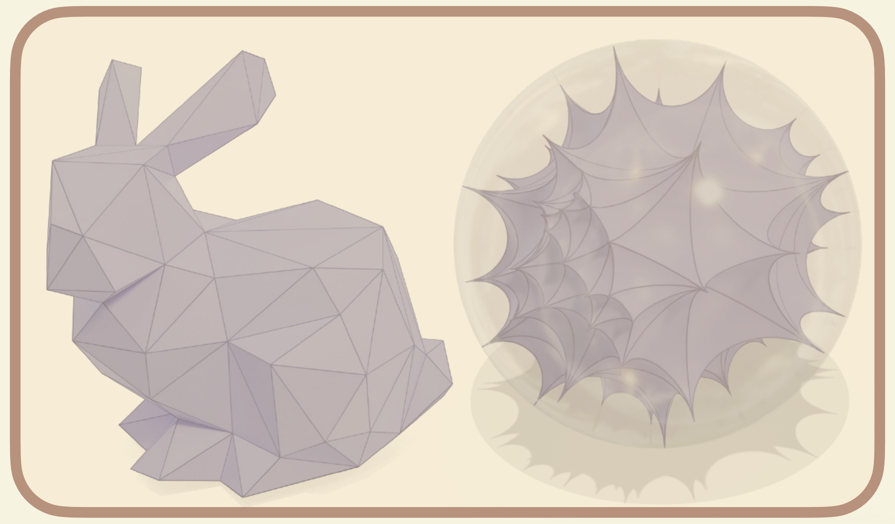
The discrete uniformization theorem

[Gu, Luo, Sun & Wu 2018; Springborn 2019]

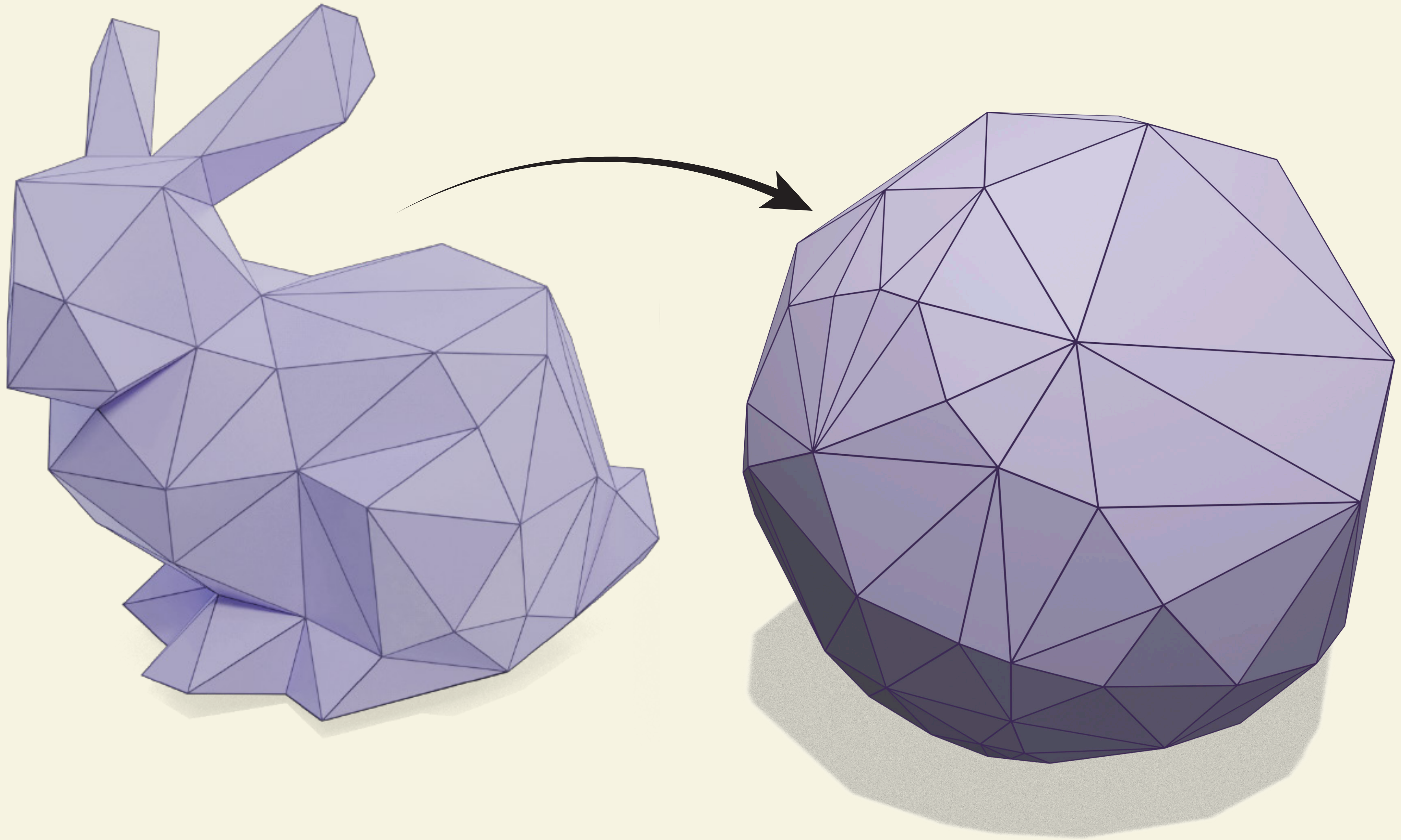


III. Parameterization

The discrete spherical uniformization theorem
[Springborn 2019]



Any simply-connected triangle mesh is discretely conformally equivalent to a mesh whose vertices lie on the unit sphere

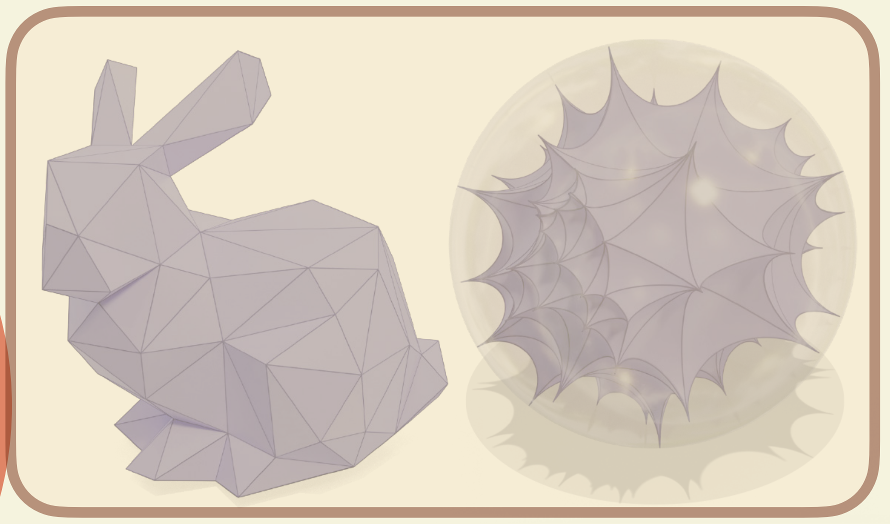


Discrete uniformization in action

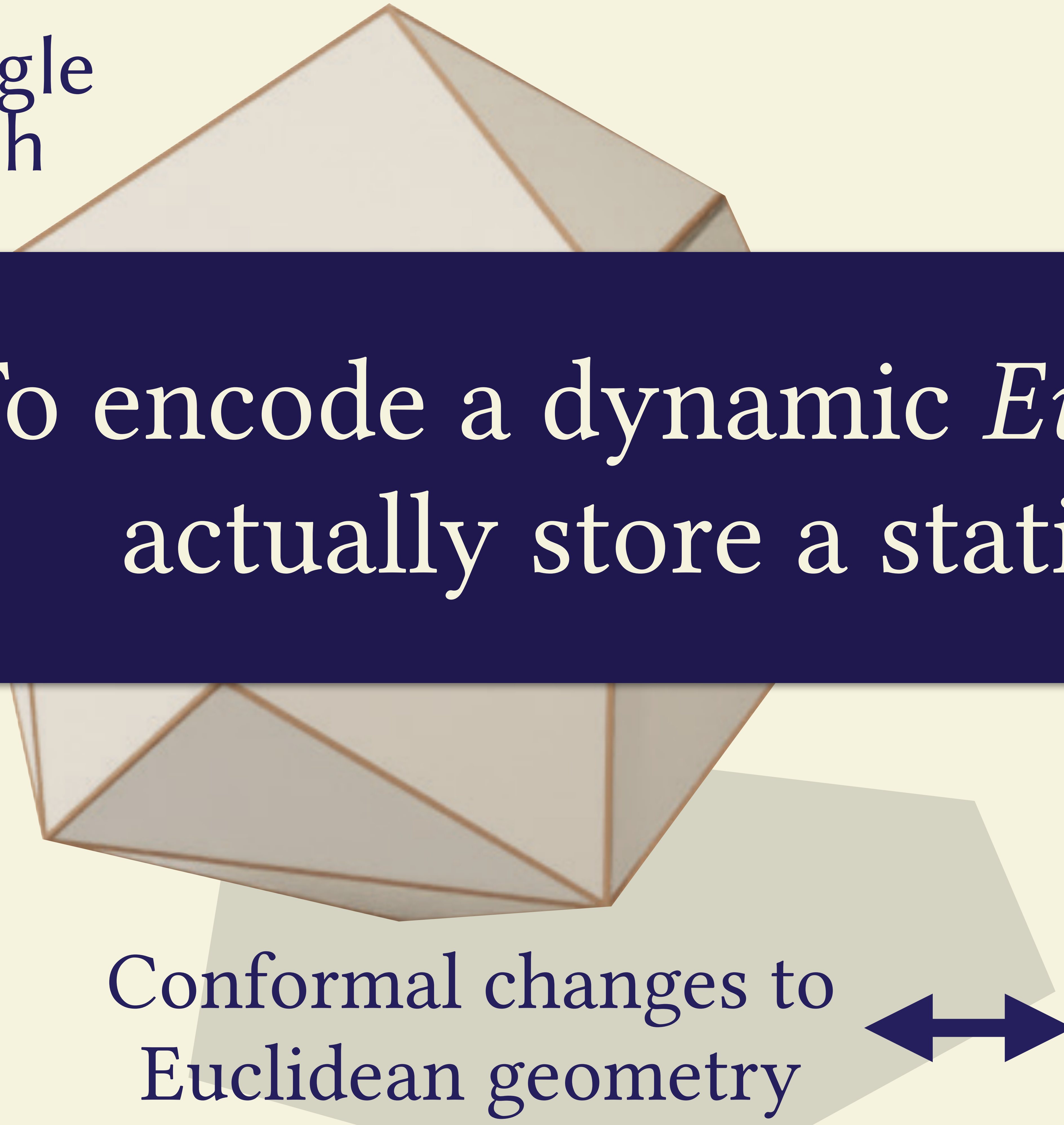
[Gillespie, Springborn, & Crane. 2021]

Triangle mesh $\longleftrightarrow$ hyperbolic polyhedron

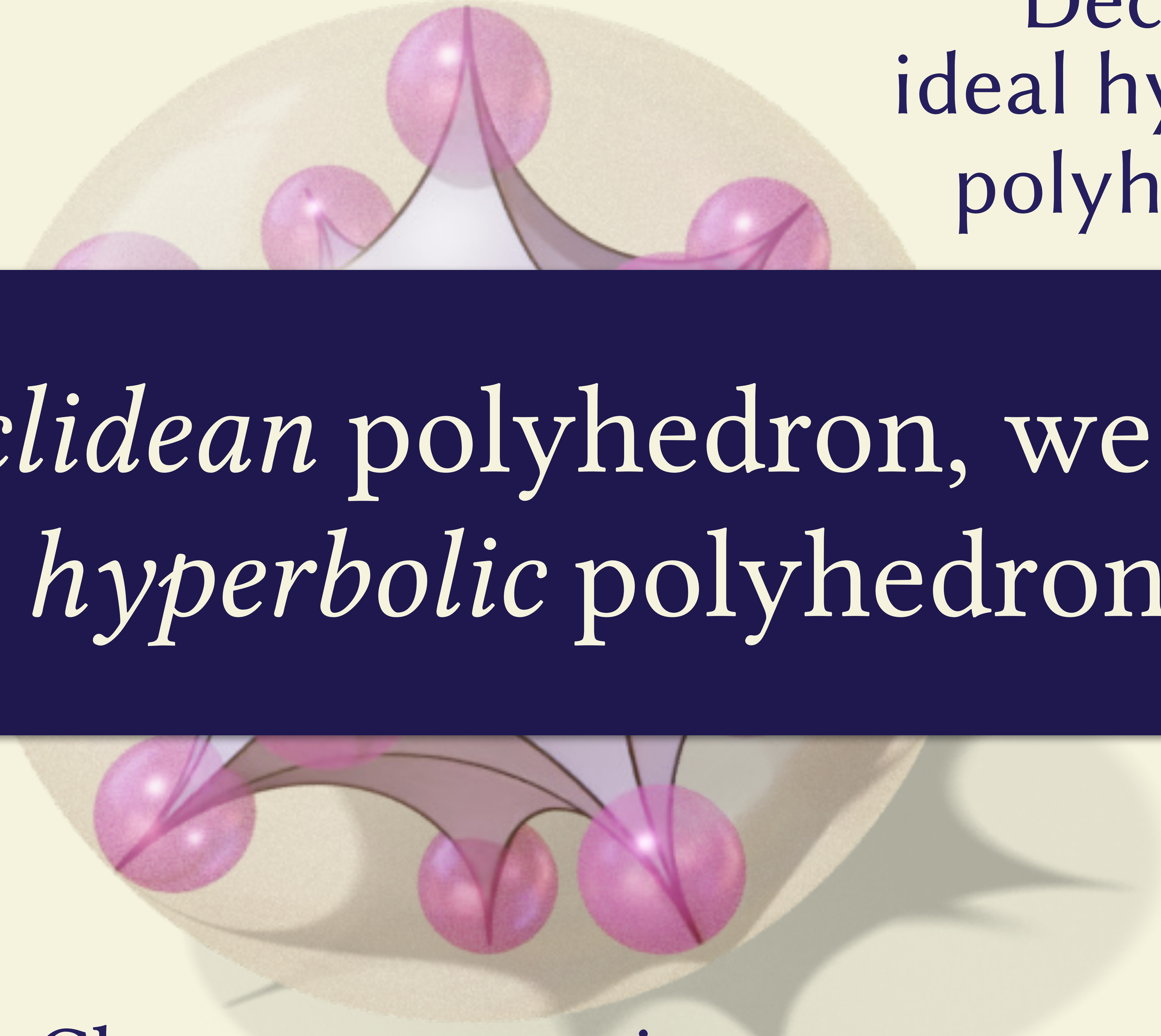
[Bobenko, Pinkall & Springborn 2010]



Triangle mesh



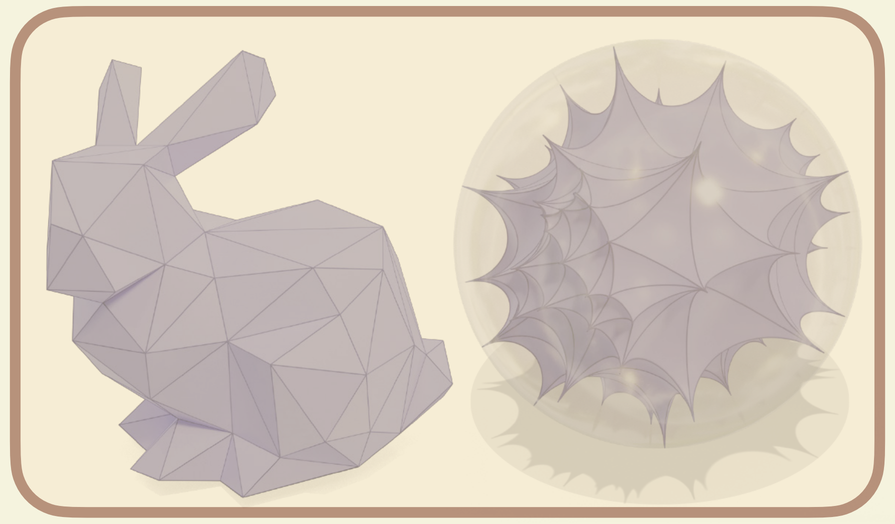
“Decorated ideal hyperbolic polyhedron”

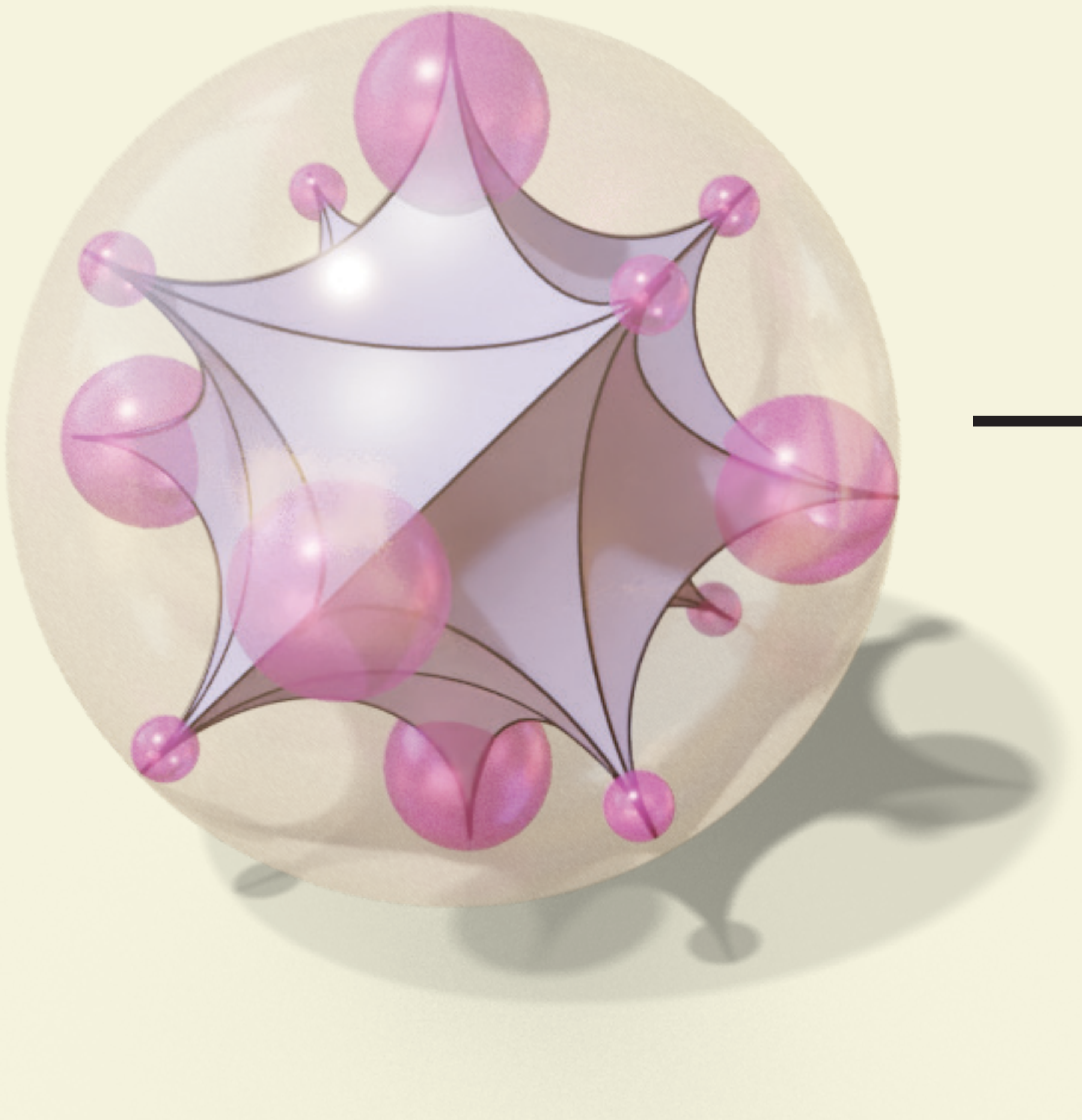


To encode a dynamic *Euclidean* polyhedron, we can actually store a static *hyperbolic* polyhedron

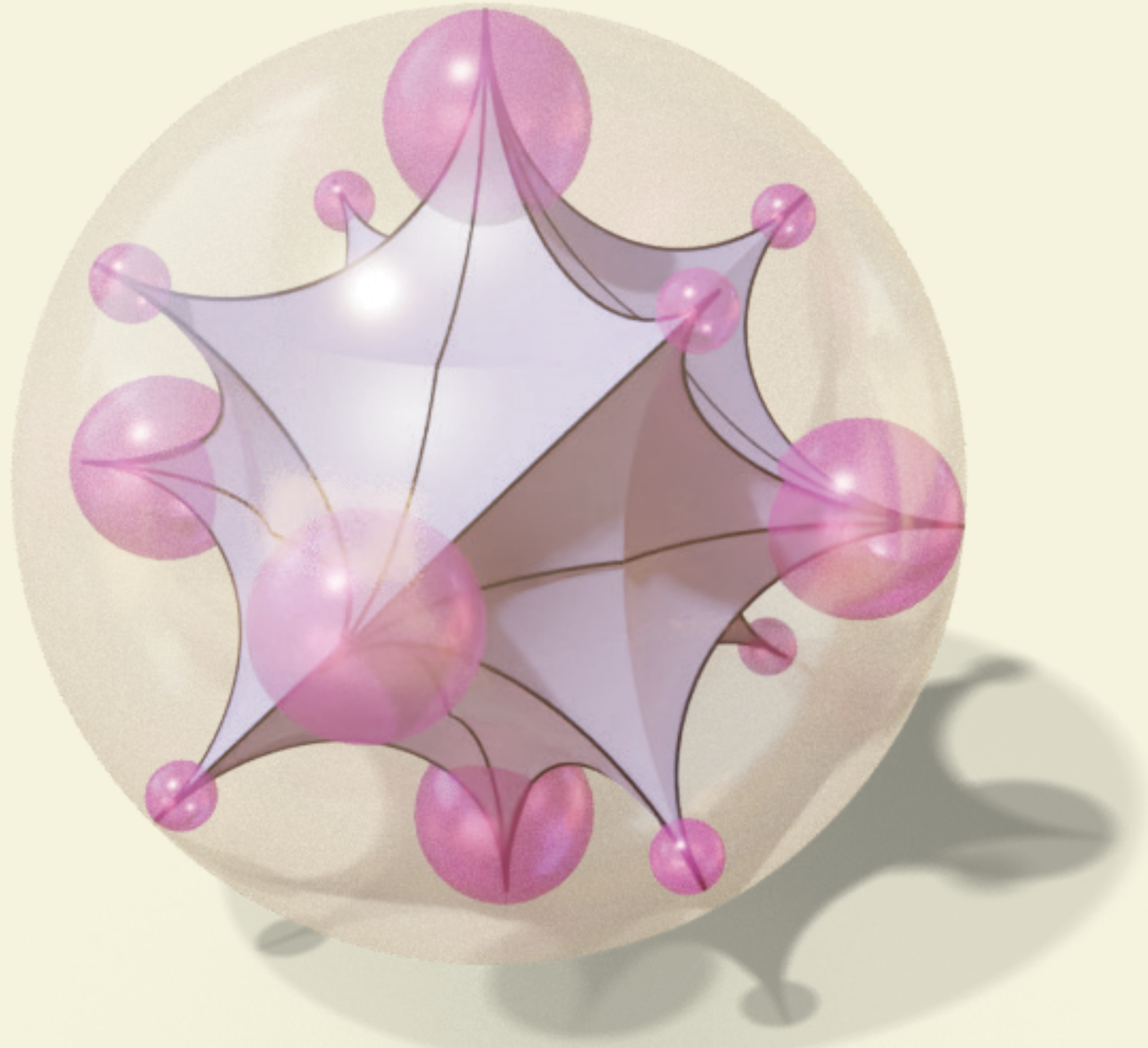
Conformal changes to Euclidean geometry $\longleftrightarrow$ Changes preserving hyperbolic geometry

Intrinsic triangulations of hyperbolic polyhedra



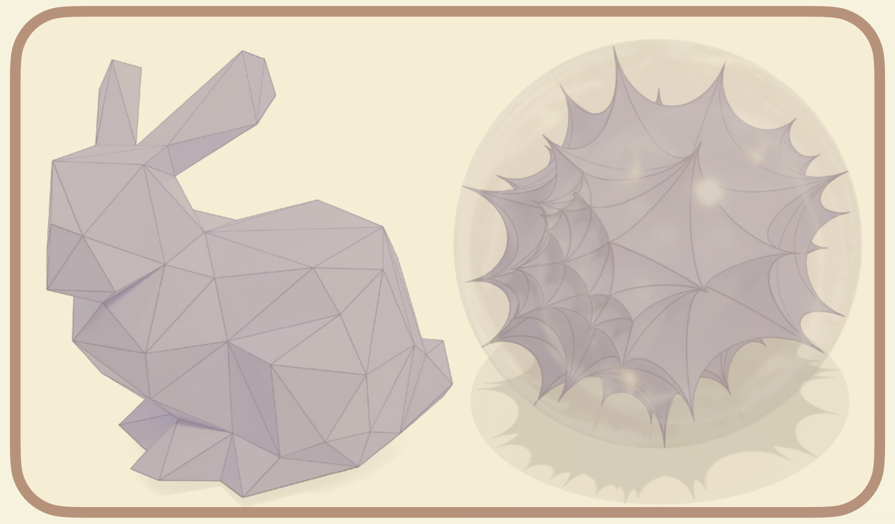


hyperbolic edge flip →



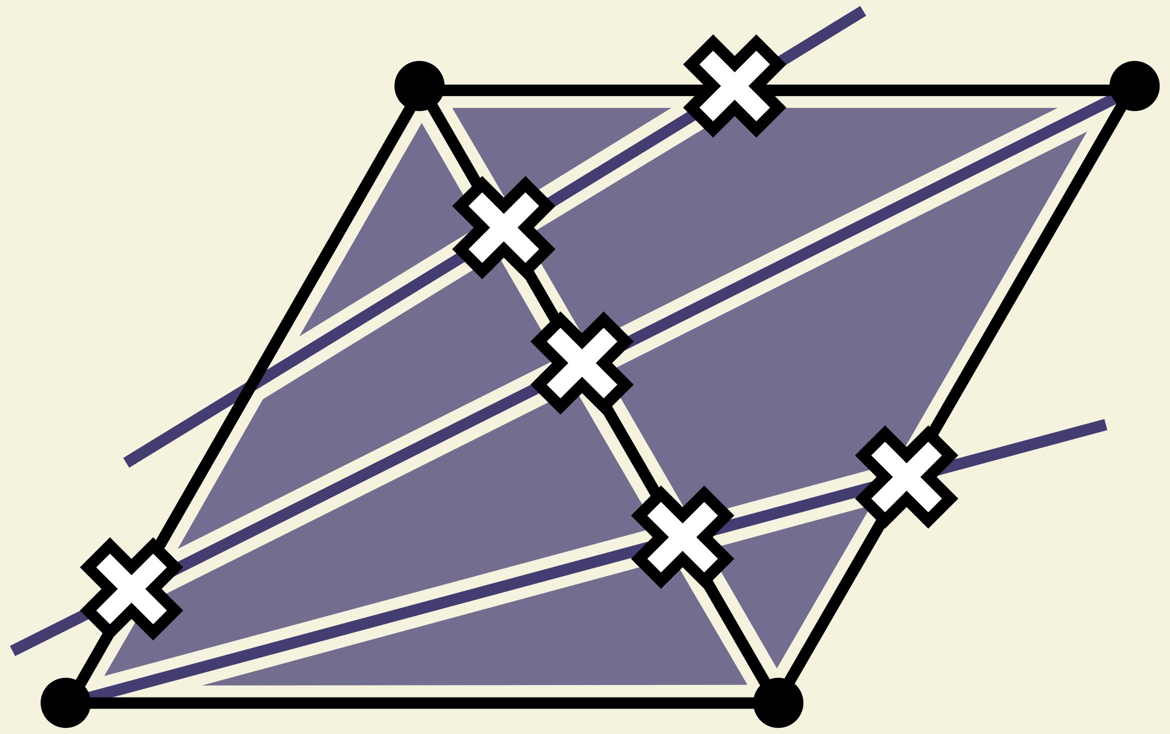
ideal Delaunay triangulation

Hyperbolic correspondence problem



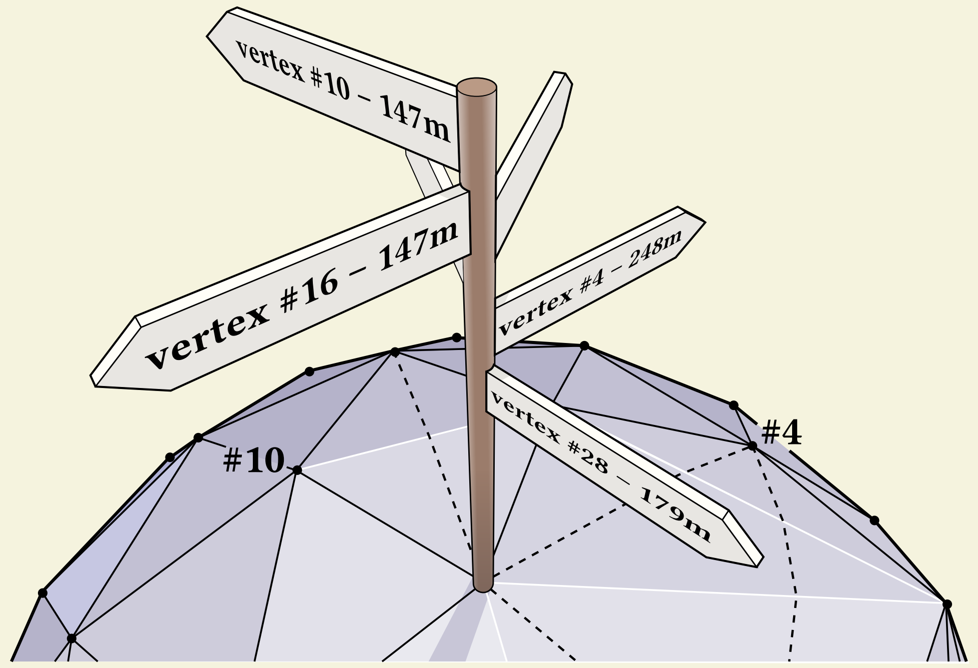
Correspondence between hyperbolic polyhedra

- Adapt Euclidean techniques to hyperbolic setting?



[Fisher, Springborn, Bobenko & Schröder 2006]

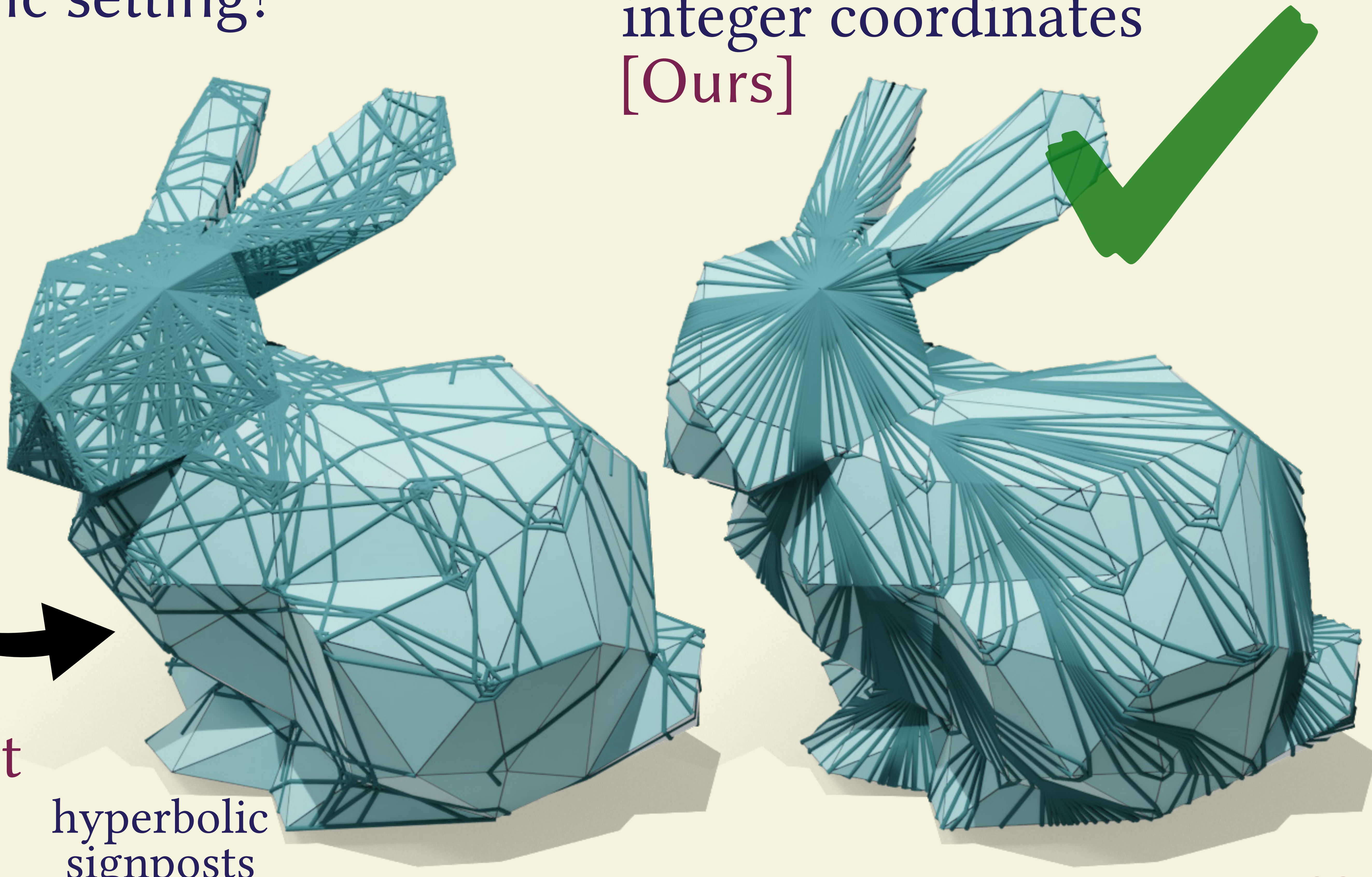
prohibitively complex



[Sharp, Soliman & Crane 2019]

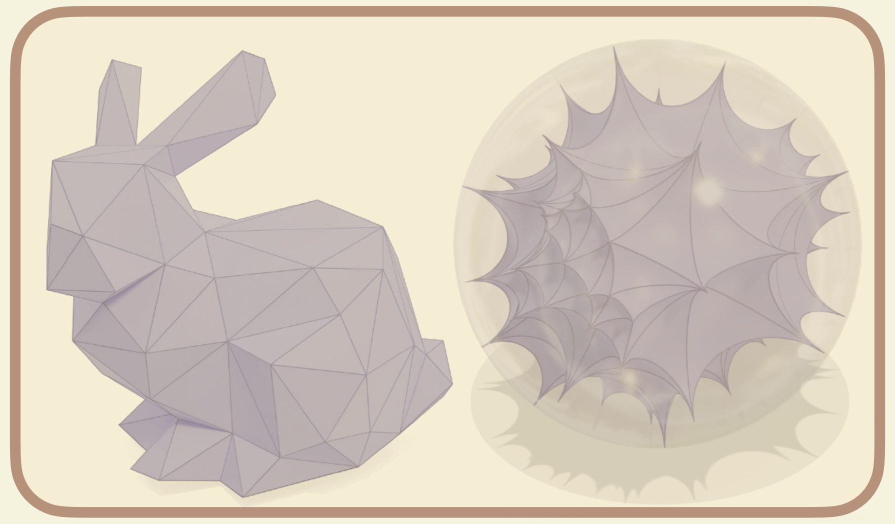
floating point errors

integer coordinates [Ours]



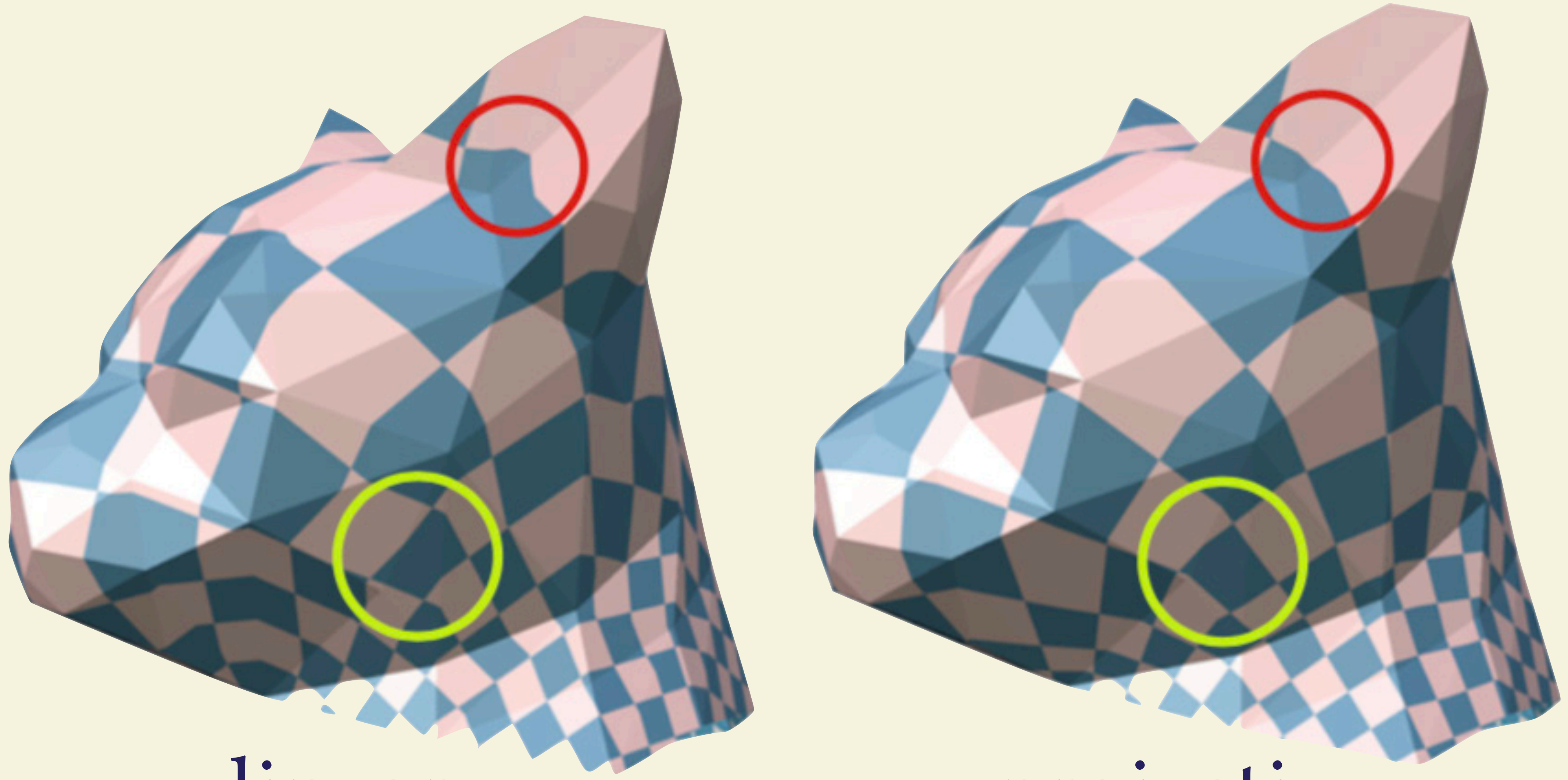
hyperbolic signposts

Projective interpolation

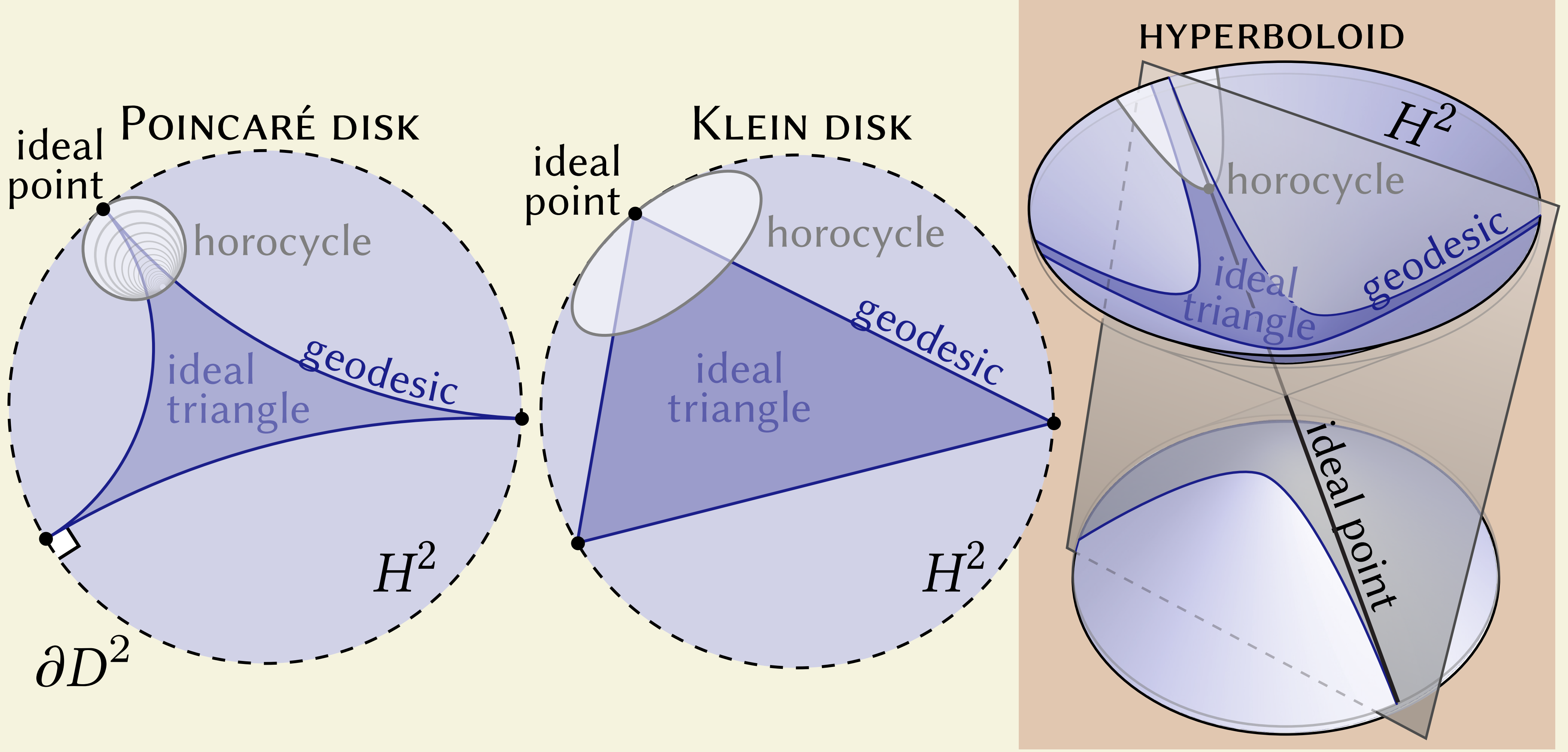


- [Springborn, Schröder & Pinkall 2008]: projective interpolation
  - Hyperbolic isometry
- [Ours]: novel projective interpolation using the hyperboloid model

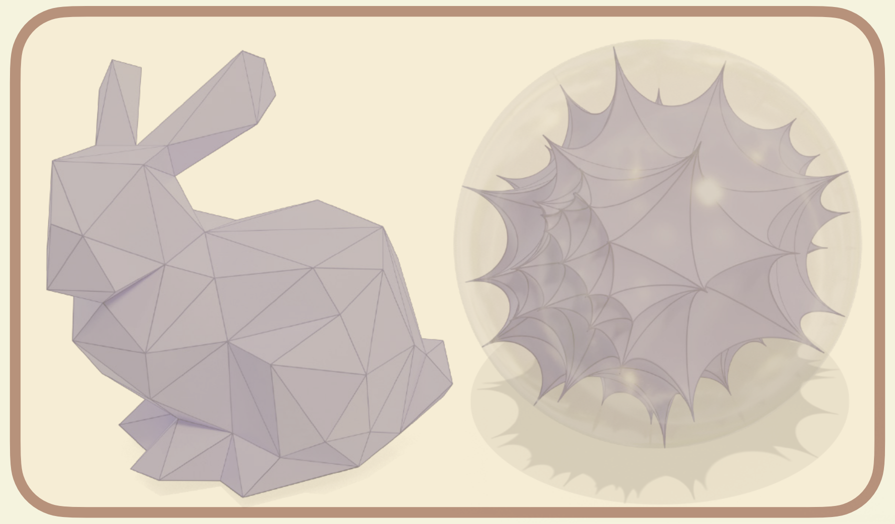
Image: [Springborn, Schröder & Pinkall 2008]

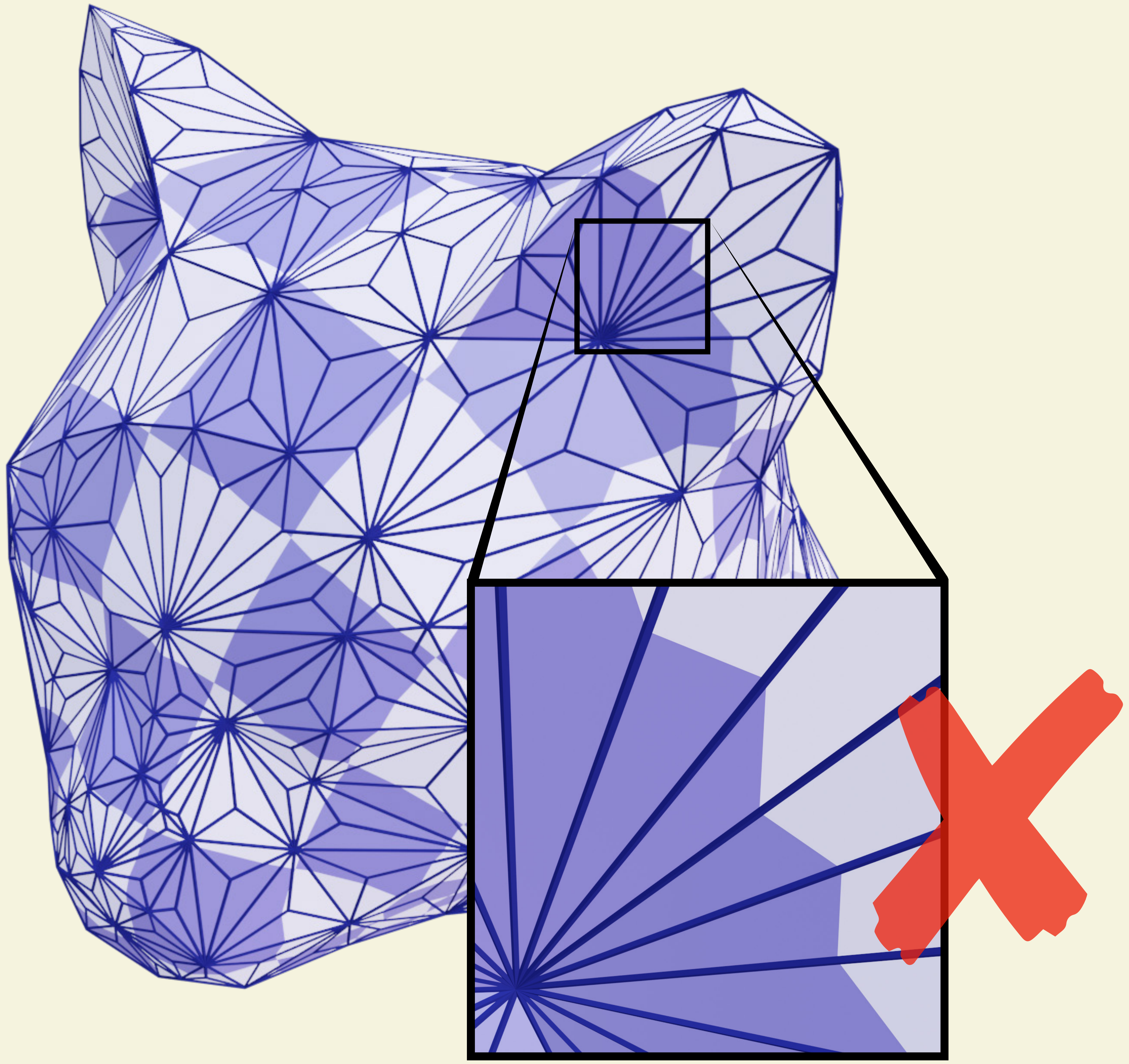


linear projective

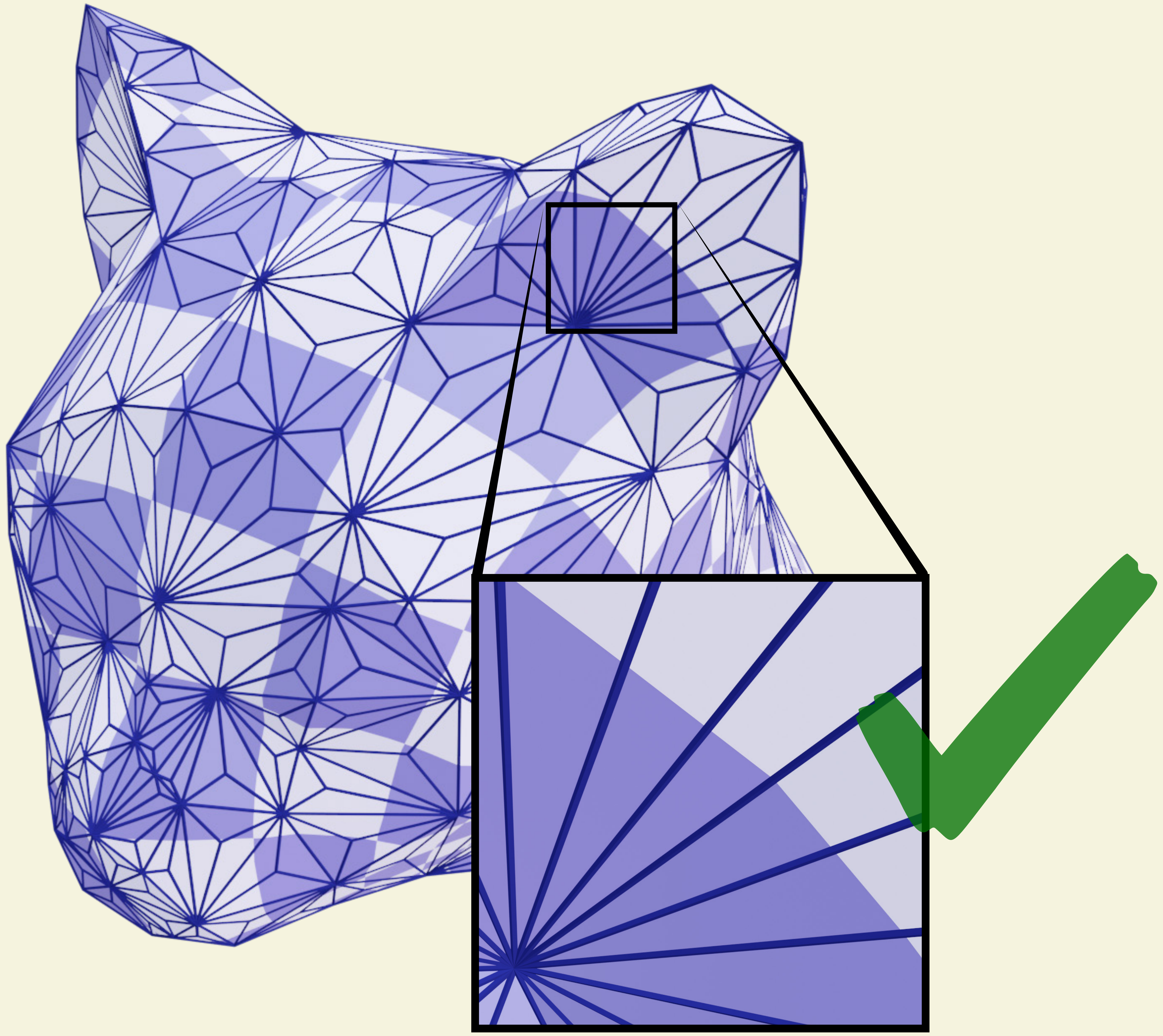


Variable triangulation > fixed triangulation



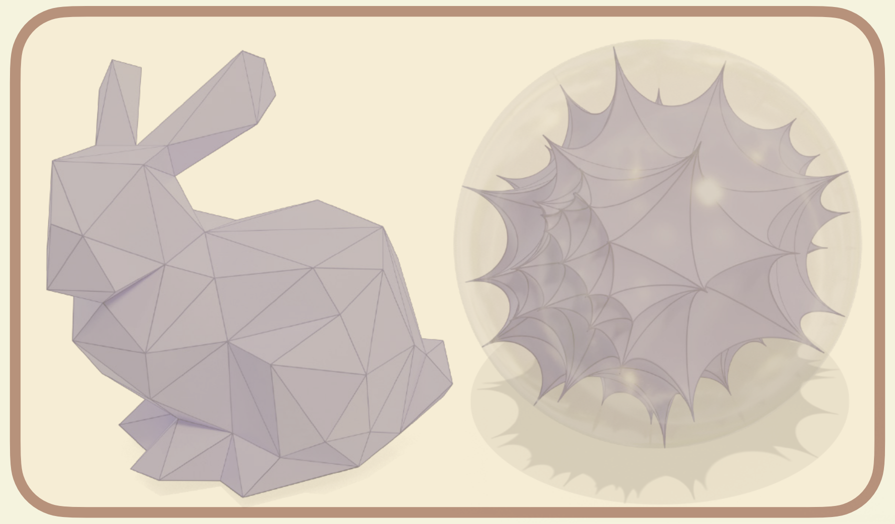


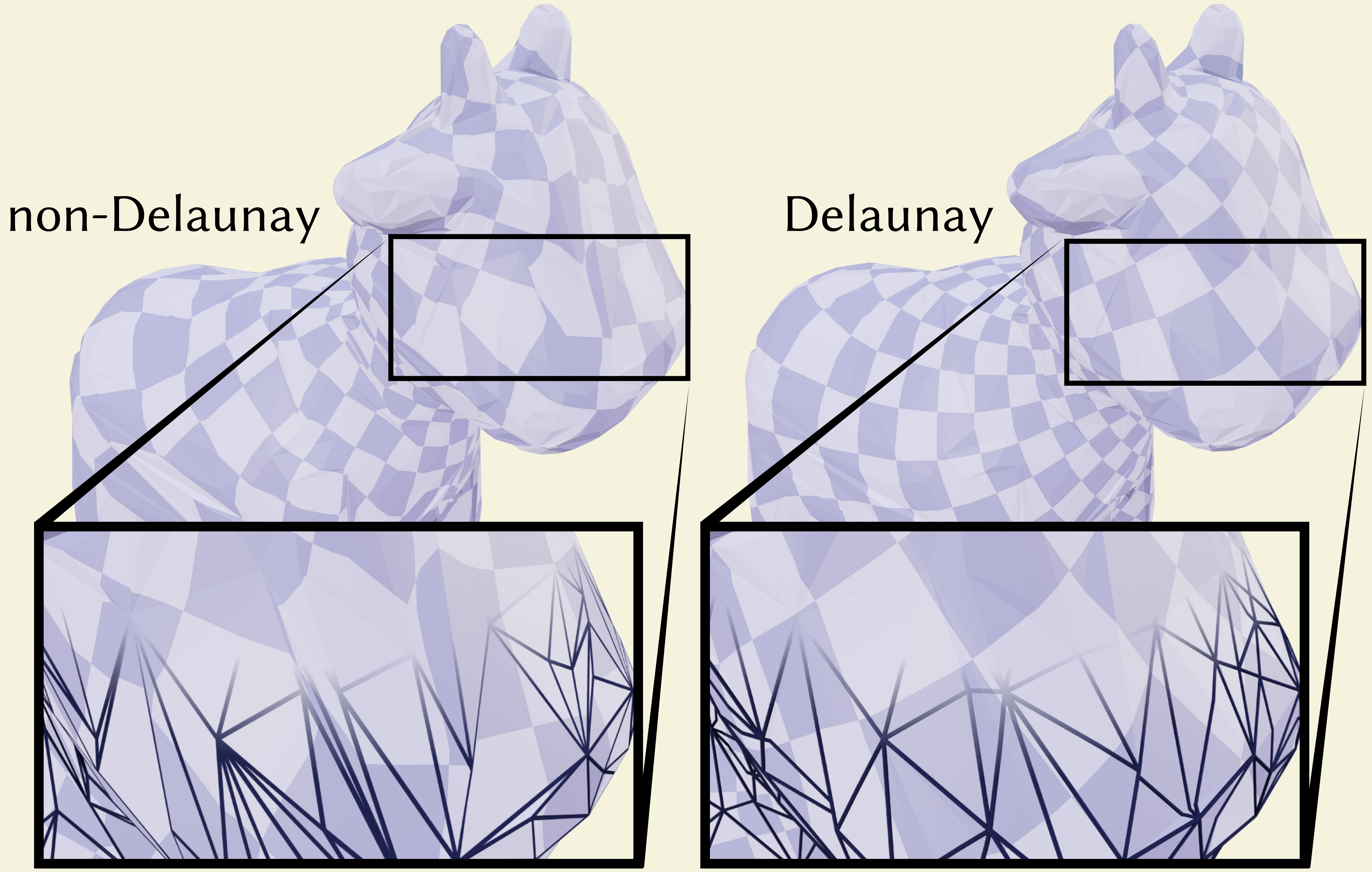
Fixed triangulation (CETM)

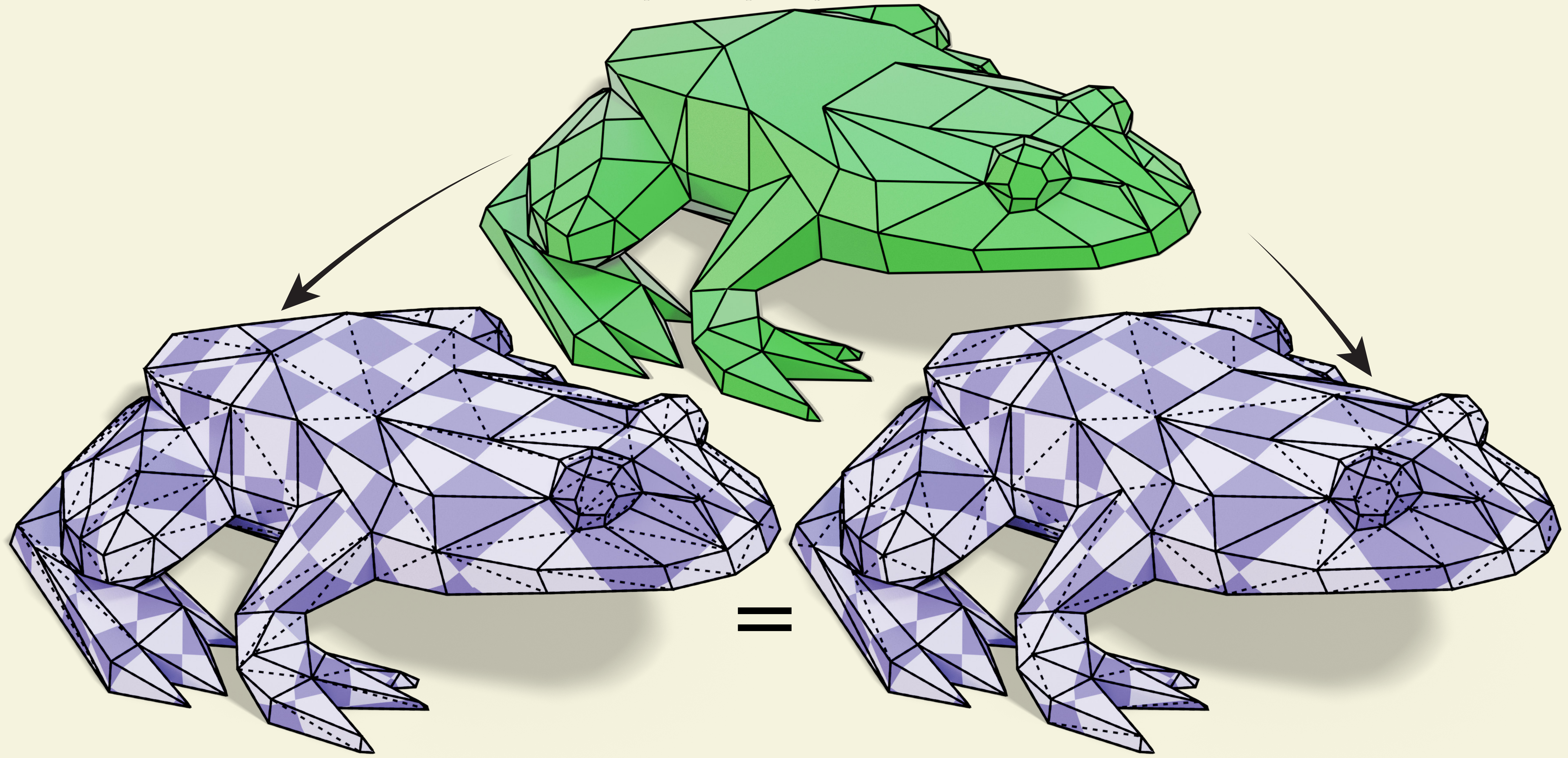


Variable triangulation (CEPS)

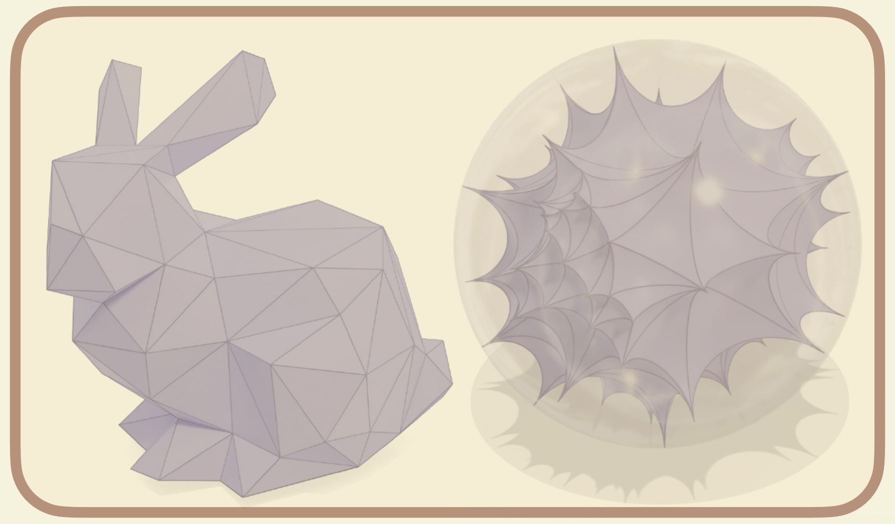
Starting from Delaunay







Final algorithm



flip to (Euclidean) Delaunay

solve for discrete conformal map

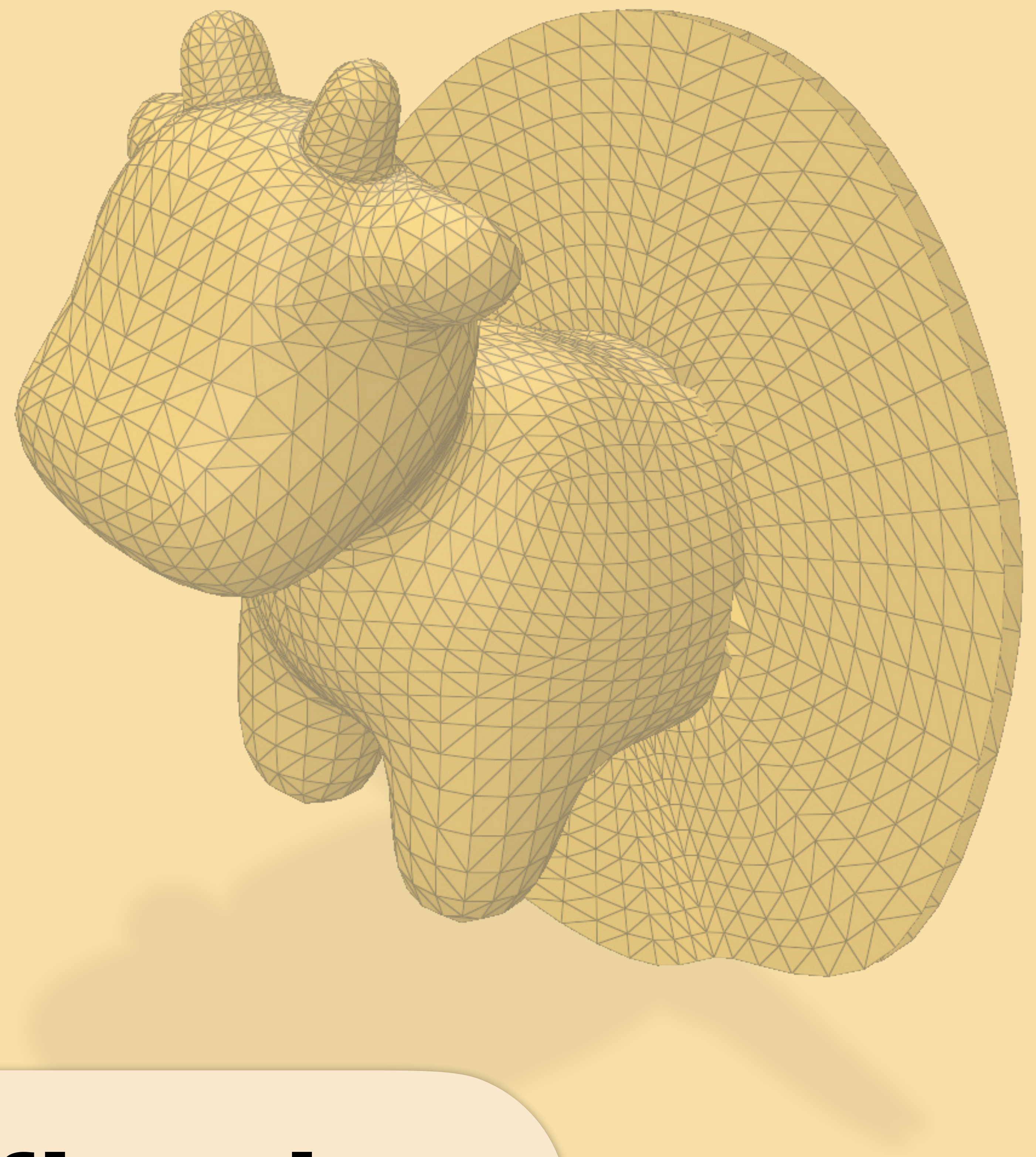
lay out in plane

extract correspondence

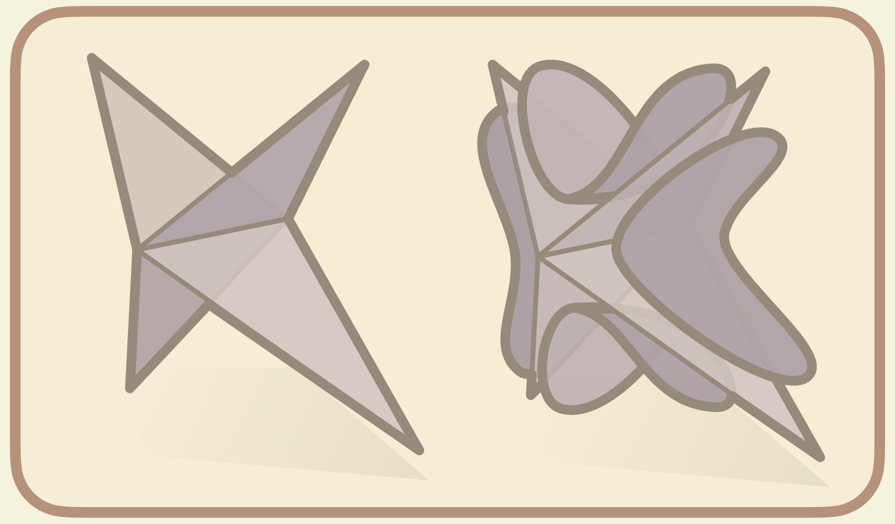
interpolate via hyperboloid

IV. Proposed work

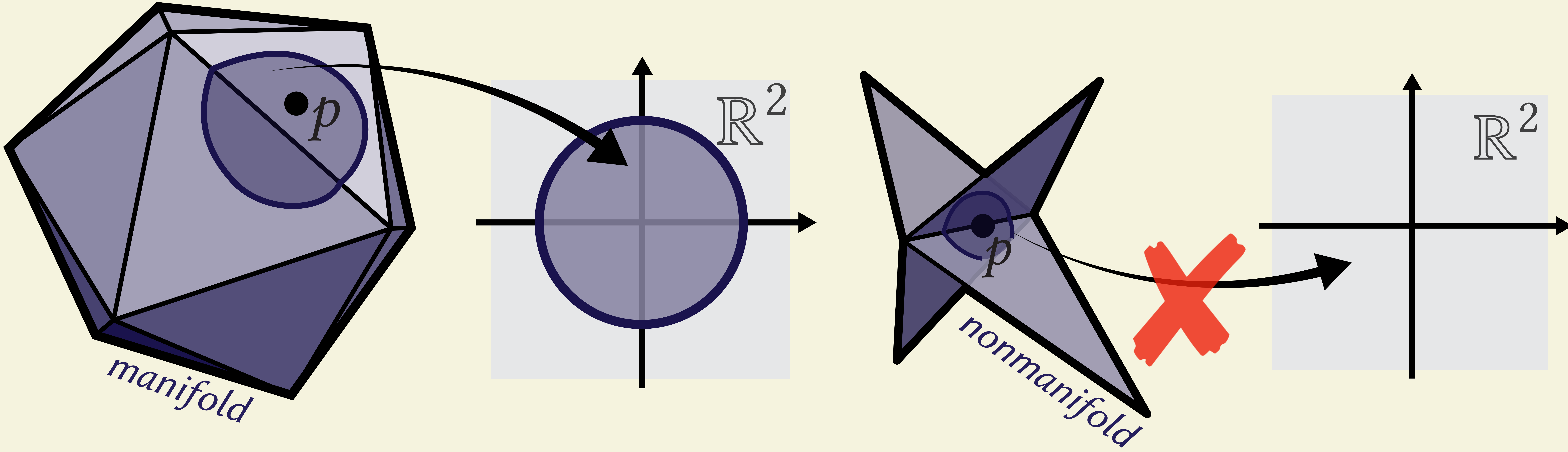
nonmanifold intrinsic simplification



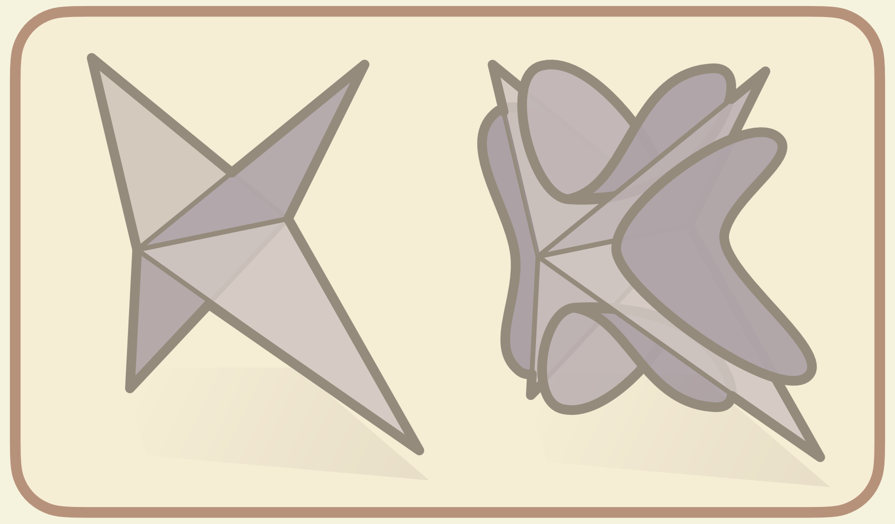
Problem: nonmanifold meshes



- *Manifold* : looks like the plane locally

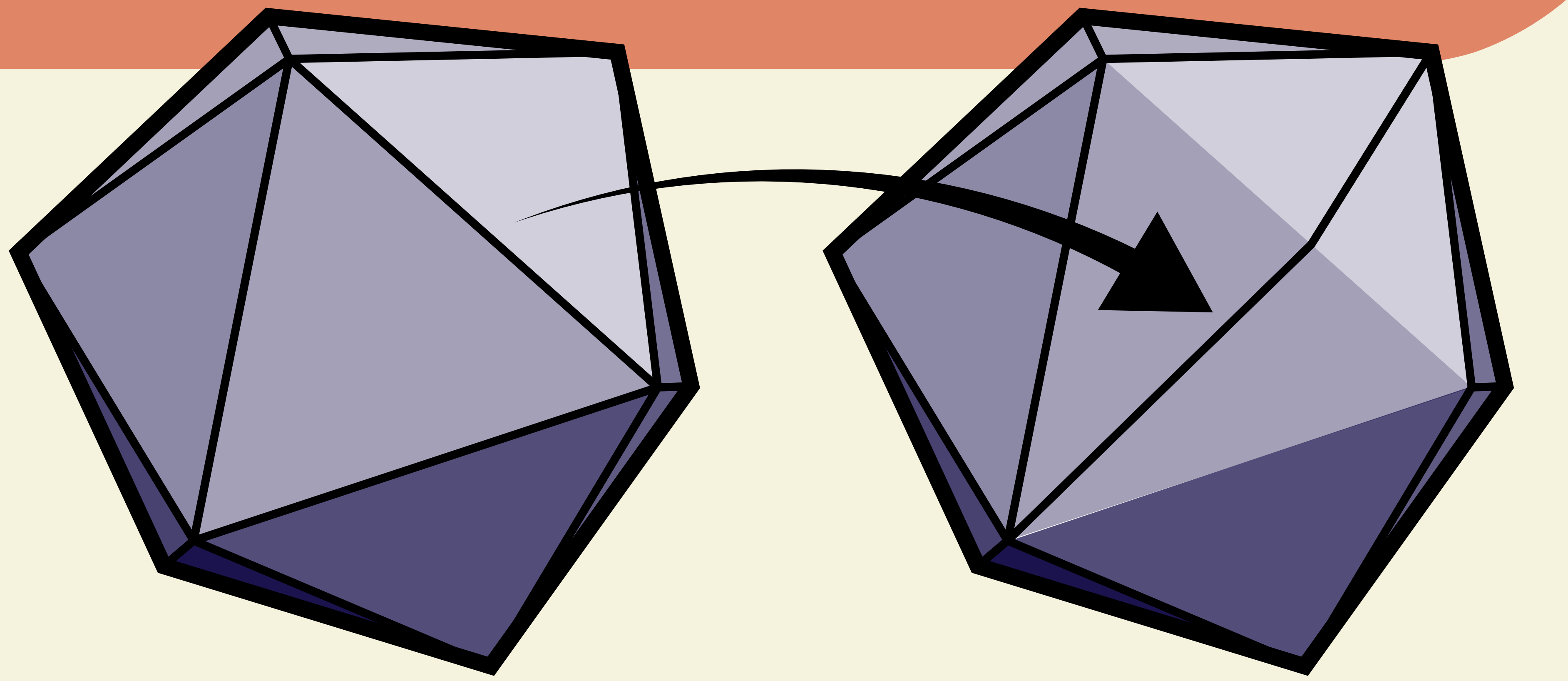


- Common simplifying assumption … but often violated in practice

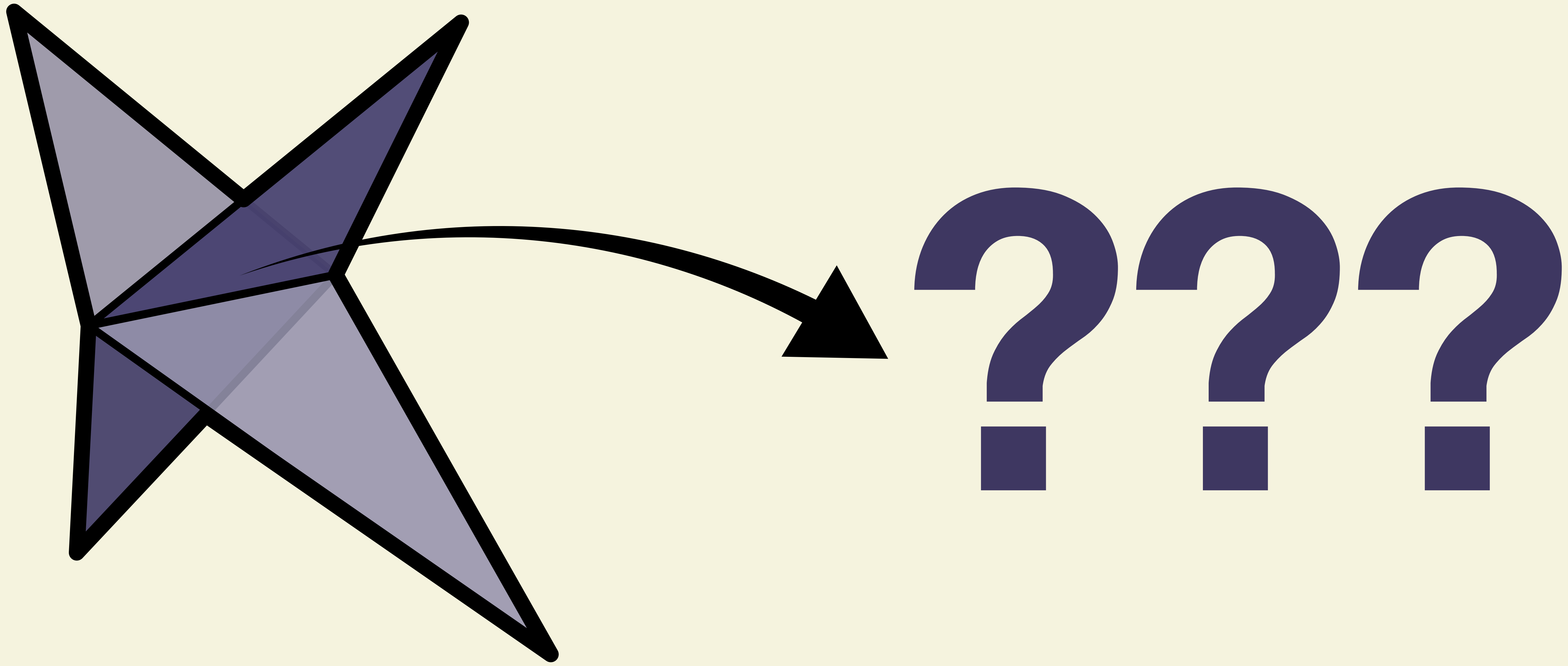


Nonmanifold meshes complicate the intrinsic picture

- Recall: edge flips

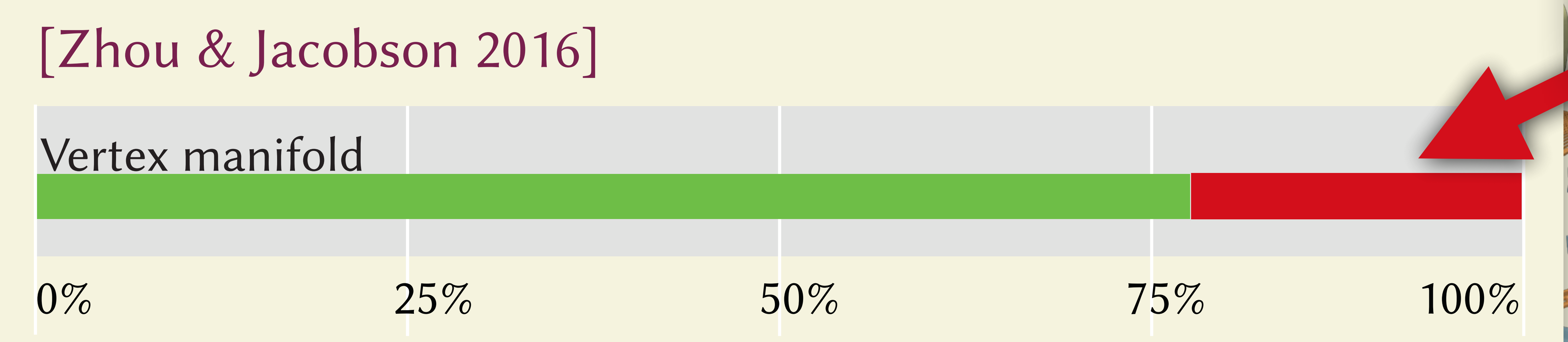


- What does this mean for nonmanifold edges?

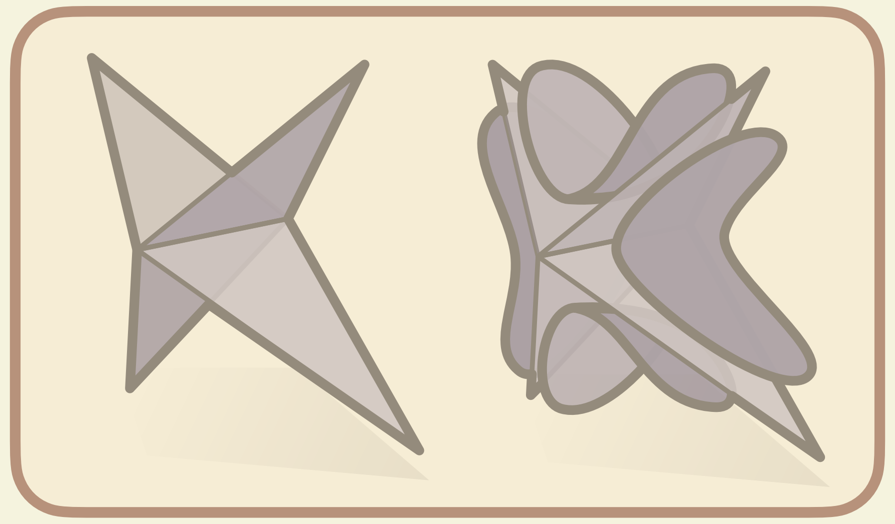


Lots of meshes are nonmanifold

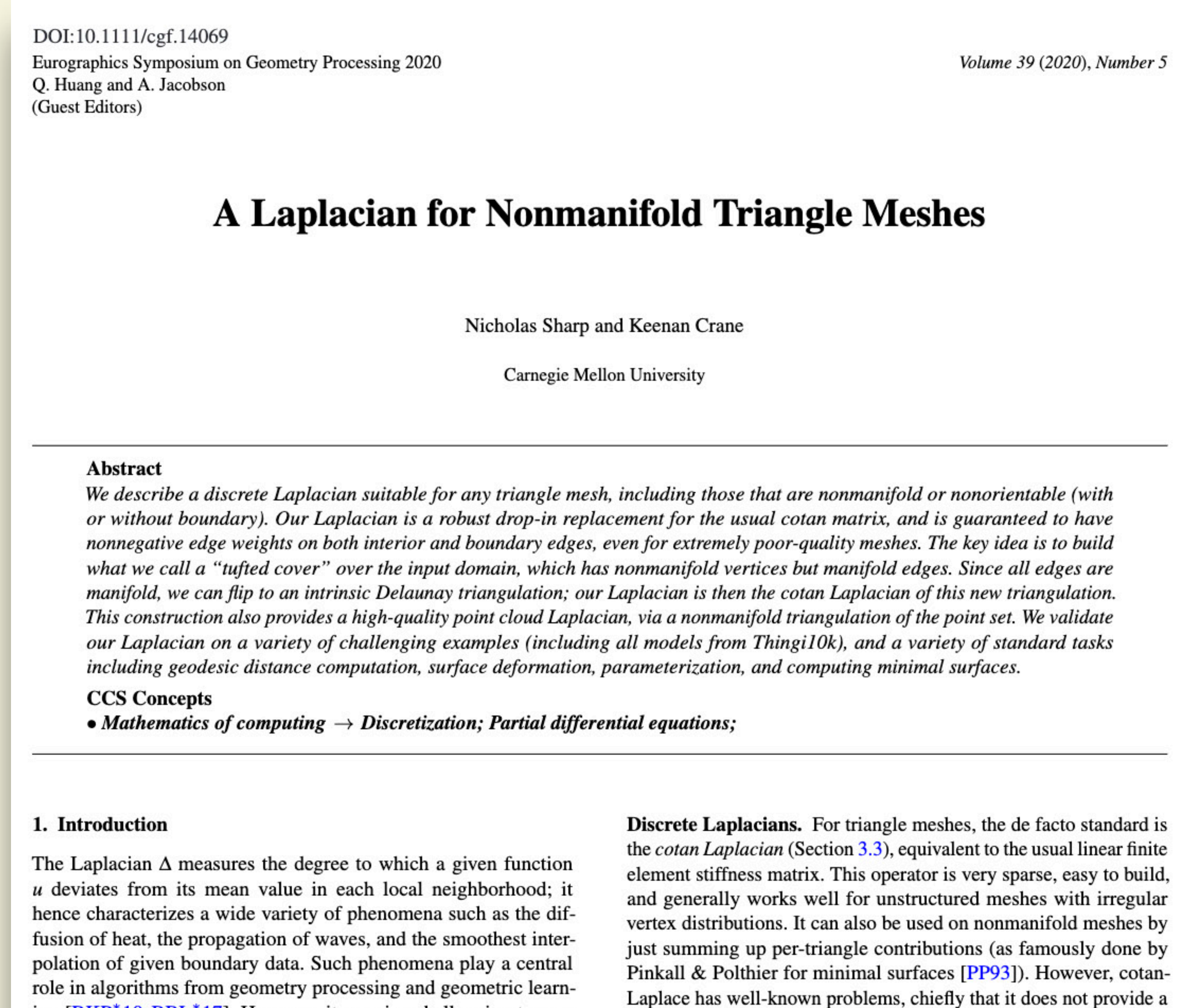
Manifold assumption fails on all these meshes



Solution: the manifold double cover



- Build associated manifold mesh to work with instead
- Follow [Sharp & Crane 2020]



DOI:10.1111/cgf.14069
Eurographics Symposium on Geometry Processing 2020
Q. Huang and A. Jacobson
(Guest Editors)

Volume 39 (2020), Number 5

A Laplacian for Nonmanifold Triangle Meshes

Nicholas Sharp and Keenan Crane

Carnegie Mellon University

Abstract

We describe a discrete Laplacian suitable for any triangle mesh, including those that are nonmanifold or nonorientable (with or without boundary). Our Laplacian is a robust drop-in replacement for the usual cotan matrix, and is guaranteed to have nonnegative edge weights on both interior and boundary edges, even for extremely poor-quality meshes. The key idea is to build what we call a "tufted cover" over the input domain, which has nonmanifold vertices but manifold edges. Since all edges are manifold, we can flip to an intrinsic Delaunay triangulation; our Laplacian is then the cotan Laplacian of this new triangulation. This construction also provides a high-quality point cloud Laplacian, via a nonmanifold triangulation of the point set. We validate our Laplacian on a variety of challenging examples (including all models from Thingi10k), and a variety of standard tasks including geodesic distance computation, surface deformation, parameterization, and computing minimal surfaces.

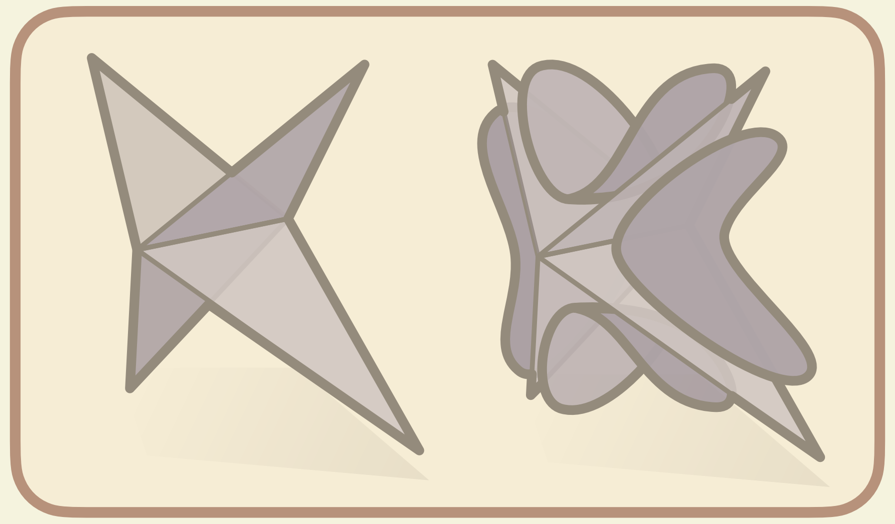
CCS Concepts
• ***Mathematics of computing*** → ***Discretization; Partial differential equations;***

1. Introduction

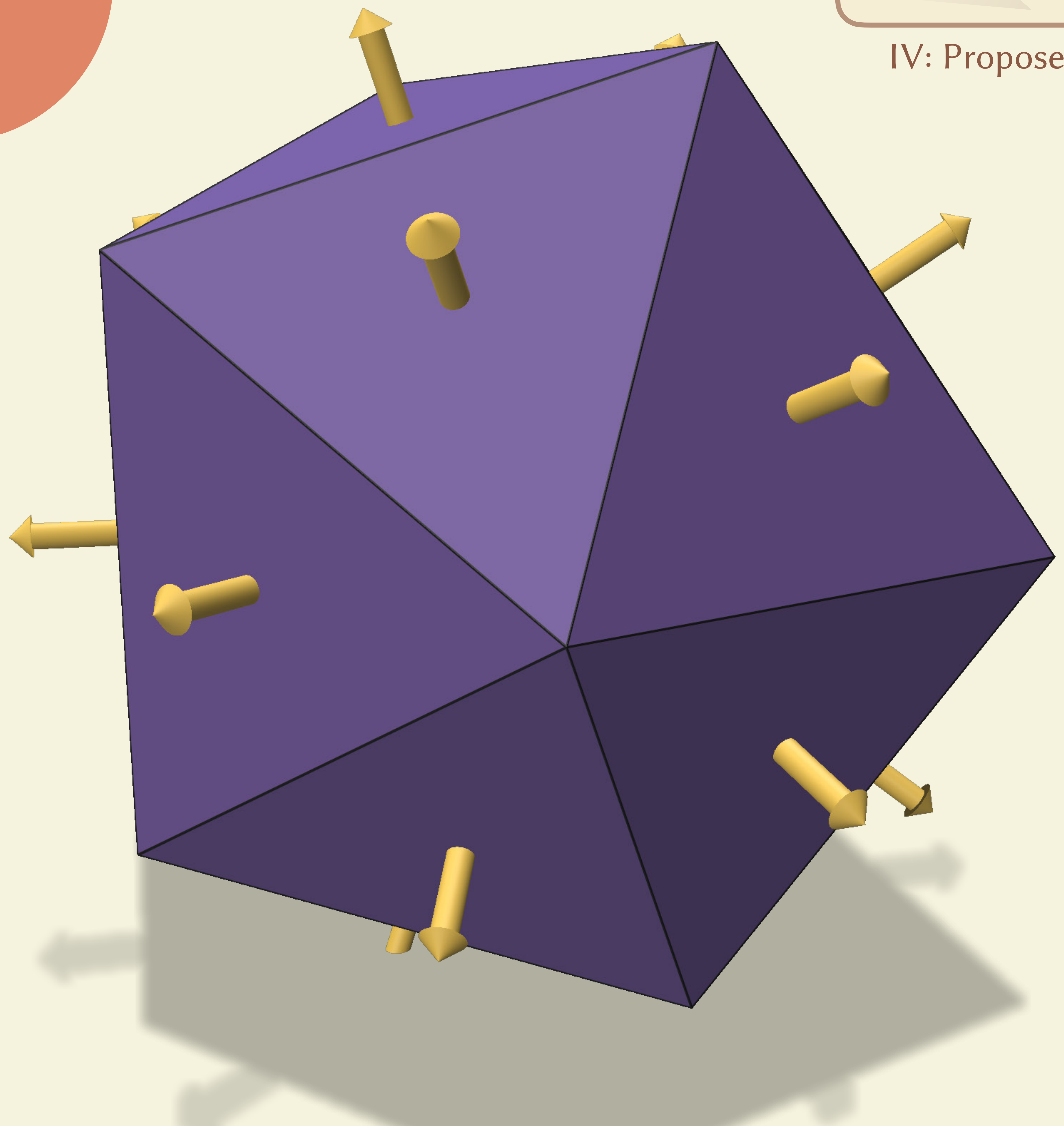
The Laplacian Δ measures the degree to which a given function *u* deviates from its mean value in each local neighborhood; it hence characterizes a wide variety of phenomena such as the diffusion of heat, the propagation of waves, and the smoothest interpolation of given boundary data. Such phenomena play a central role in algorithms from geometry processing and geometric learn-

Discrete Laplacians. For triangle meshes, the de facto standard is the *cotan Laplacian* (Section 3.3), equivalent to the usual linear finite element stiffness matrix. This operator is very sparse, easy to build, and generally works well for unstructured meshes with irregular vertex distributions. It can also be used on nonmanifold meshes by just summing up per-triangle contributions (as famously done by Pinkall & Polthier for minimal surfaces [PP93]). However, cotan-Laplace has well-known problems, chiefly that it does not provide a

Prelude: orienting nonorientable meshes



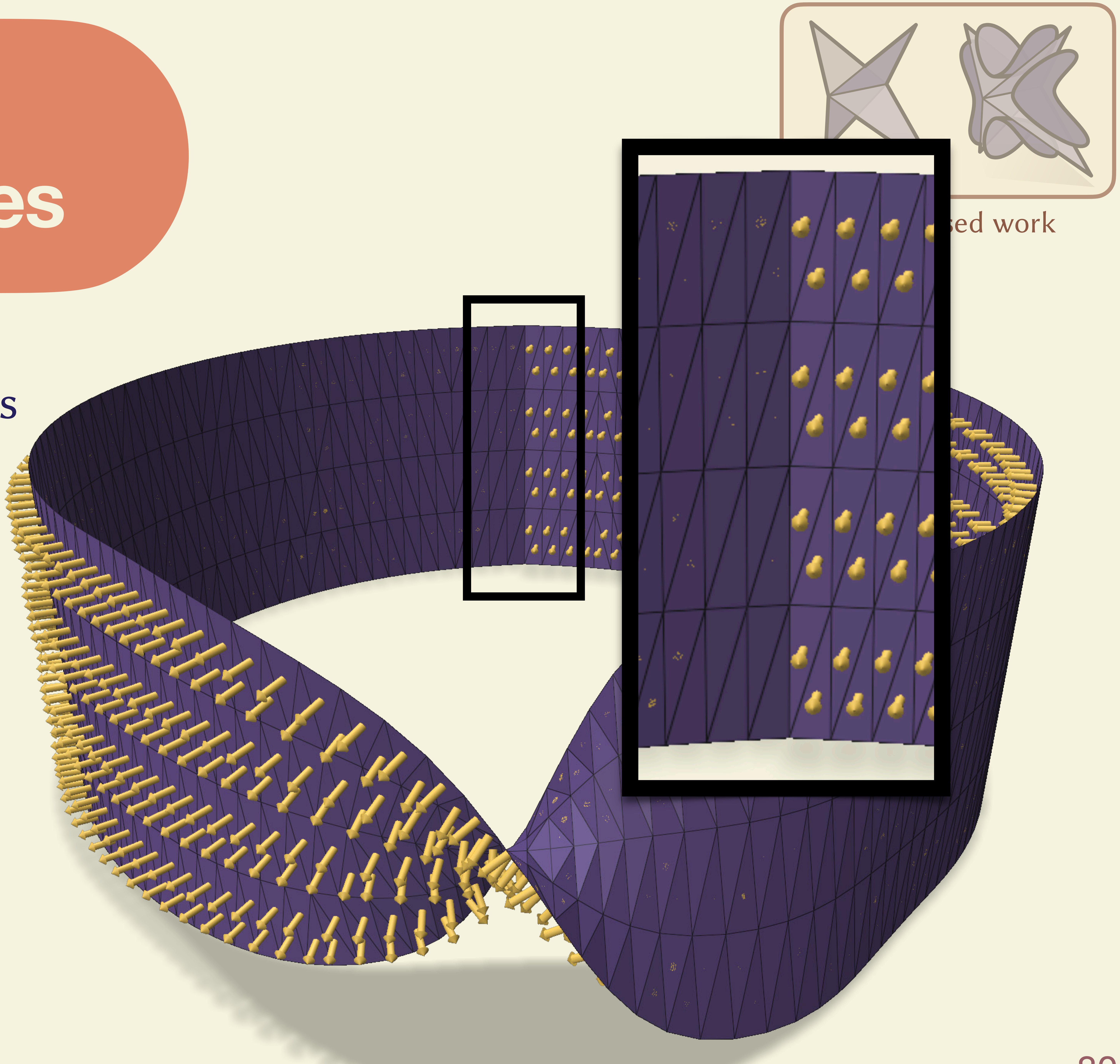
- Orientation distinguishes two sides
- Visualize with arrows



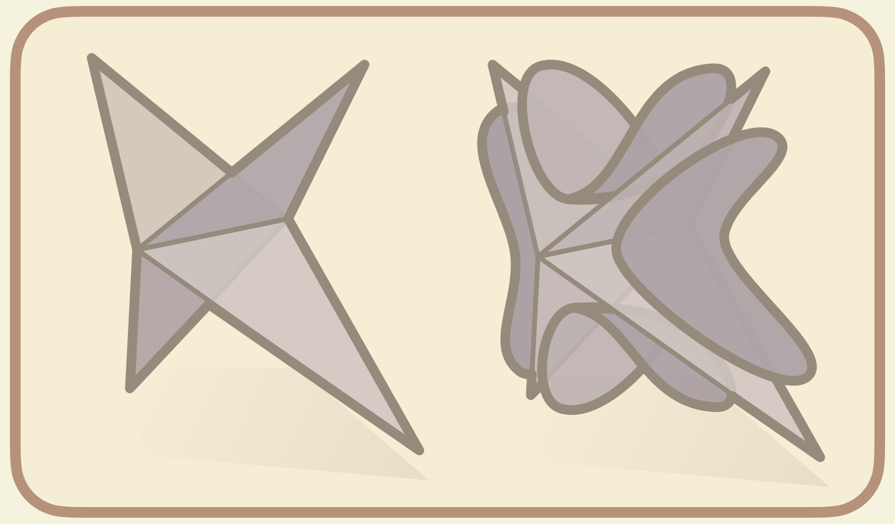
Prelude: orienting nonorientable meshes

- Orientation distinguishes two sides
- Visualize with arrows
- Not every surface is orientable

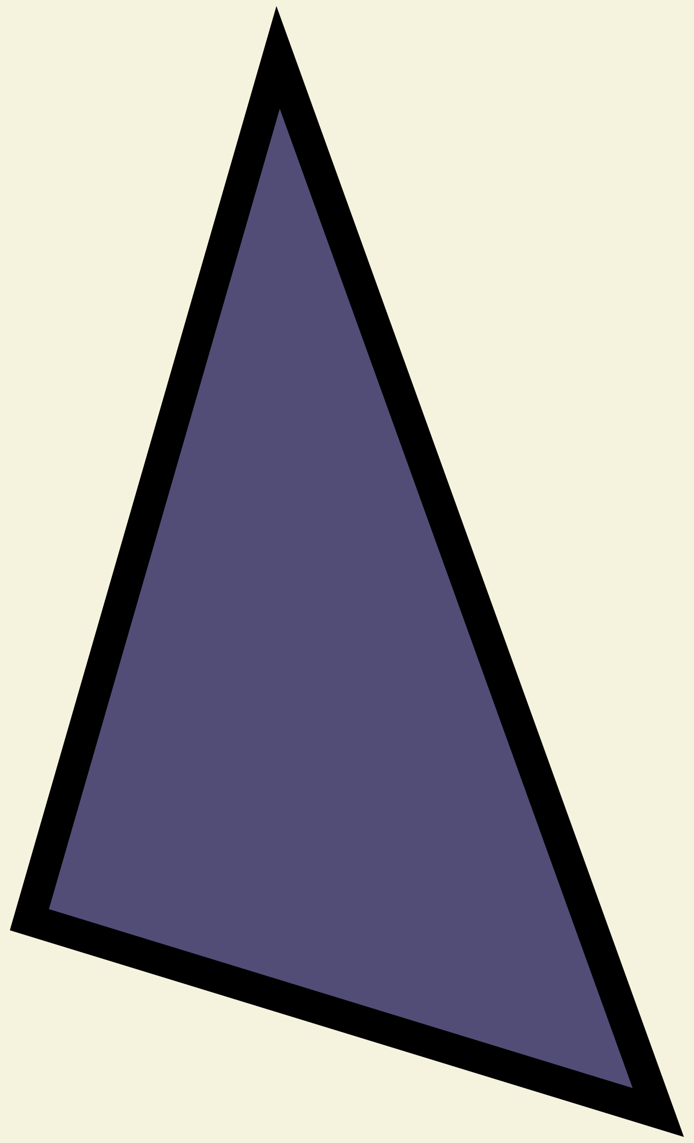
Problem: no consistent choice of arrow for all faces

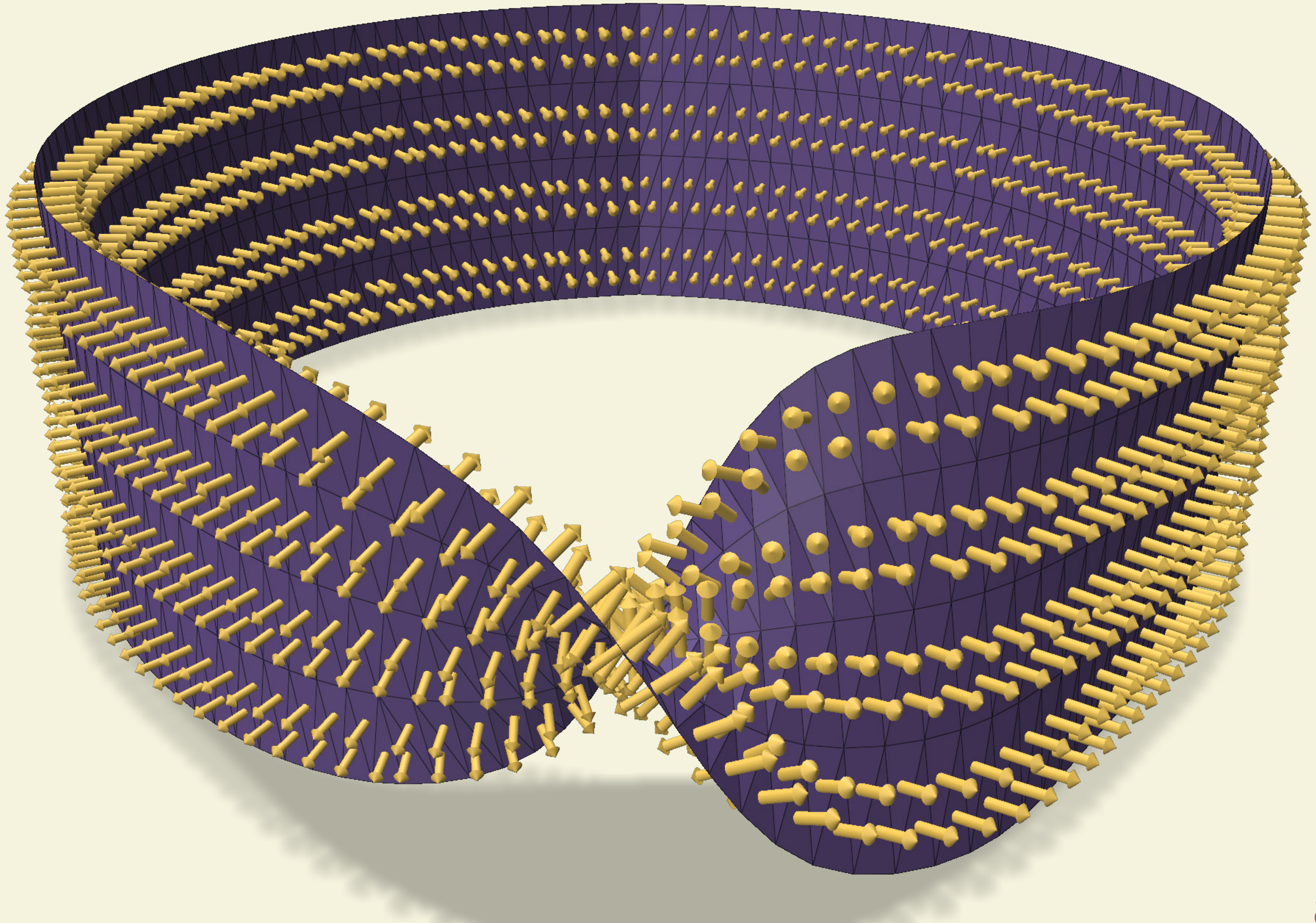


Easy “solution”: draw both arrows

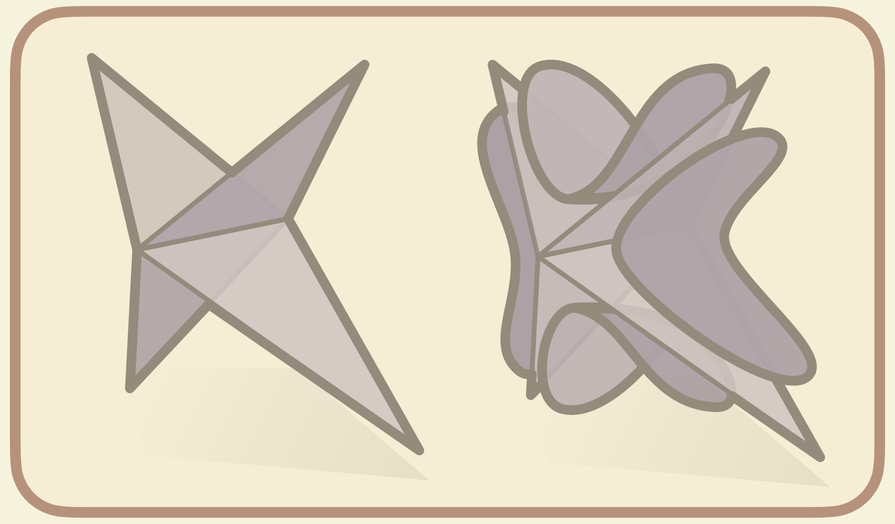


- Sounds like cheating…
  - … but contains a good idea

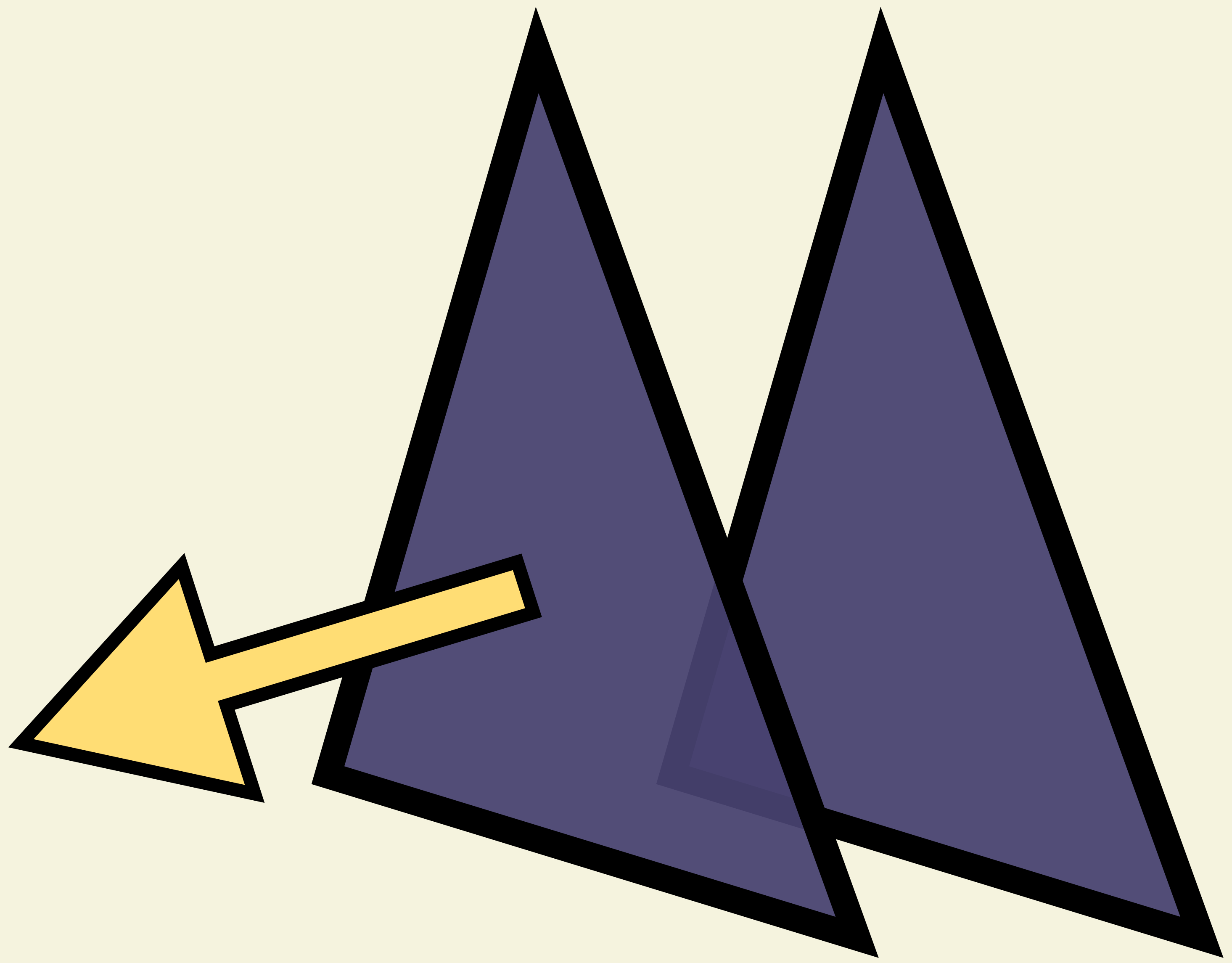


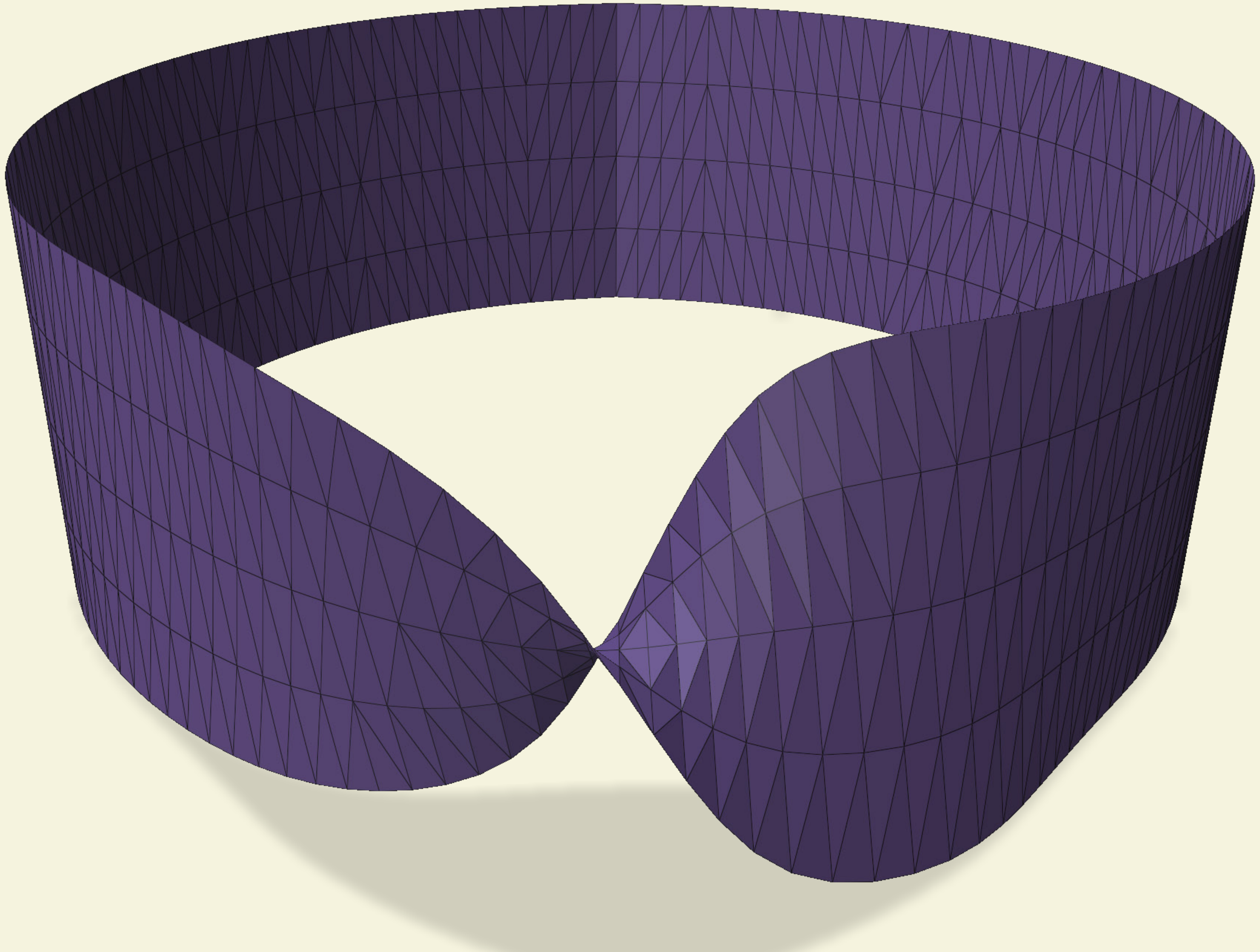


Easy "solution": draw both arrows



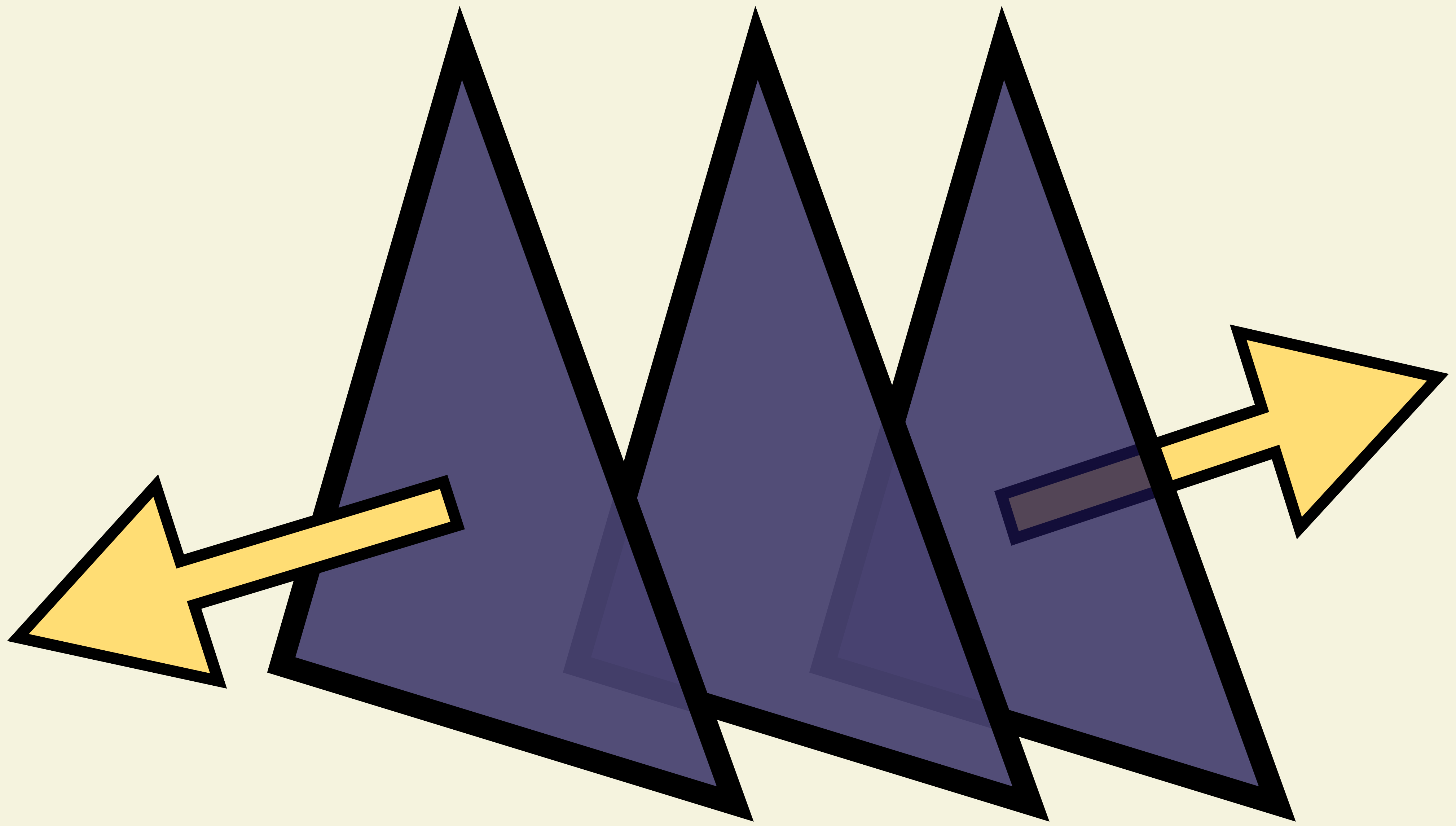
- Sounds like cheating...
    - ... but contains a good idea

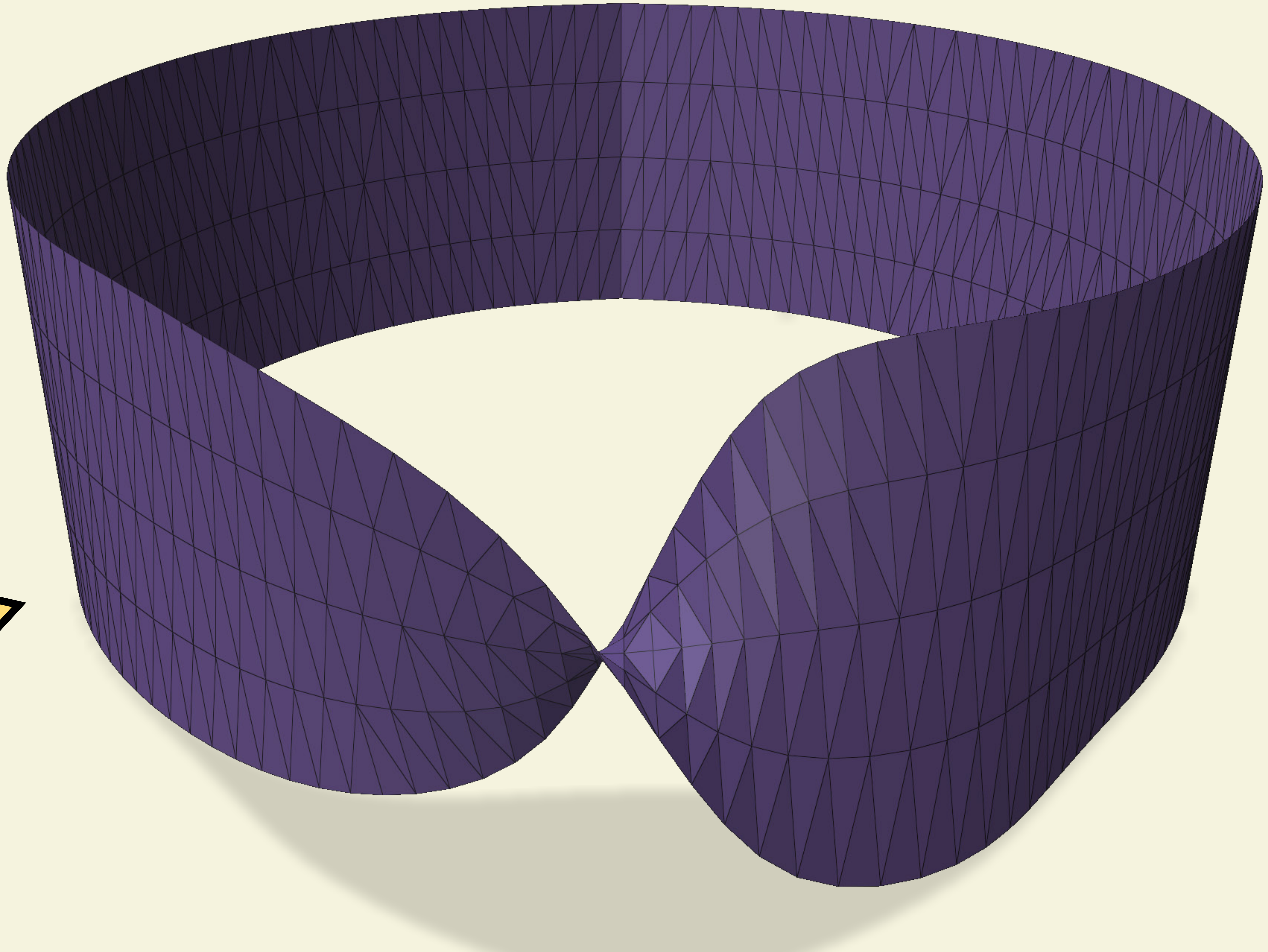




Easy "solution": draw both arrows

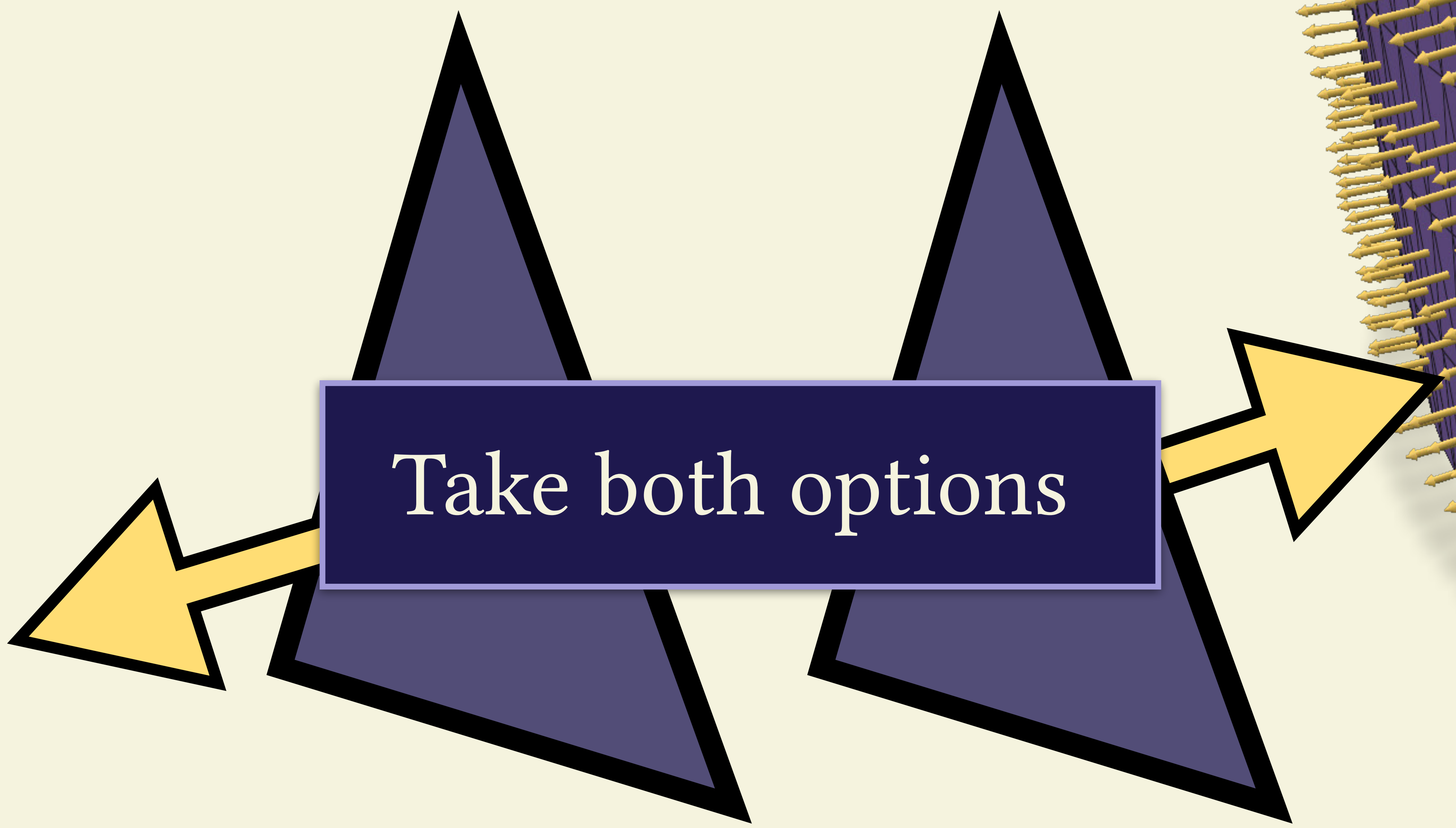
- Sounds like cheating…
  - … but contains a good idea

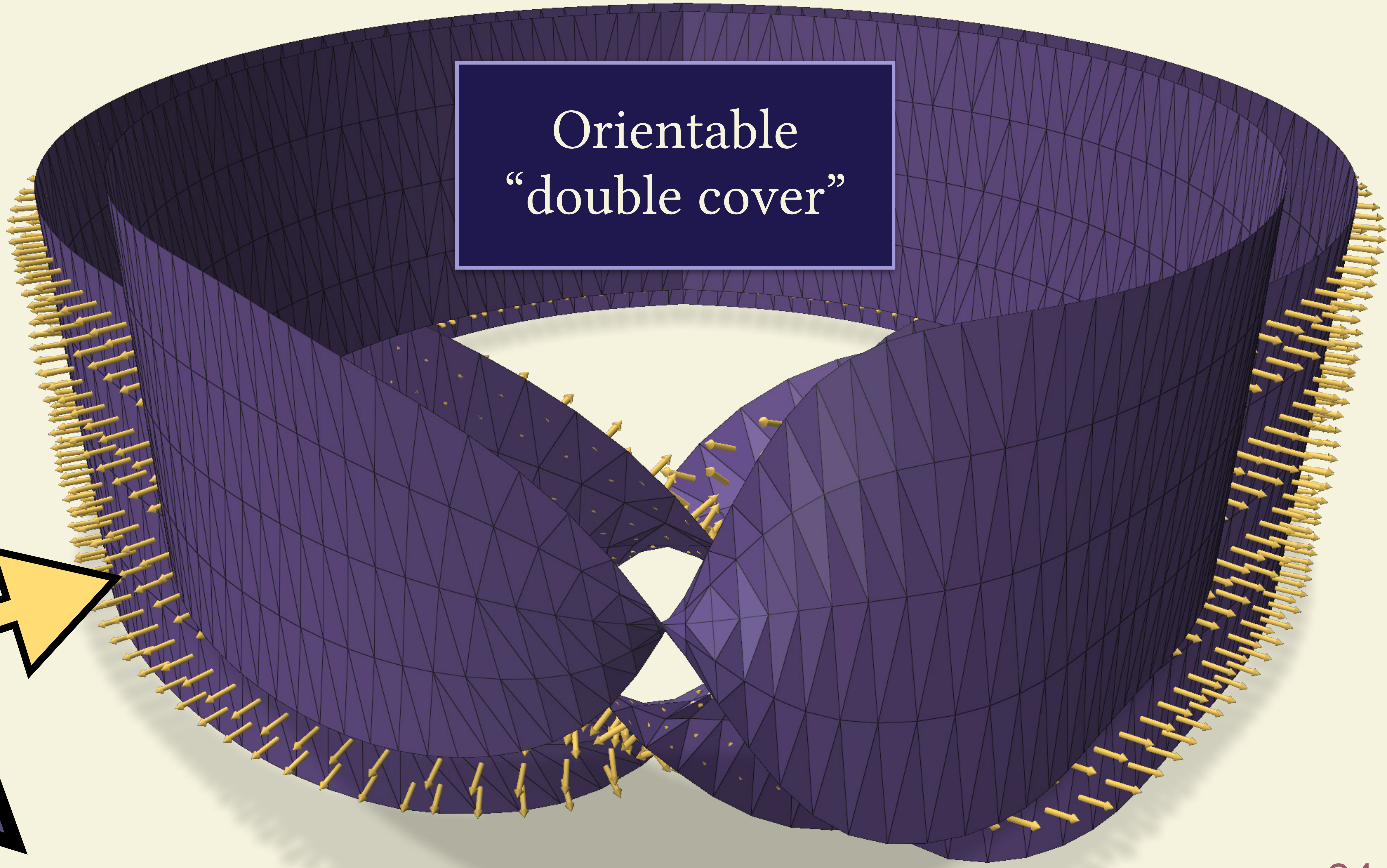




Easy "solution": draw both arrows

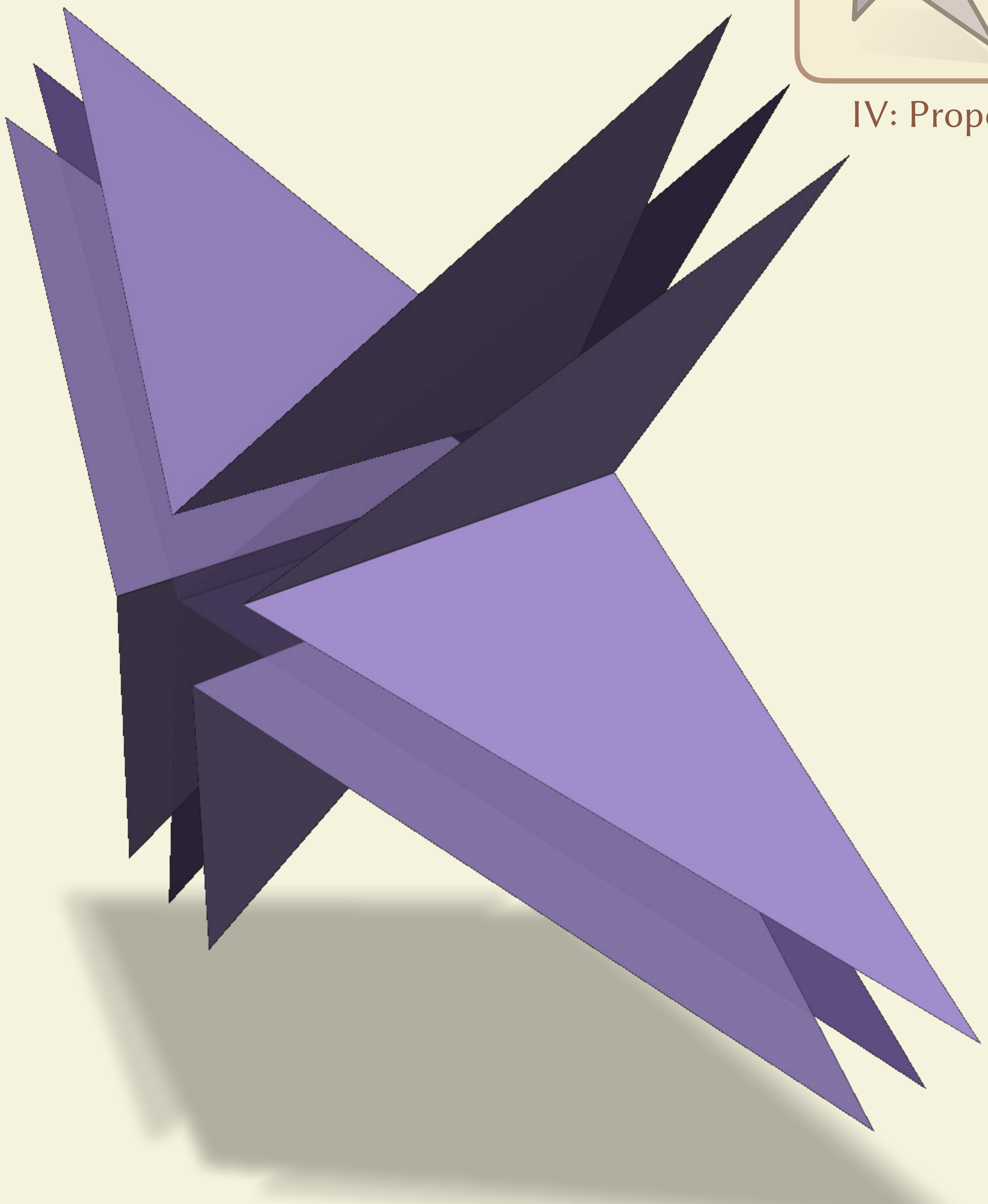
- Sounds like cheating...
  - ... but contains a good idea

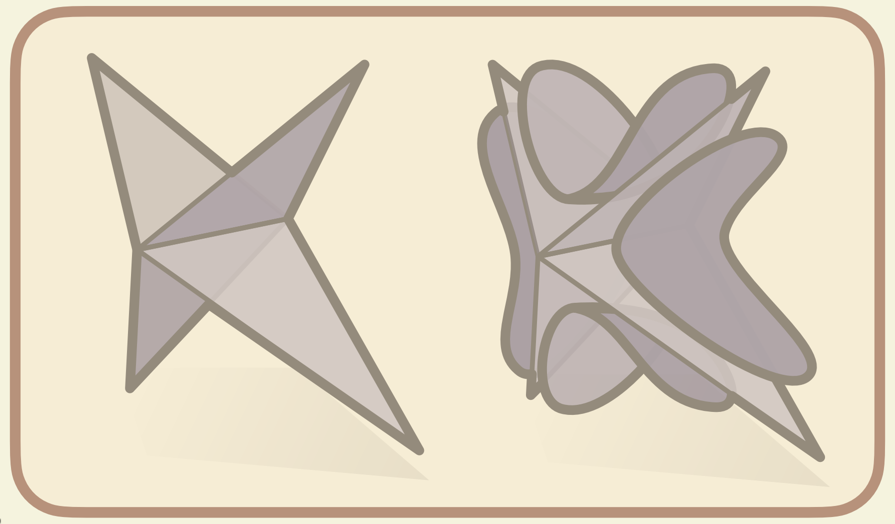




And it also works on nonmanifold meshes

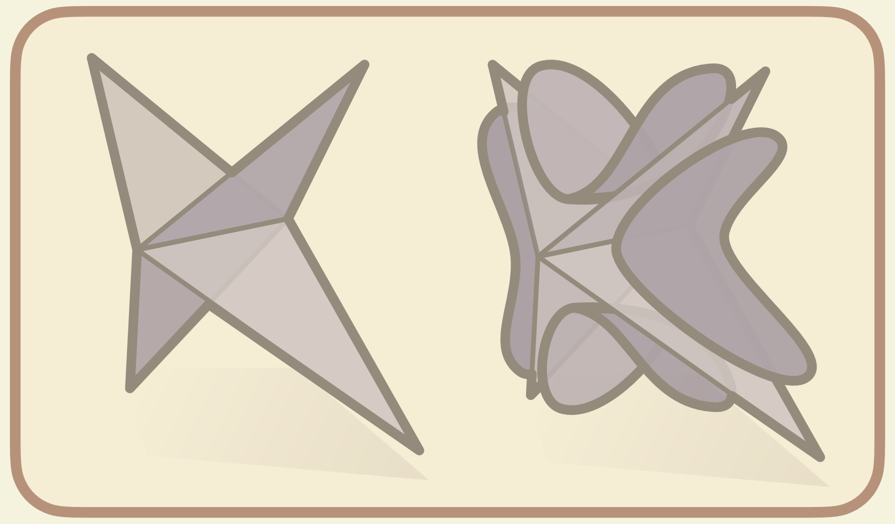
- Just make two copies of each face
- General strategy for nonmanifold geometry processing:
    1. Build manifold double cover
    2. Do manifold geometry processing





IV: Proposed work

Intrinsic simplification of nonmanifold meshes



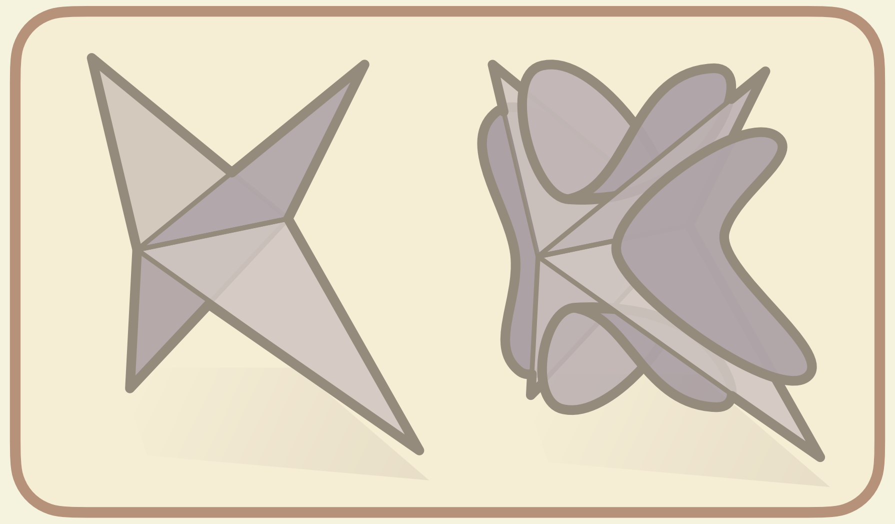
nonmanifold mesh

1. build double cover

2. simplify intrinsically

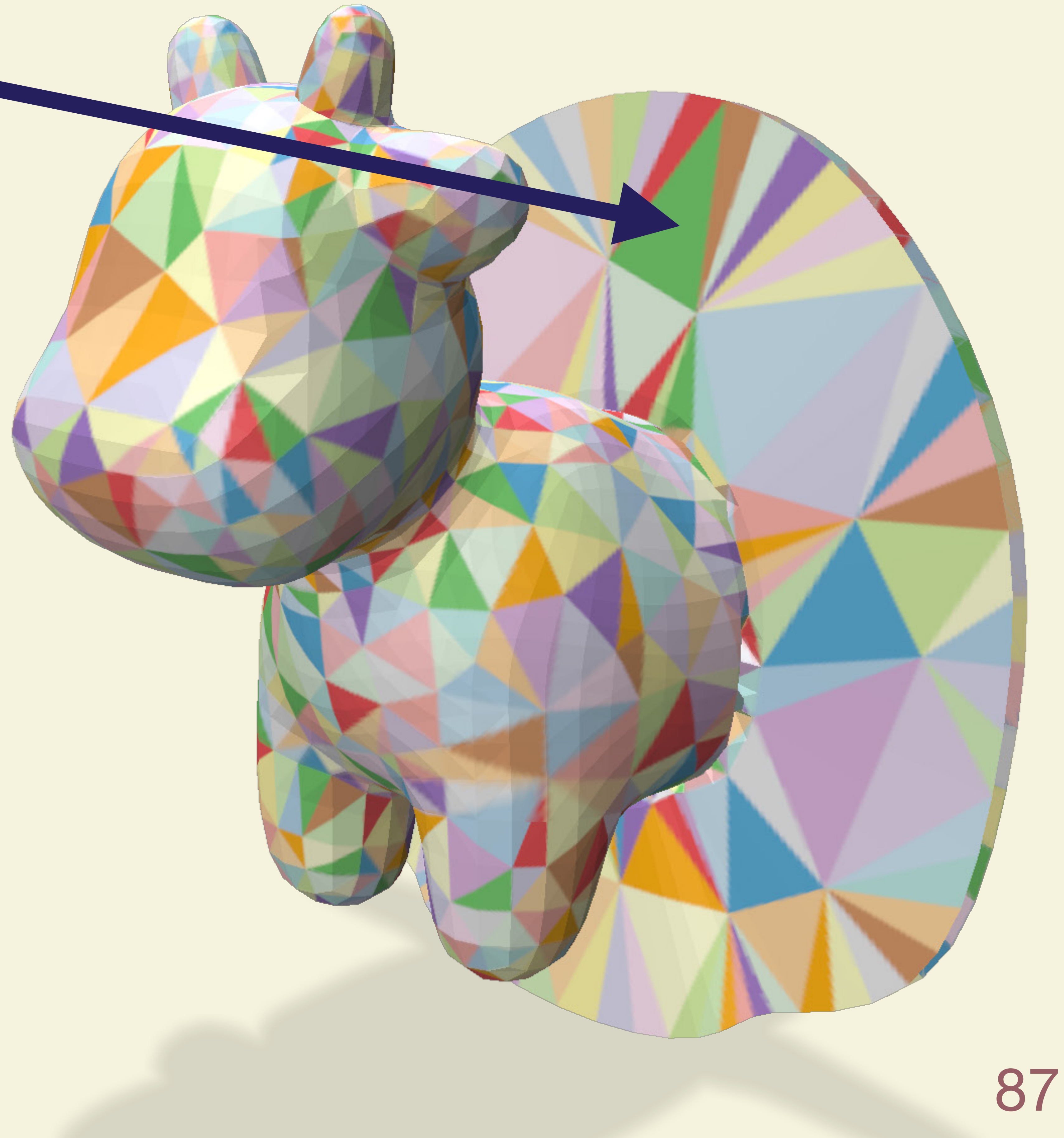
Questions to explore:

compatibility between sheets



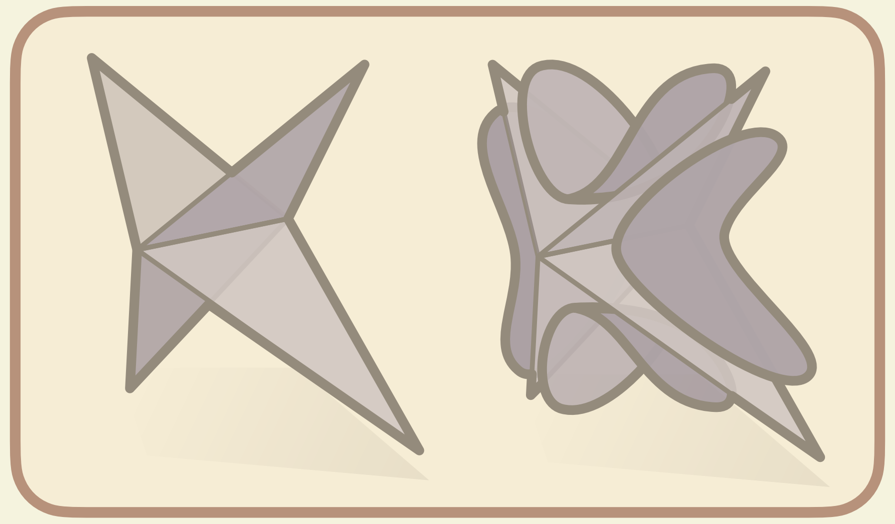
removed different vertices

does this matter?

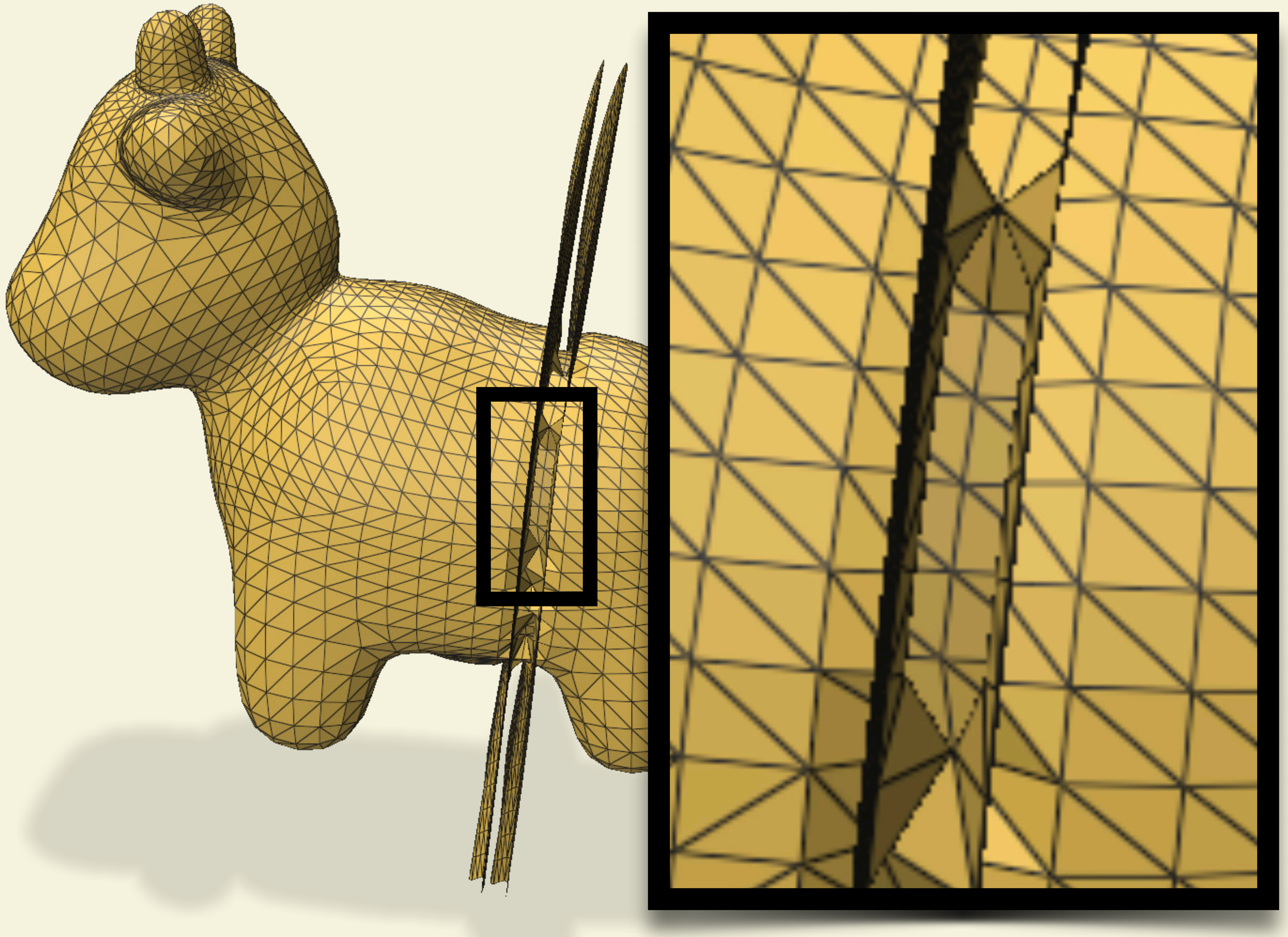


Questions to explore:

choice of double cover

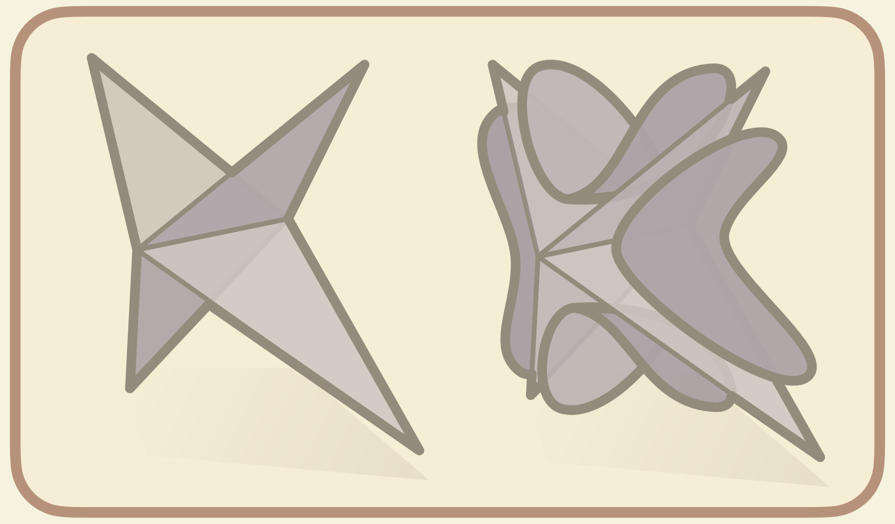


Multiple options for double cover



which is best?
does it matter?

Timeline:

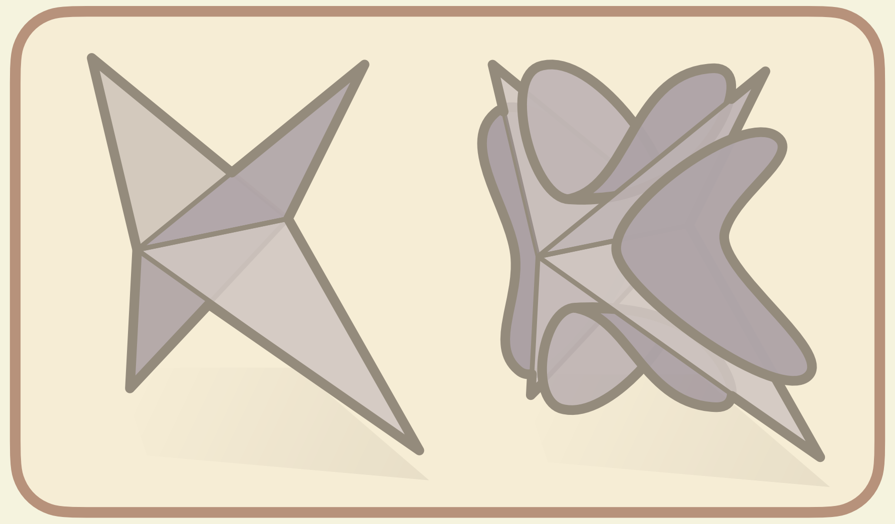


Dec	Jan	Feb	Mar	Apr	May	Jun

Dec. 2023 – Jan. 2024:

- finish ongoing work (Harnack tracing); submit to Siggraph

Timeline:



Dec	Jan	Feb	Mar	Apr	May	Jun

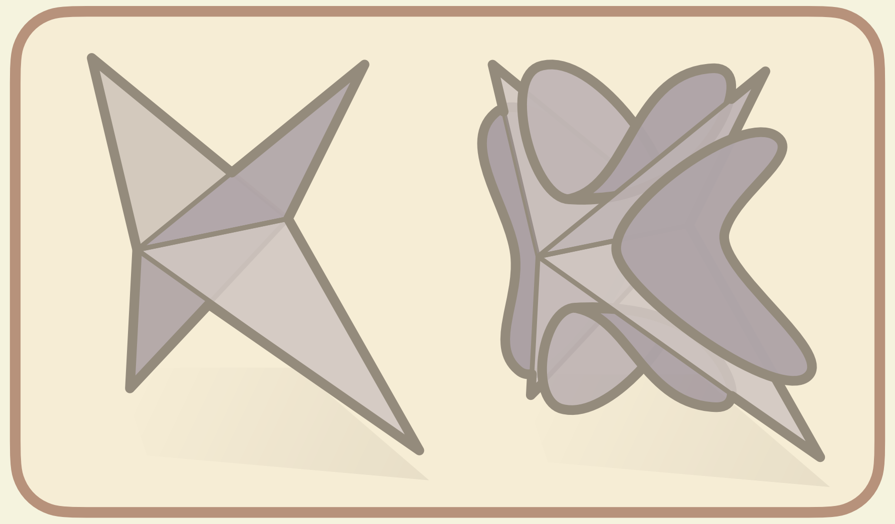
Dec. 2023 – Jan. 2024:

- finish ongoing work (Harnack tracing); submit to Siggraph

Jan. – Apr. 2024:

- nonmanifold intrinsic simplification
- finish up an unrelated project on "circular arc triangulations"

Timeline:



Dec	Jan	Feb	Mar	Apr	May	Jun

Dec. 2023 – Jan. 2024:

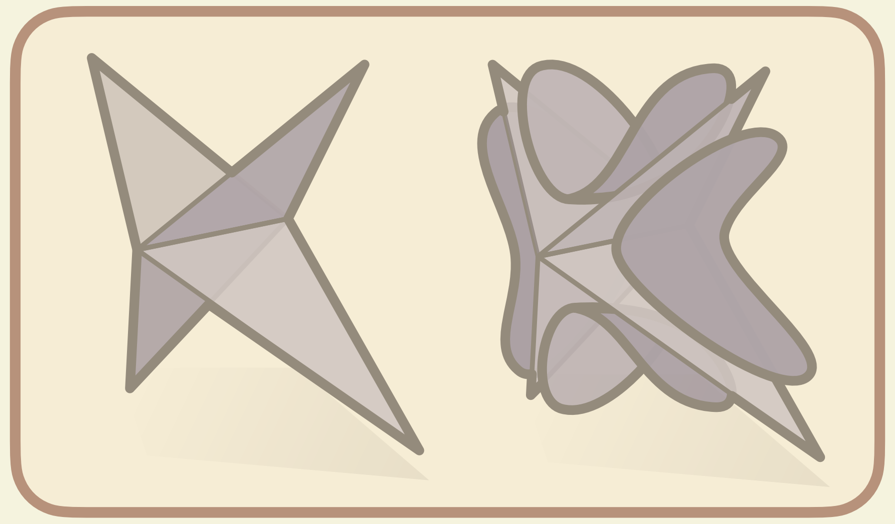
- finish ongoing work (Harnack tracing); submit to Siggraph

Jan. – Apr. 2024:

- nonmanifold intrinsic simplification
- finish up an unrelated project on "circular arc triangulations"

Apr. – Jun. 2024:

- Write thesis



Timeline:

Dec	Jan	Feb	Mar	Apr	May	Jun

Dec. 2023 – Jan. 2024:

- finish ongoing work (Harnack tracing); submit to Siggraph

Jan. – Apr. 2024:

- nonmanifold intrinsic simplification
- finish up an unrelated project on “circular arc triangulations”

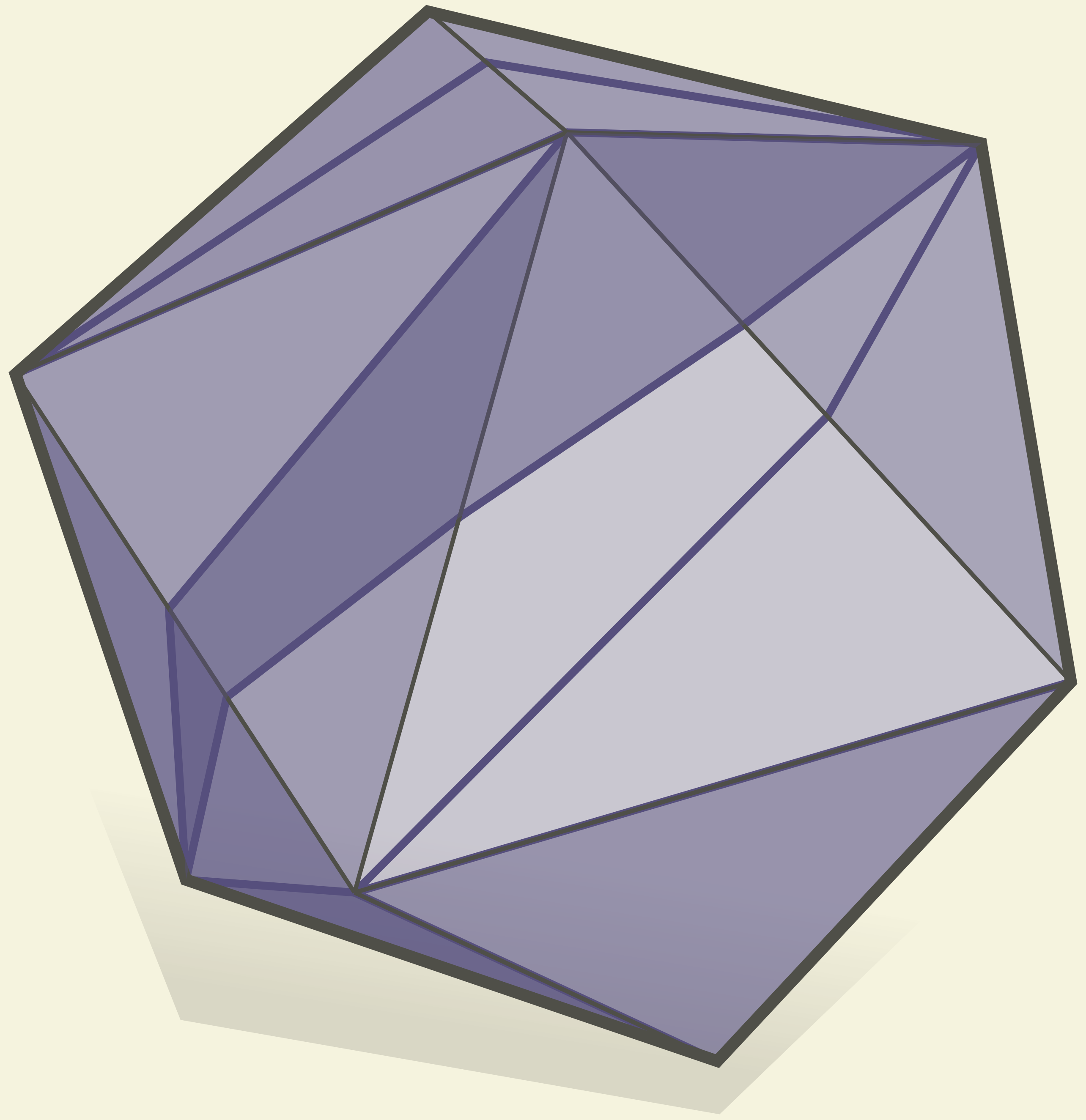
Apr. – Jun. 2024:

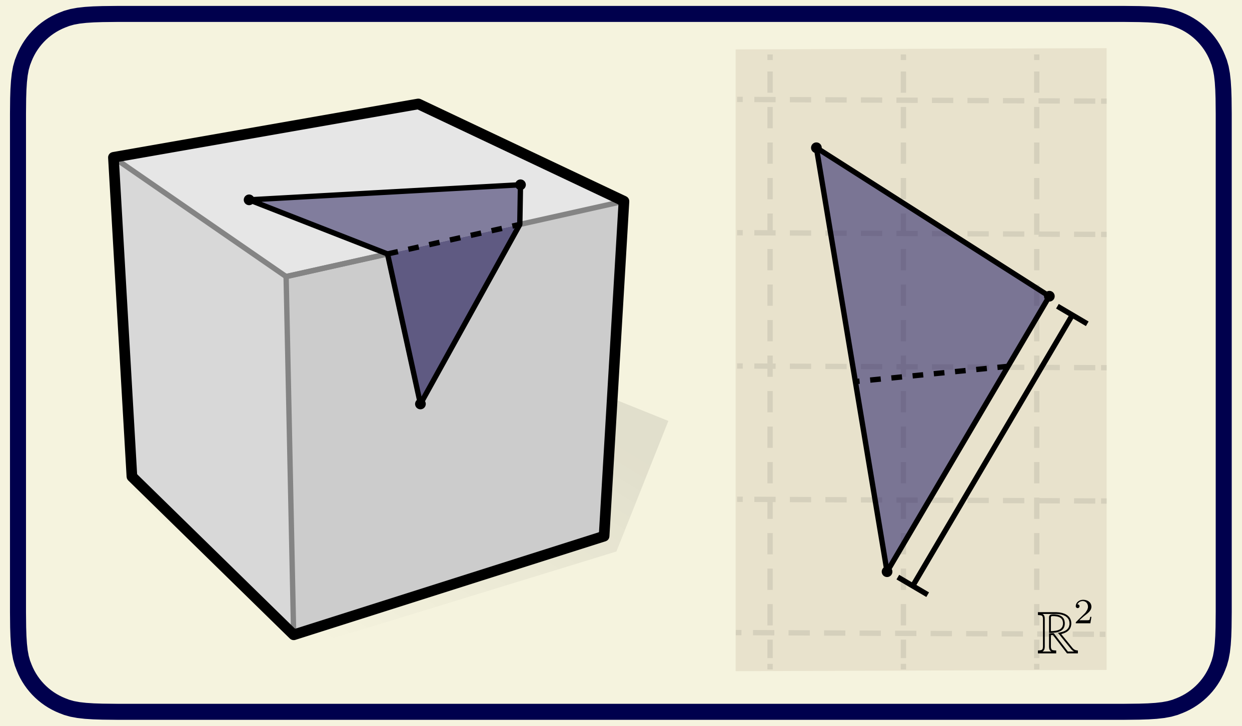
- Write thesis

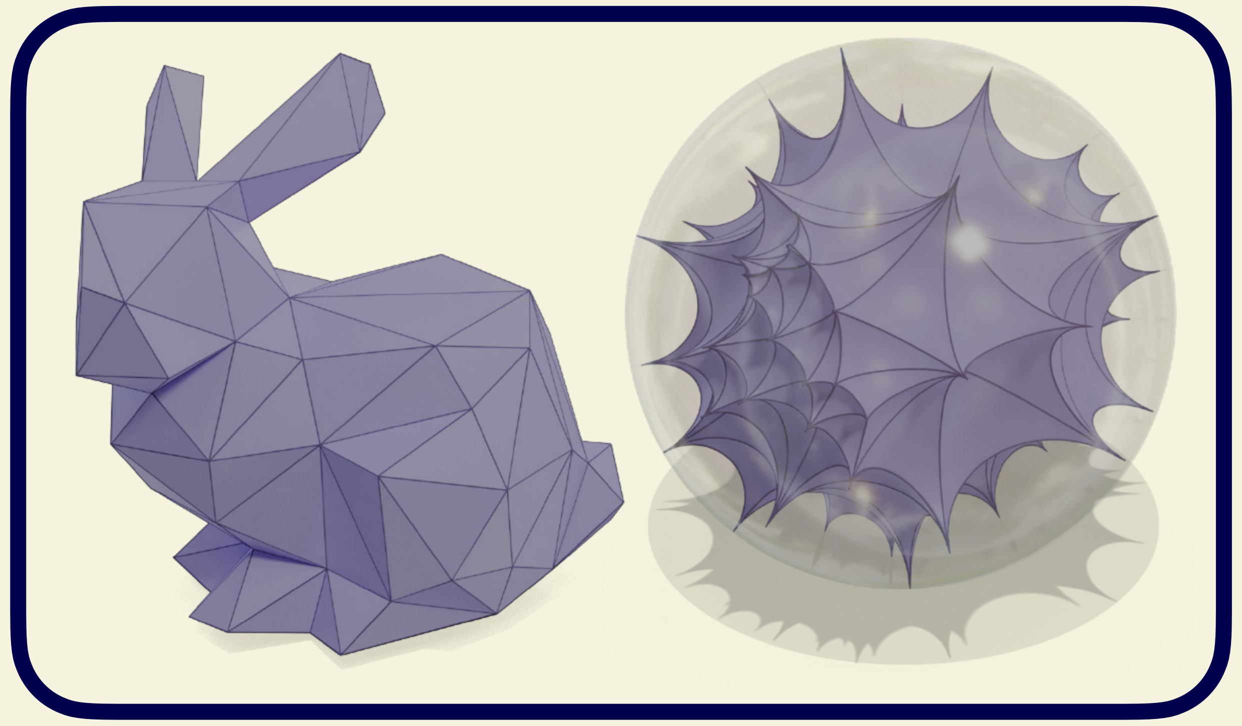
Jun. 2024:

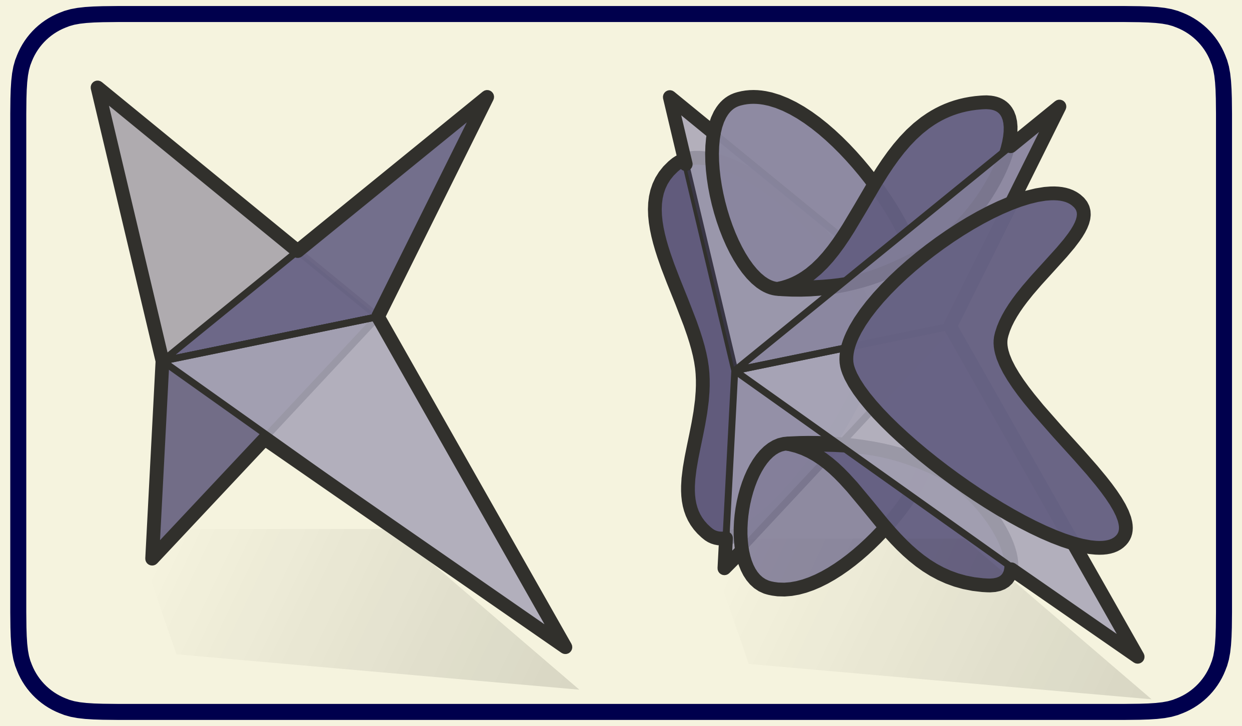
- Defend thesis

Thanks for listening









Supplemental Slides

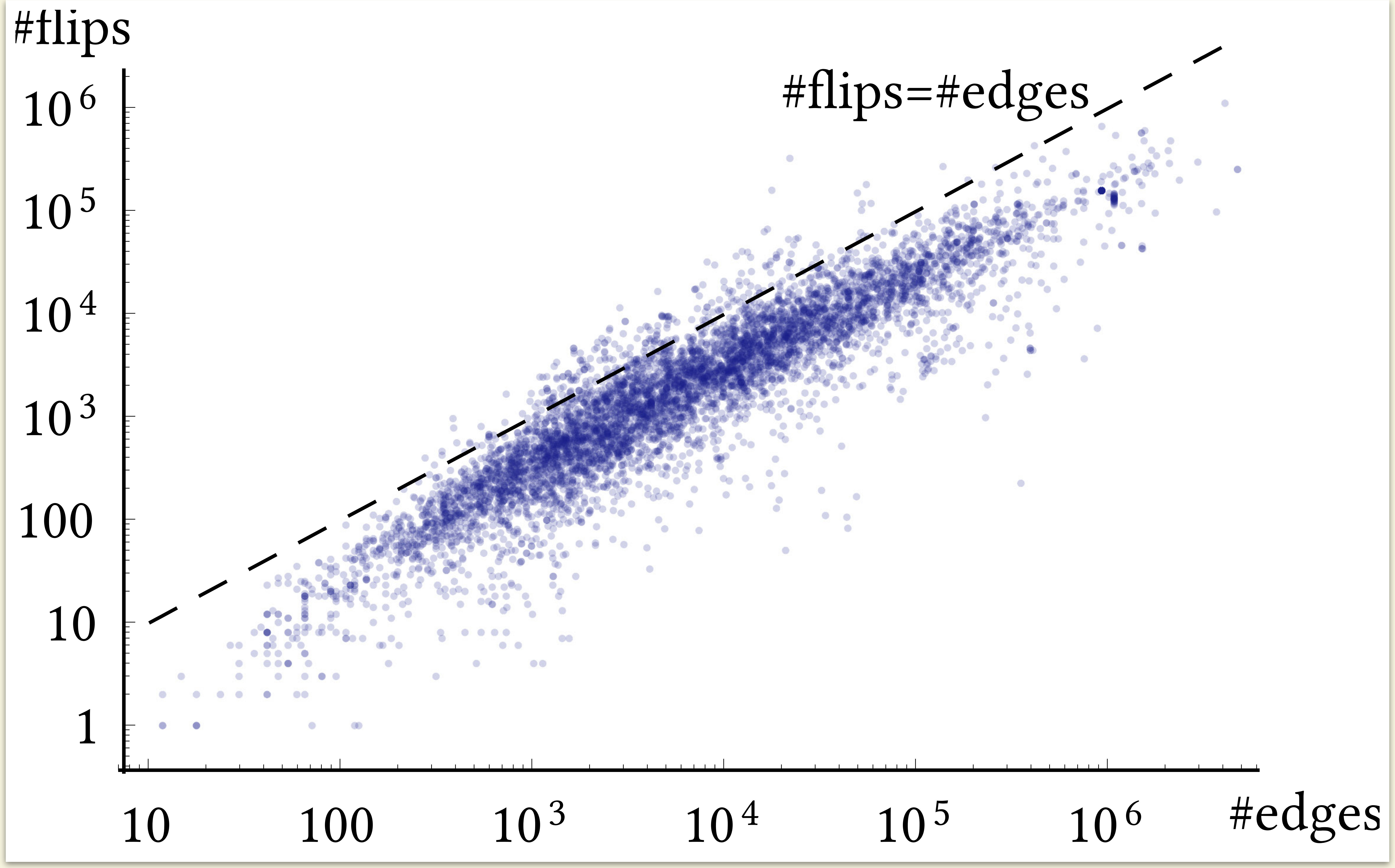
Bad basis functions

Input basis function

Intrinsic basis function

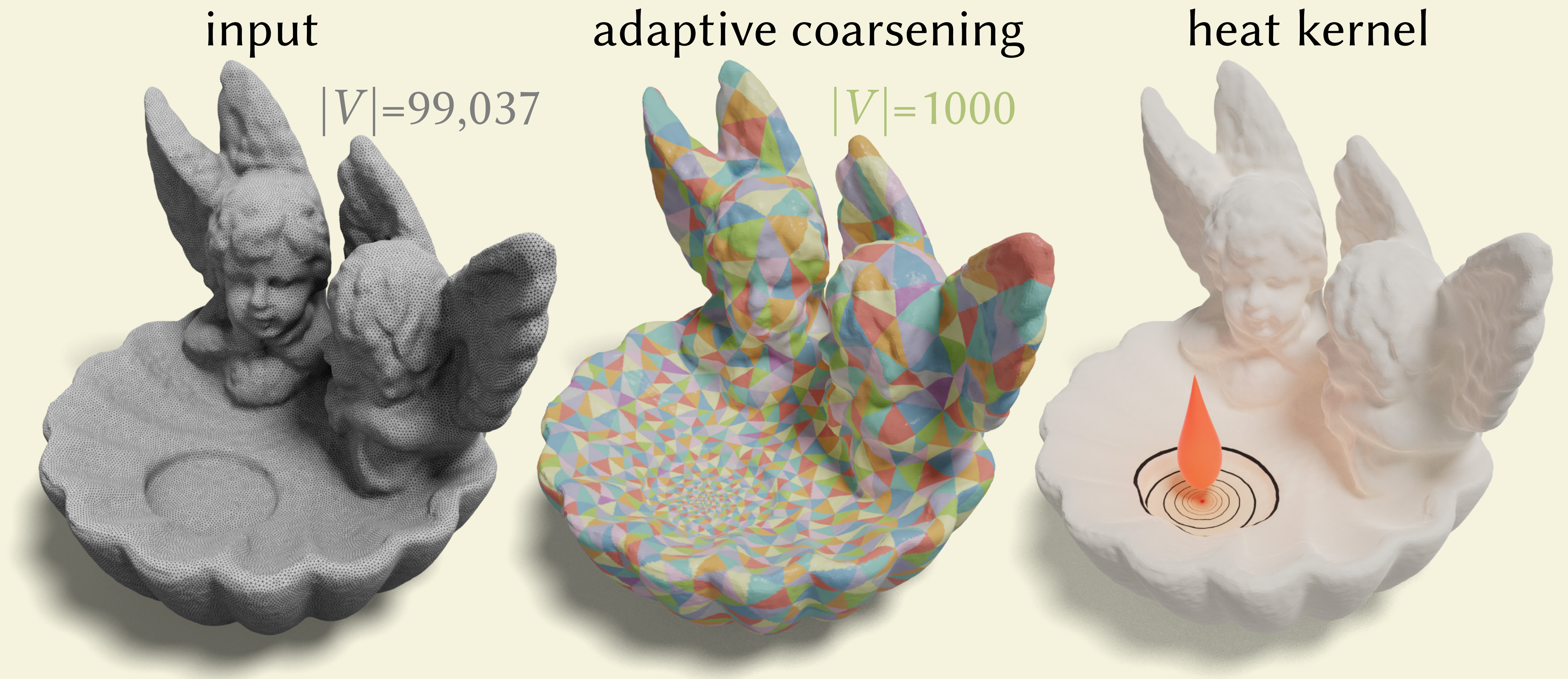
[Sharp, Soliman & Crane 2019]

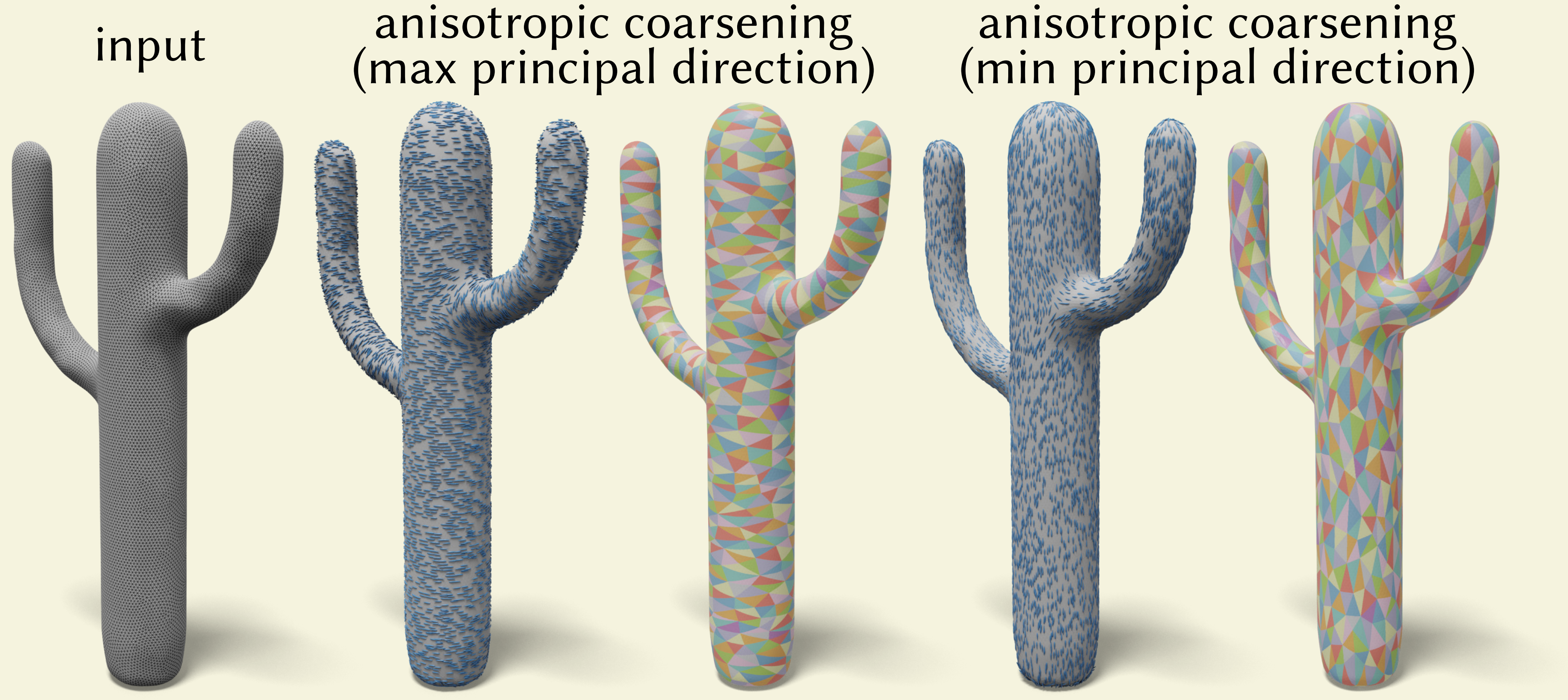
Delaunay flip complexity



[Sharp, Soliman & Crane 2019]

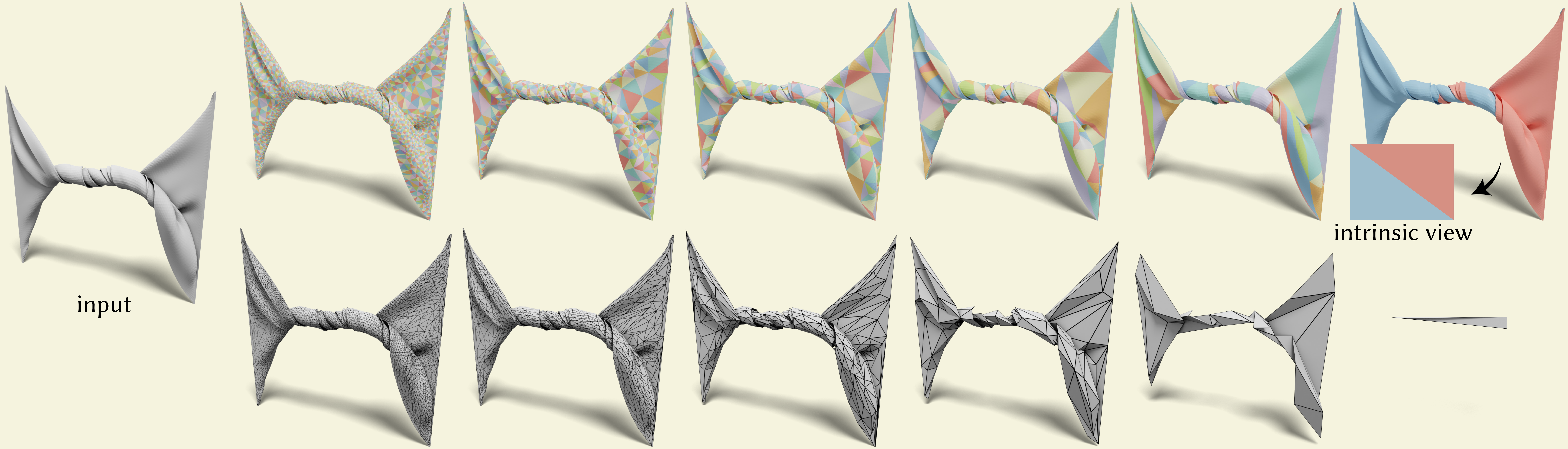
Adaptive simplification





Near-developable surfaces

intrinsic simplification



extrinsic simplification