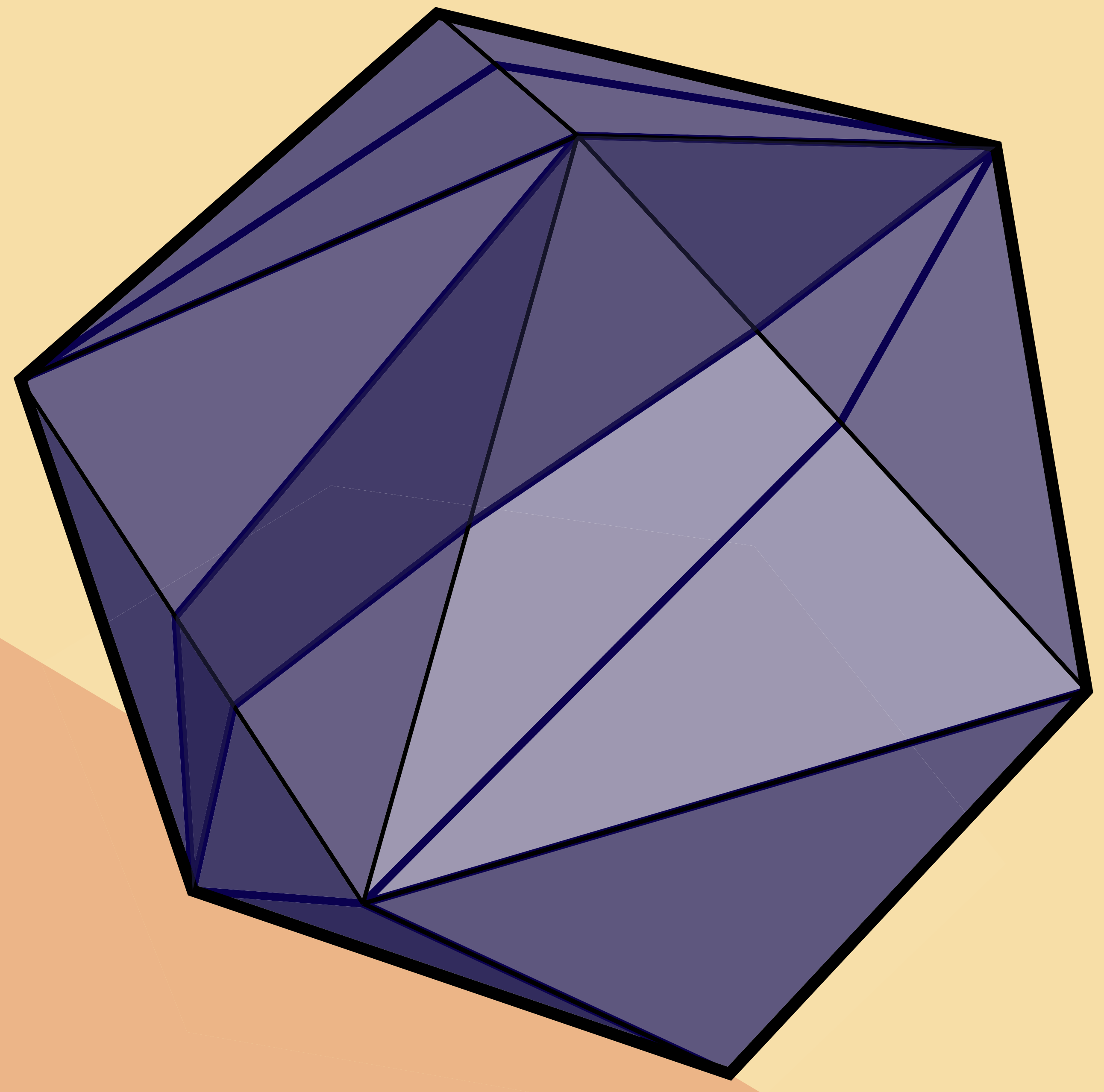
Evolving Intrinsic Triangulations

Mark Gillespie, Carnegie Mellon University

Committee:

Keenan Crane (Chair)
James McCann
Ioannis Gkioulekas
Boris Springborn

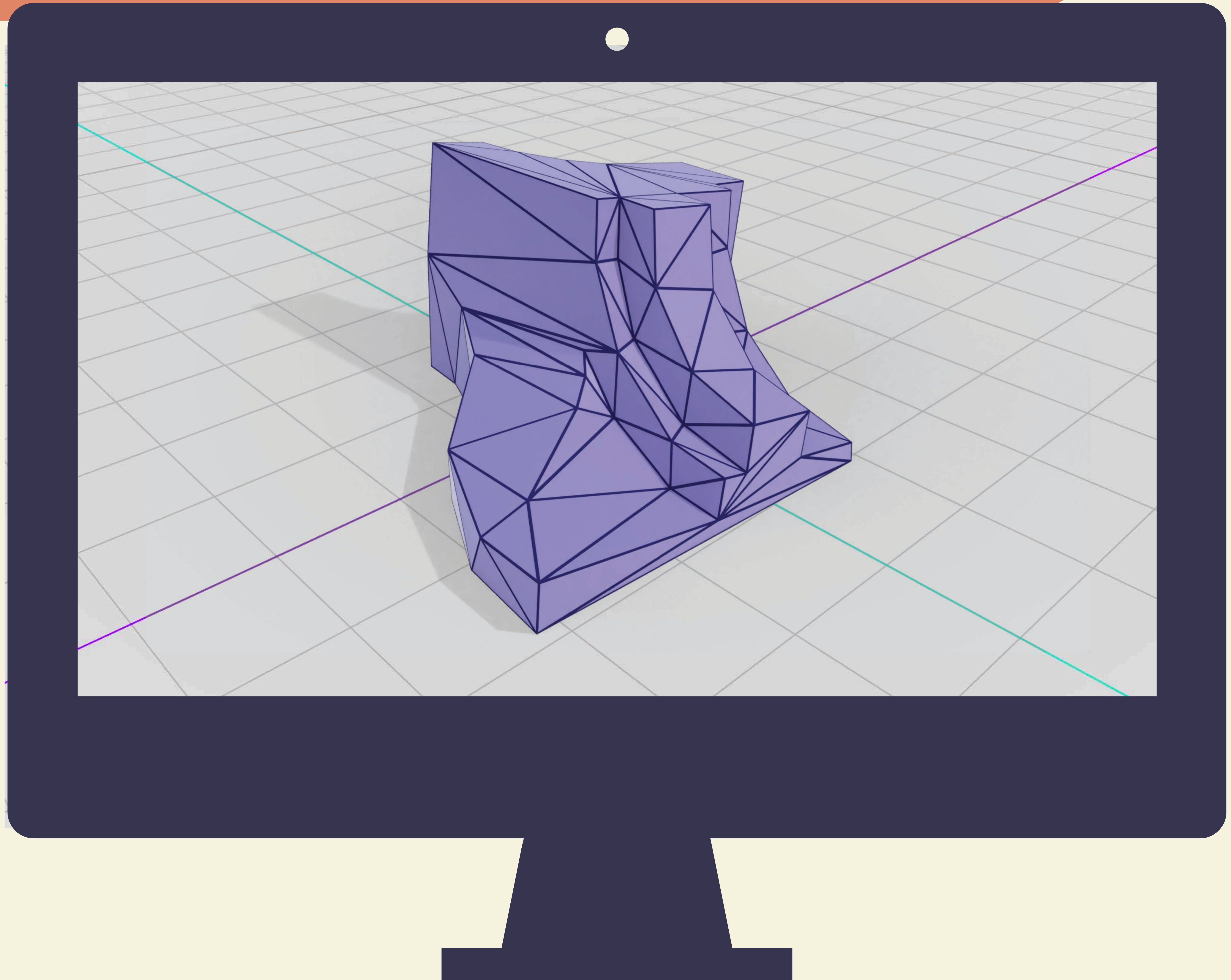


Thesis Defense

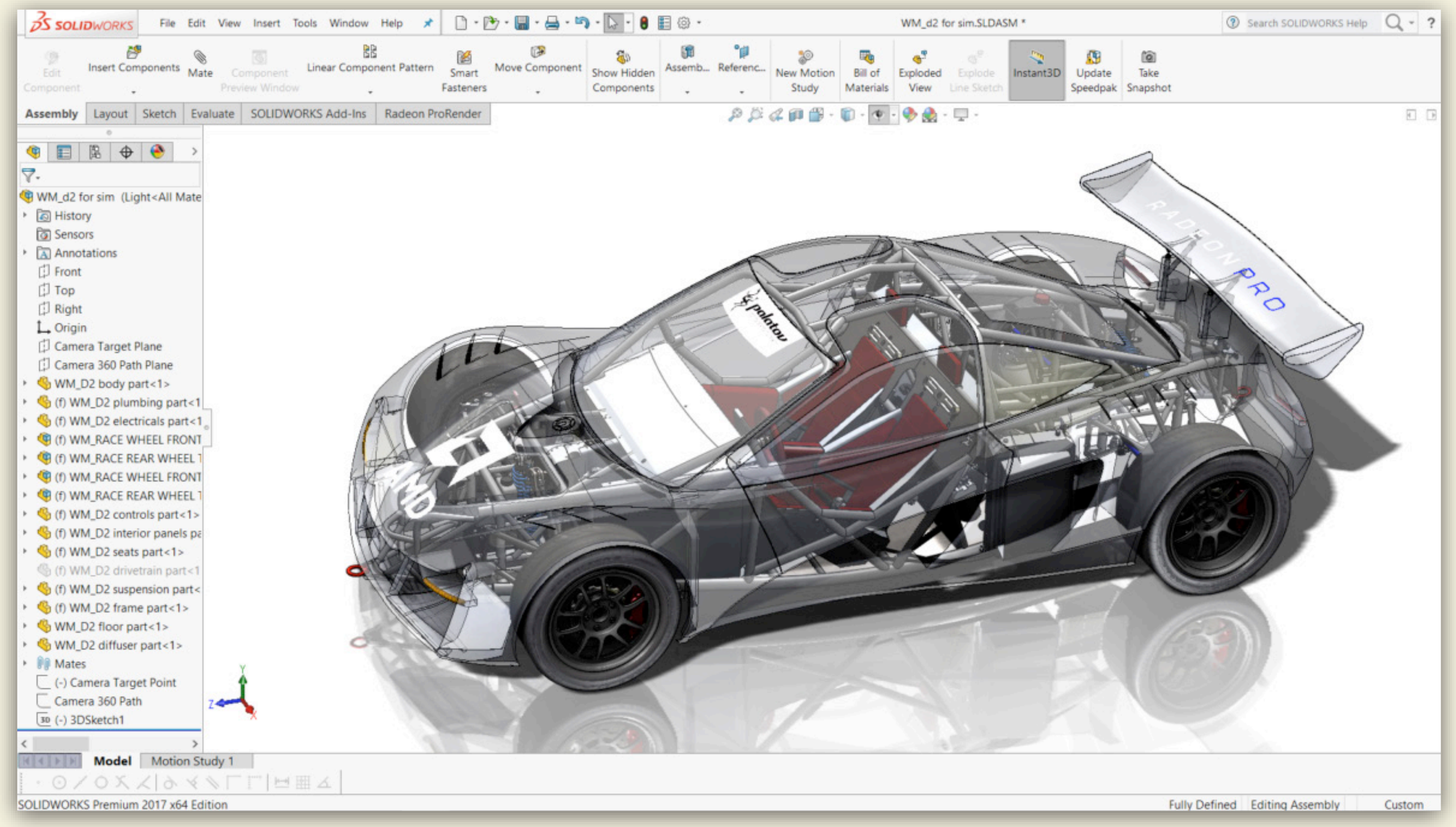
Monday, 22 April 2024

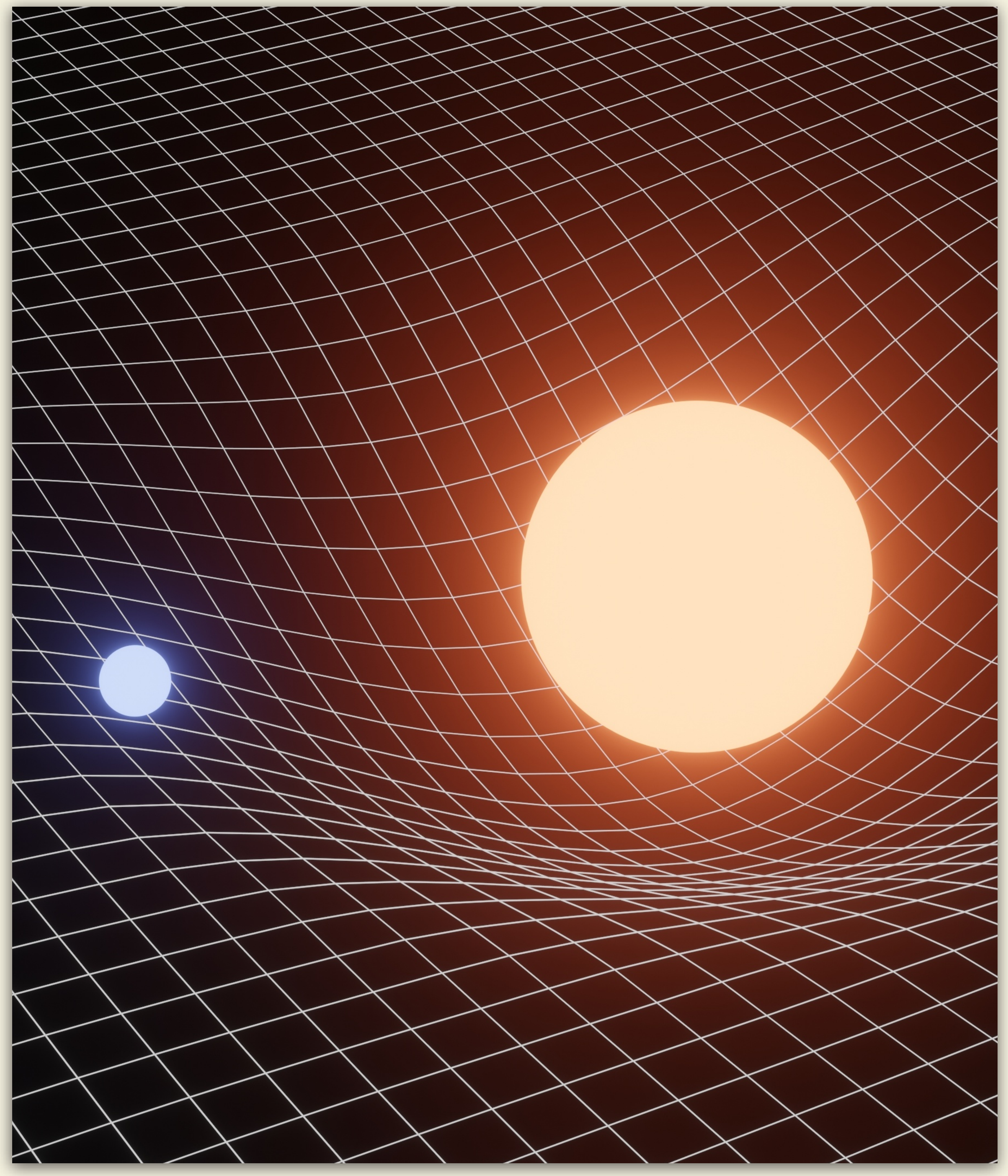
My field: geometry processing

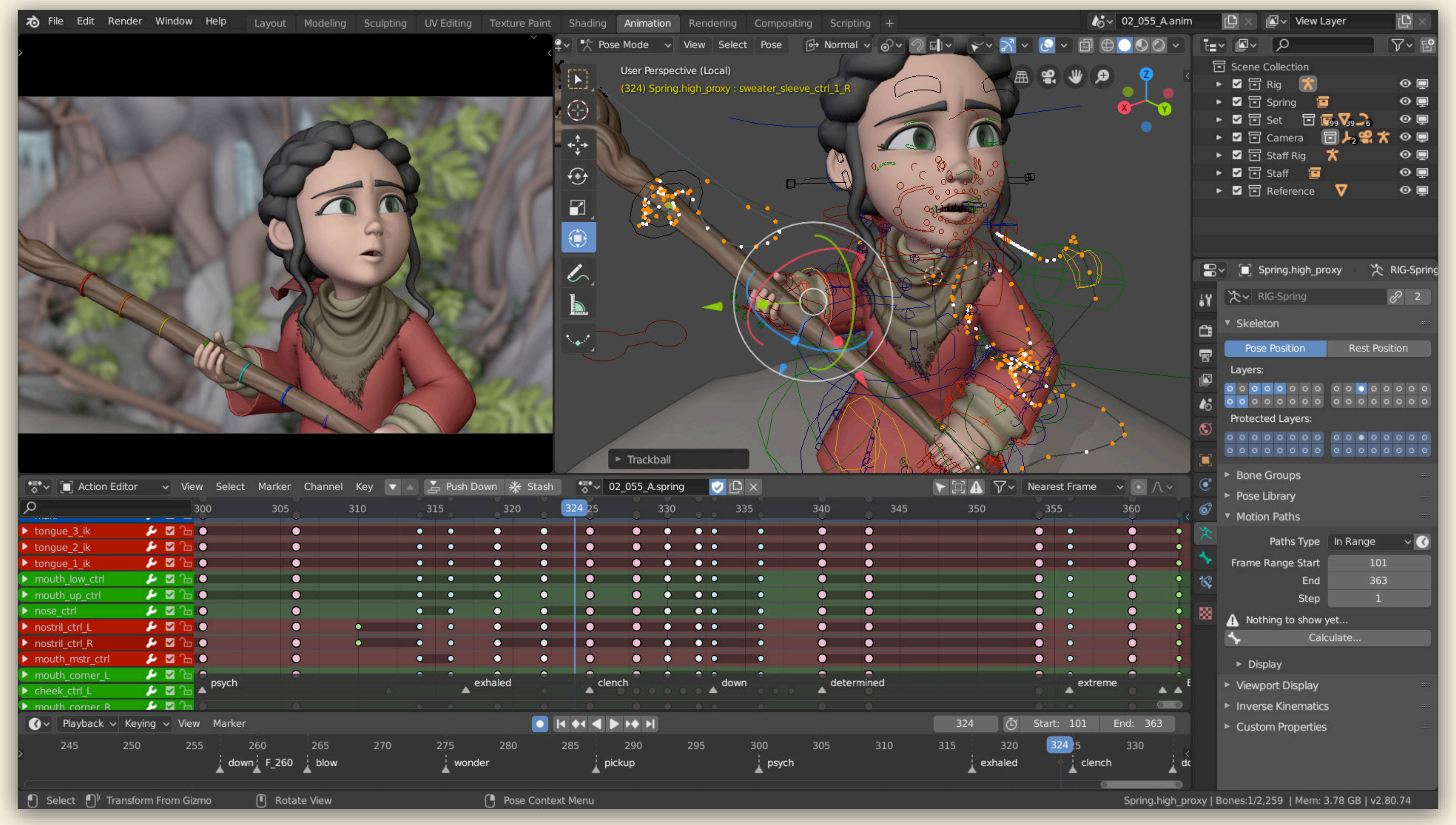
My field: geometry processing

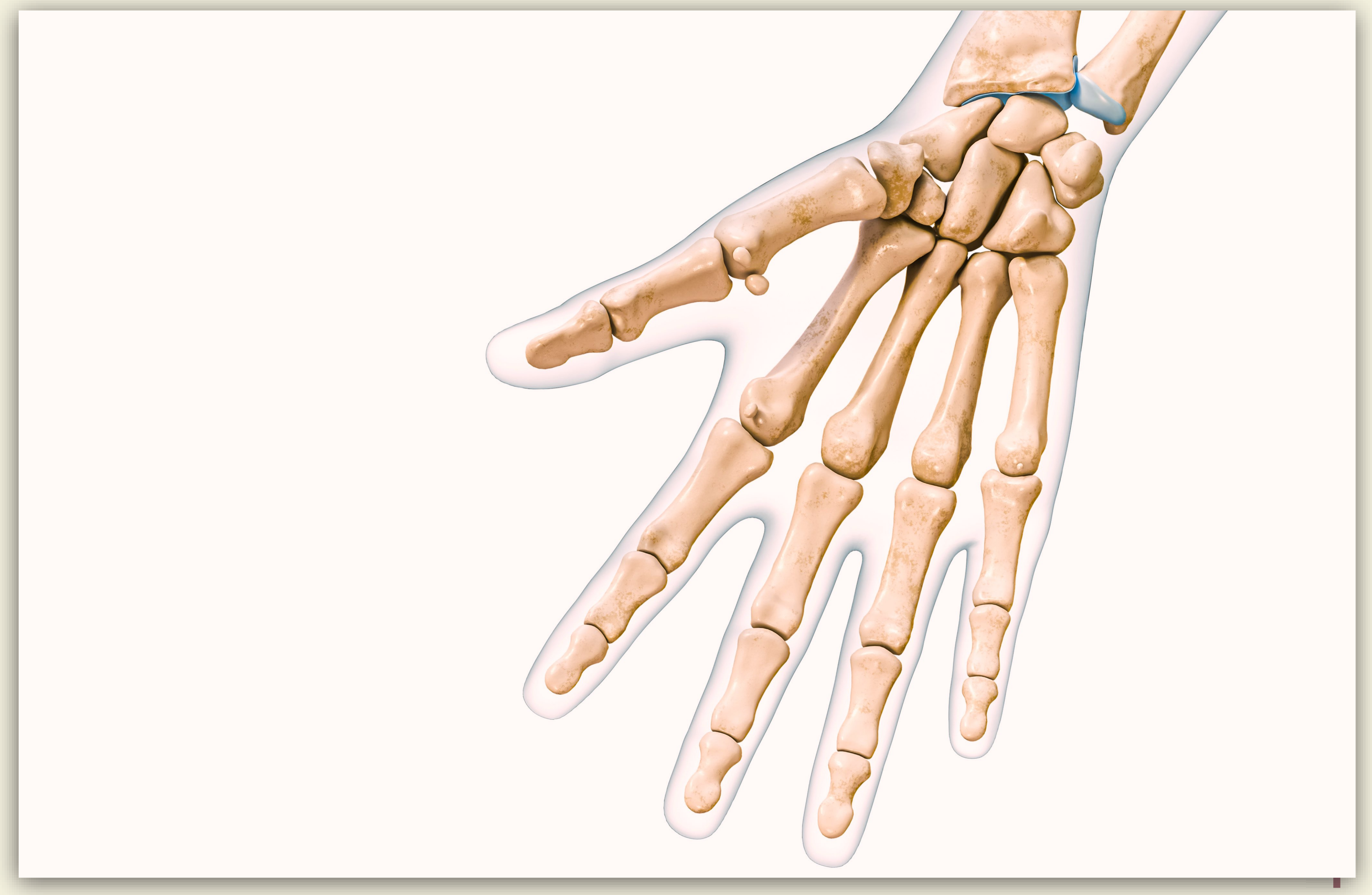


Geometric data is all around us

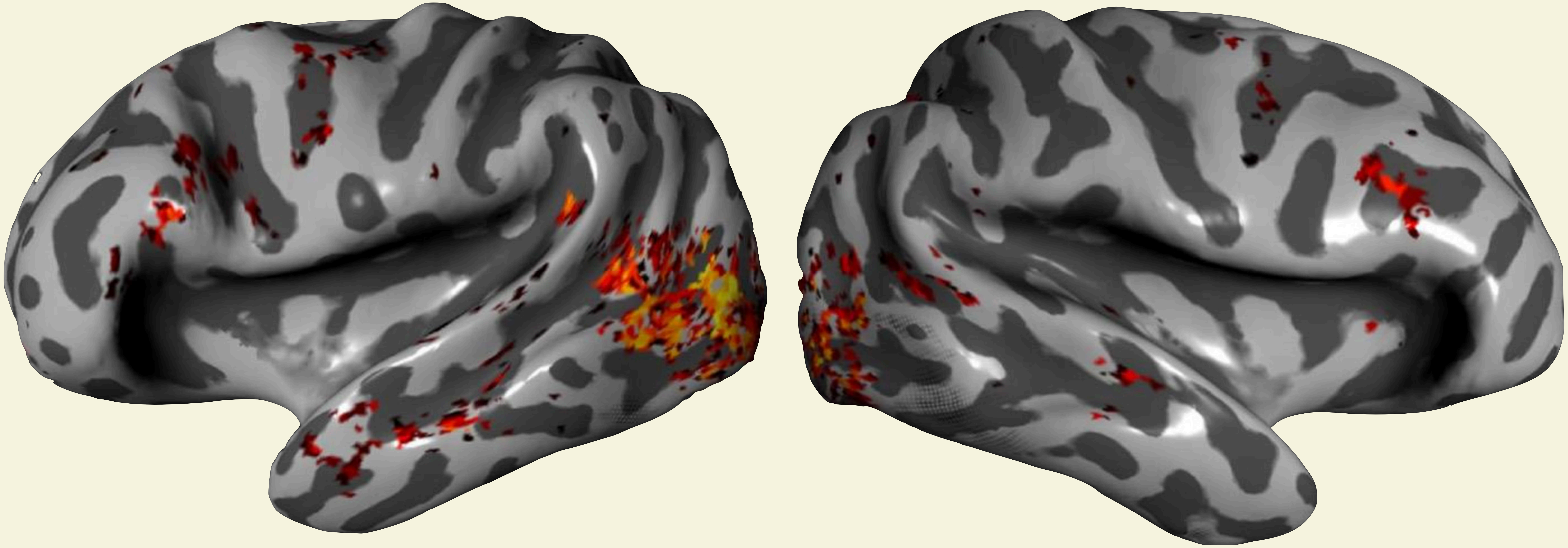






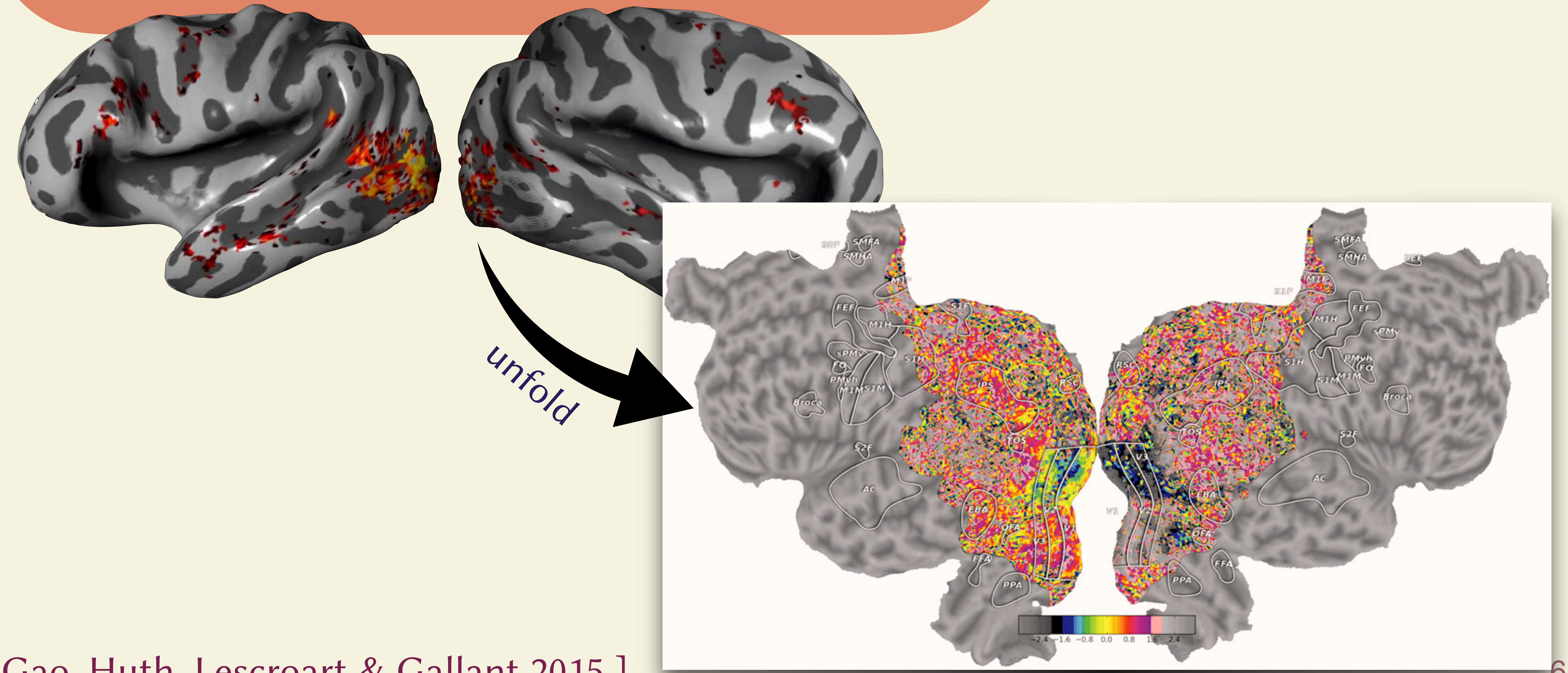


Computation is essential



[Gao, Huth, Lescroart & Gallant 2015]

Computation is essential

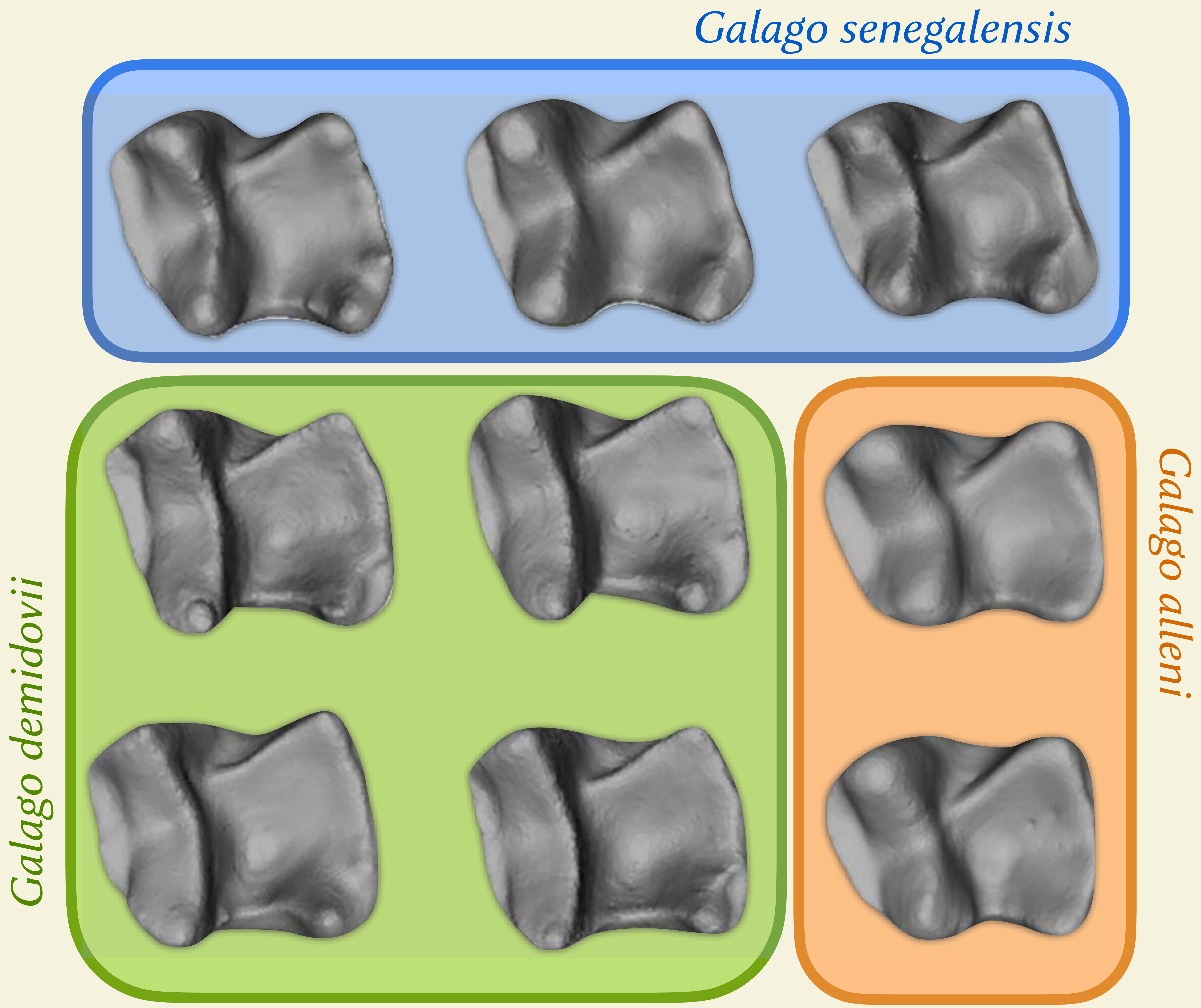


[Gao, Huth, Lescroart & Gallant 2015]

Computation is essential

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

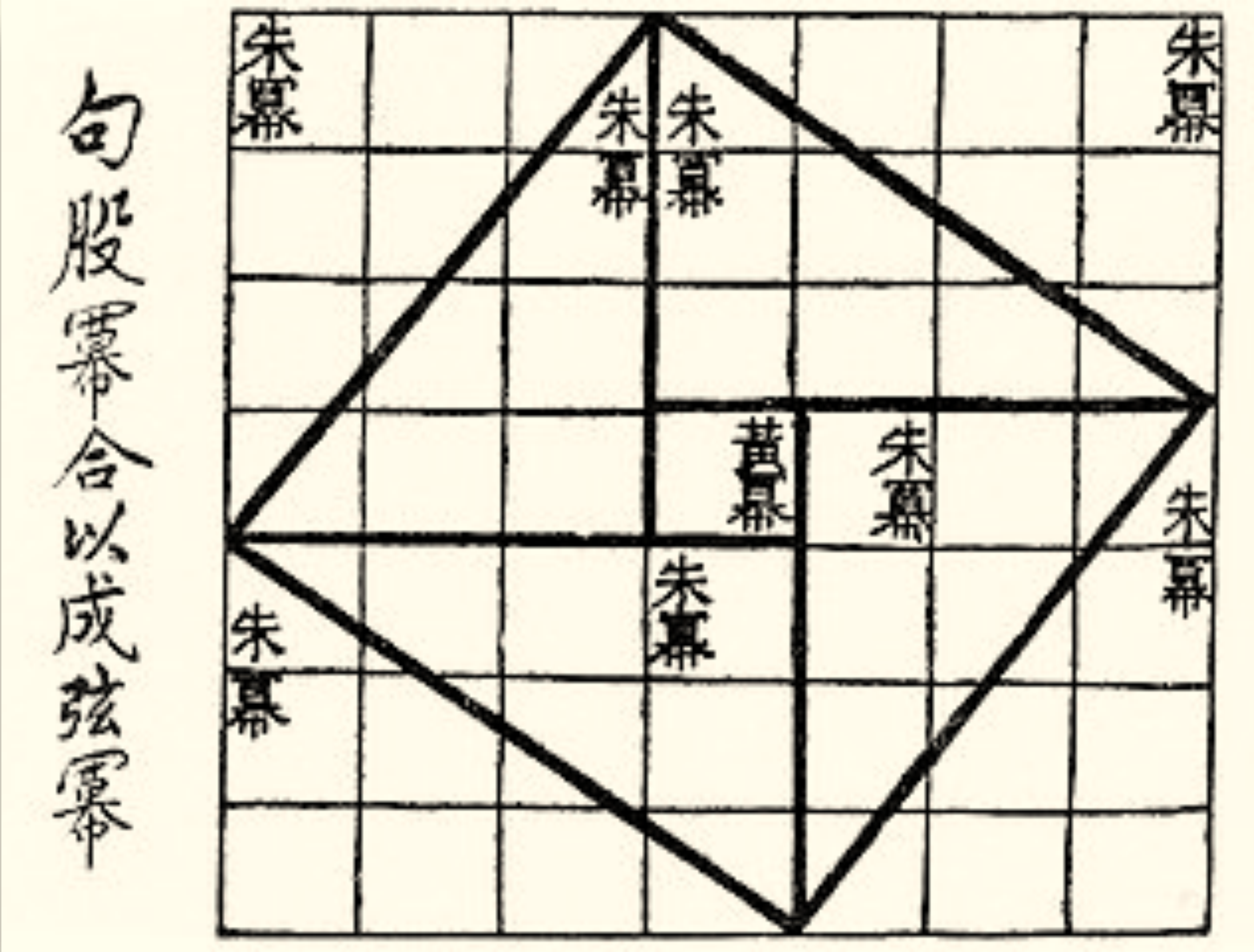
Computation is essential



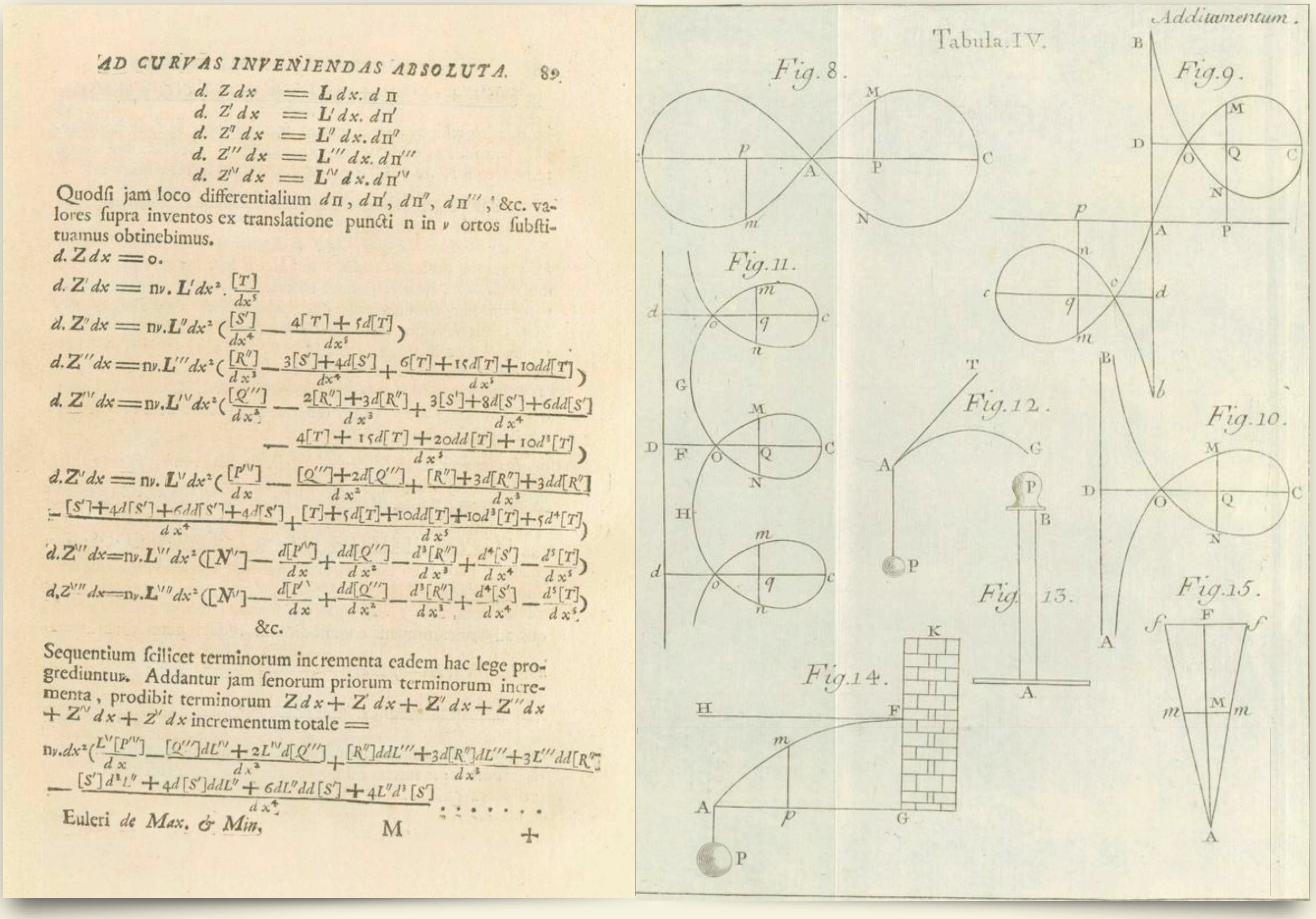
[Boyer, Lipman,
St. Clair, et al. 2011]

Key tool: differential geometry

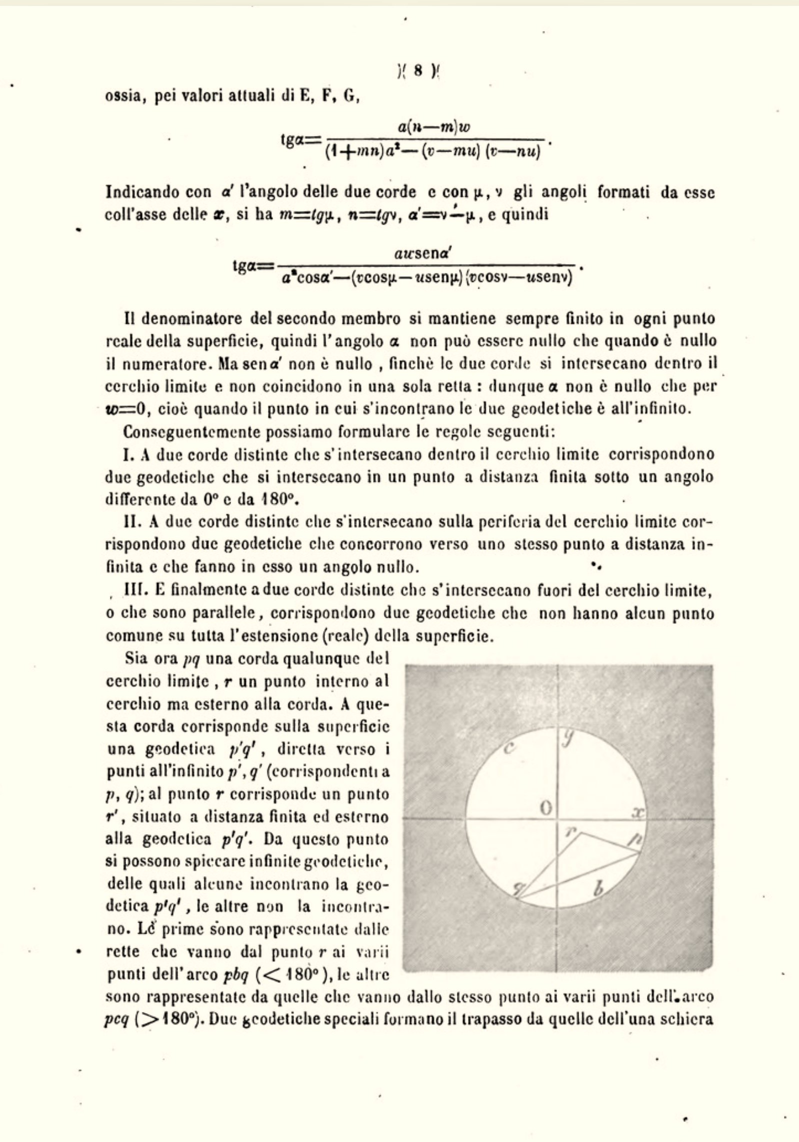
[Babylonian table, c.1800 BCE]



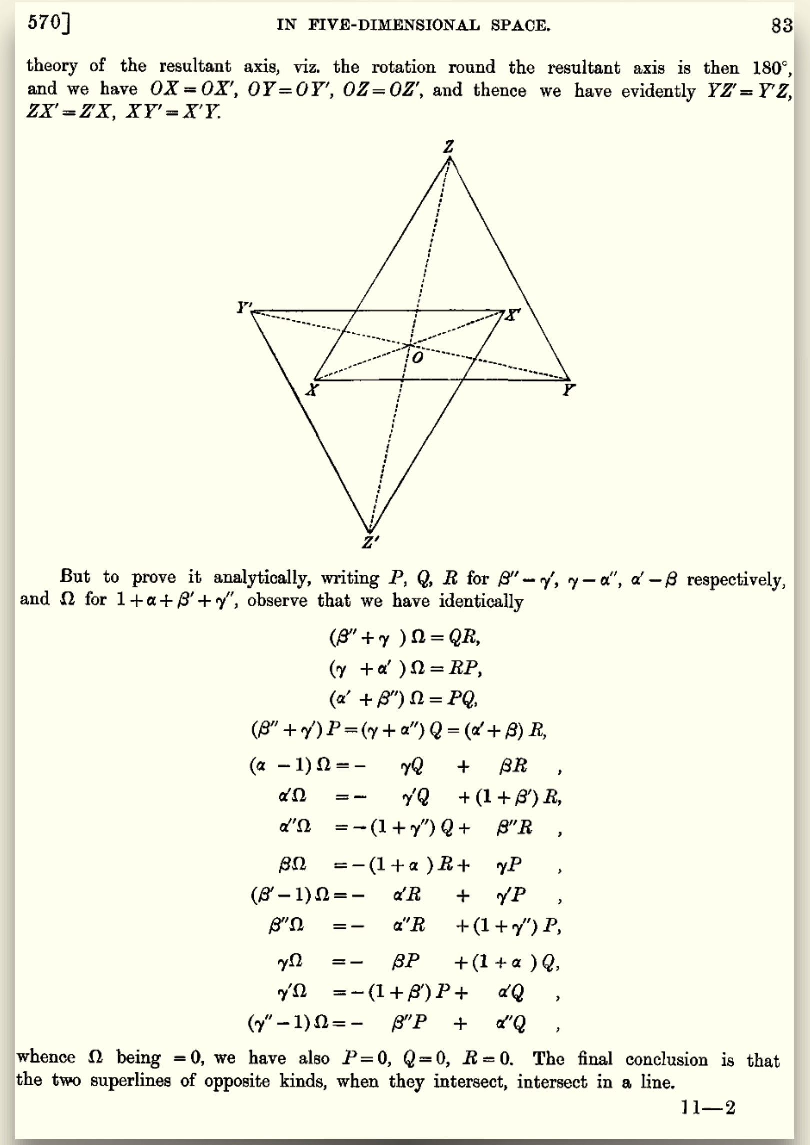
[Zhou, c. 200]



[Euler, 1744]



[Beltrami, 1868]

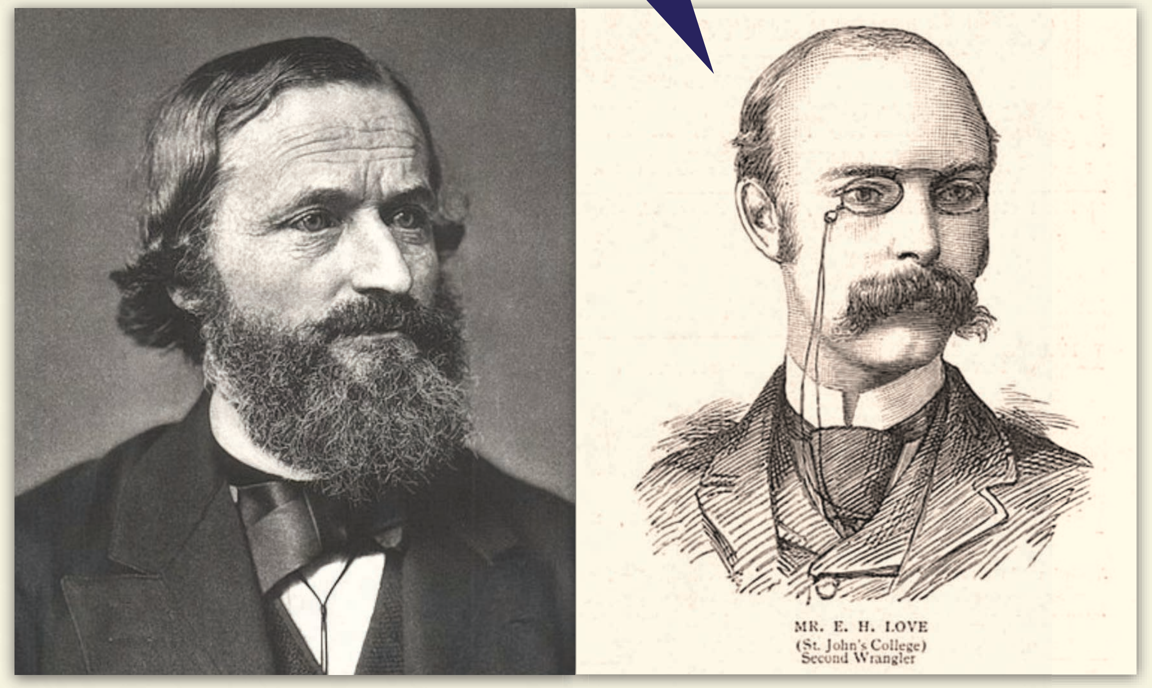


[Cayley, 1896]

Working with 3D shapes is hard

Example: predict sound by finding *vibrational modes*

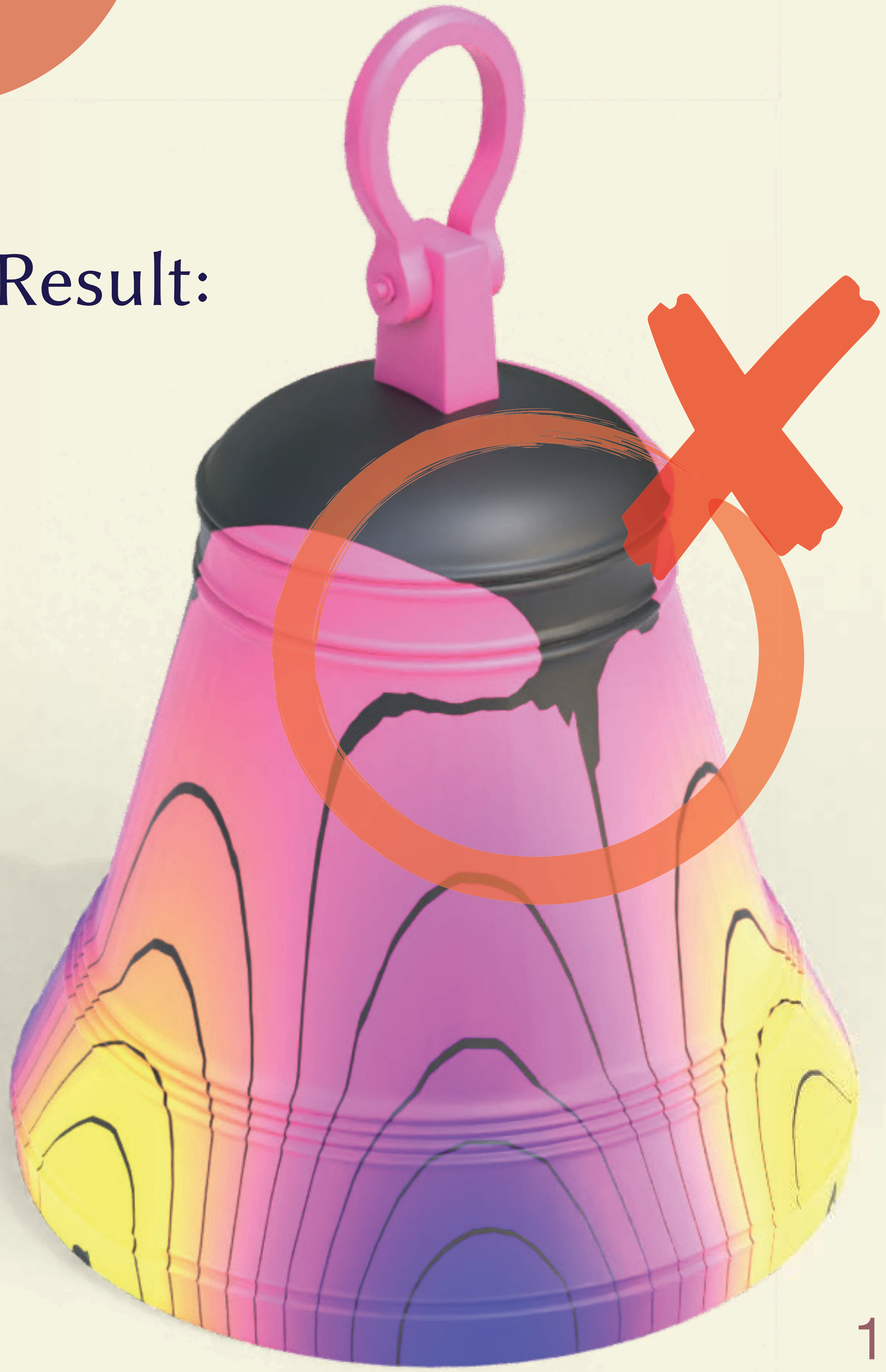
build *bilaplacian* matrix and find eigenvectors



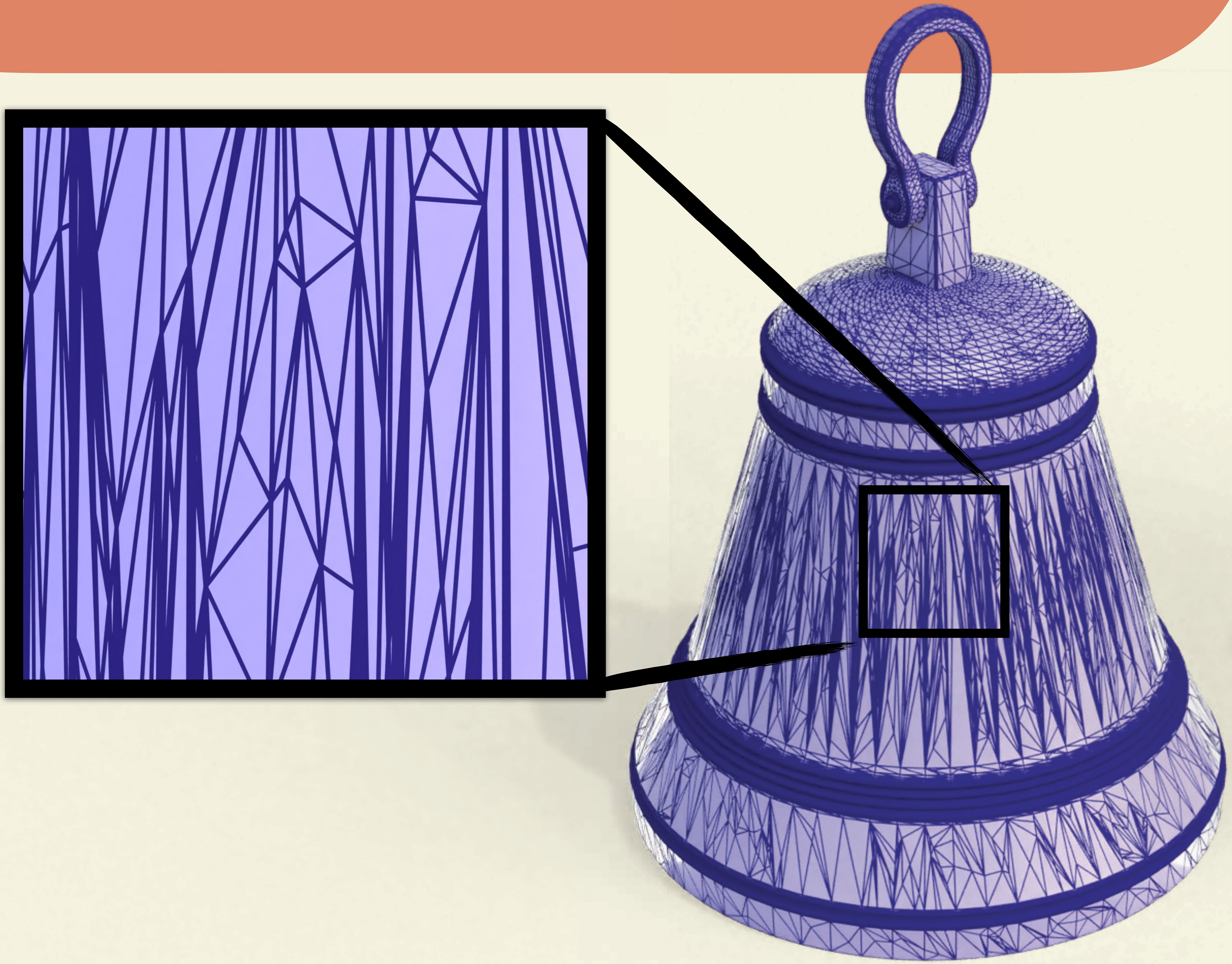
Working with 3D shapes is hard

Example: 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

4x lower error

measured with *a posteriori* error estimates

Problem: triangle quality

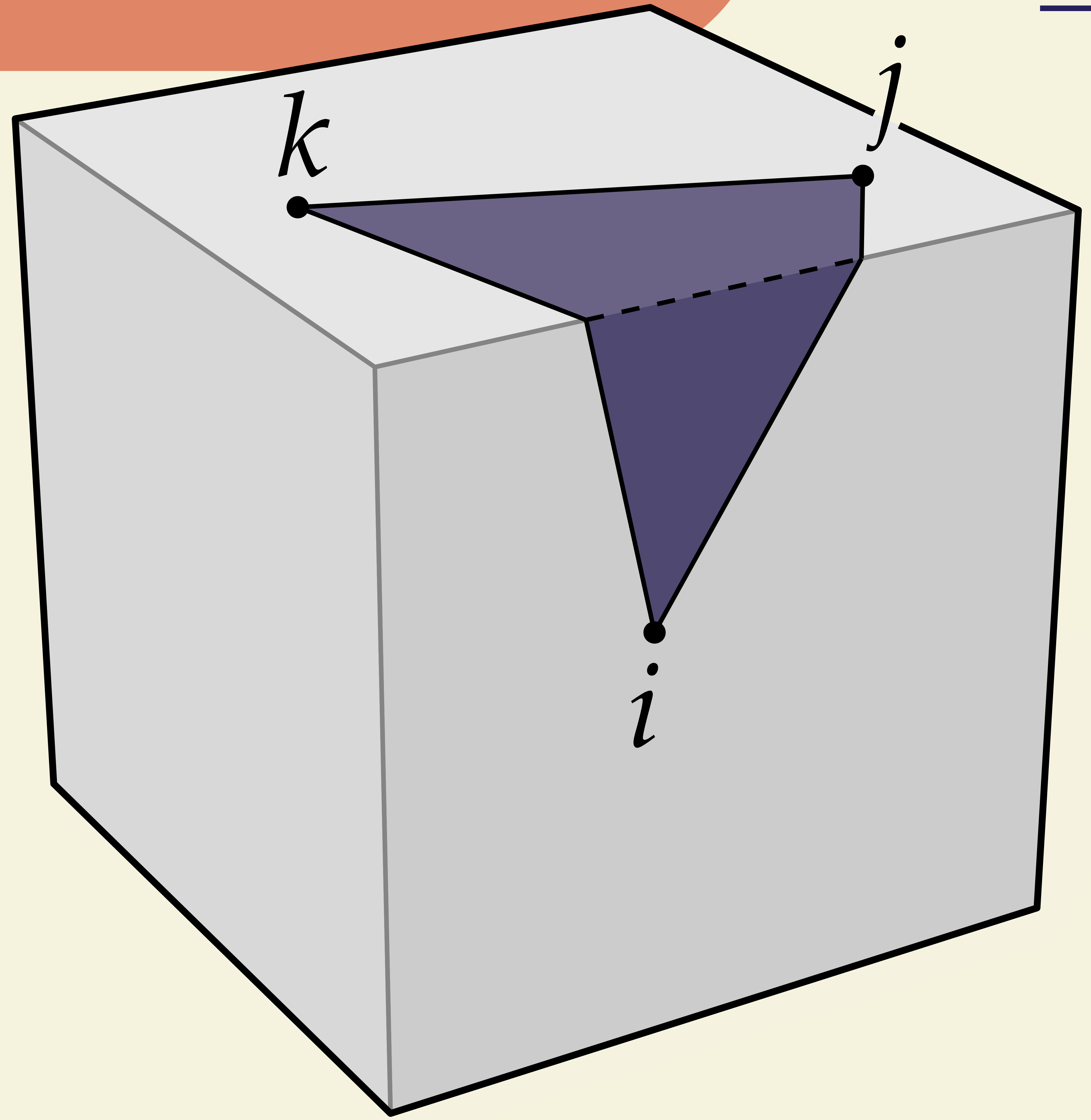
Problem. Our triangles play two roles. They encode both:

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

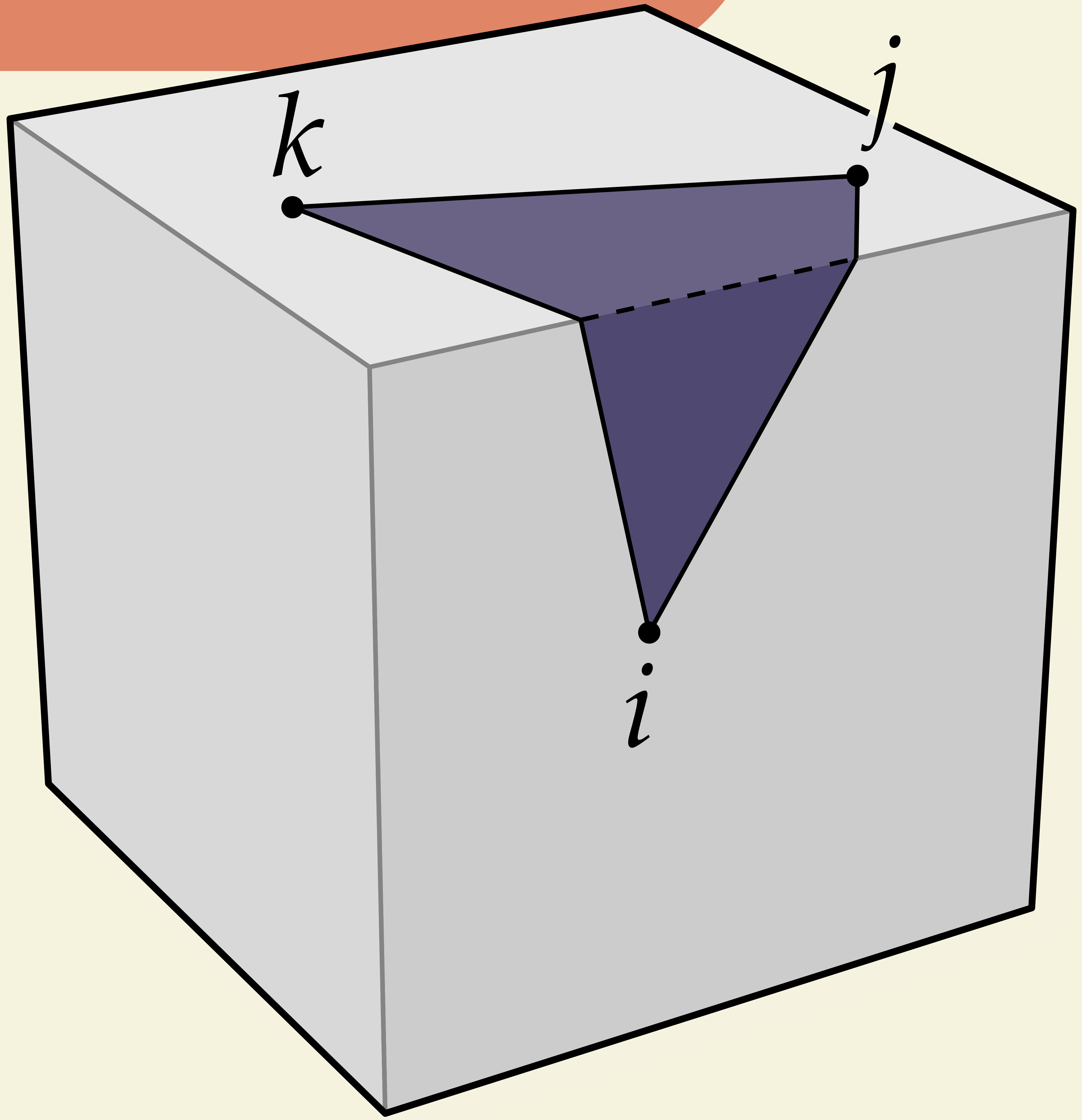
Intrinsic triangles

broadening our idea of what a triangle is

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



Intrinsic triangles



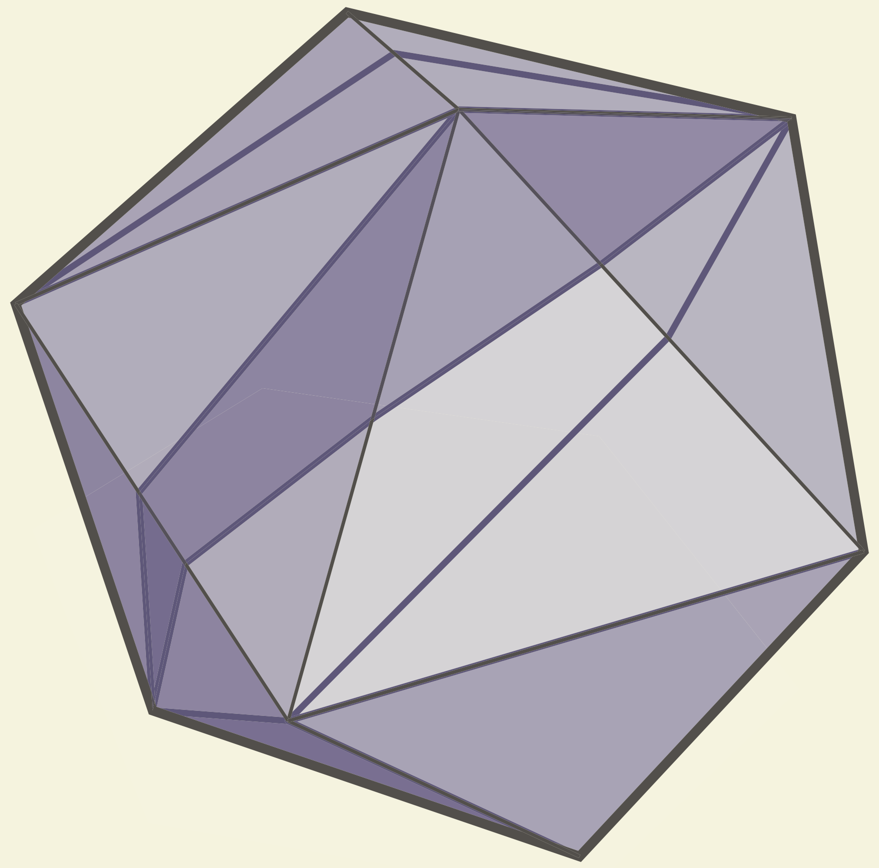
Clean solution to triangle quality issues ***if*** you have a fixed background surface

What if there is no fixed background surface?

What if our geometry changes over time?

Evolving intrinsic triangulations

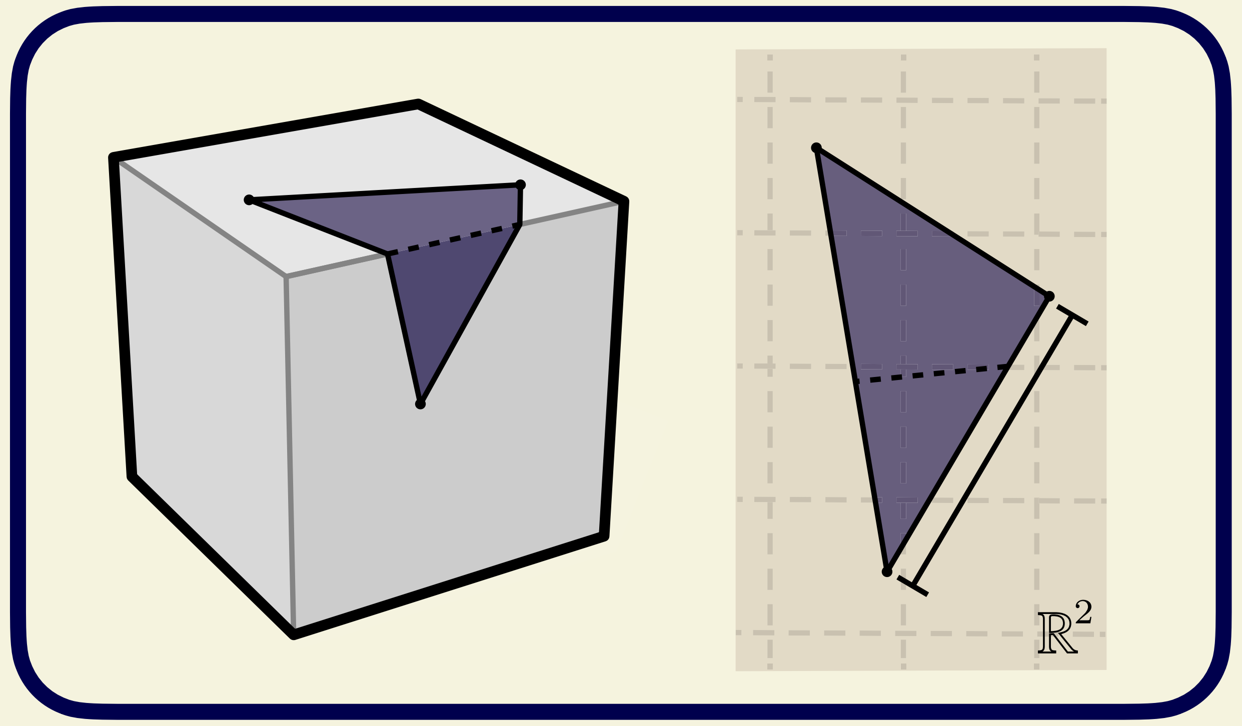
In my thesis,
I present data structures & algorithms
for using intrinsic triangulations
to describe time-evolving surfaces



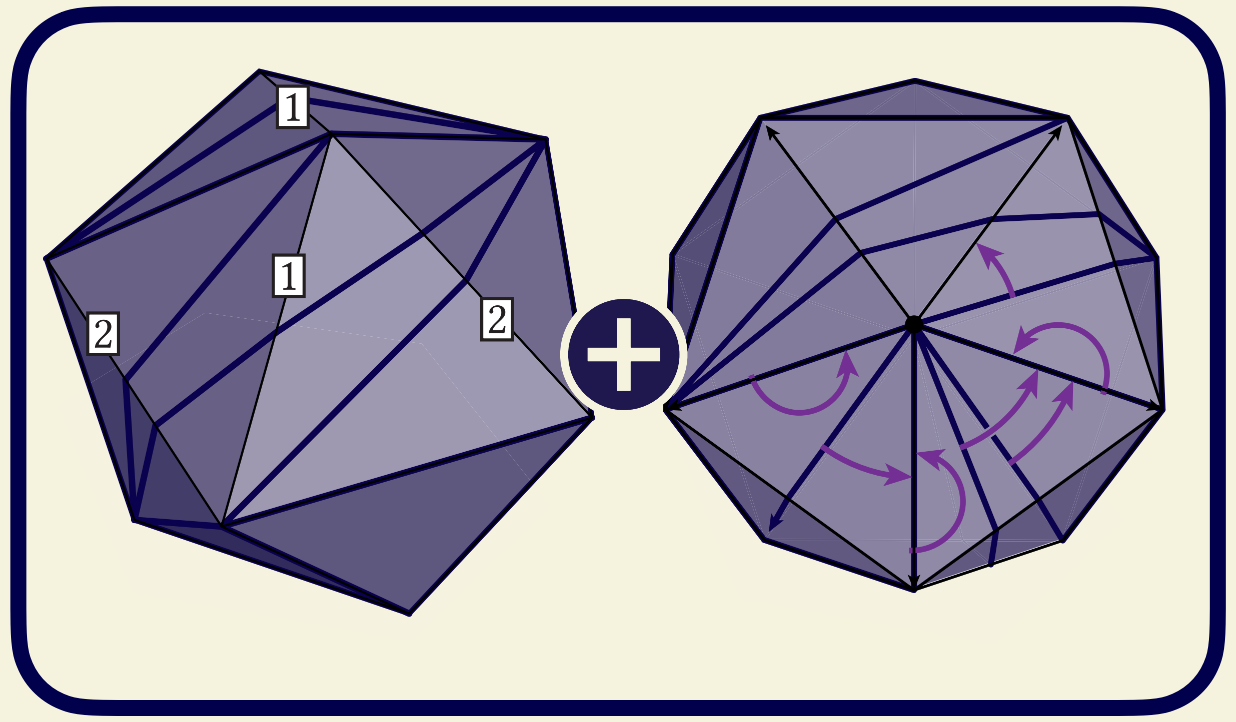
Outline

evolving surfaces

I. Background



II. Data Structures



[**Gillespie**, Sharp, & Crane. 2021. Integer coordinates for intrinsic geometry processing. *ACM TOG*]

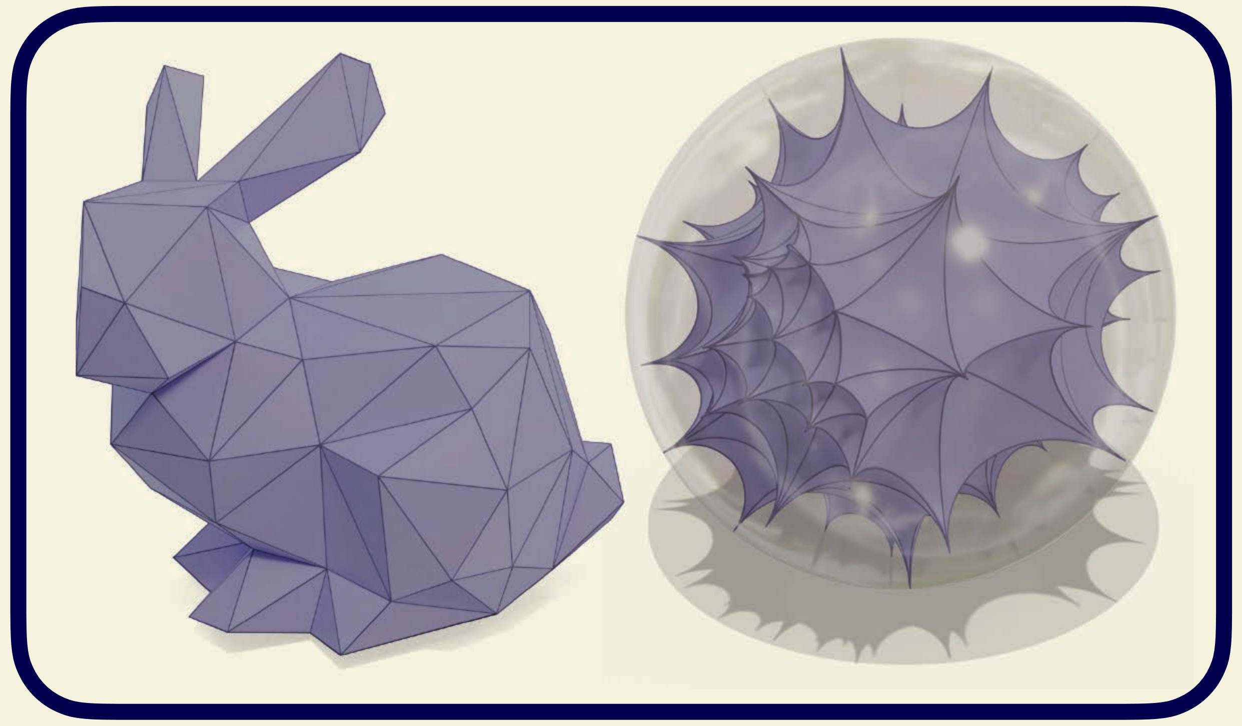
Encode an intrinsic triangulation on a surface

III. Simplification

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

Track intrinsic triangulation while *simplifying* a surface

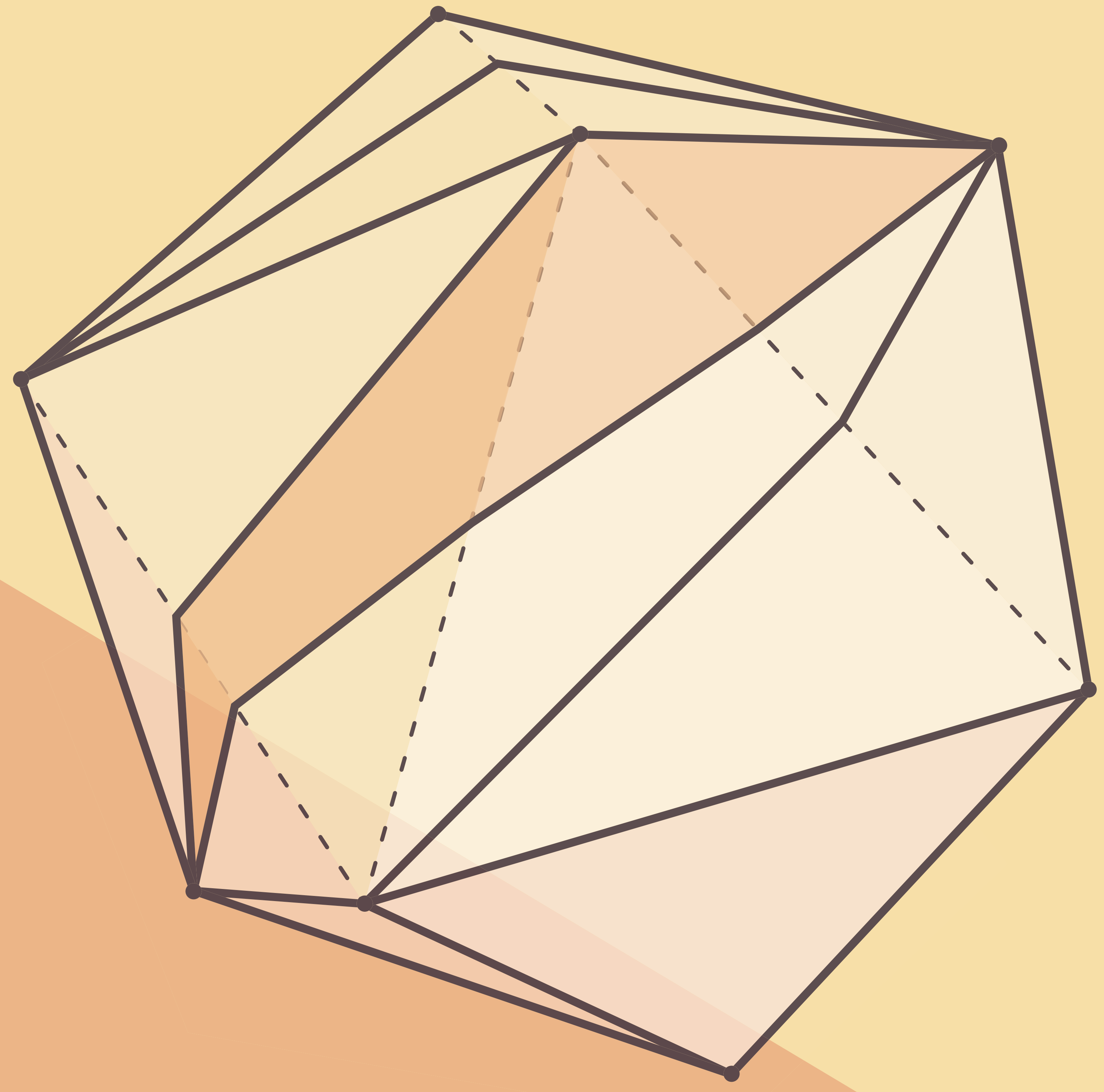
IV. Parameterization



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

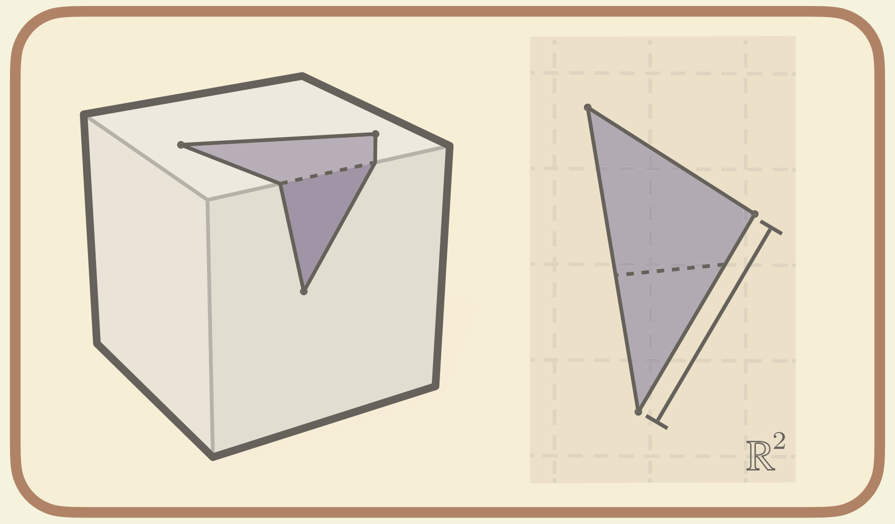
Track intrinsic triangulation while *flattening* a surface

I. Background

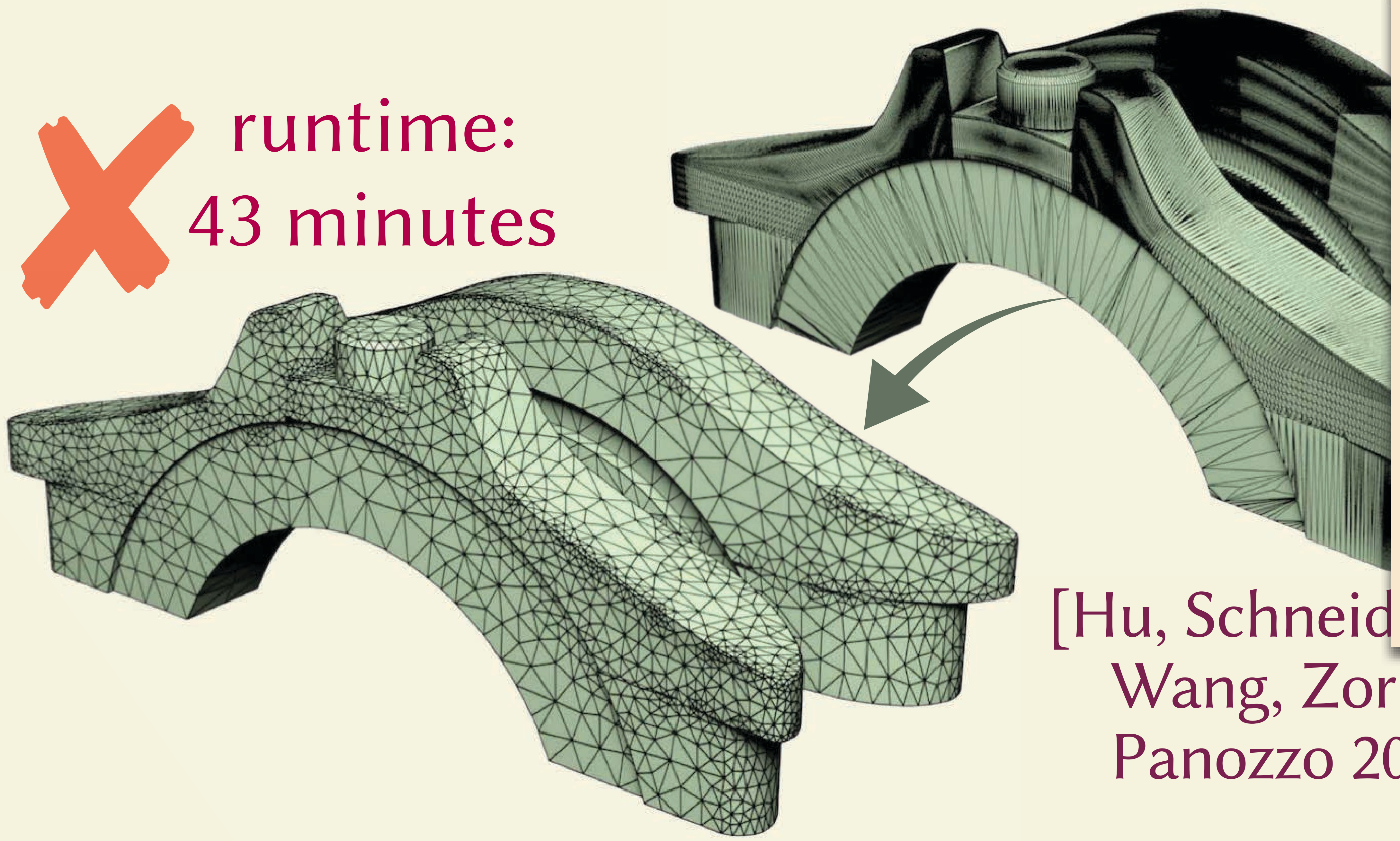


Status quo: remeshing

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



runtime:
47 minutes



runtime:
43 minutes

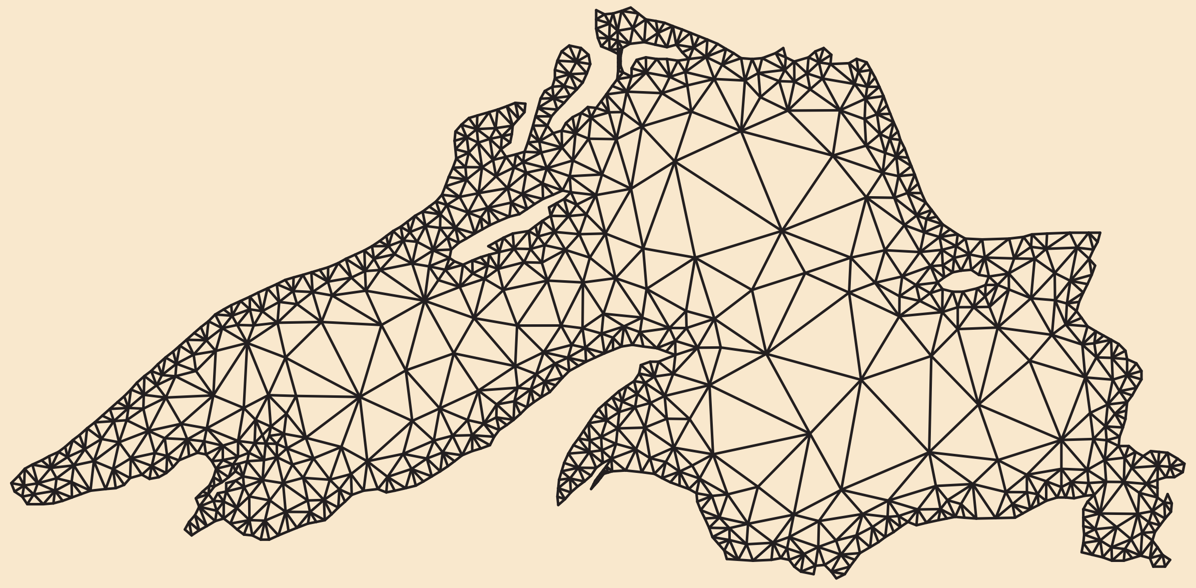
[Hu, Schneid
Wang, Zorin &
Panozzo 2020]

Meshing is much easier in 2D

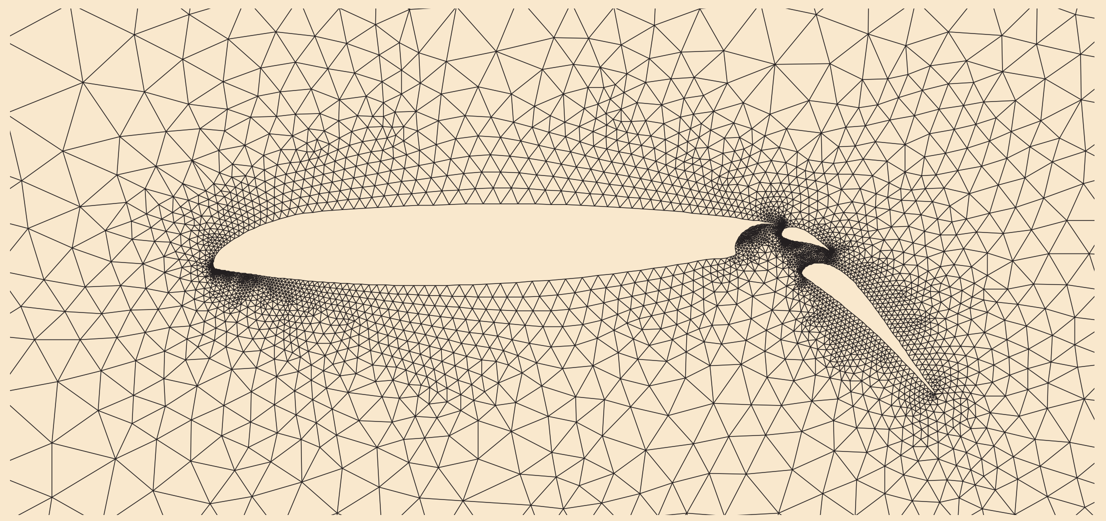
has theoretical guarantees!

Generate high-quality meshes in milliseconds

(using Delaunay triangulations with refinement
[Chew 1993; Shewchuk 1997])



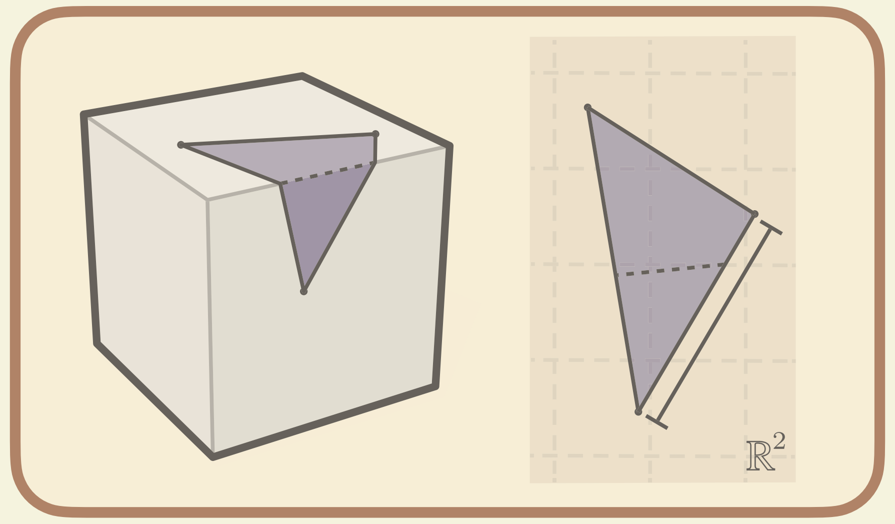
80 milliseconds



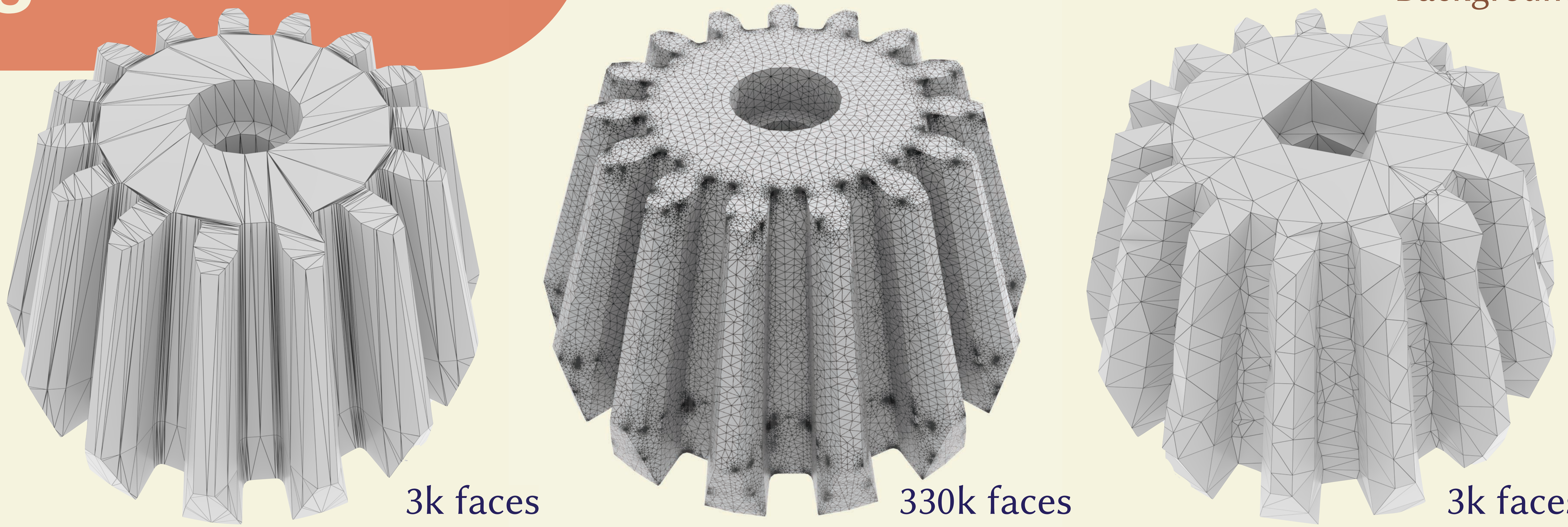
70 milliseconds

[Shewchuk 1997]

Trade offs of extrinsic remeshing in 3D



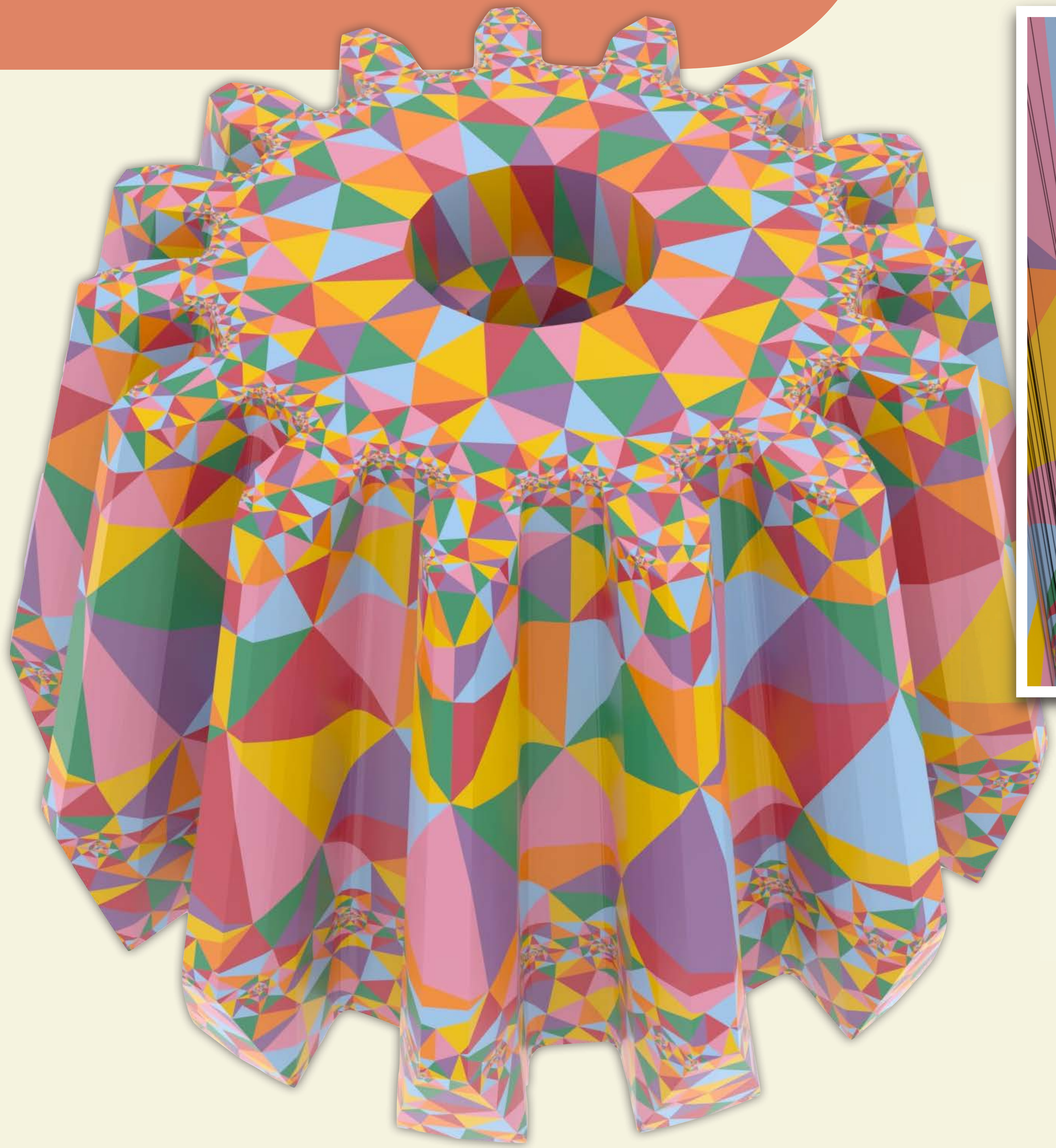
Background

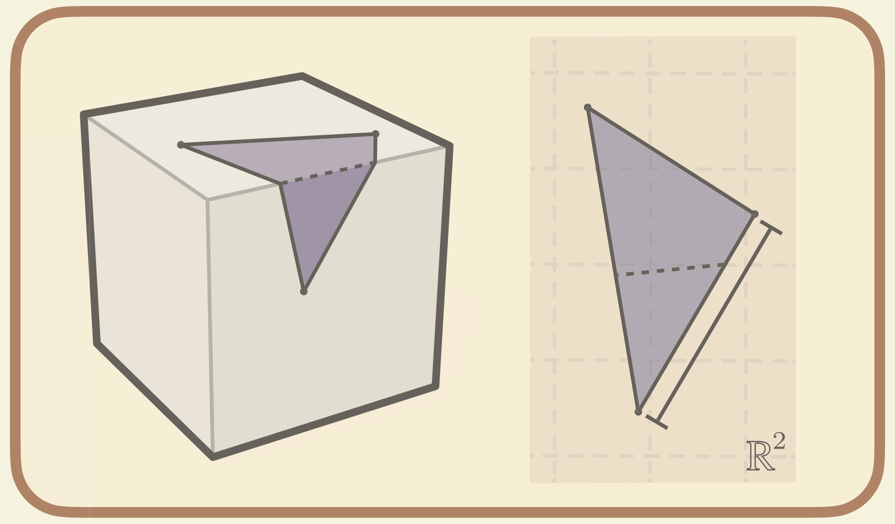


3k faces | 330k faces | 3k faces

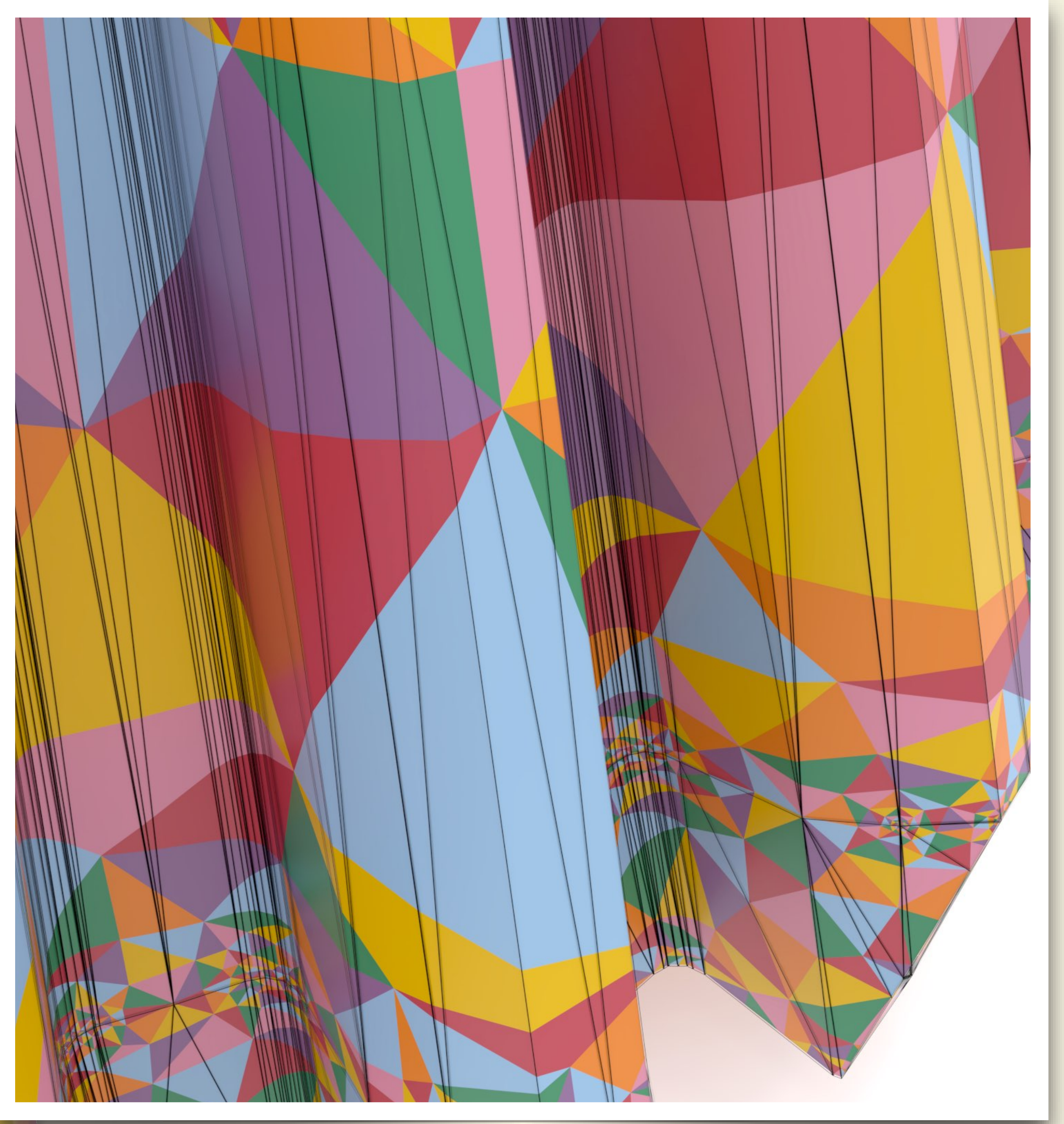
triangle quality	✘	✔	✔
mesh size	✔	✘	✔
geometric fidelity	✔	✔	✘

Intrinsic triangulations sidestep the trade off



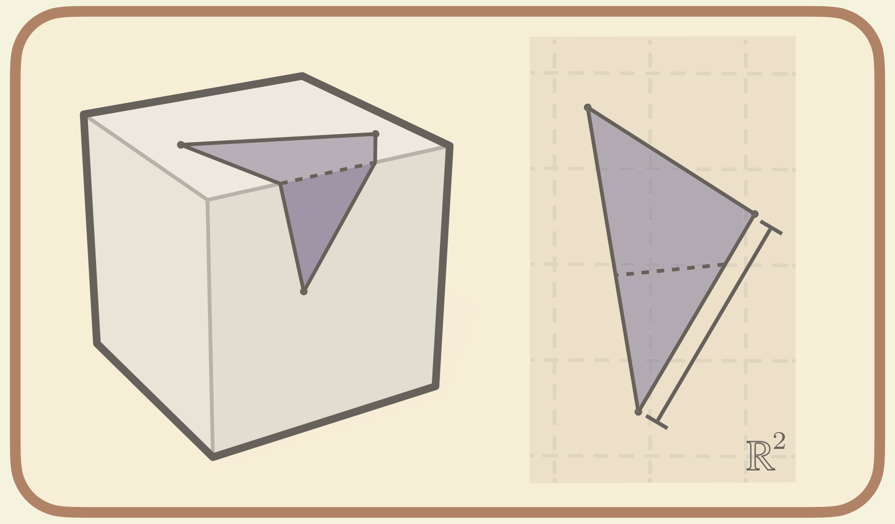


Background

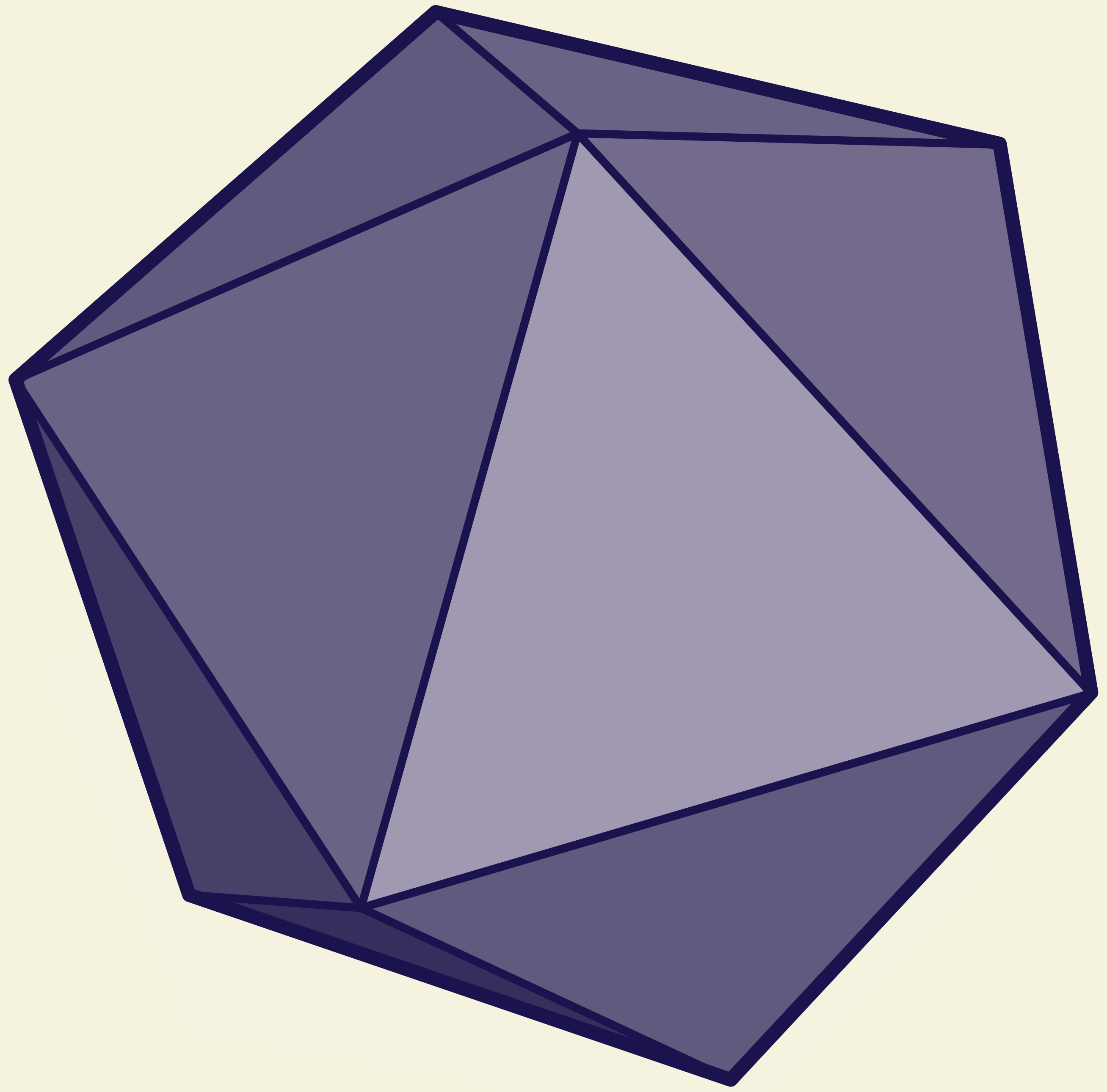


takes a fraction of a second

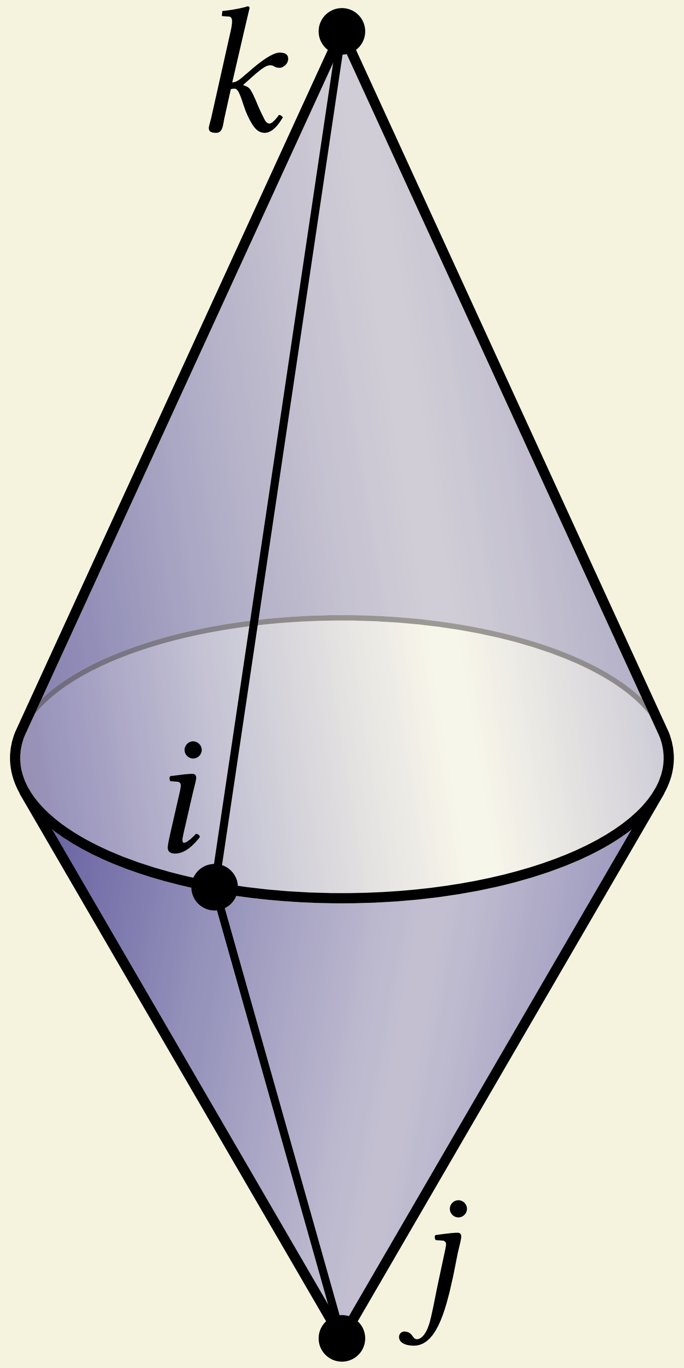
Triangulations



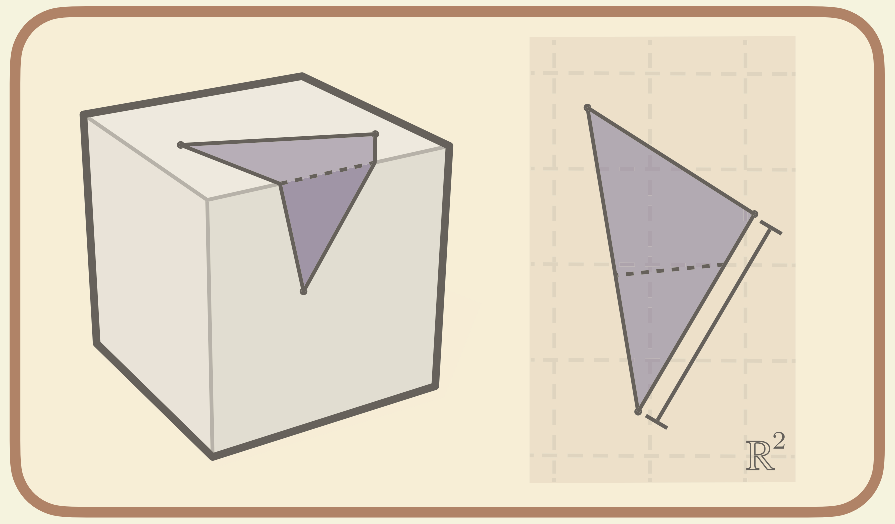
A *triangulation* is a collection of triangles glued together along their edges to form a surface



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



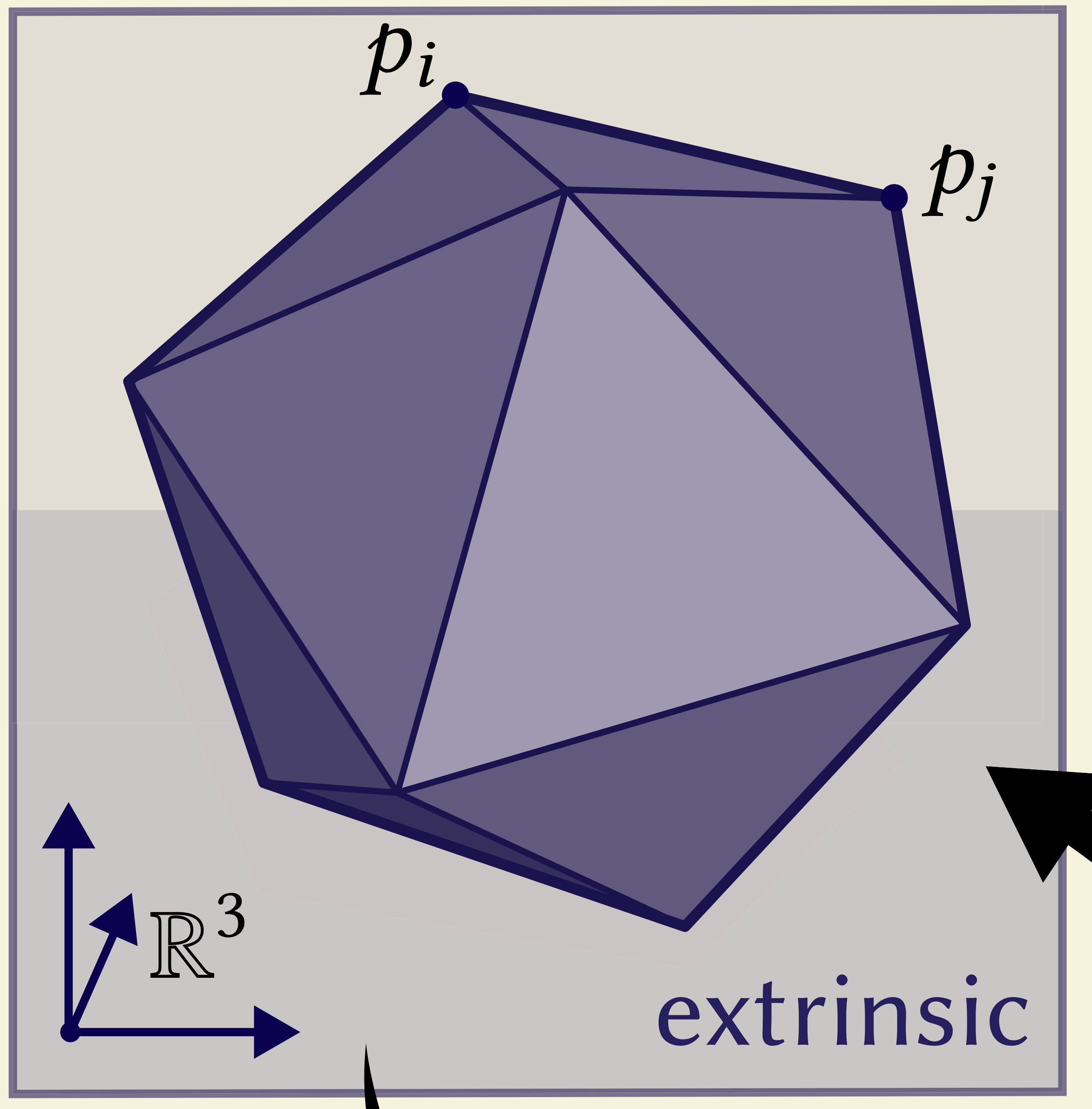
Extrinsic and intrinsic triangulations

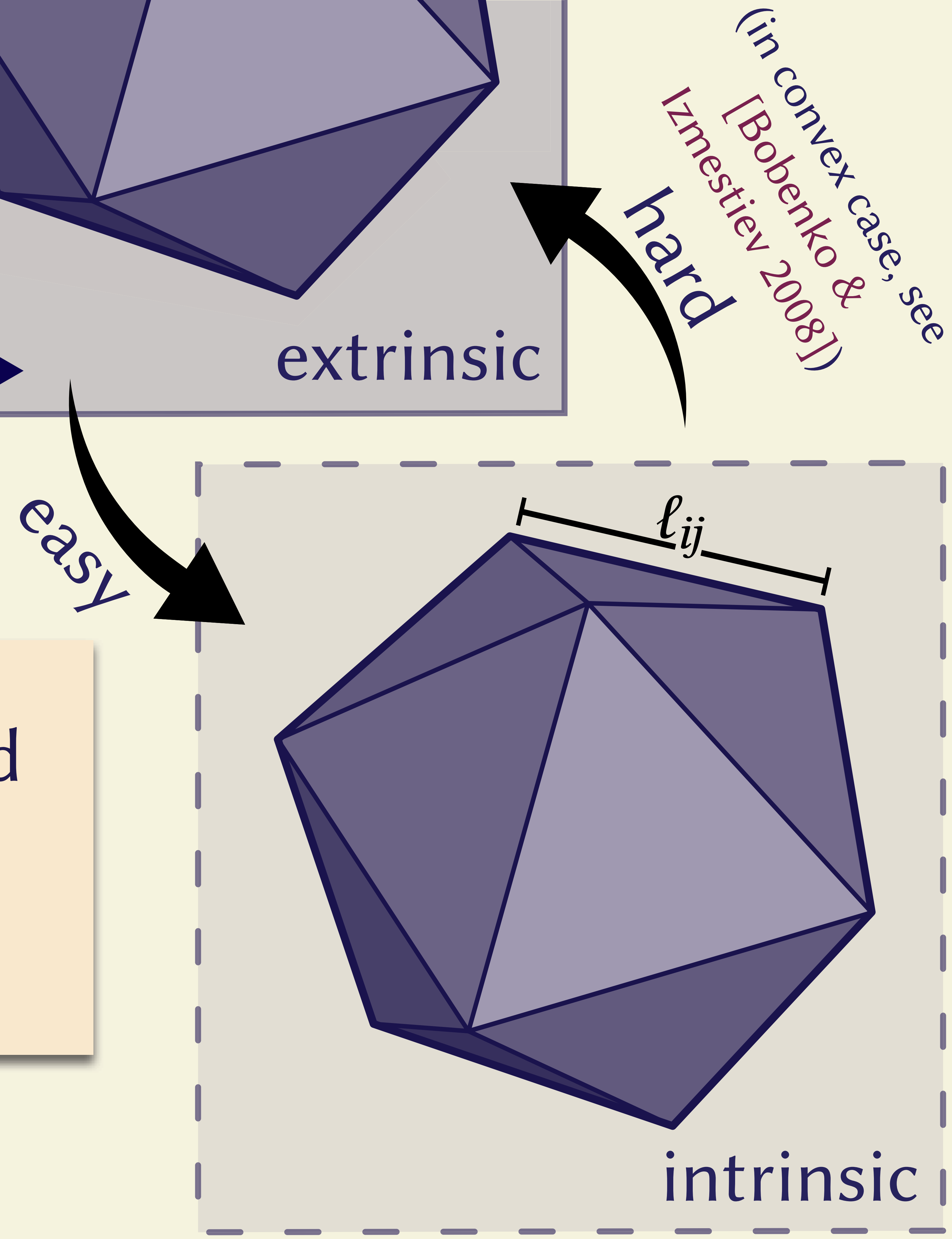


Background

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

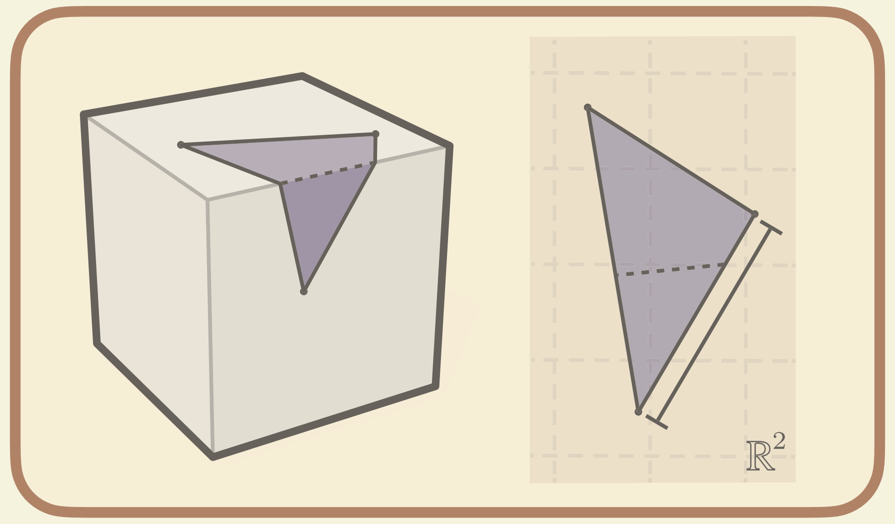
An *intrinsic triangulation* is a triangulation equipped with positive edge lengths $\ell : E \to \mathbb{R}_{>0}$ satisfying the triangle inequality





I'll refer to both as "triangle meshes"

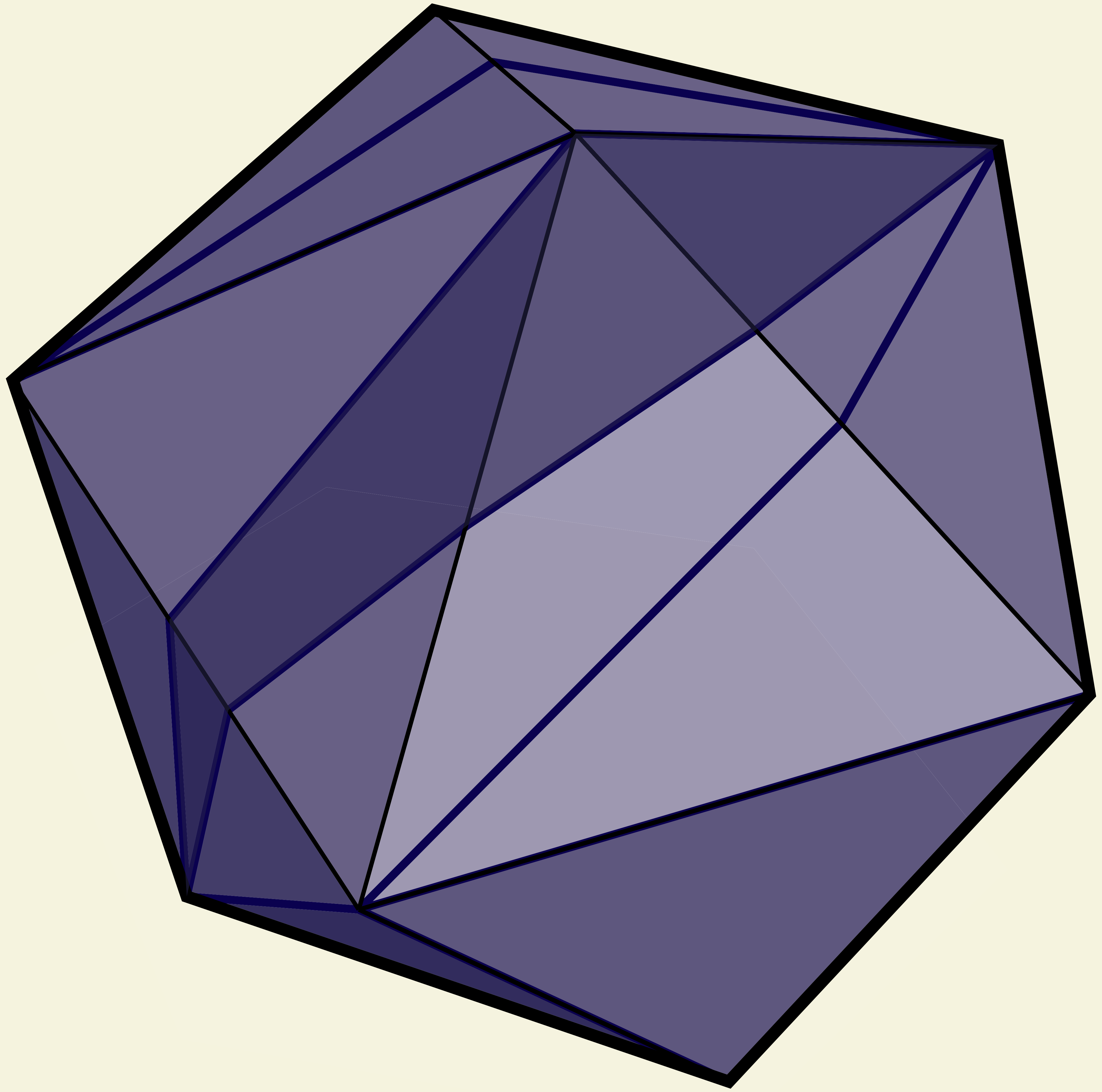
Correspondence



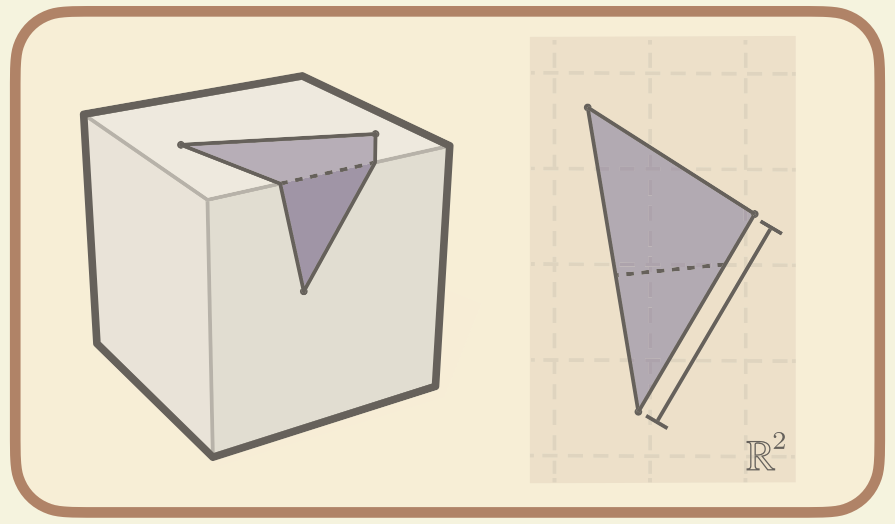
Background

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

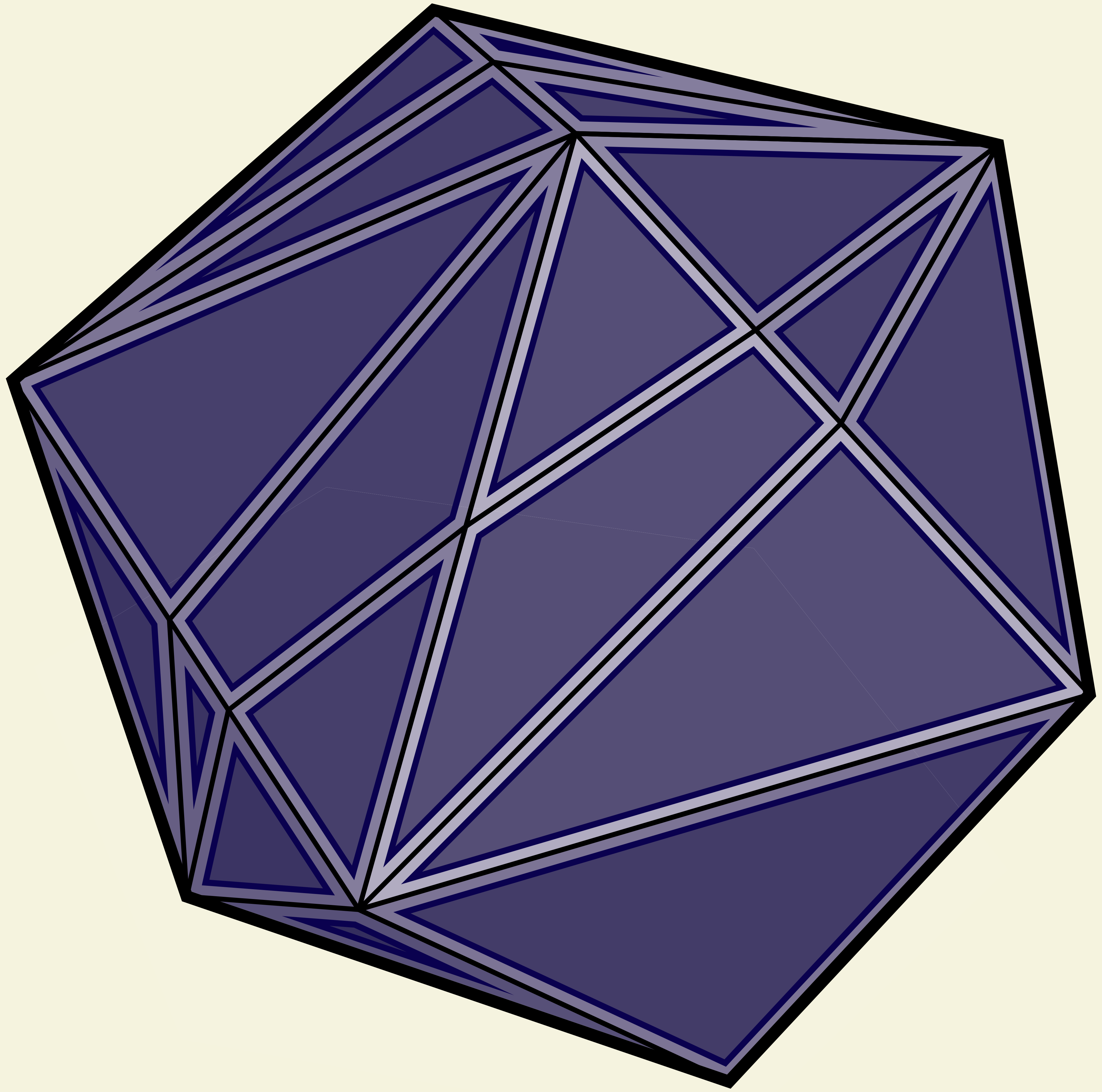


Common subdivision

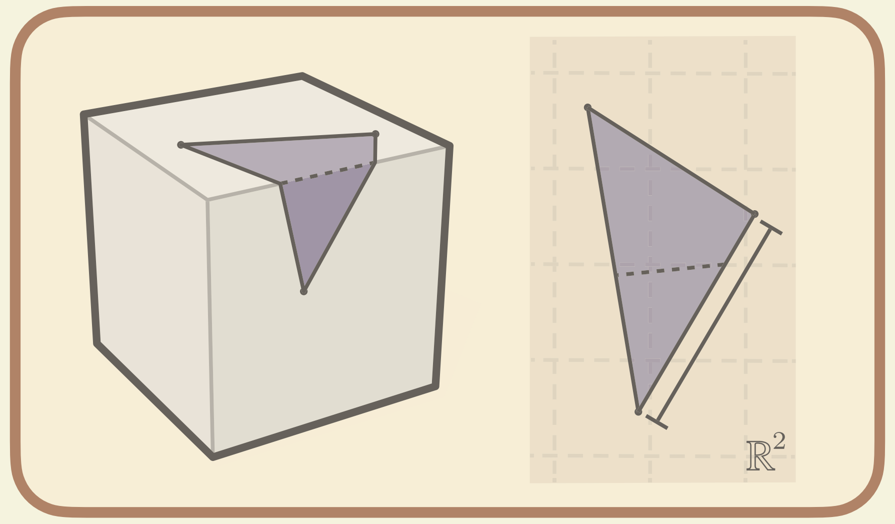


The *common subdivision* of two triangulations is the result of cutting one triangulation along the edges of the other

- Contains both triangulations



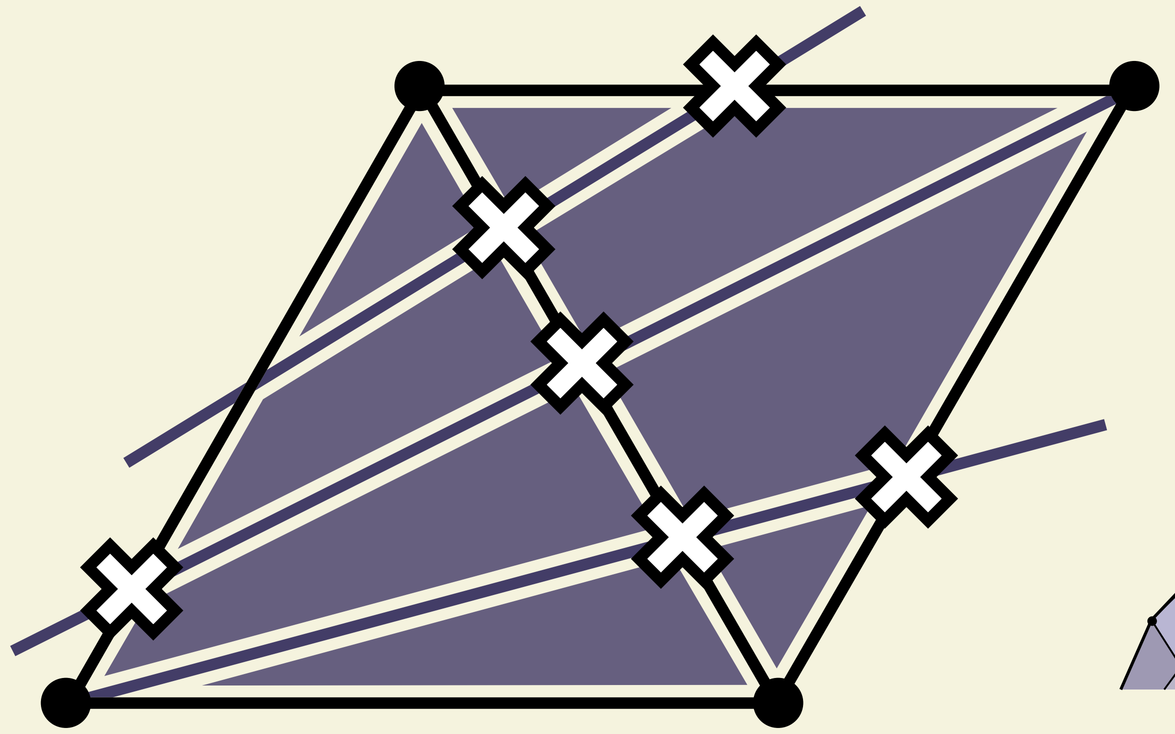
The challenge of evolving intrinsic triangulations



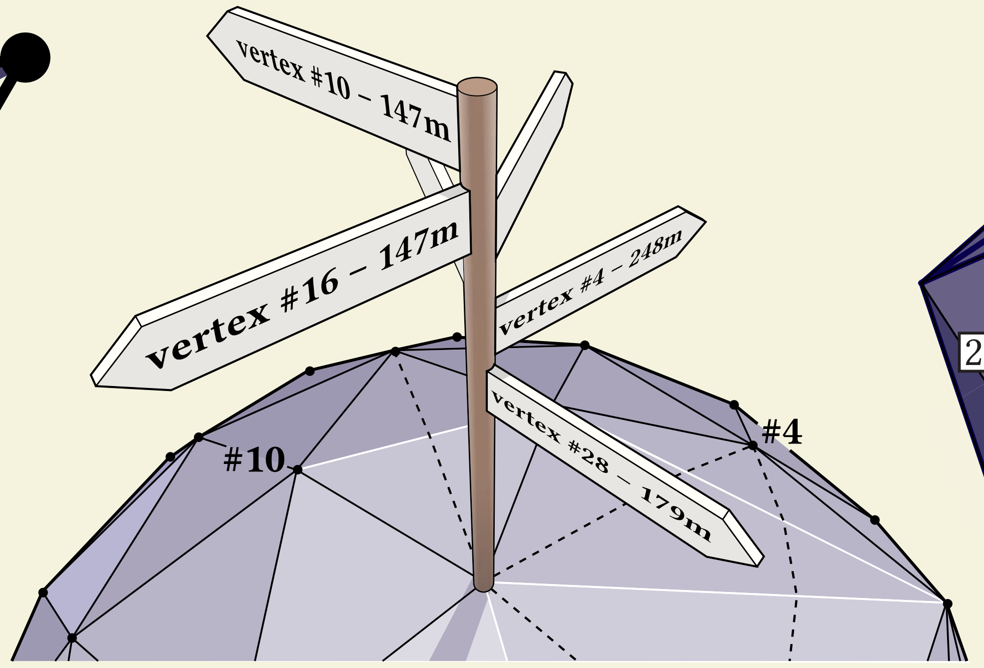
Background

- Tracking correspondence between meshes with different geometry

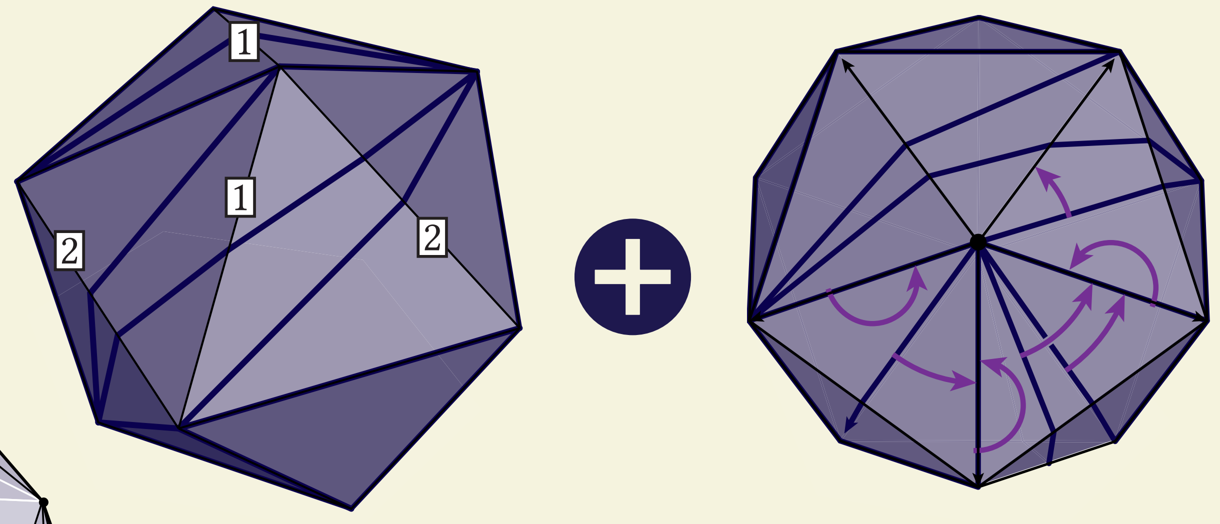
CORRESPONDENCE WITH SAME GEOMETRY



[Fisher, Springborn, Bobenko & Schröder 2006]



[Sharp, Soliman & Crane 2019]

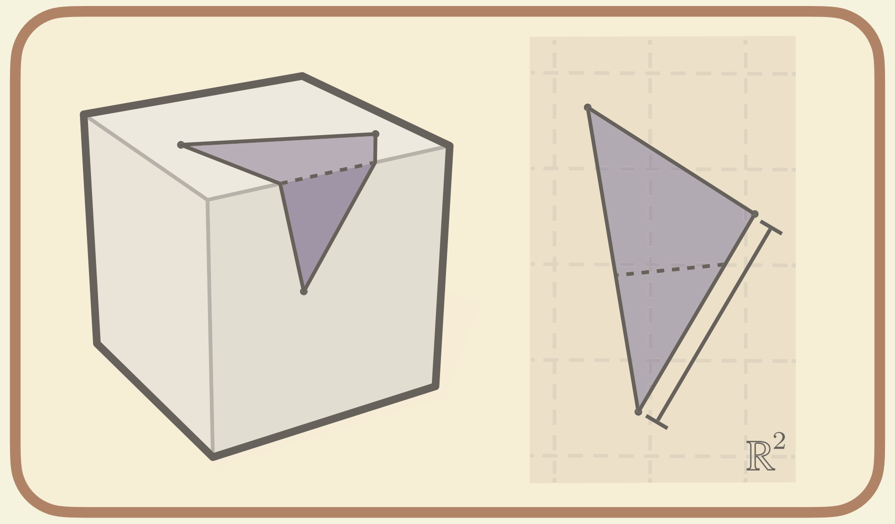


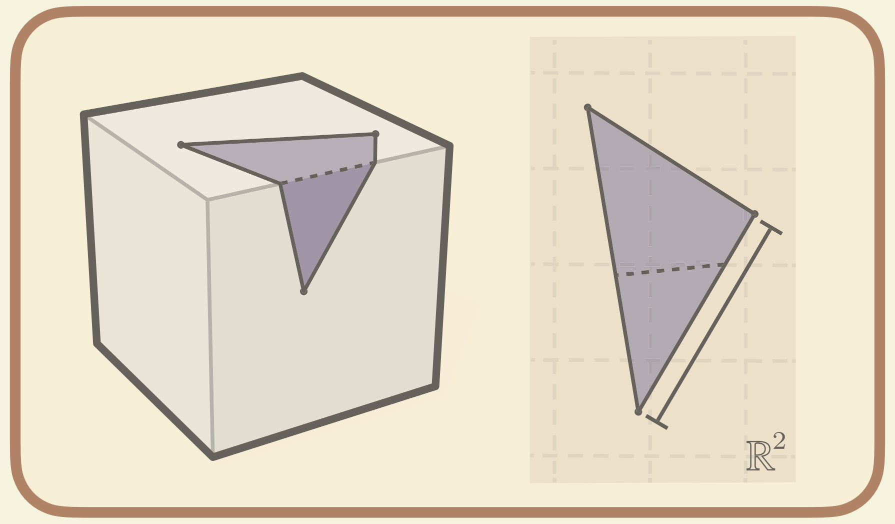
[Gillespie, Sharp & Crane 2021]

CORRESPONDENCE WITH DIFFERENT GEOMETRY

???

The space of intrinsic triangulations is large

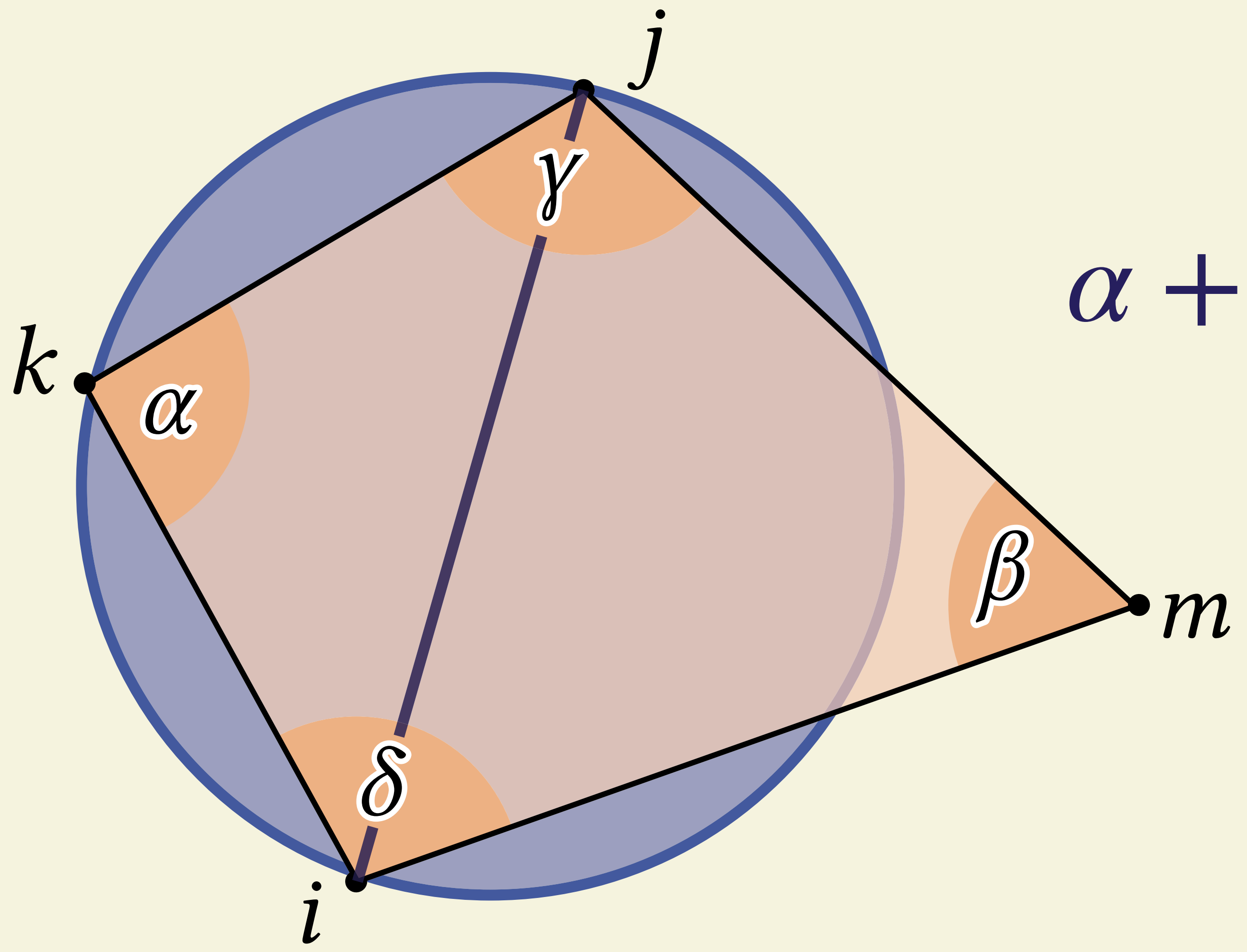




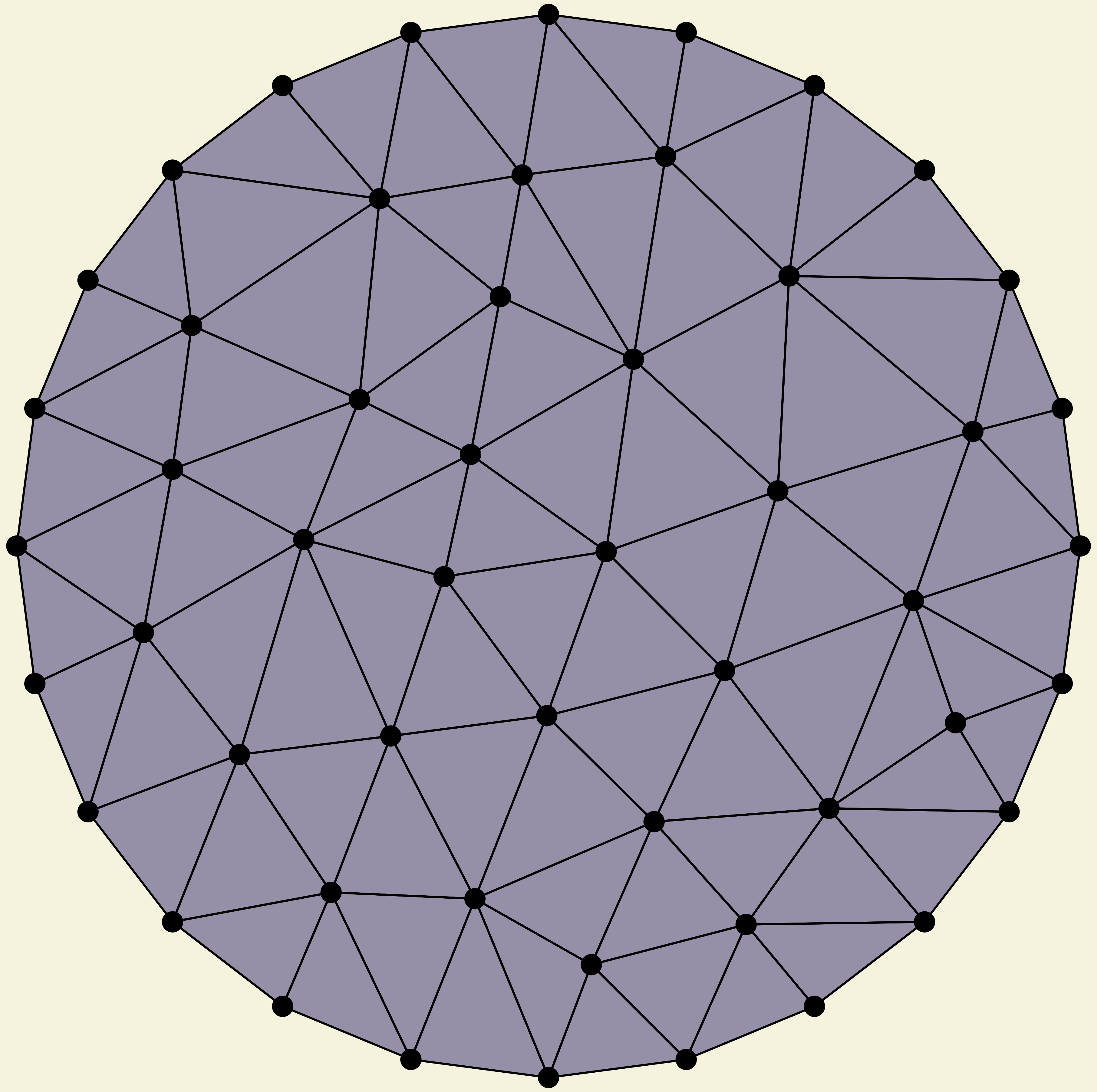
Background

Delaunay triangulations

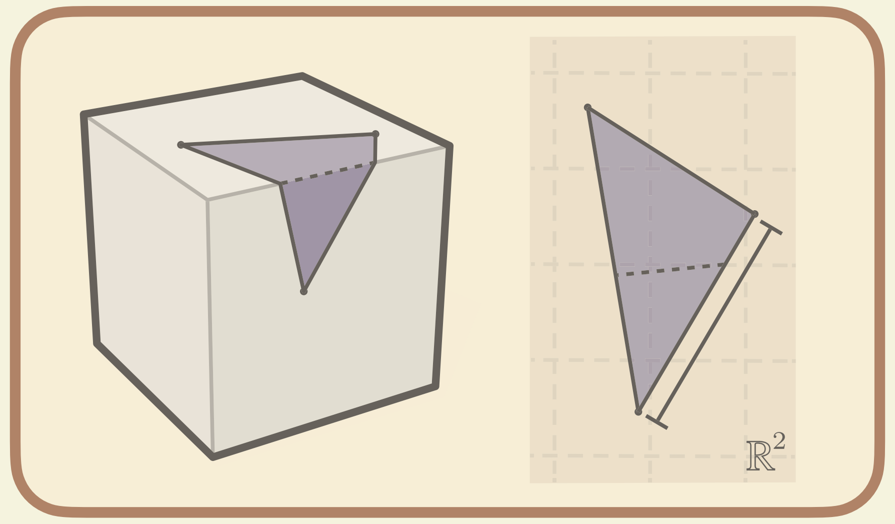
- Countless useful properties:
  - Essentially unique, maximize angles lexicographically, minimize spectrum lexicographically, smoothest interpolation, positive cotan weights...
- Characterized by *empty circumcircle condition*



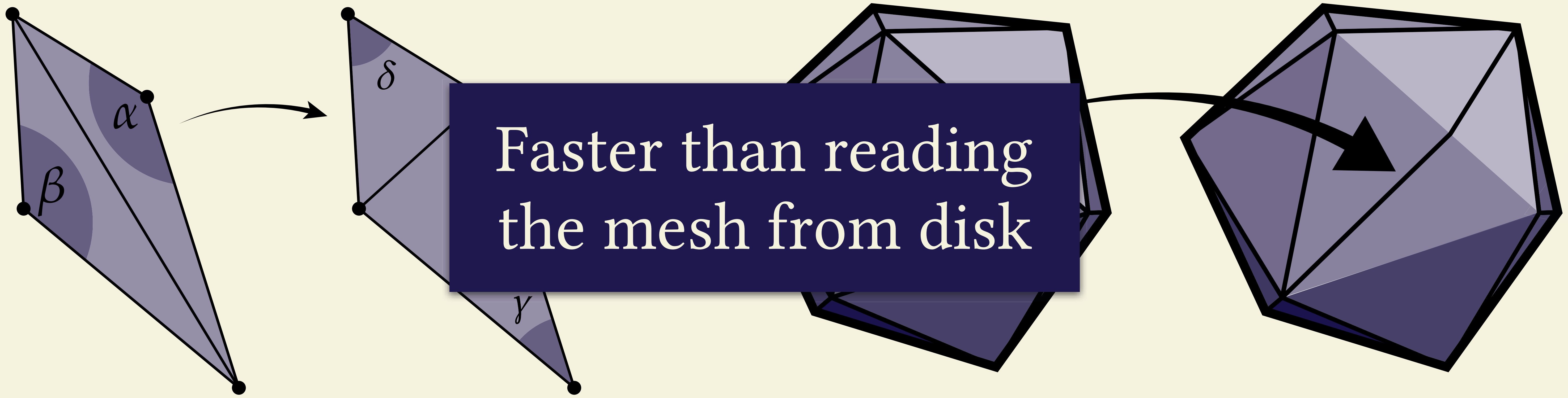
$$\alpha + \beta \leq \gamma + \delta$$

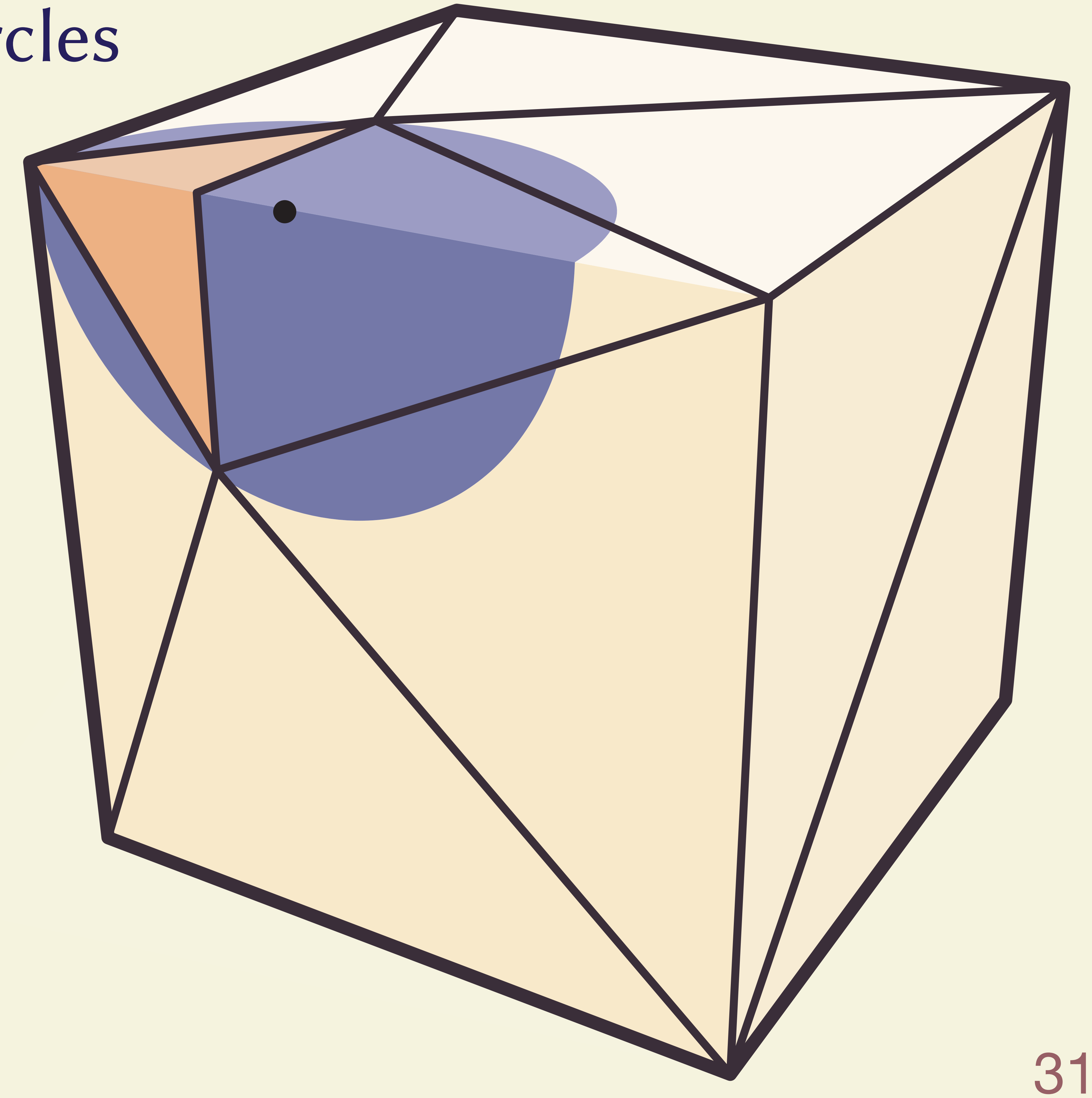


Intrinsic Delaunay triangulations

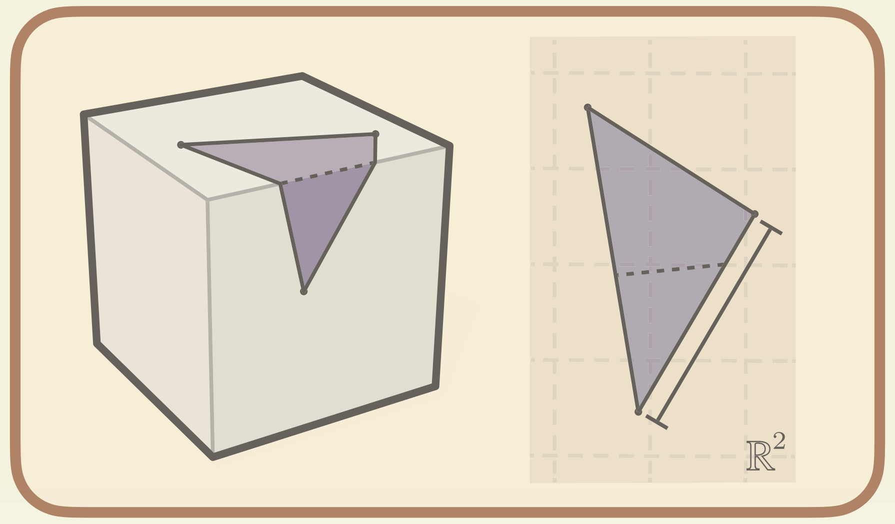


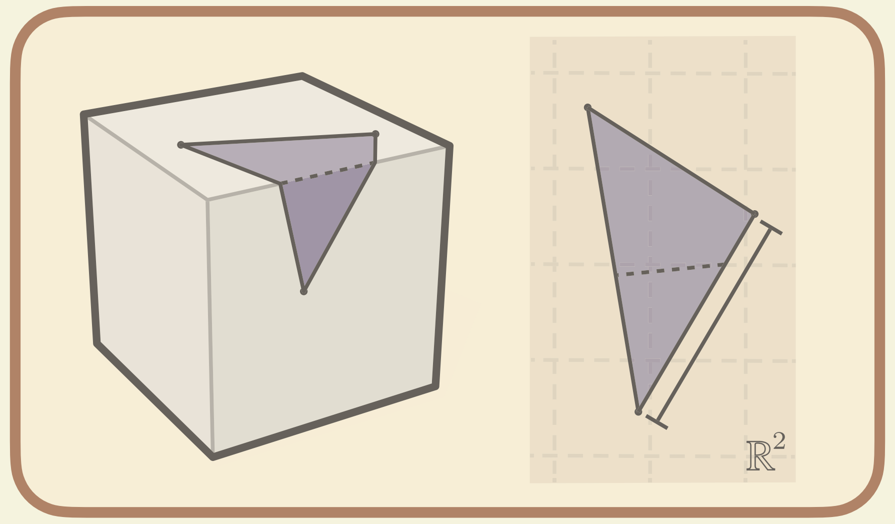
- [Indermitte, Liebling, Troyanov & Clemençon 2001, Bobenko & Springborn 2007]: empty intrinsic circumcircles
  - Maintain many nice properties. [Sharp, **Gillespie** & Crane 2021; §4.1.1]
- Compute by a simple algorithm:
  - Flip any non-Delaunay edge until none remain





Intrinsic Delaunay triangulations provide good function spaces





Background

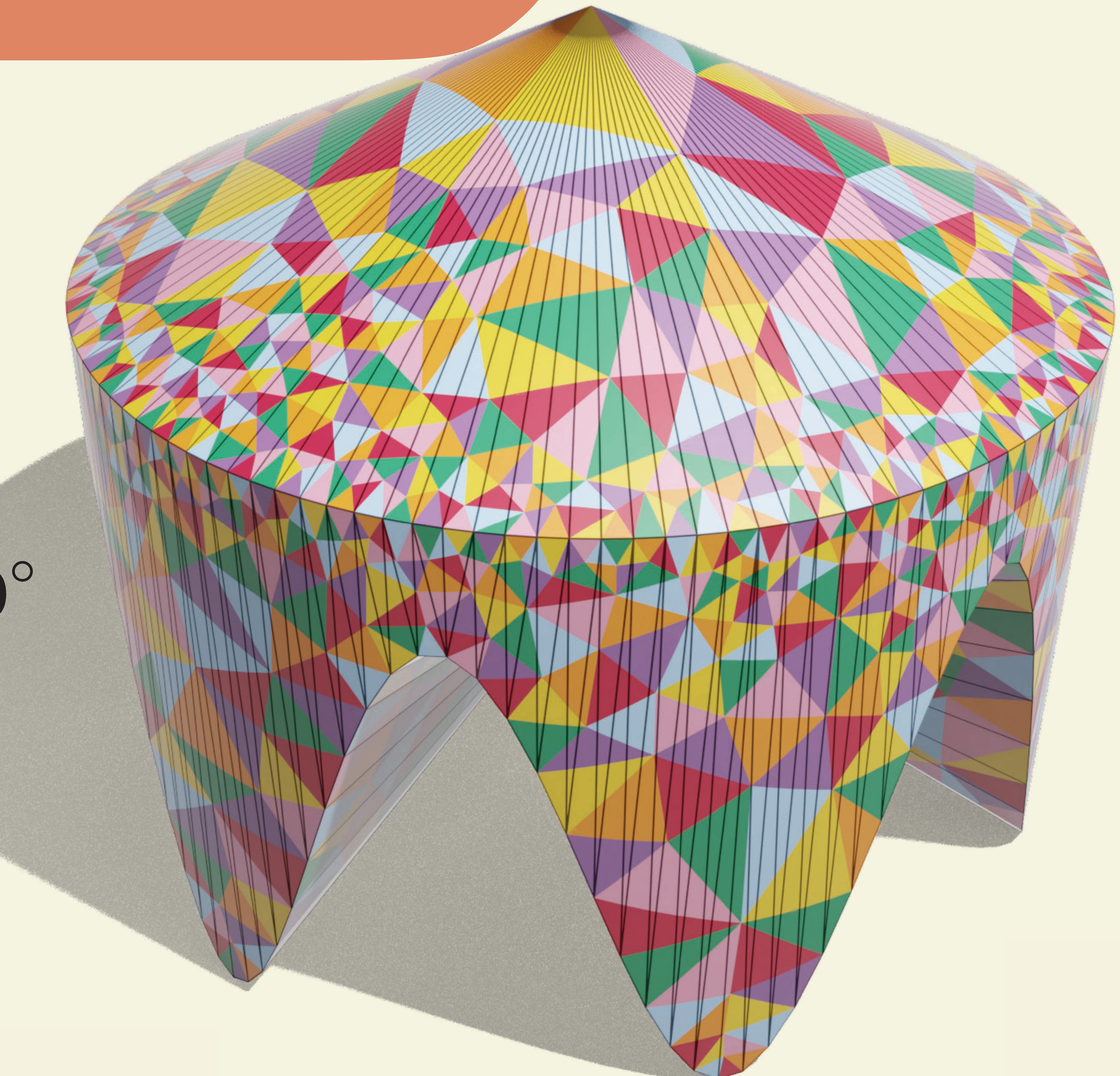
Intrinsic Delaunay refinement

[Sharp, Soliman & Crane 2019]

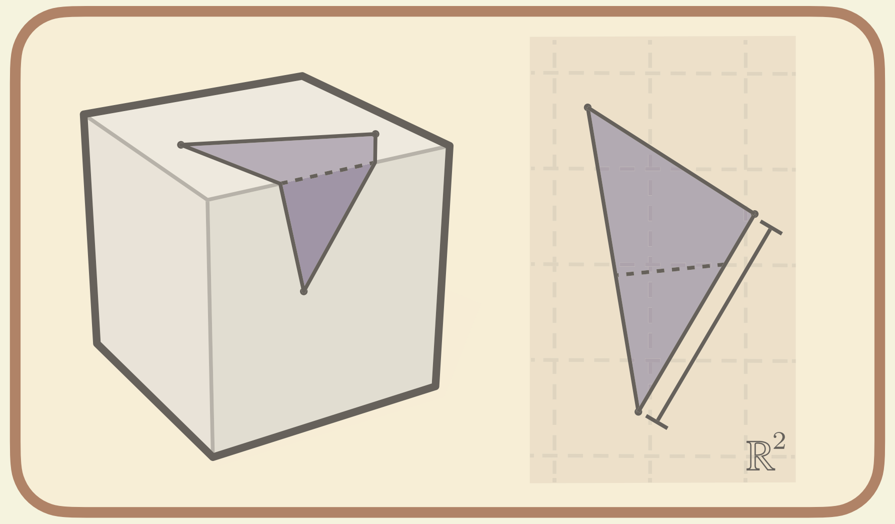
Add vertices intrinsically
to improve quality

Theorem [G., Sharp & Crane 2021]

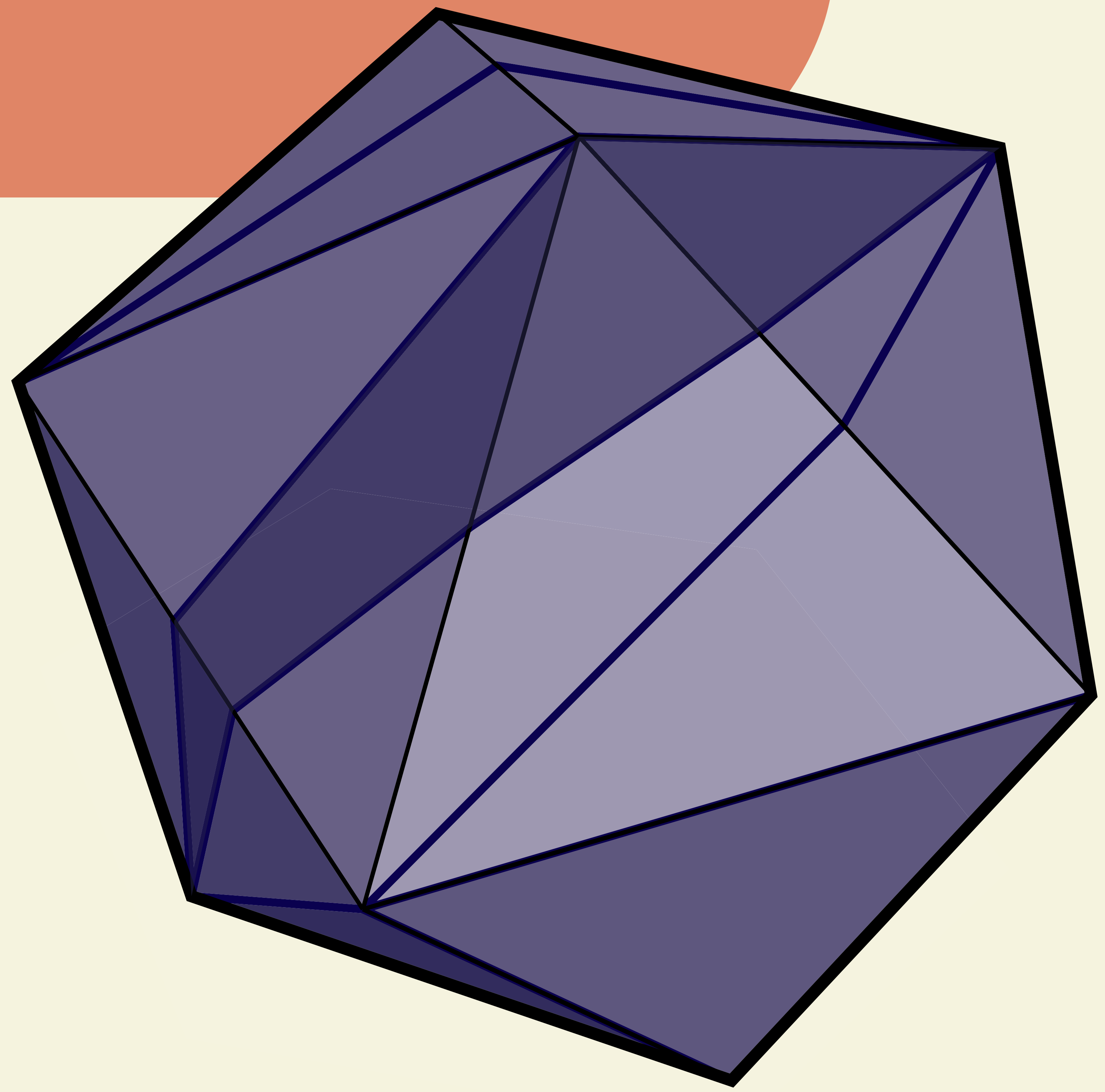
Let M be a mesh without boundary whose cone angles are all at least 60°. Then intrinsic Delaunay refinement produces a Delaunay mesh with triangle corner angles at least 30°



A brief history of intrinsic triangulations



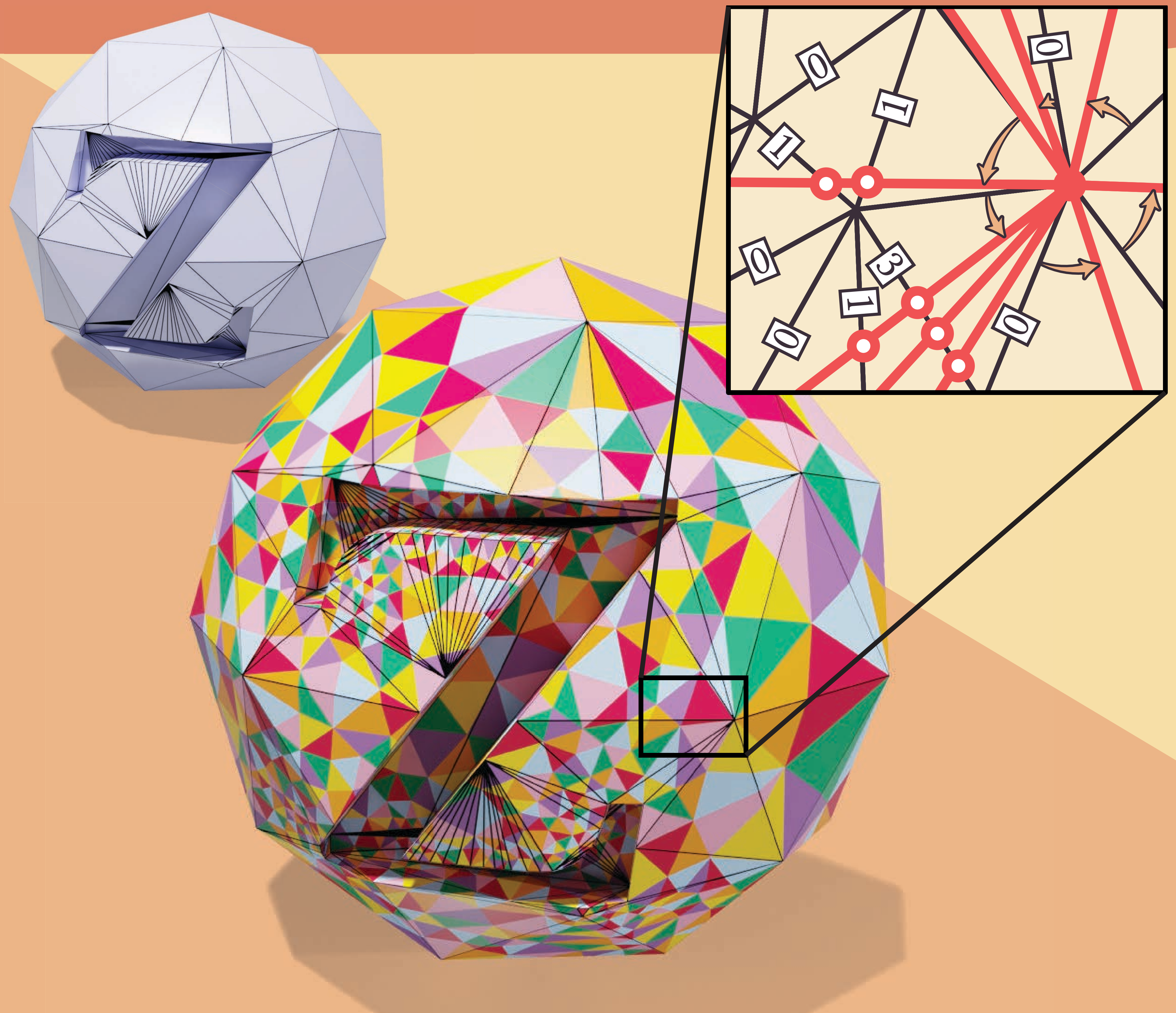
Background



Foundations: [Alexandrov 1948; Regge 1961]

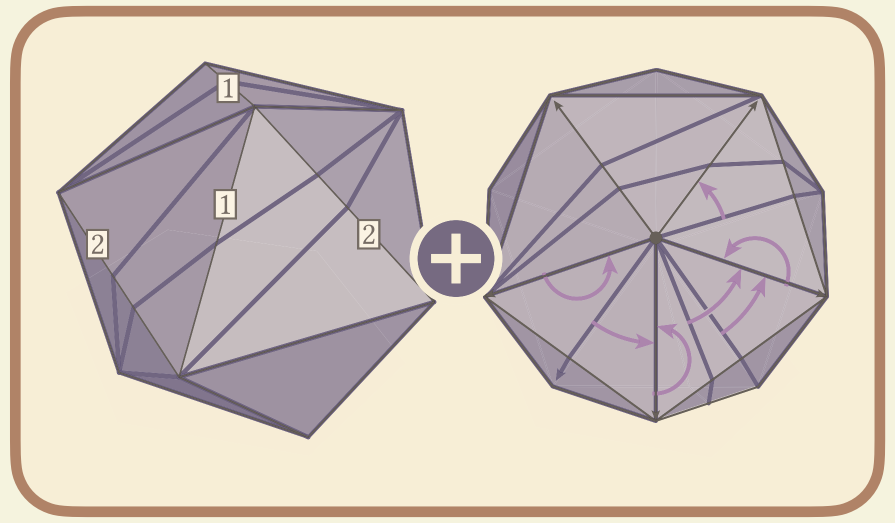
Geometry Processing: [Fisher, Springborn, Bobenko & Schröder 2006; Bobenko & Springborn 2007, Bobenko & Izmestiev 2008; Sun, Wu, Gu & Luo 2015; Sharp, Soliman & Crane 2019; Fumero, Möller & Rodolà 2020; Gillespie, Springborn & Crane 2021; Finnendahl, Schwartz & Alexa 2023]

II. Data Structures for Intrinsic Triangulations



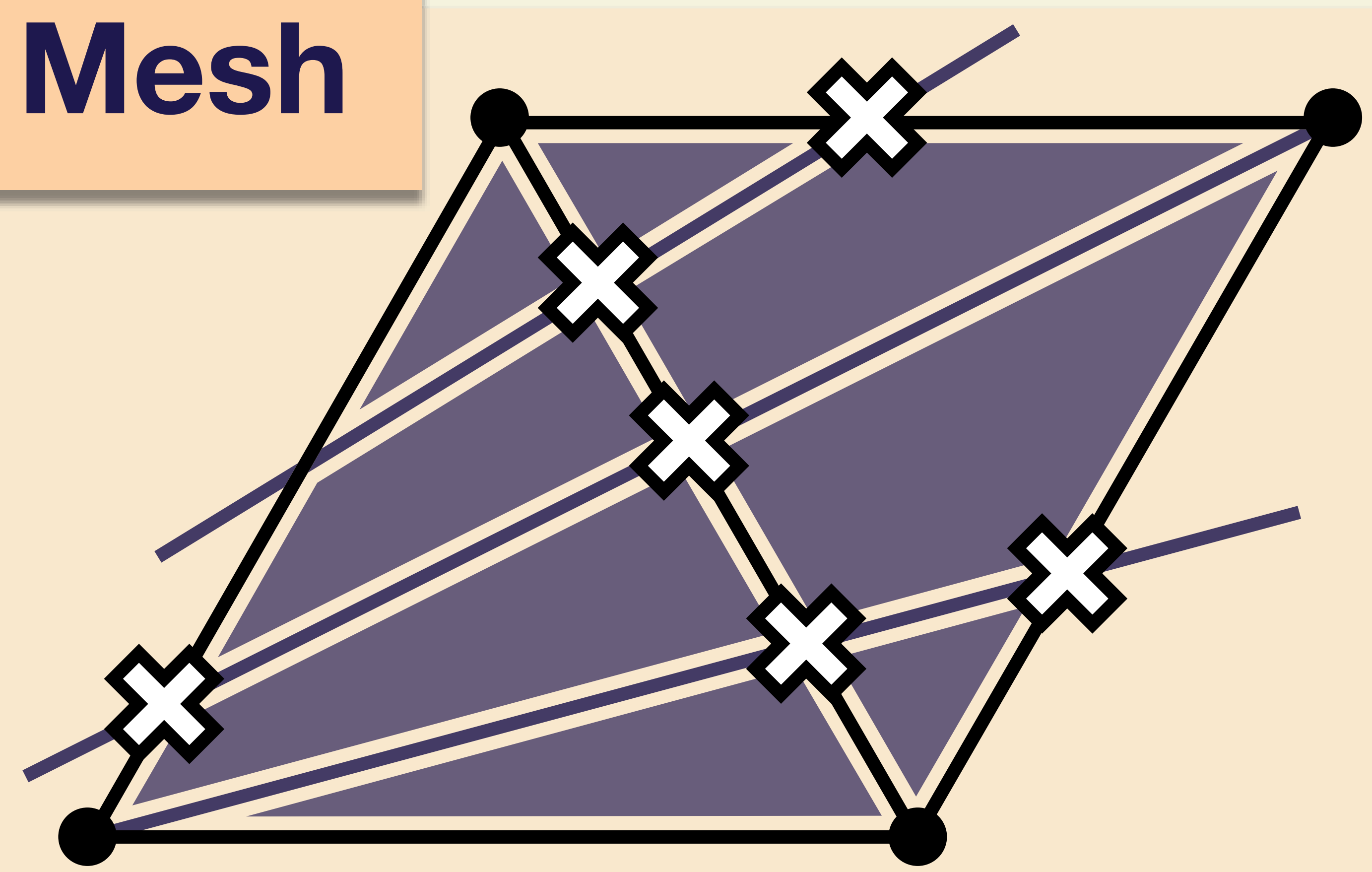
[**Gillespie**, Sharp, & Crane. 2021. Integer coordinates for intrinsic geometry processing. *ACM Transactions on Graphics*]

Correspondence data structures



Integer coordinates for intrinsic triangulations

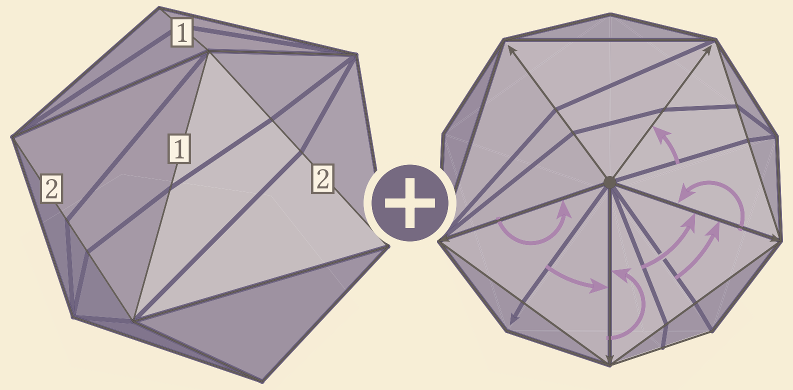
Overlay Mesh



[Fisher, Springborn, Bobenko & Schröder 2006]

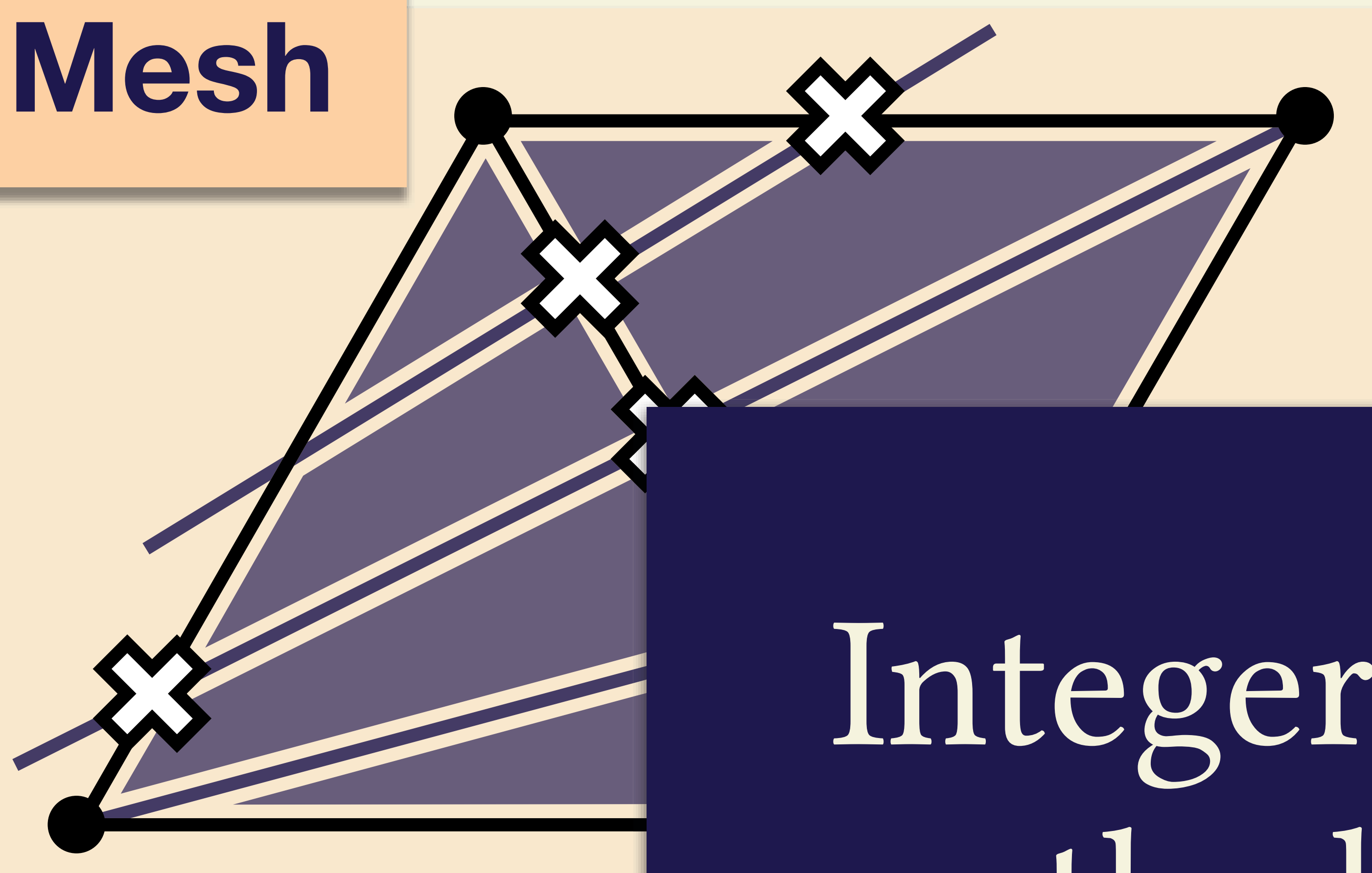
- Explicit mesh of common subdivision
- Edge flips nonlocal & expensive
  - **No further operations**

Correspondence data structures



Integer coordinates for intrinsic triangulations

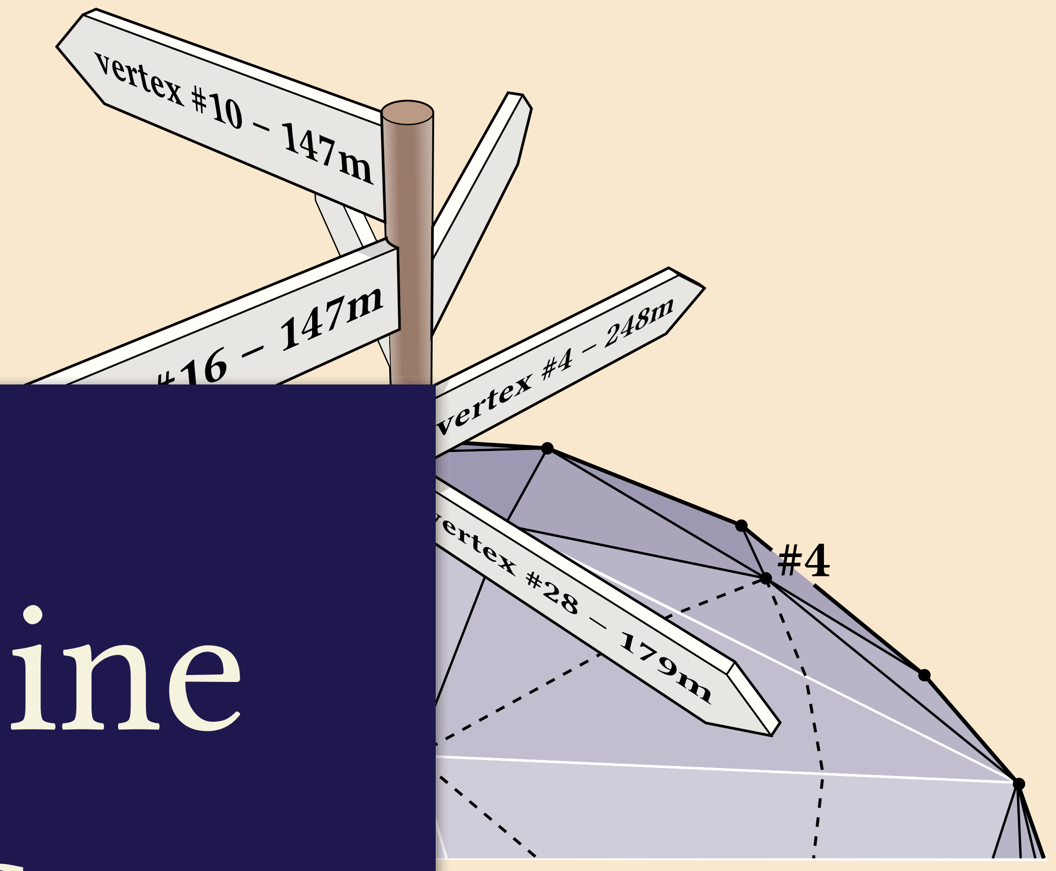
Overlay Mesh



[Fisher, Springb
& Schröd

- Explicit mesh of common subdivision
- Edge flips nonlocal & expensive
  - **No further operations**

Signposts

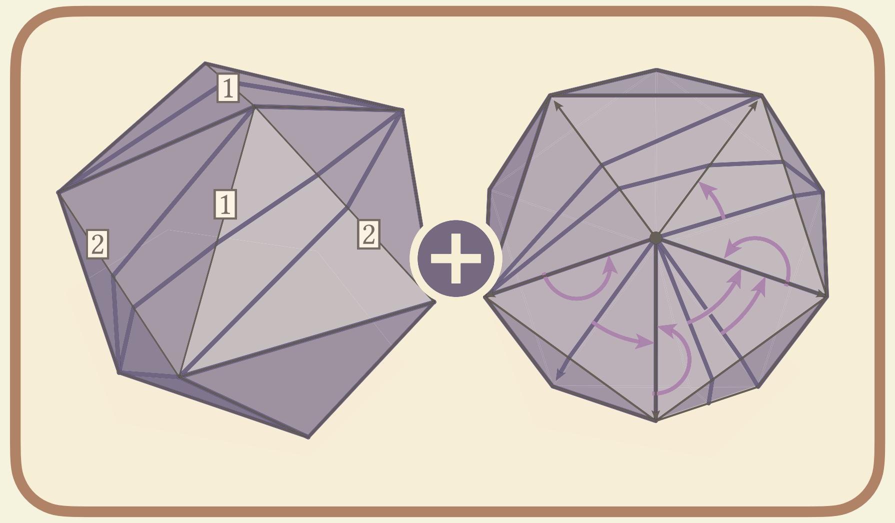


n & Crane 2019]

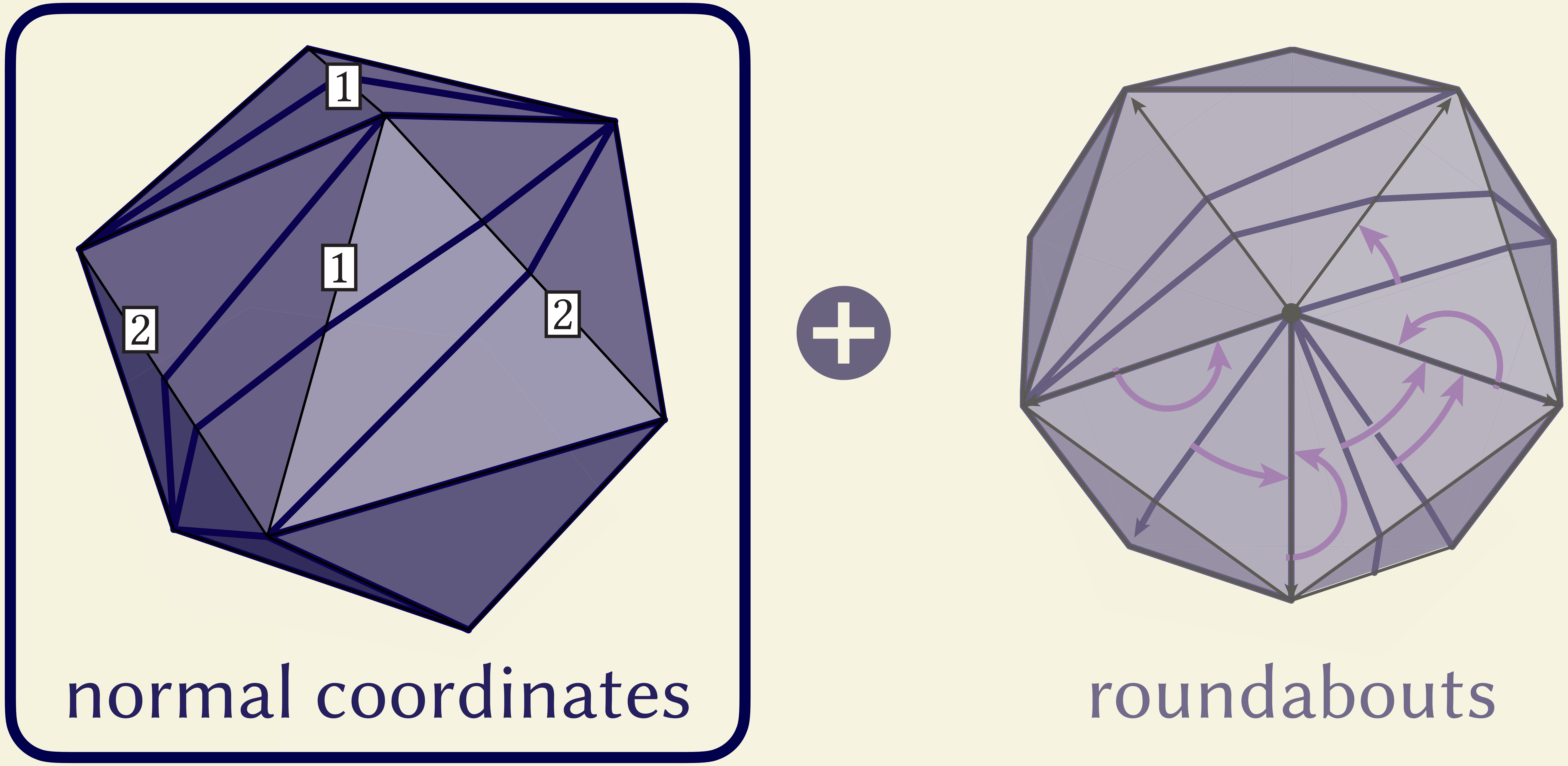
- Floating point quantities stored at vertices
- **Supports many local mesh operations**
- Common subdivision connectivity may be invalid

Integer coordinates combine the best of both worlds

The integer coordinates data structure

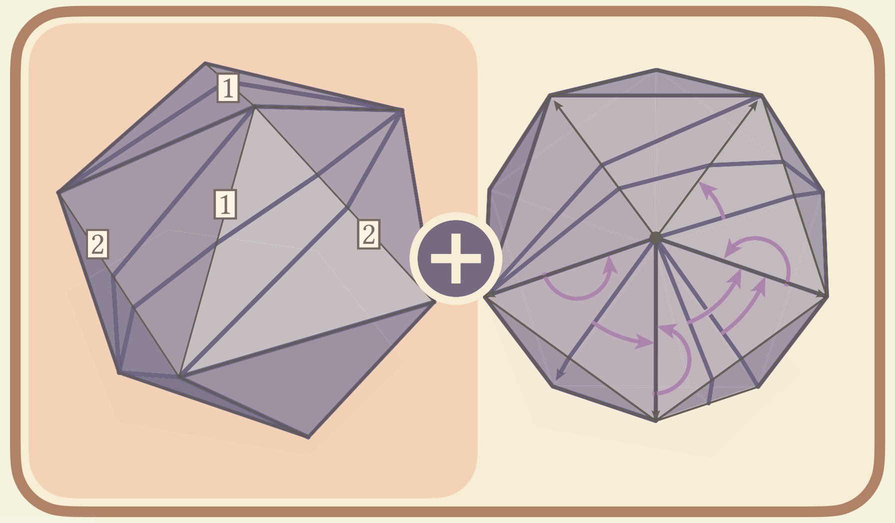


Integer coordinates for intrinsic triangulations

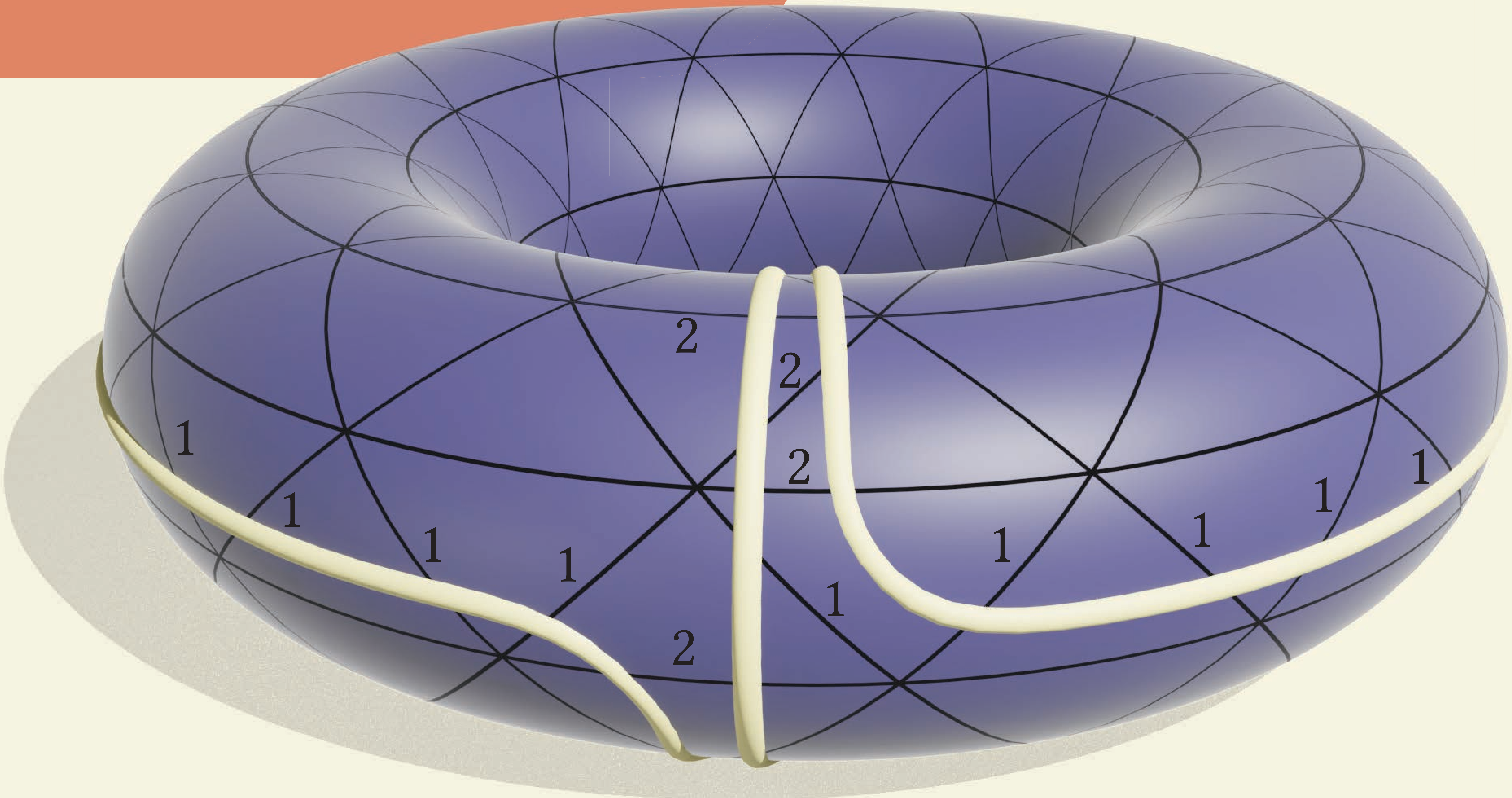


(concretely, just 3 integers per edge)

Normal coordinates



Integer coordinates for intrinsic triangulations

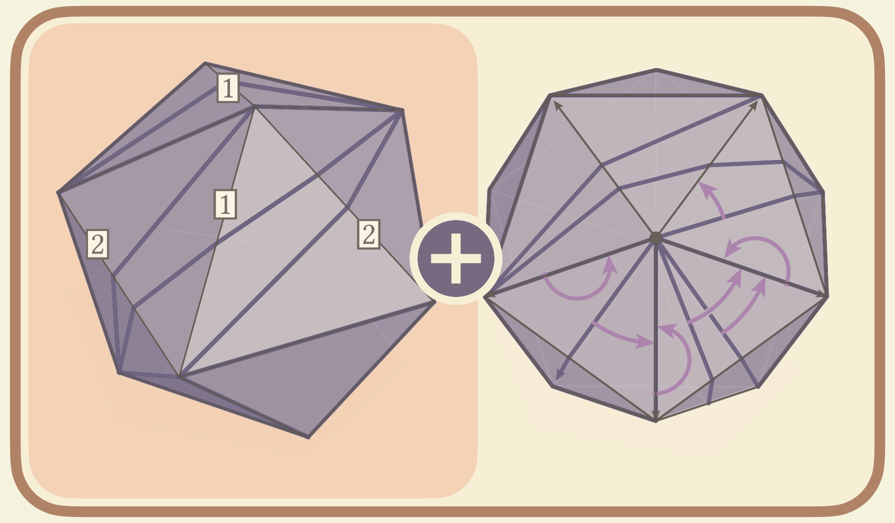


Foundations: [Kneser 1929; Haken 1961]

Computational Topology: [Schaefer+ 2008; Erickson & Nayyeri 2013]

Geometry Processing: [Hass & Trnkova 2020]

Encoding a curve with normal coordinates

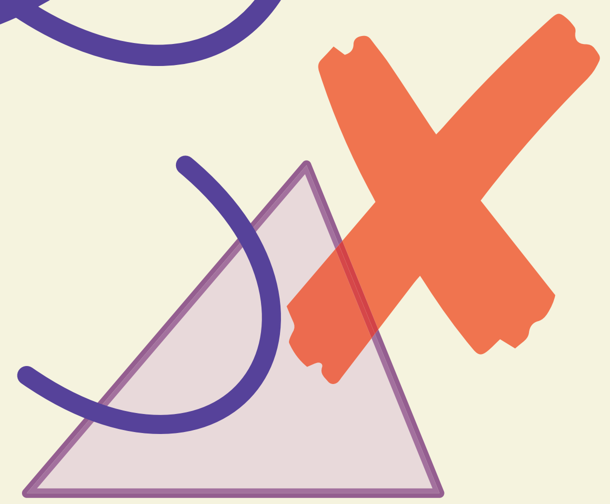


Integer coordinates for intrinsic triangulations

- Just count intersections

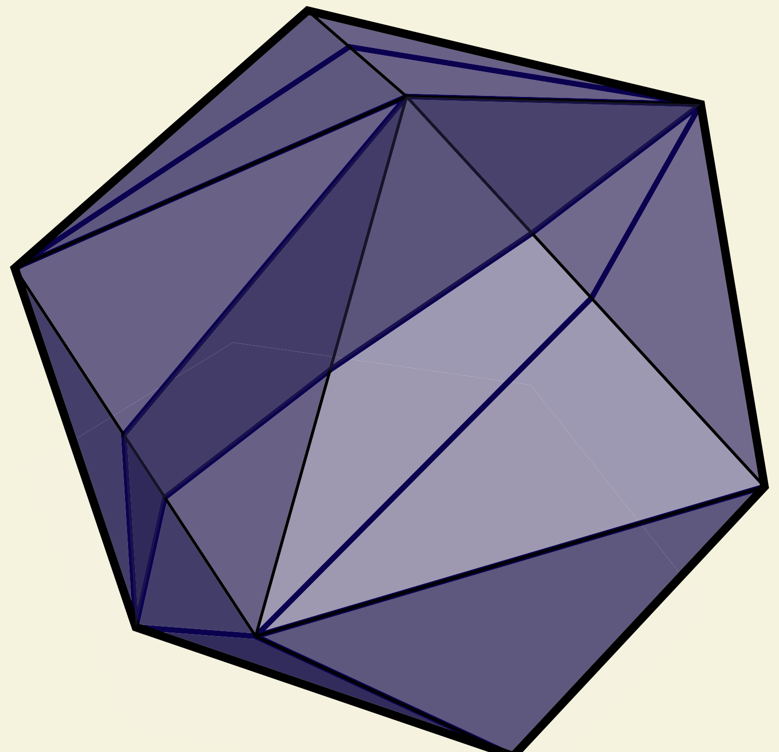
Rules

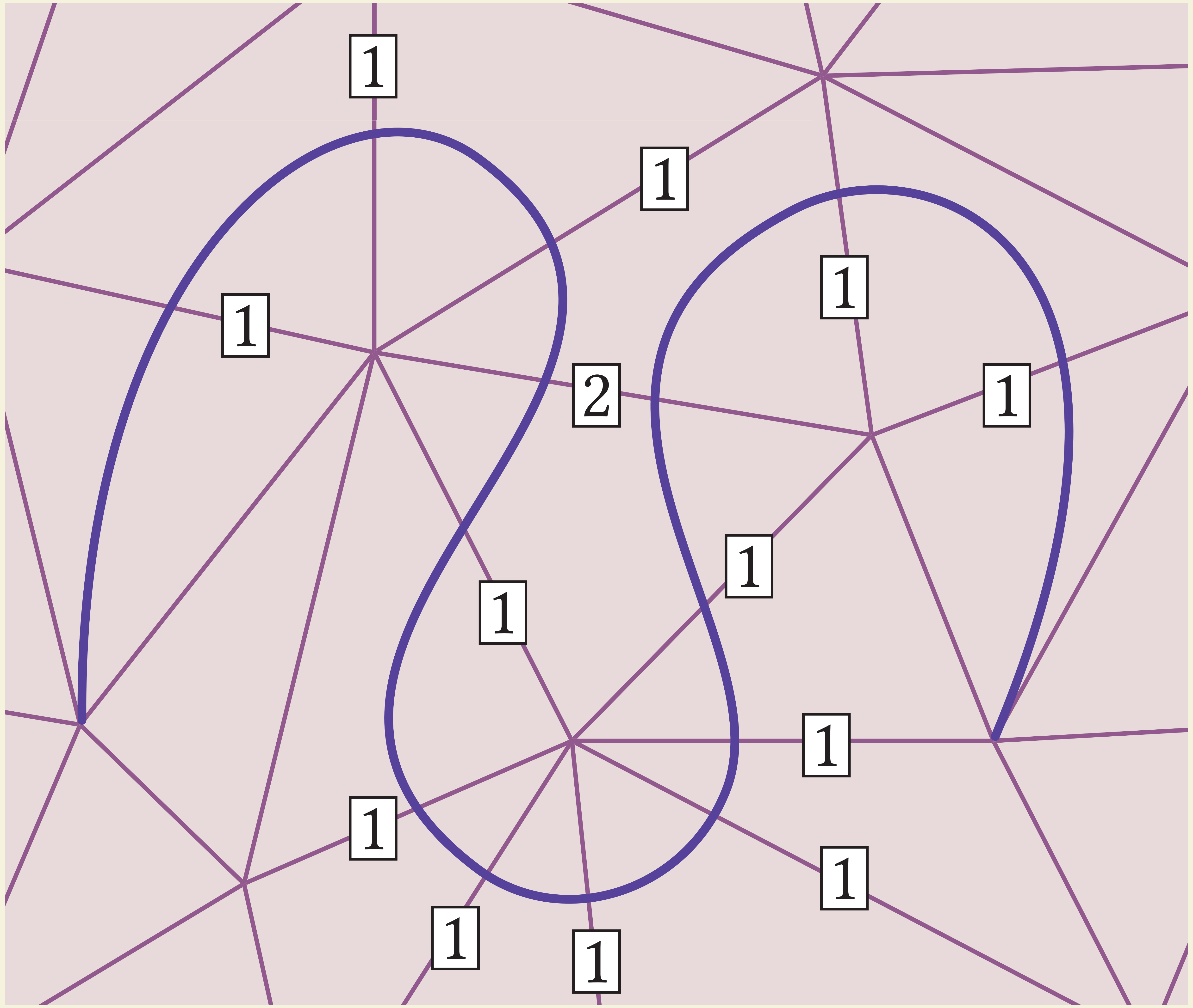
1. No self-crossings
2. No U-turns



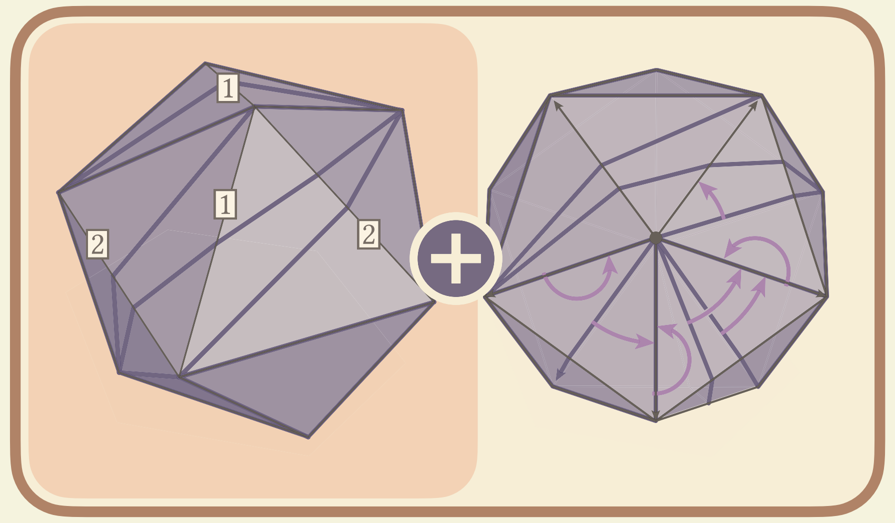
(also curves may only start or end at vertices of the triangulation)

automatically satisfied for our triangulations

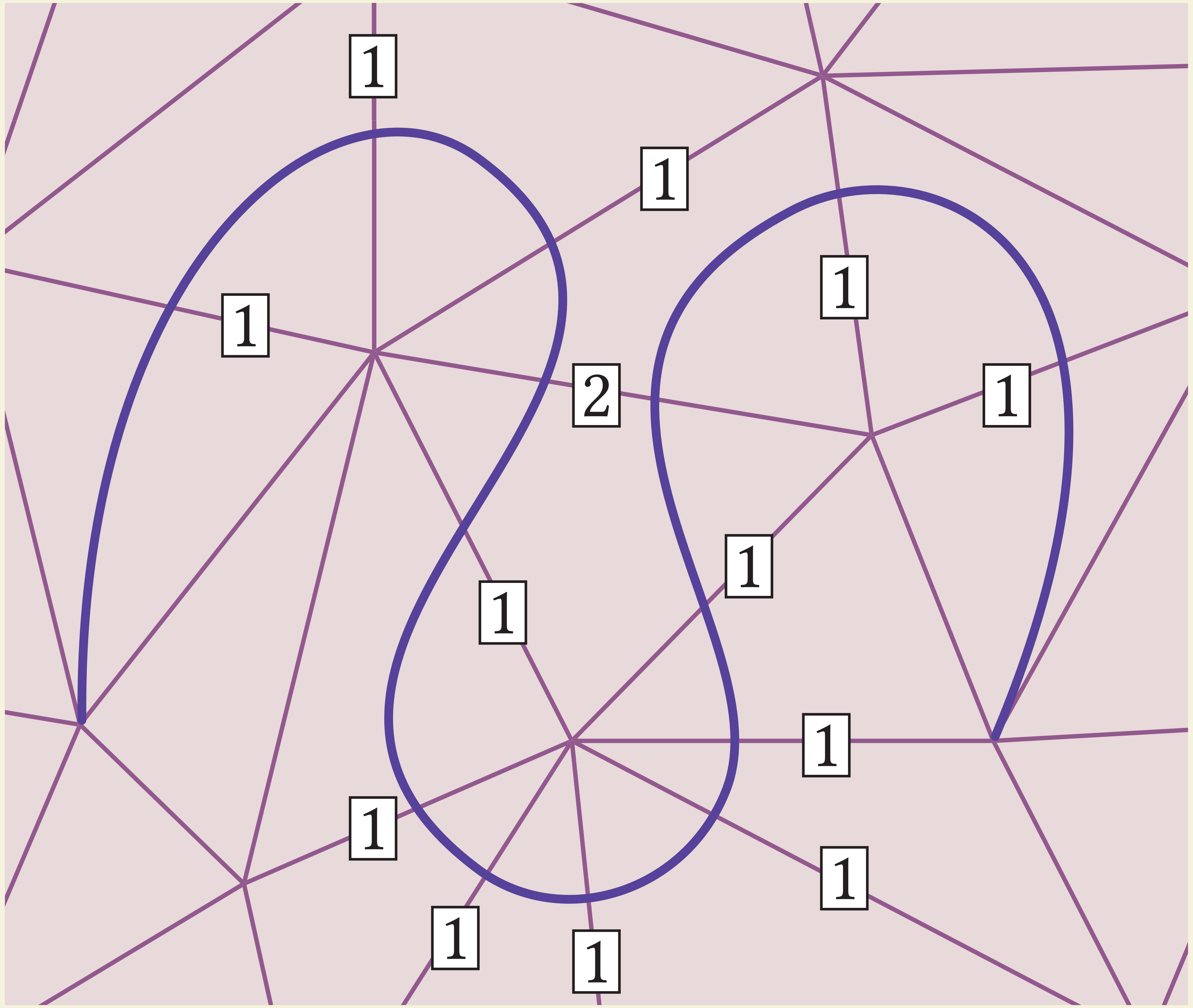




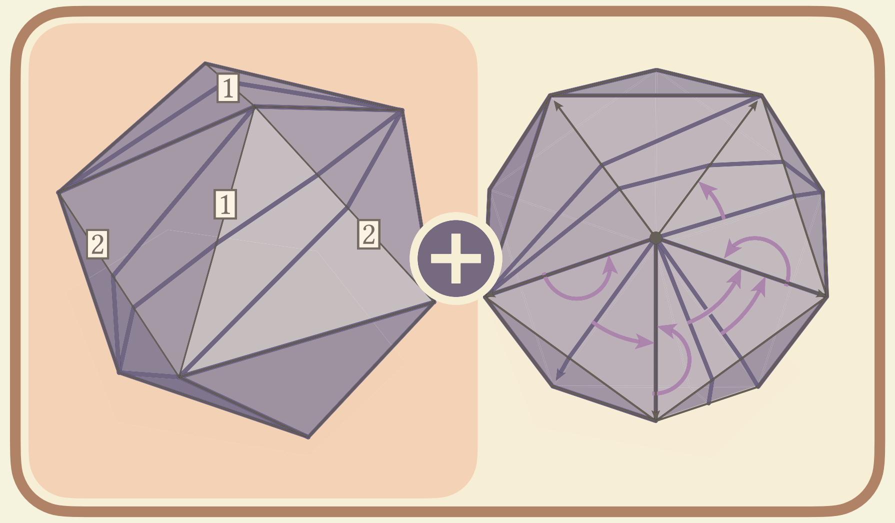
How much do normal coordinates tell us?


Integer coordinates for intrinsic triangulations

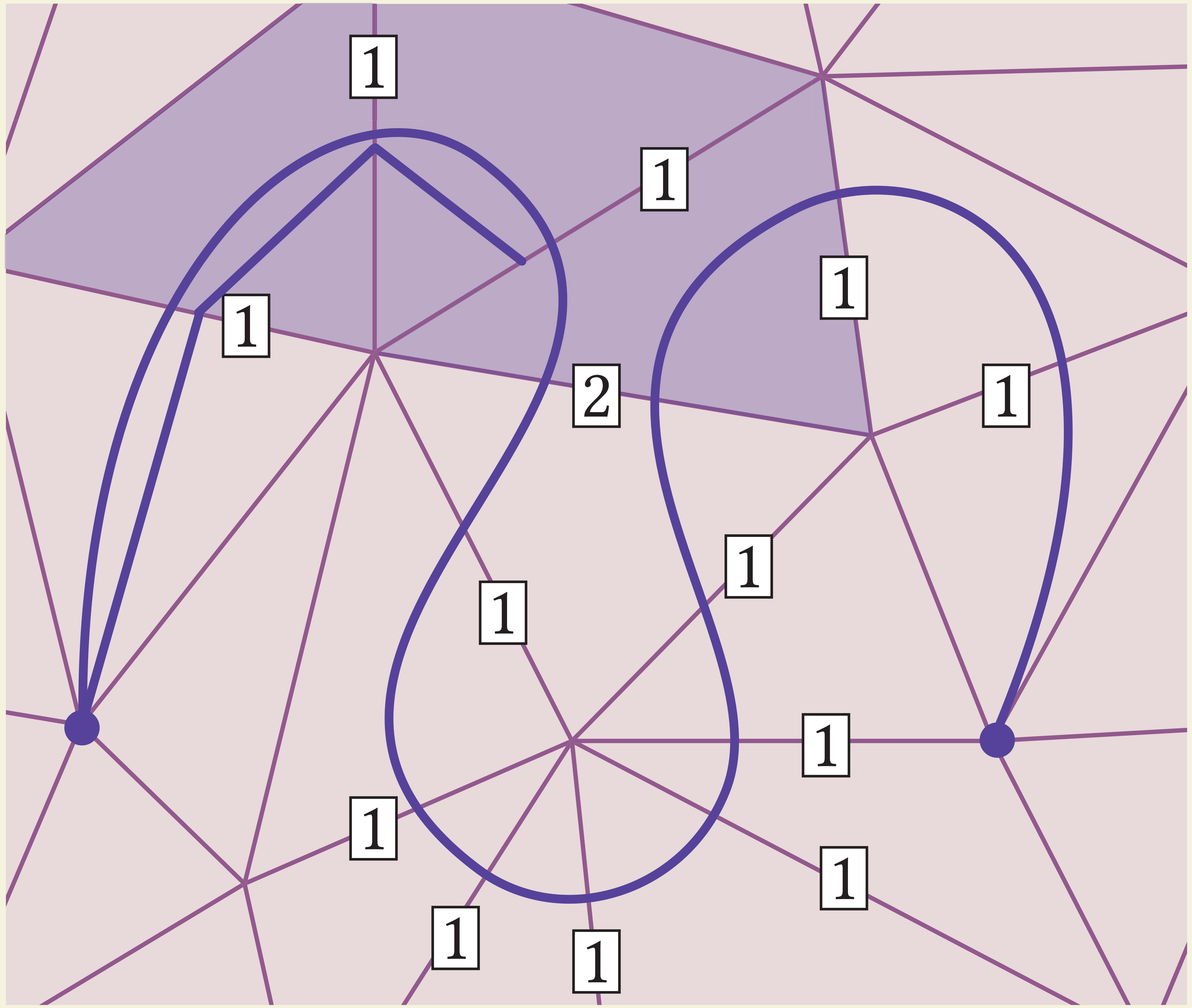
- Encodes sequence of triangles



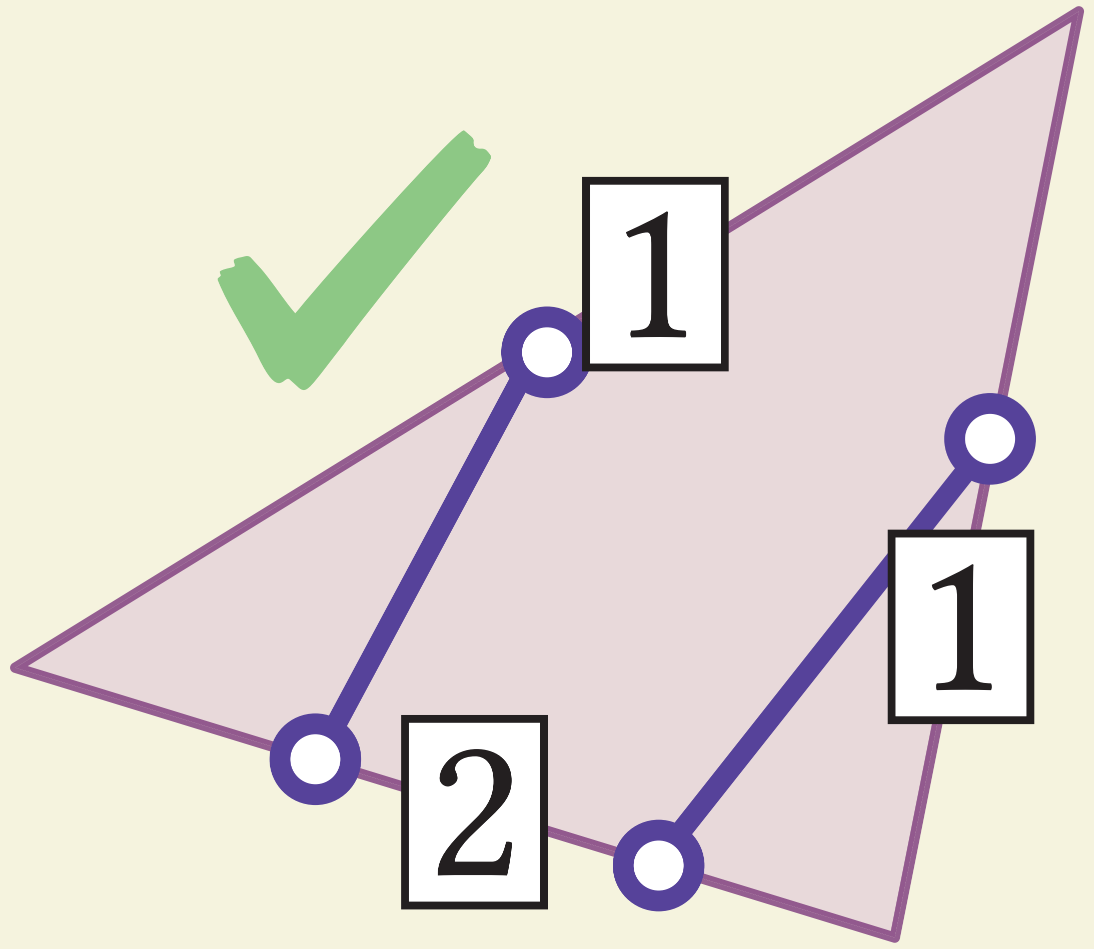
Reconstructing the curve

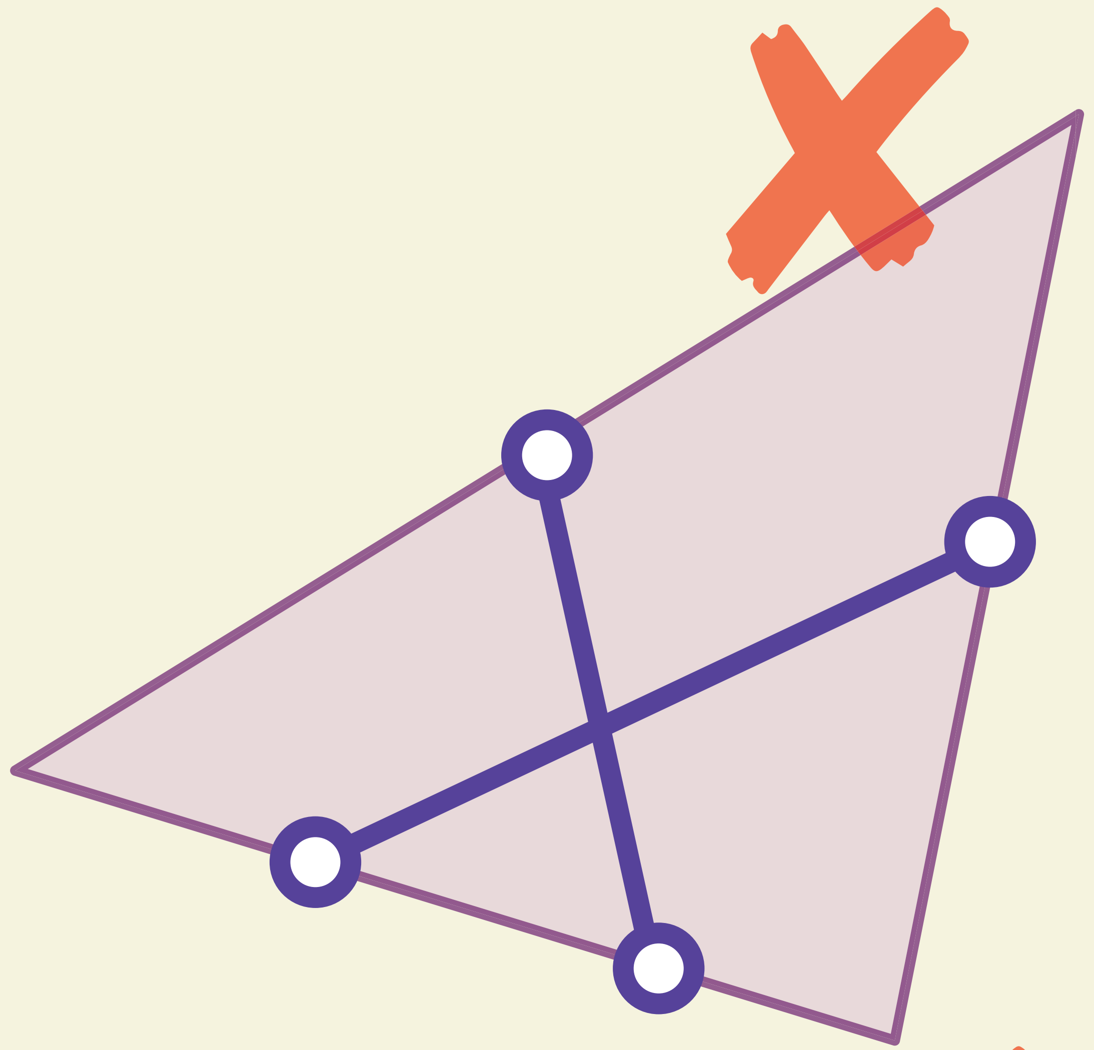


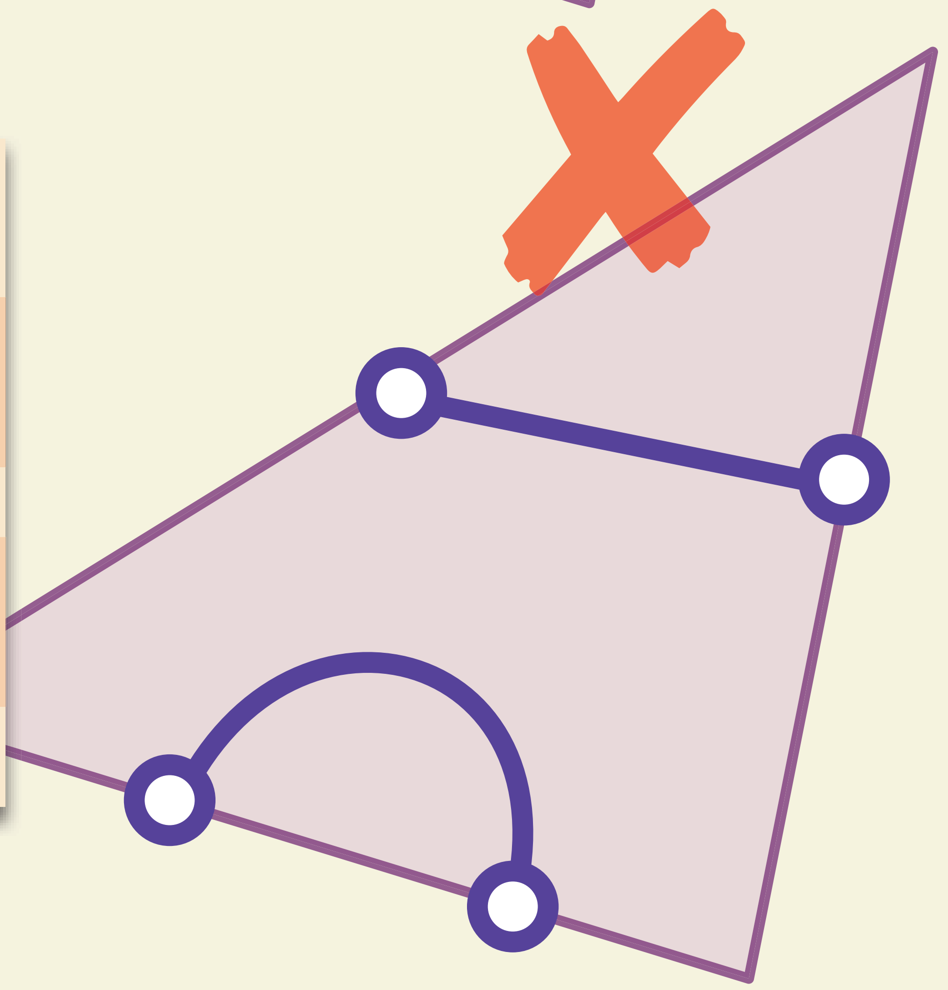
Integer coordinates for intrinsic triangulations



Reconstructing the curve

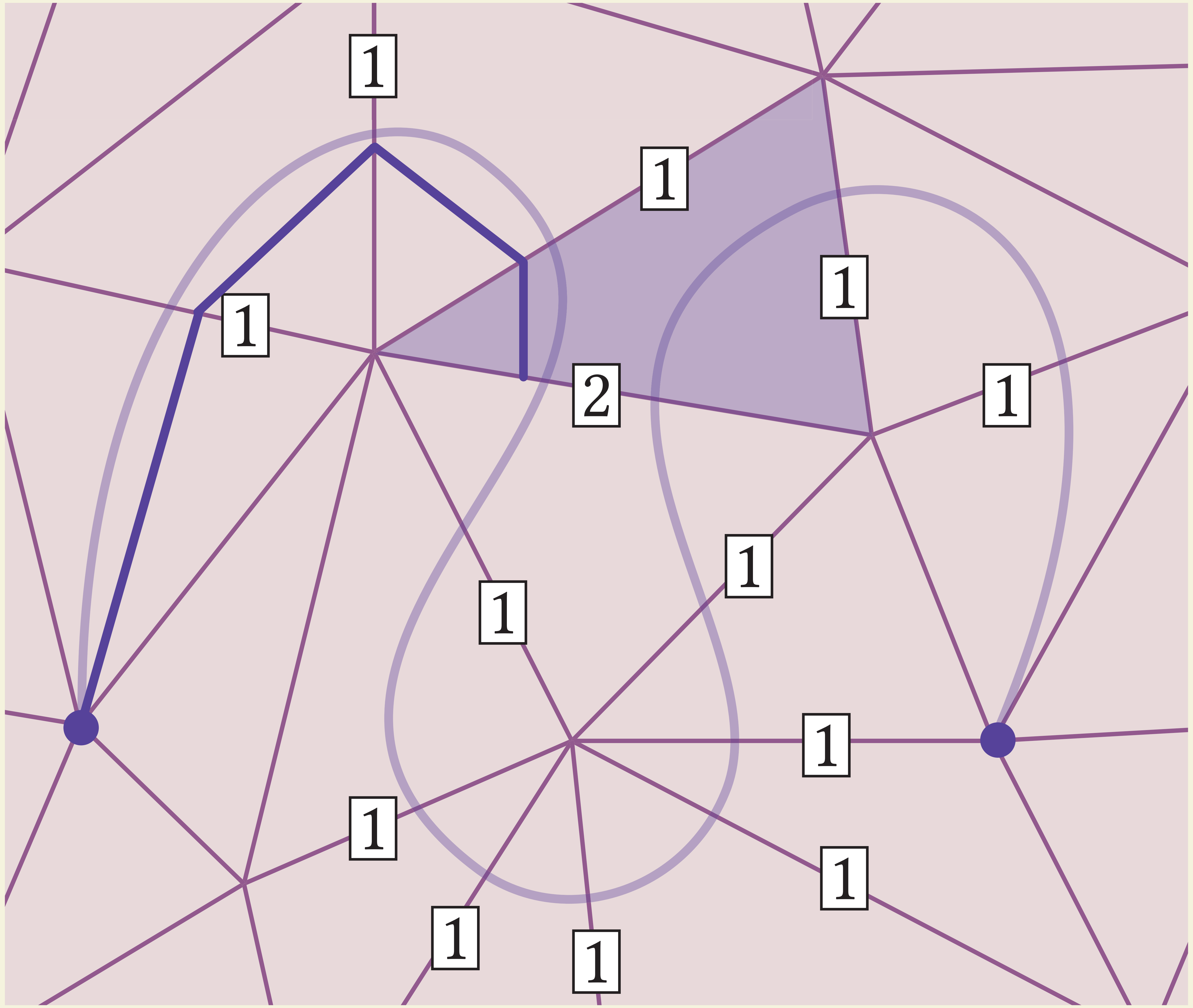


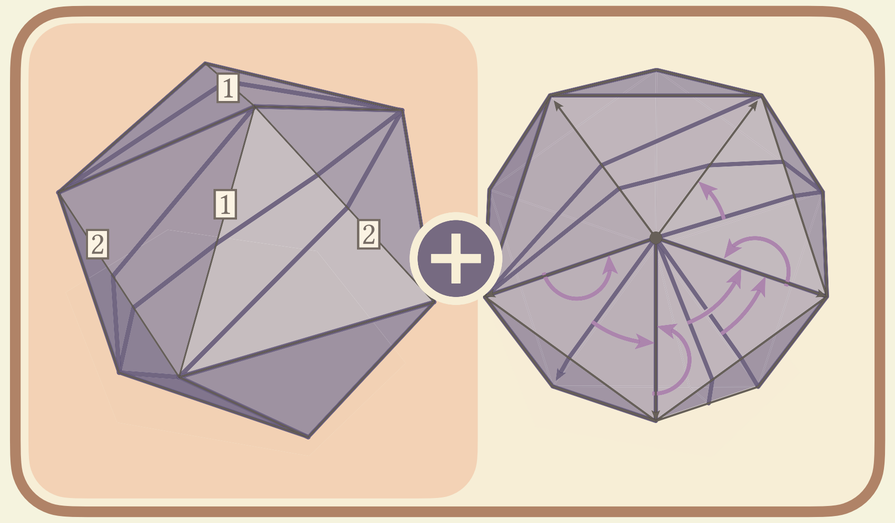




Rules

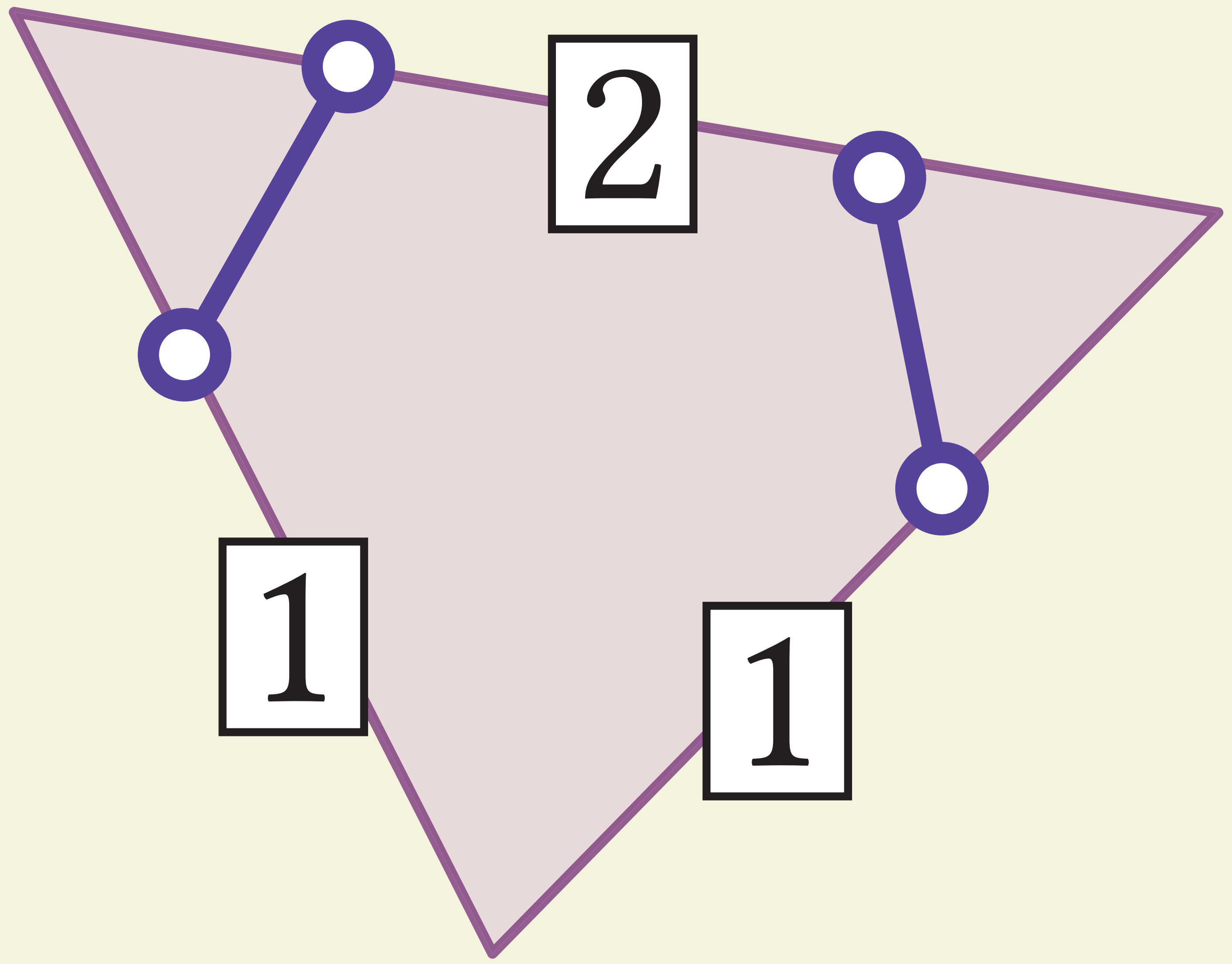
1. No self-crossings
2. No U-turns

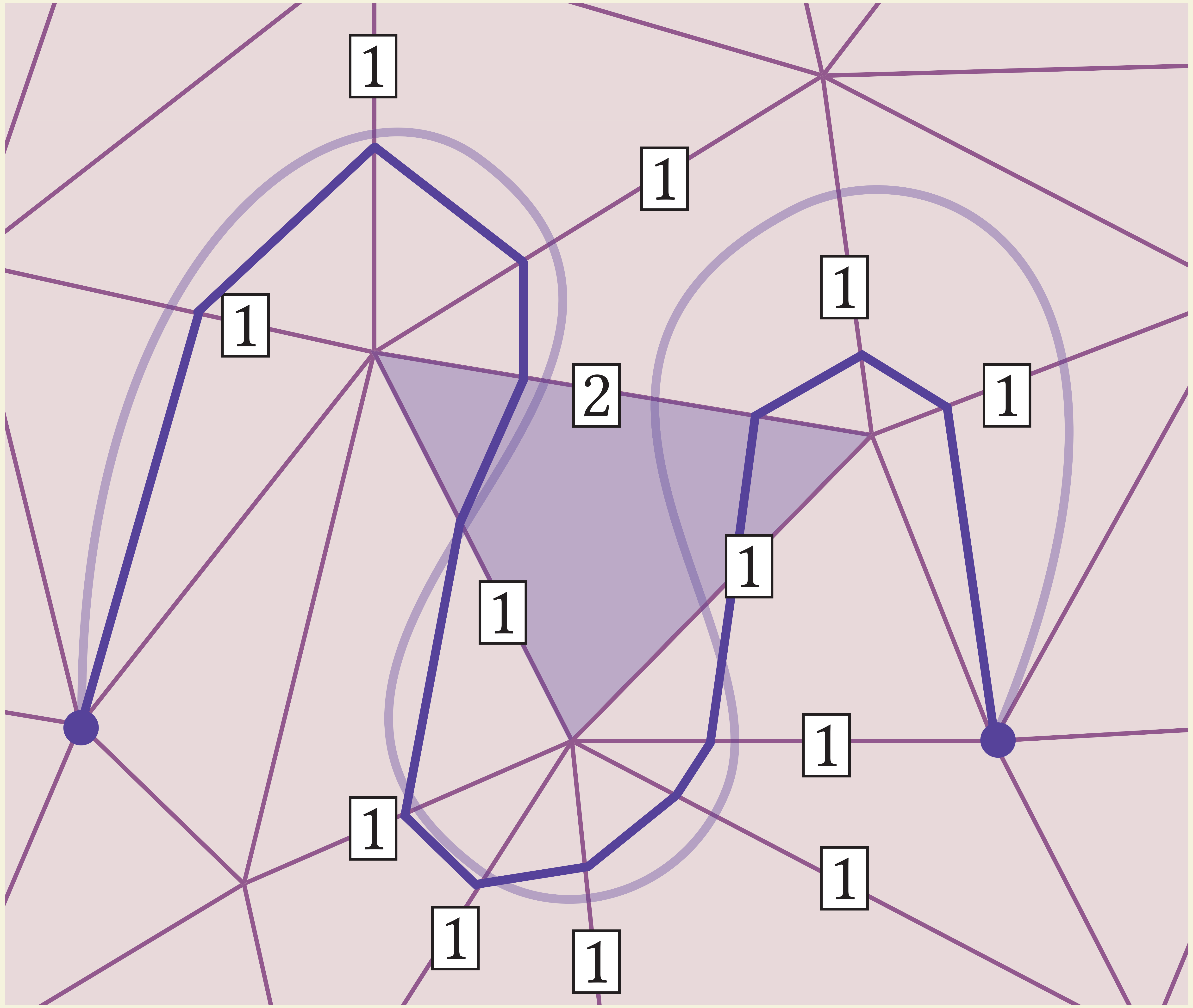


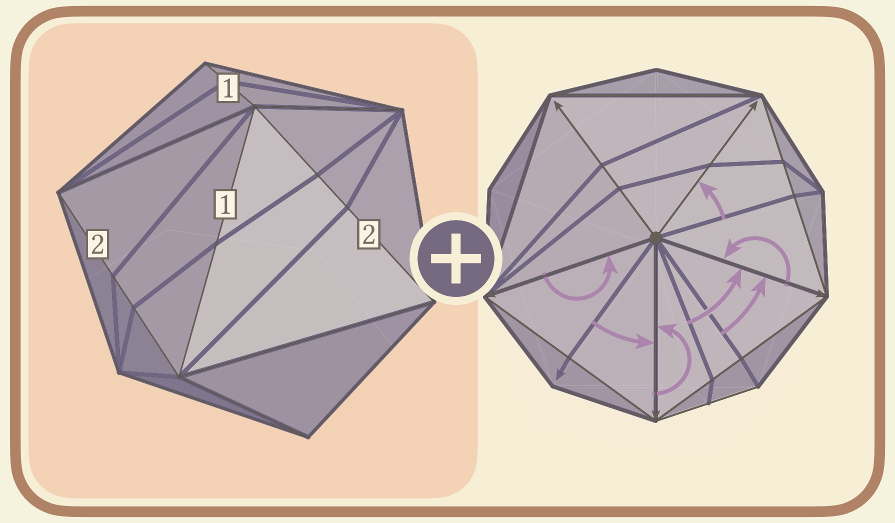


Integer coordinates for intrinsic triangulations

Reconstructing the curve

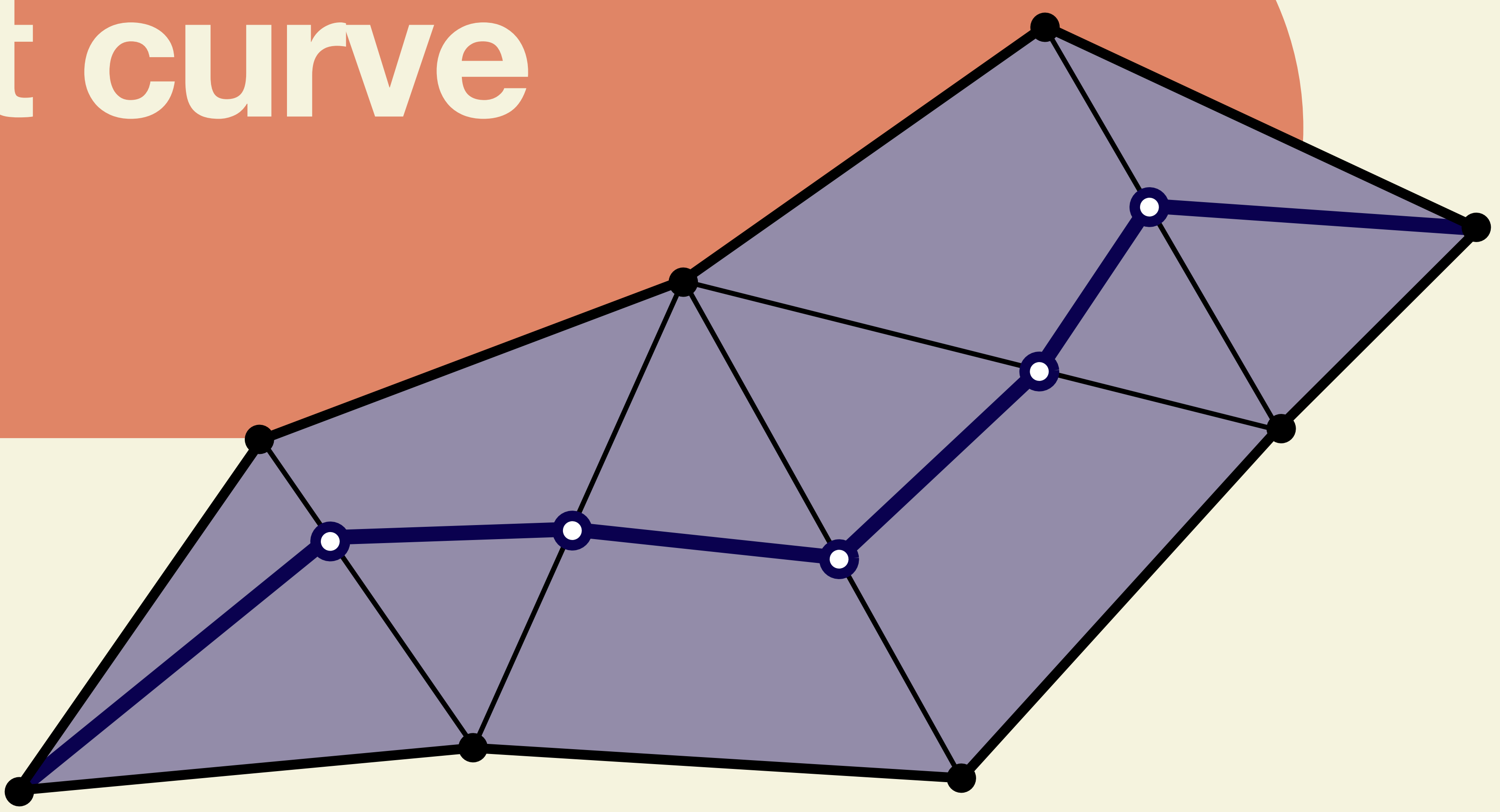




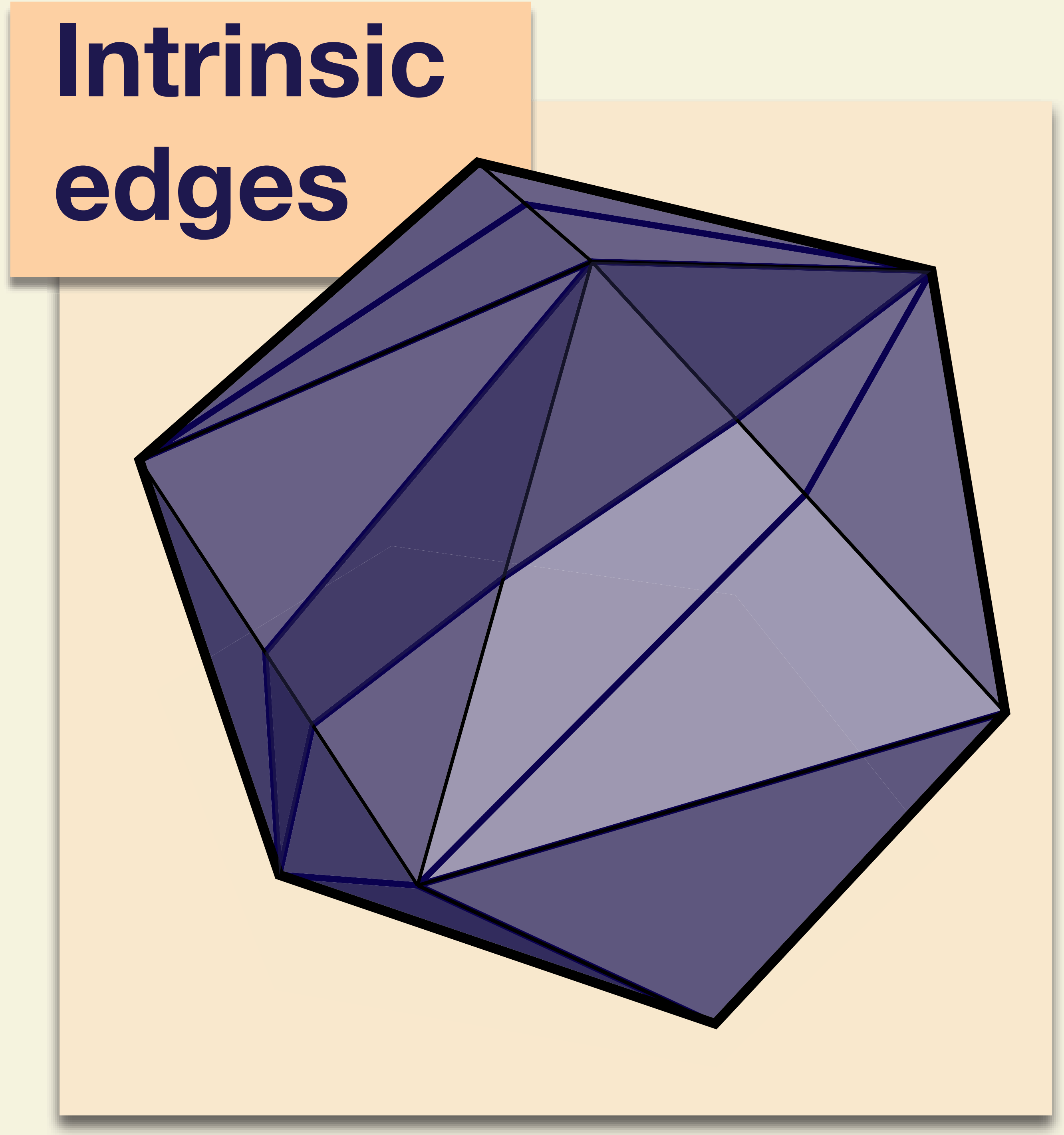


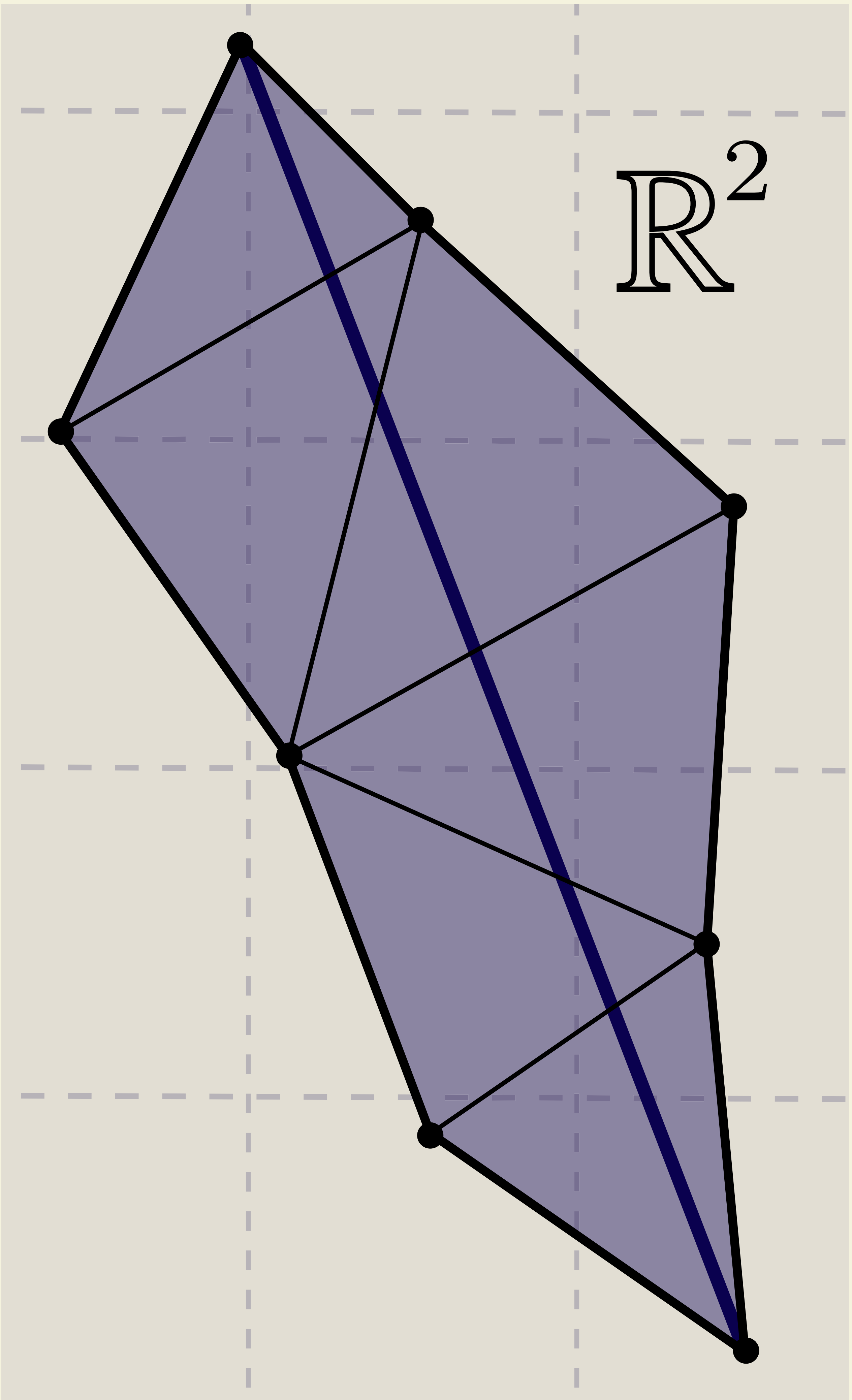
Integer coordinates for intrinsic triangulations

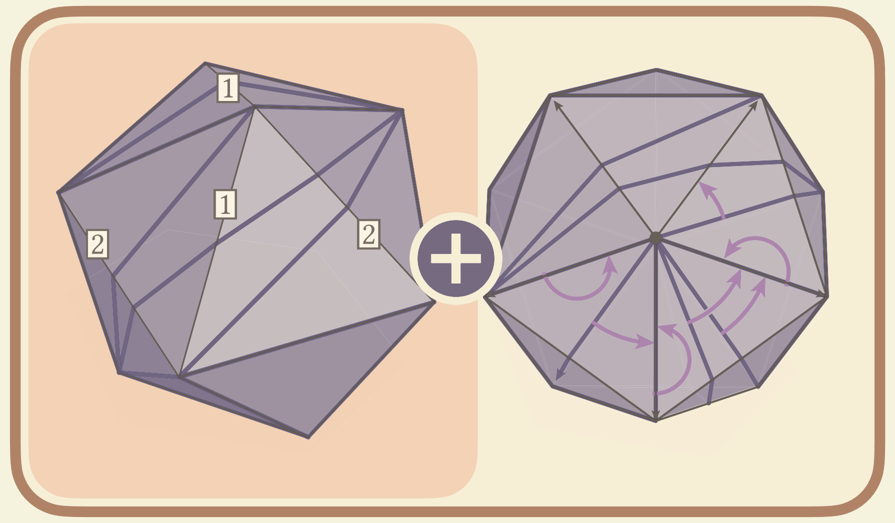
Finding the exact curve geometry



- So far: sequence of triangles
- True curve is a *straightest path*
  - Lay out in plane to find exact curve
- Normal coordinates determine edges exactly

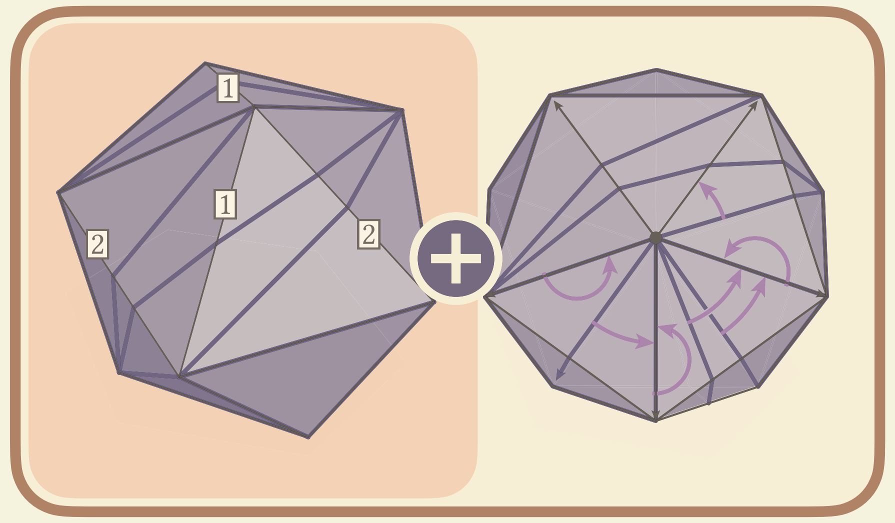






Integer coordinates for intrinsic triangulations

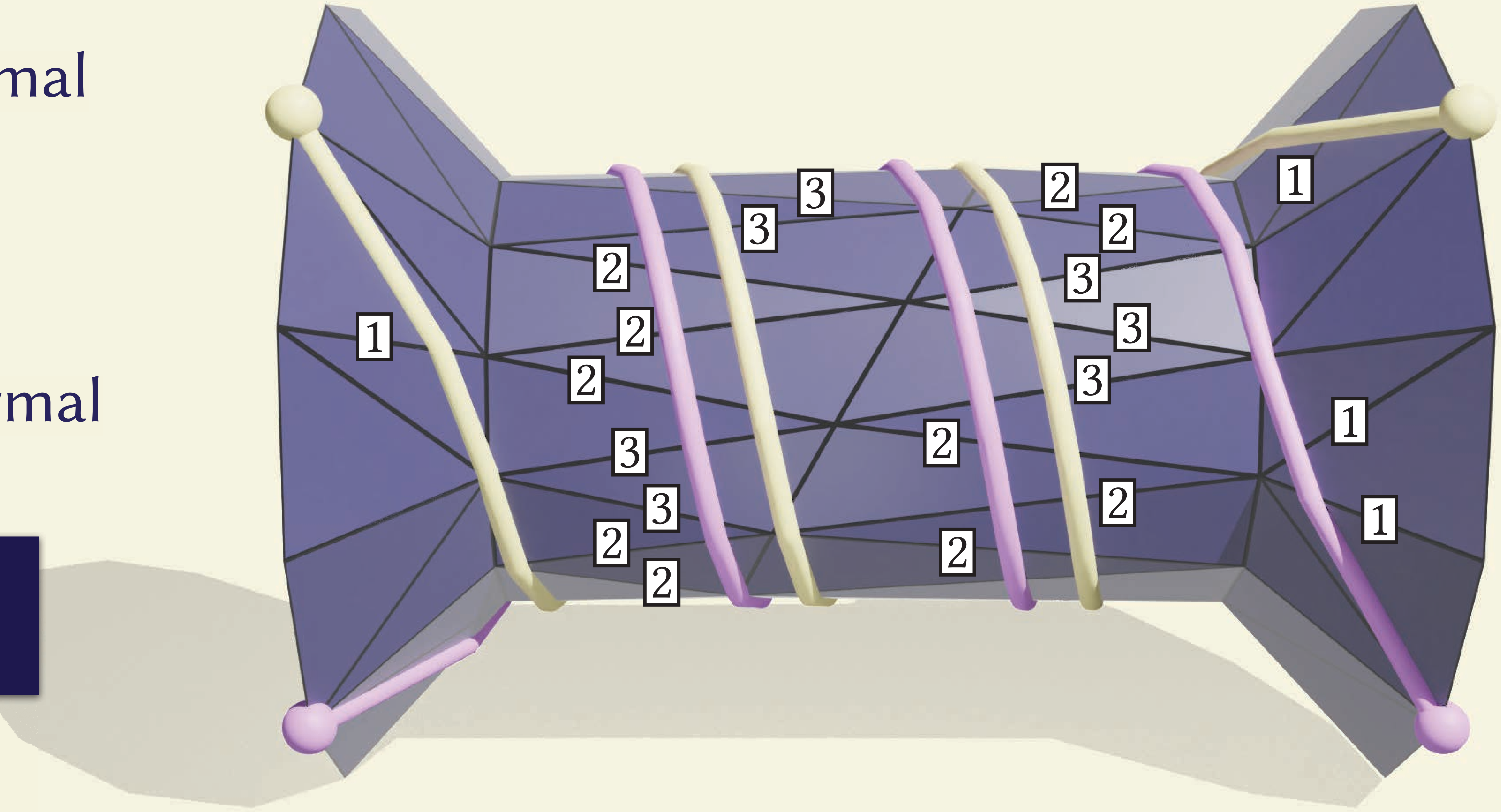
Collections of Curves



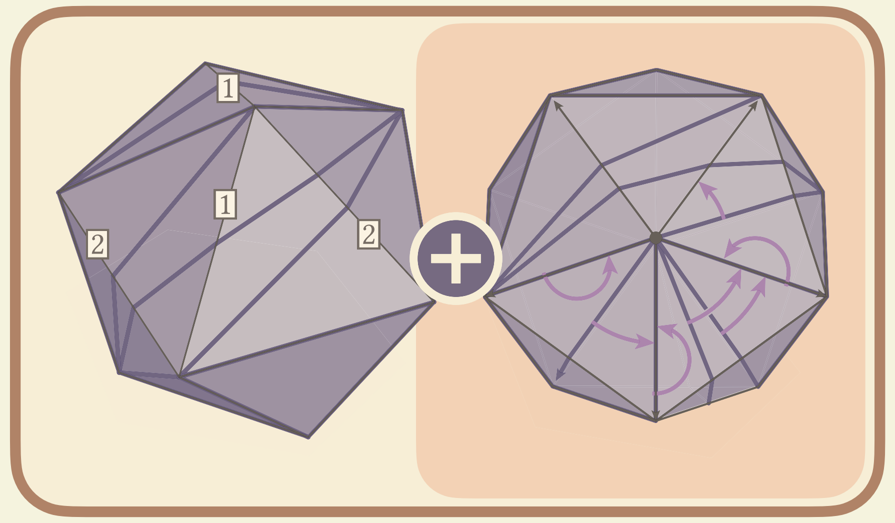
Integer coordinates for intrinsic triangulations

- *e.g.* edges of a triangulation
- Could store multiple sets of normal coordinates
  - Expensive ✗
- Instead, just store one set of normal coordinates

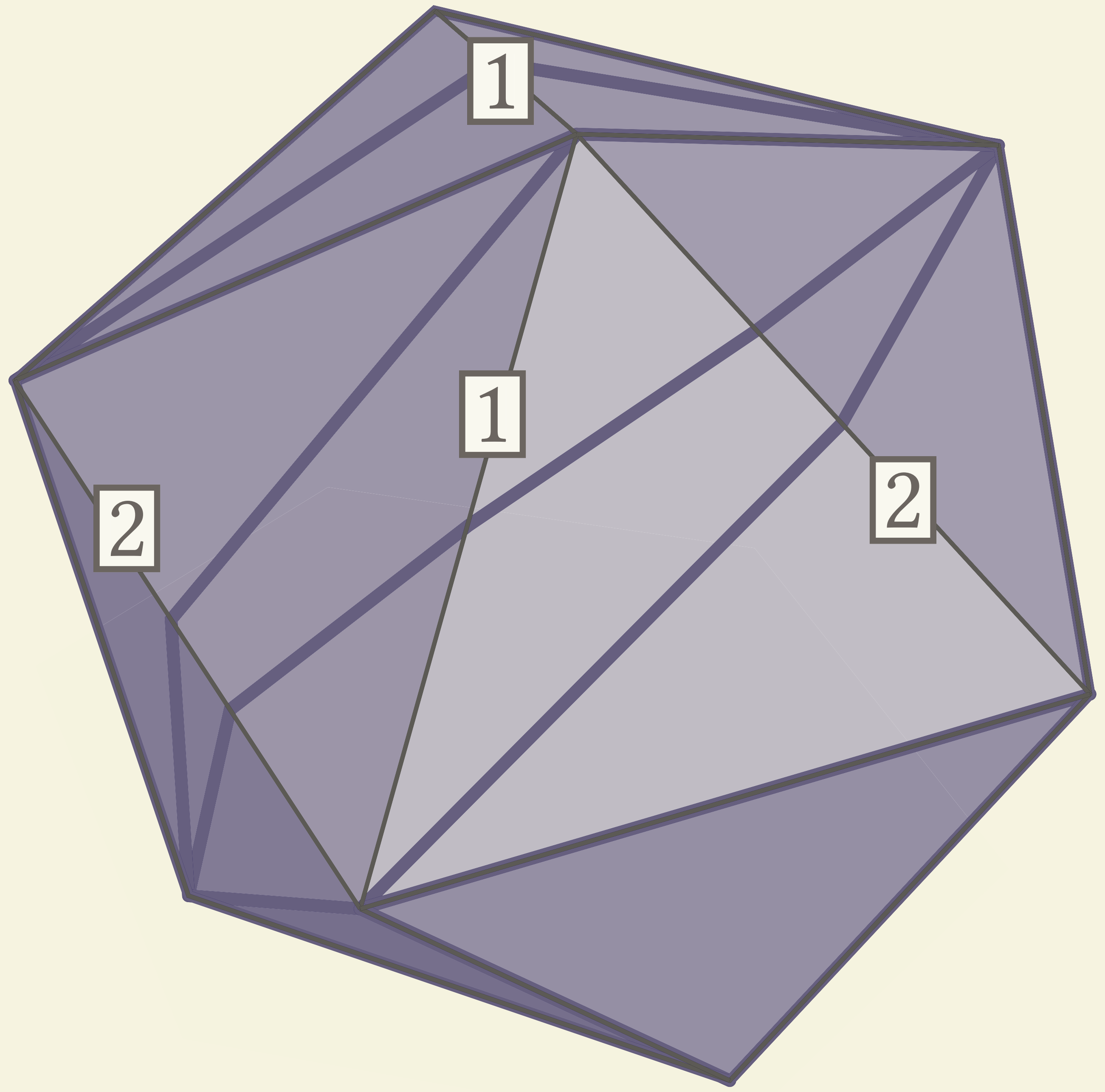
Store just one integer per edge



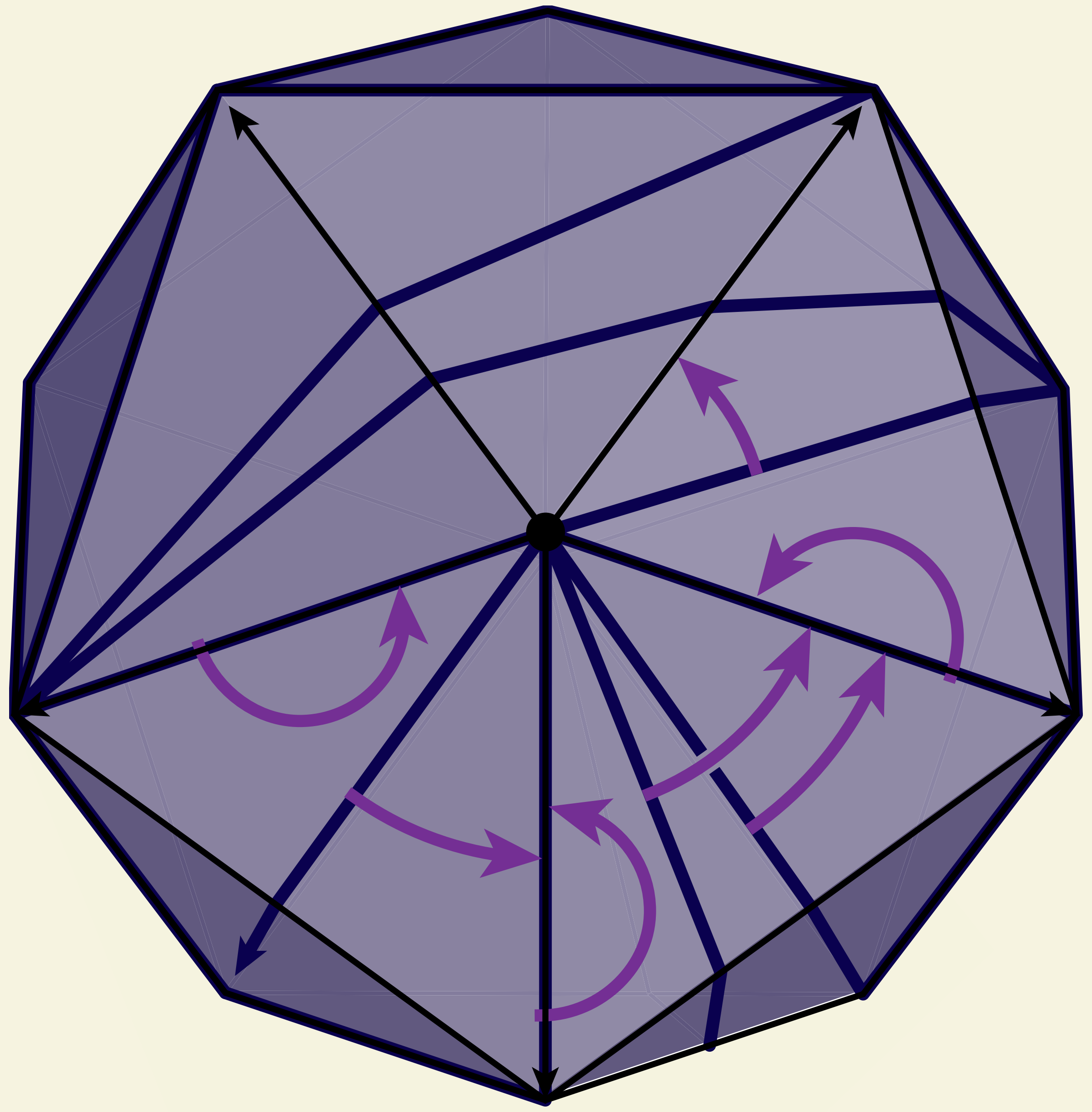
The integer coordinates data structure



Integer coordinates for intrinsic triangulations

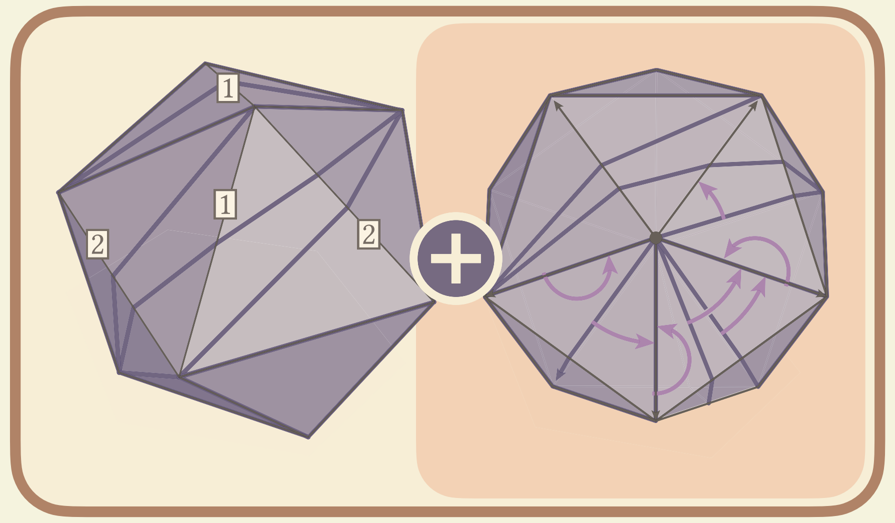


normal coordinates



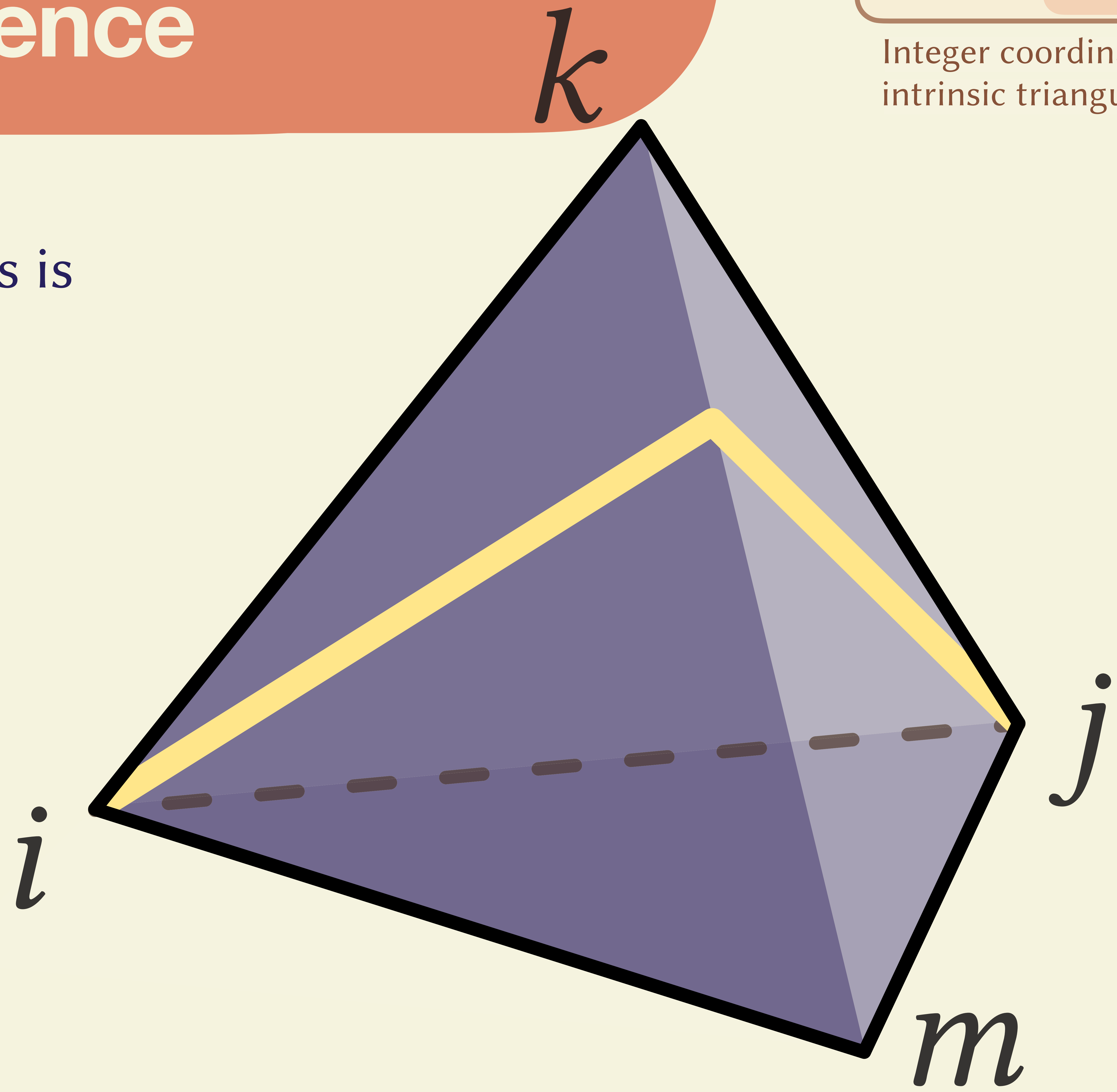
roundabouts

Normal coordinates are not enough to encode correspondence

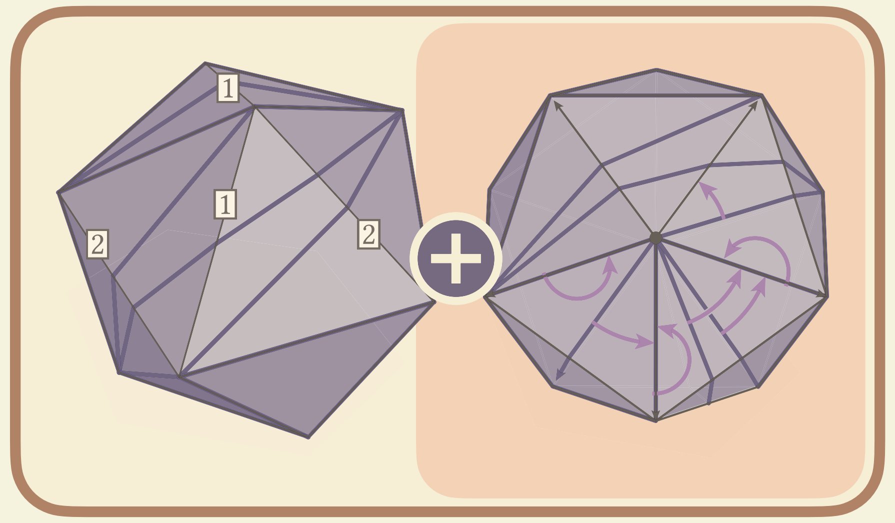


Integer coordinates for intrinsic triangulations

- Can’t immediately tell which edge this is

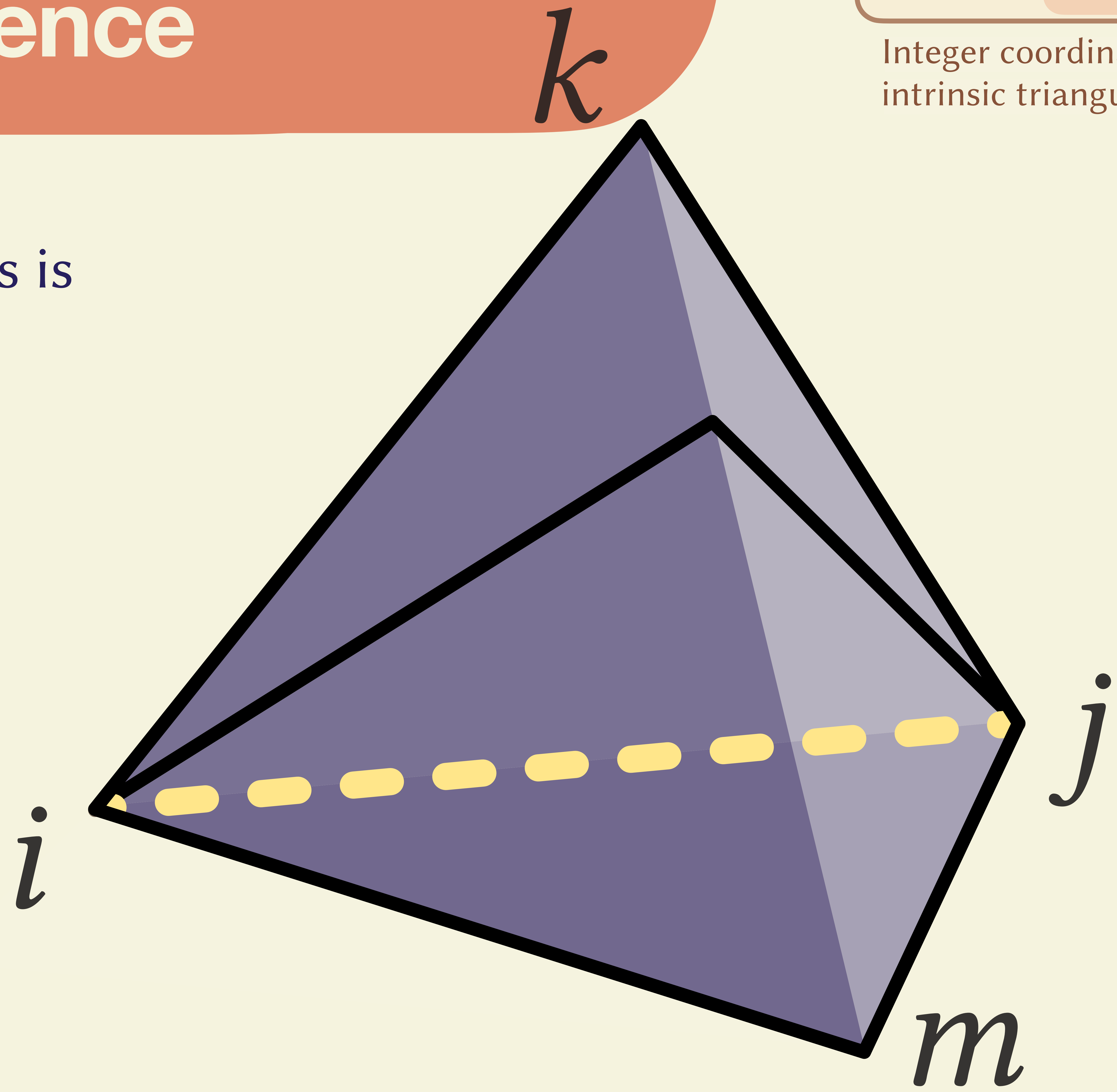


Normal coordinates are not enough to encode correspondence



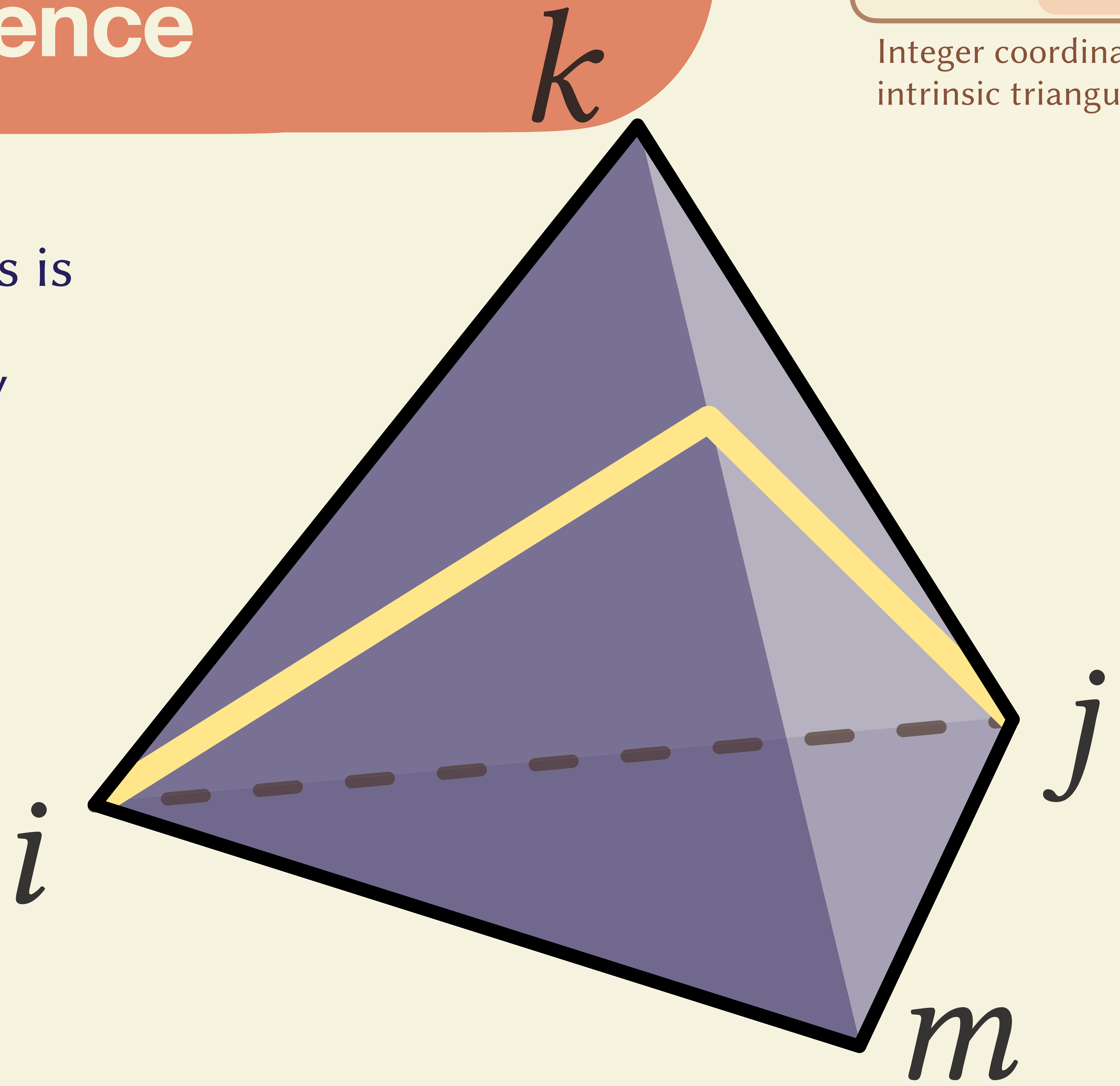
Integer coordinates for intrinsic triangulations

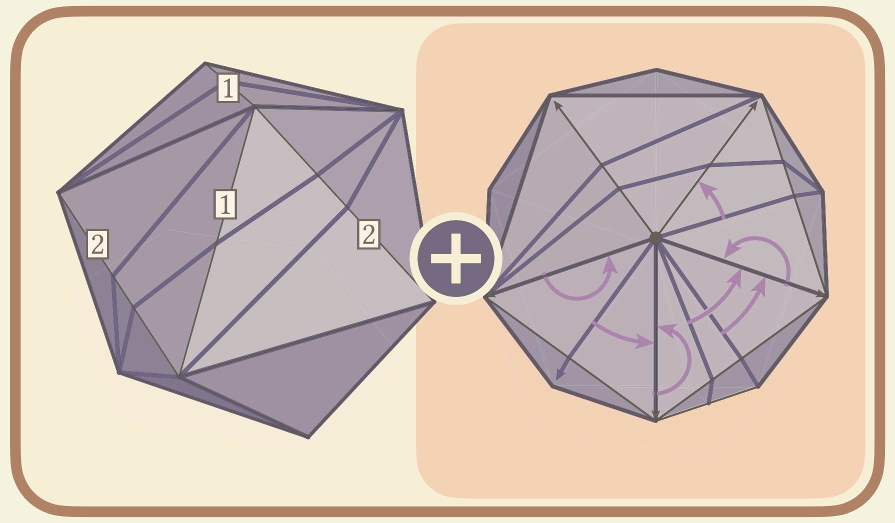
- Can't immediately tell which edge this is



Normal coordinates are not enough to encode correspondence

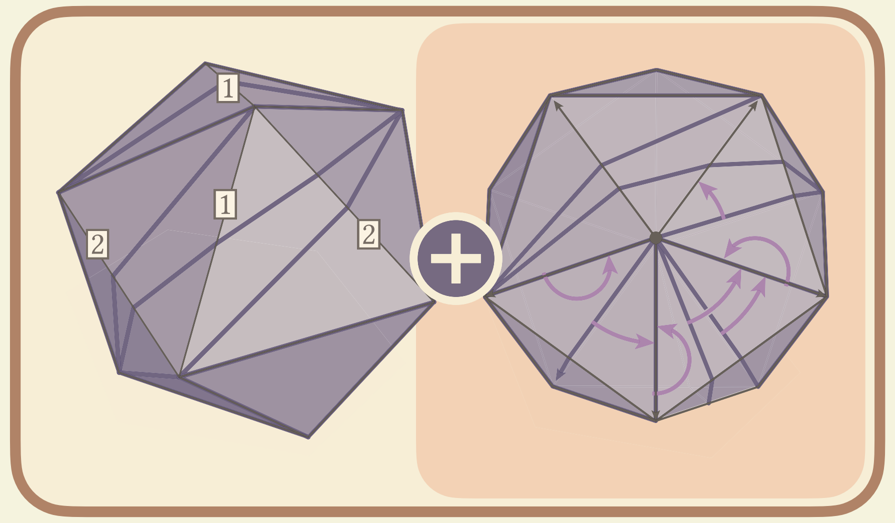
- Can’t immediately tell which edge this is
  - Roundabouts resolve this ambiguity



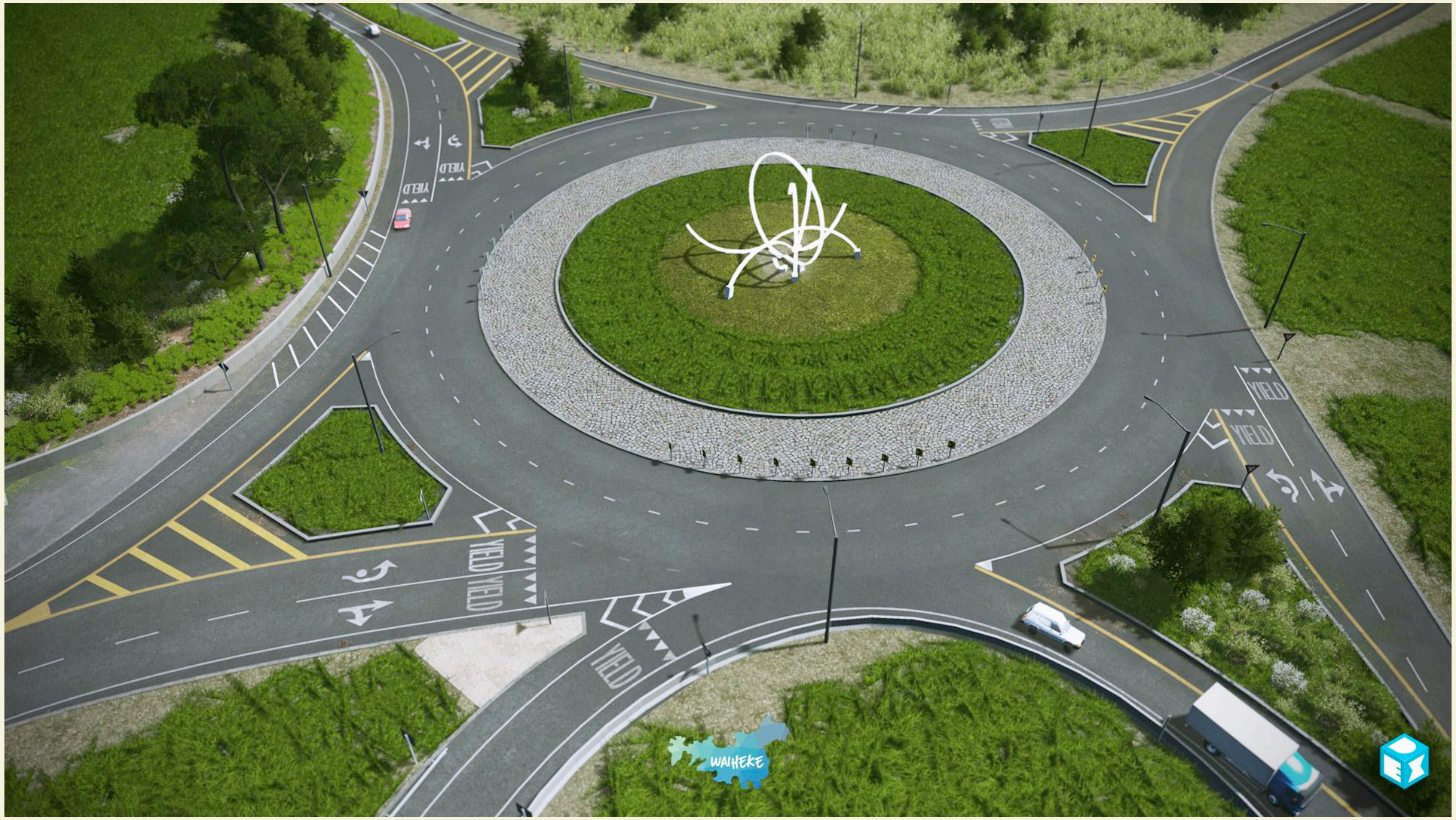


Integer coordinates for intrinsic triangulations

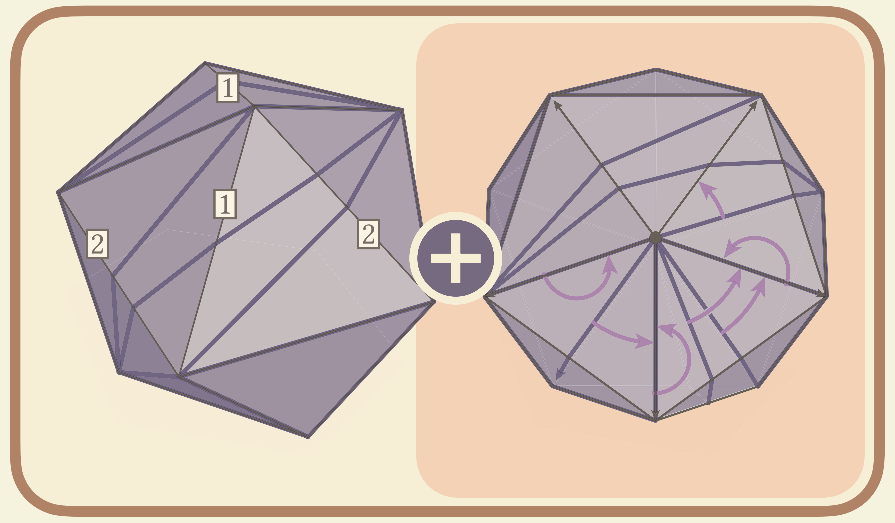
Roundabouts



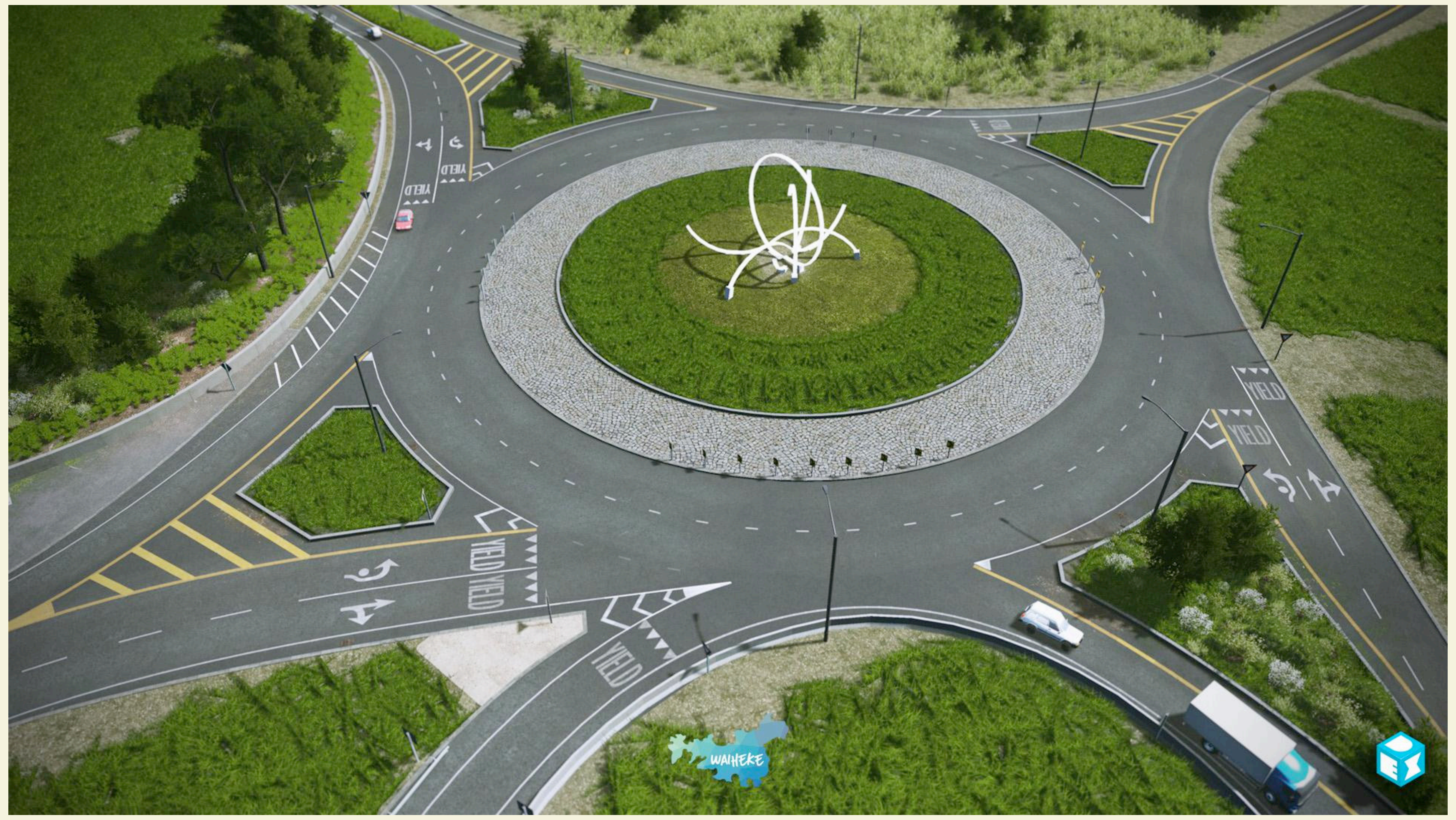
Integer coordinates for intrinsic triangulations

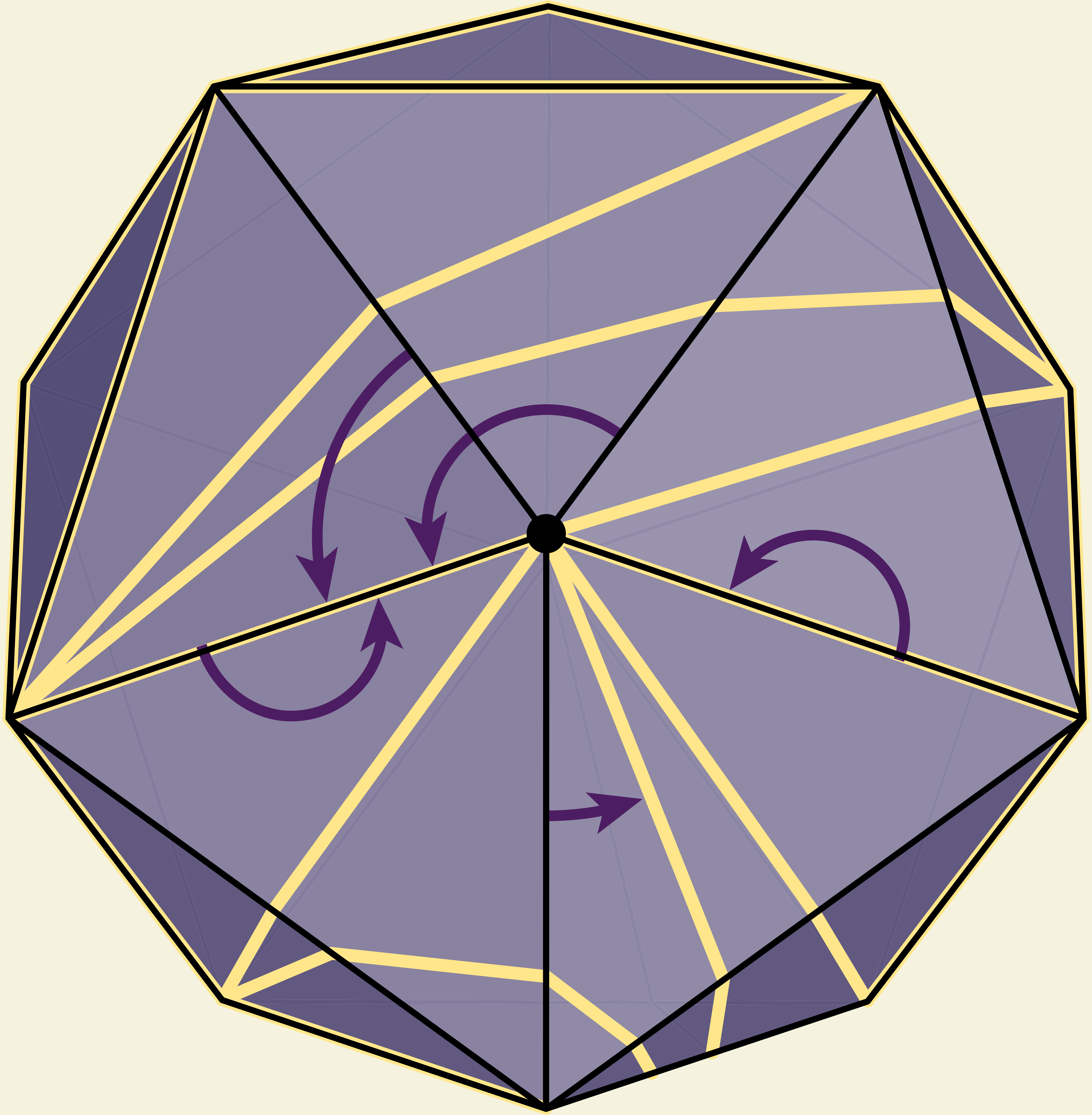


Roundabouts

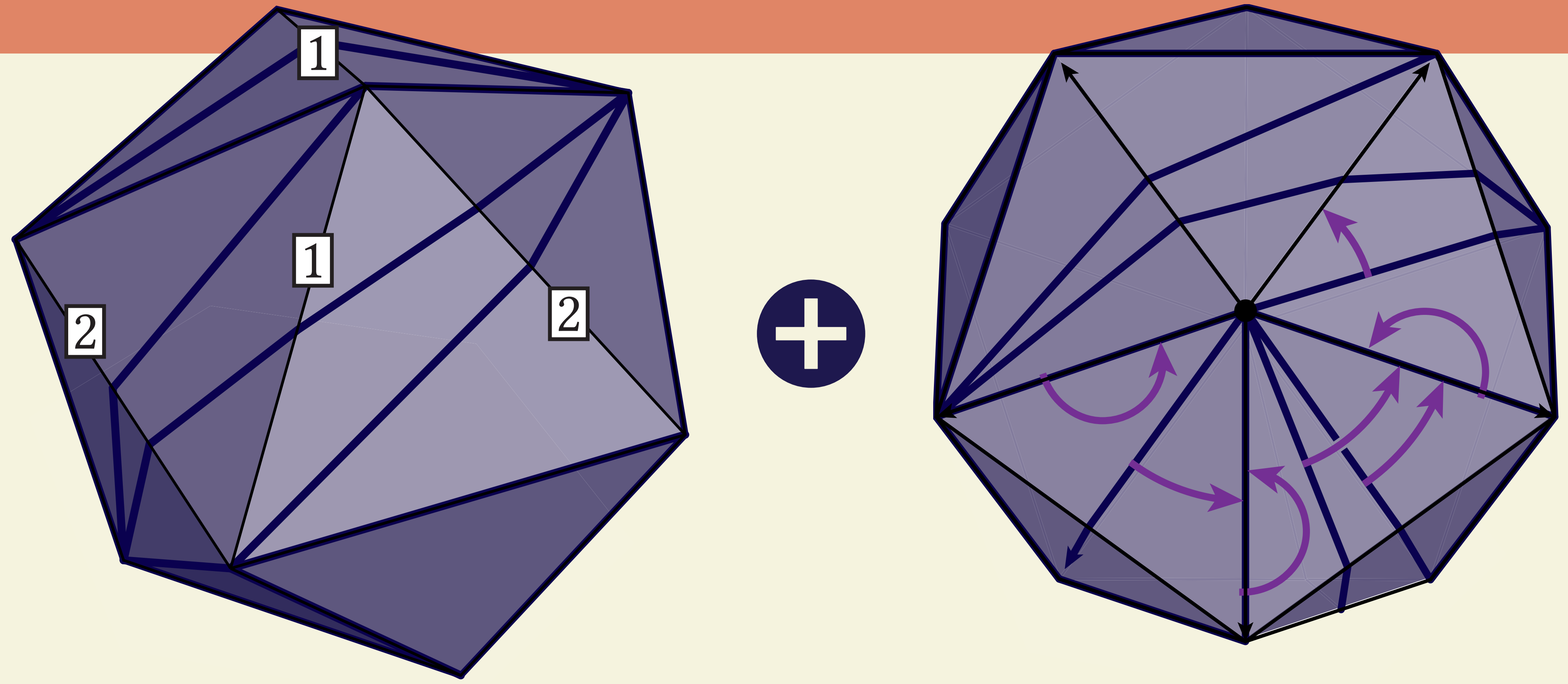


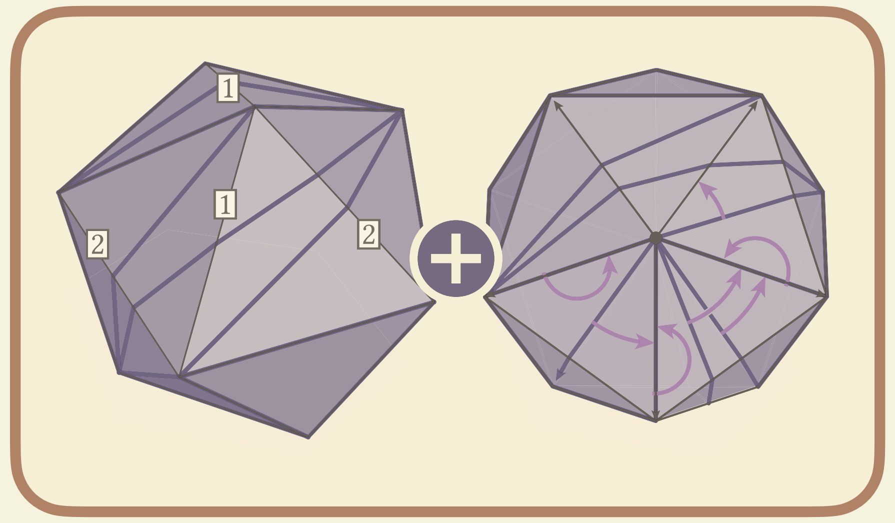
- Record how edges interleave
- Store pointer to next edge
- Resolves all ambiguity





Data structure operations

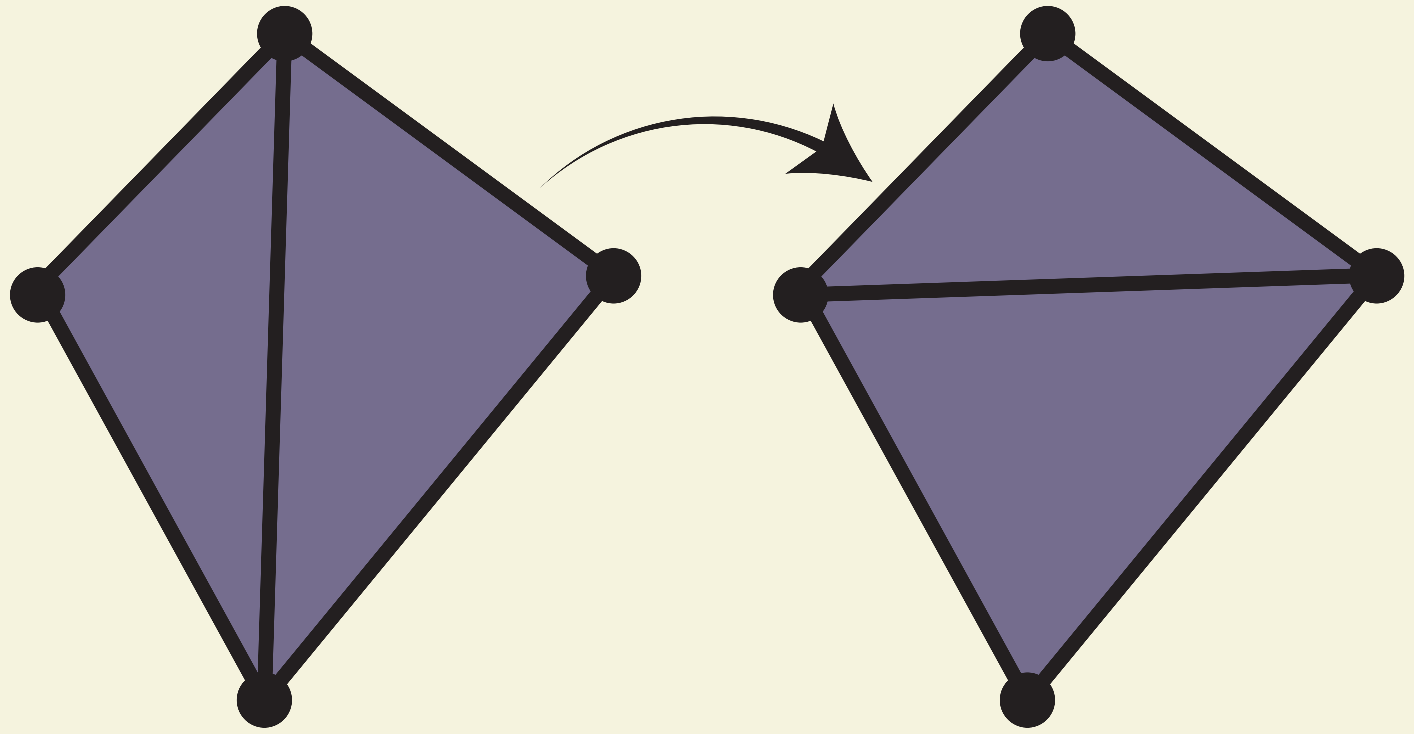




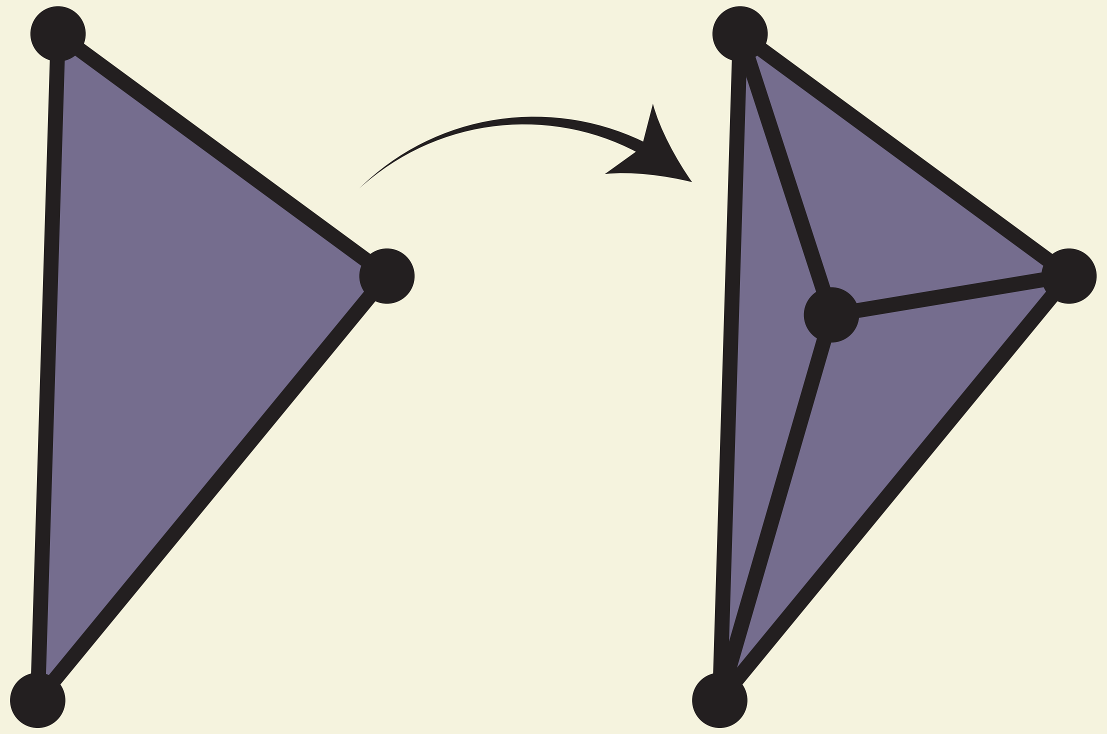
Integer coordinates for intrinsic triangulations

a complete description of correspondence

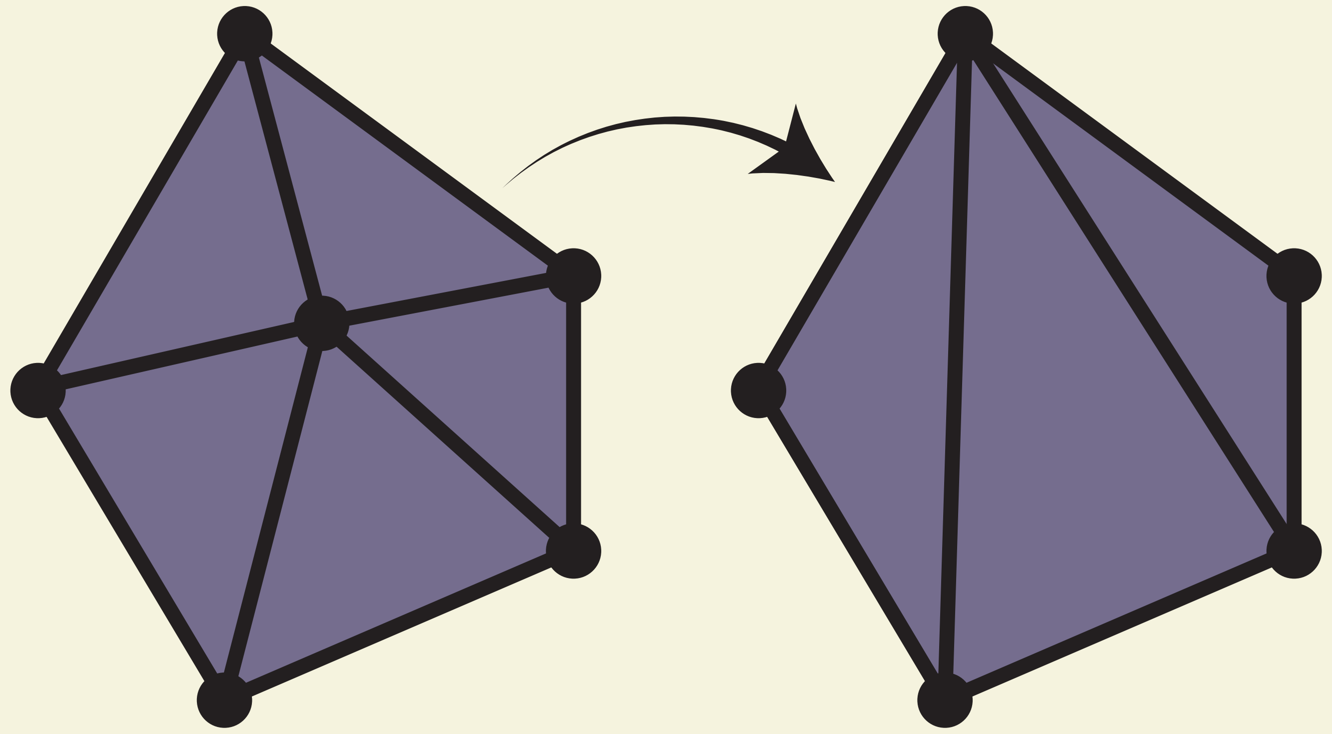
- Supports a variety of connectivity changes:



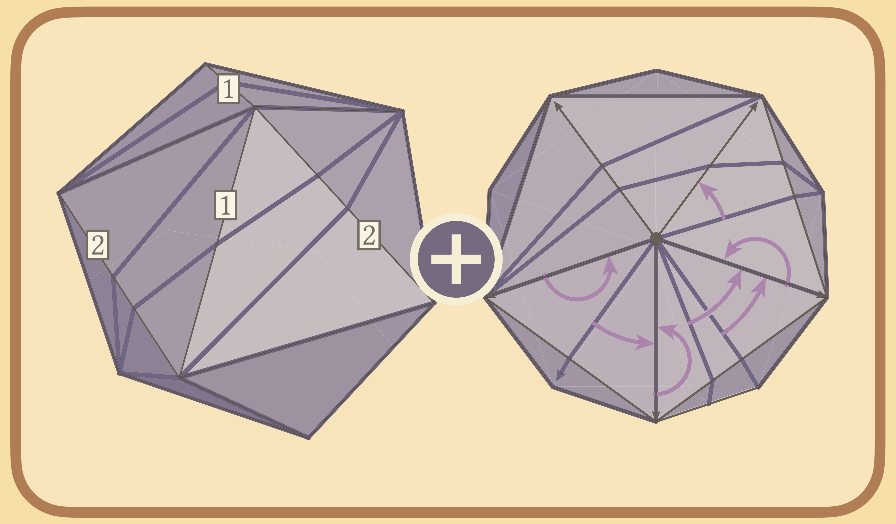
edge flips



vertex insertion

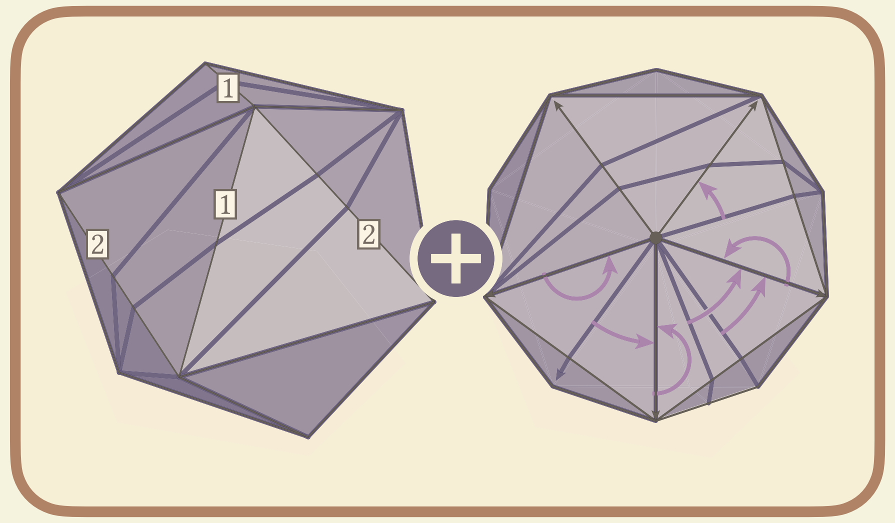


flat vertex removal



Integer coordinates for intrinsic triangulations

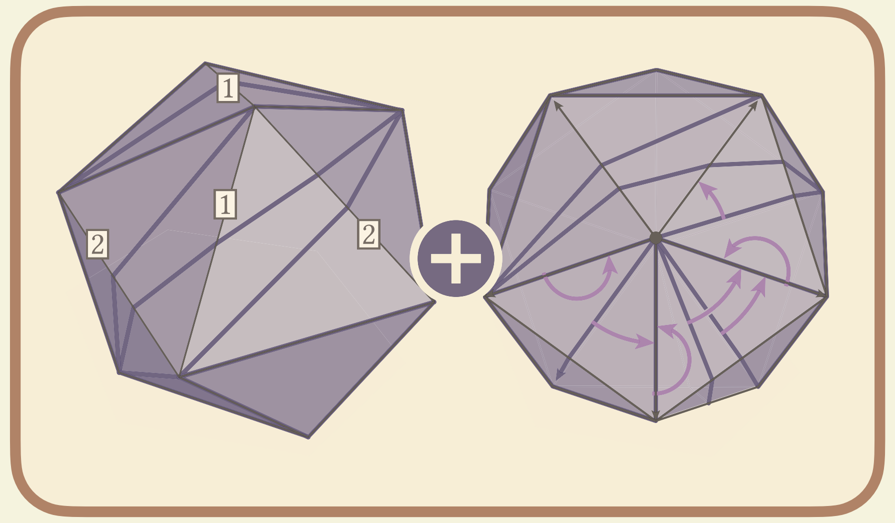
Applications



Intrinsic Delaunay refinement

- Recall: improve mesh quality by inserting vertices

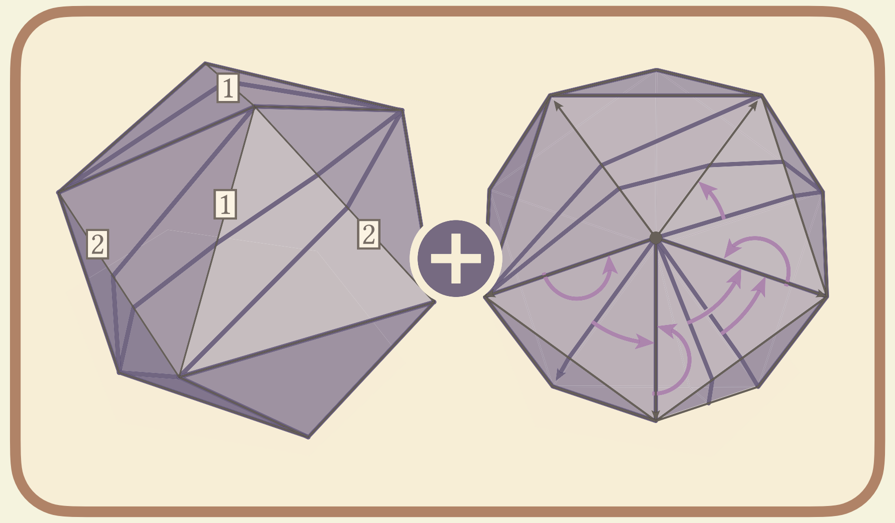
Common subdivisions of intrinsic Delaunay refinements



Integer coordinates for intrinsic triangulations
‣ *applications*

- Integer coordinates can be crucial to recovering the common subdivision

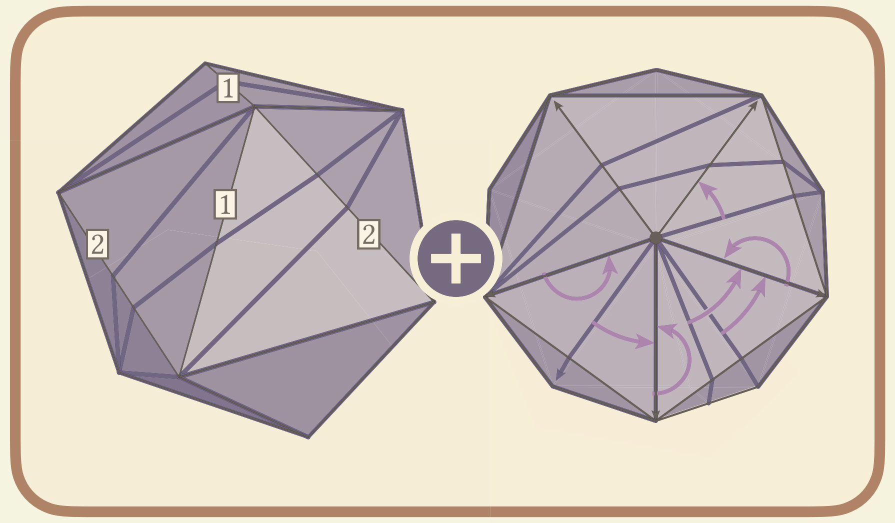
Intrinsic Delaunay refinement — validation



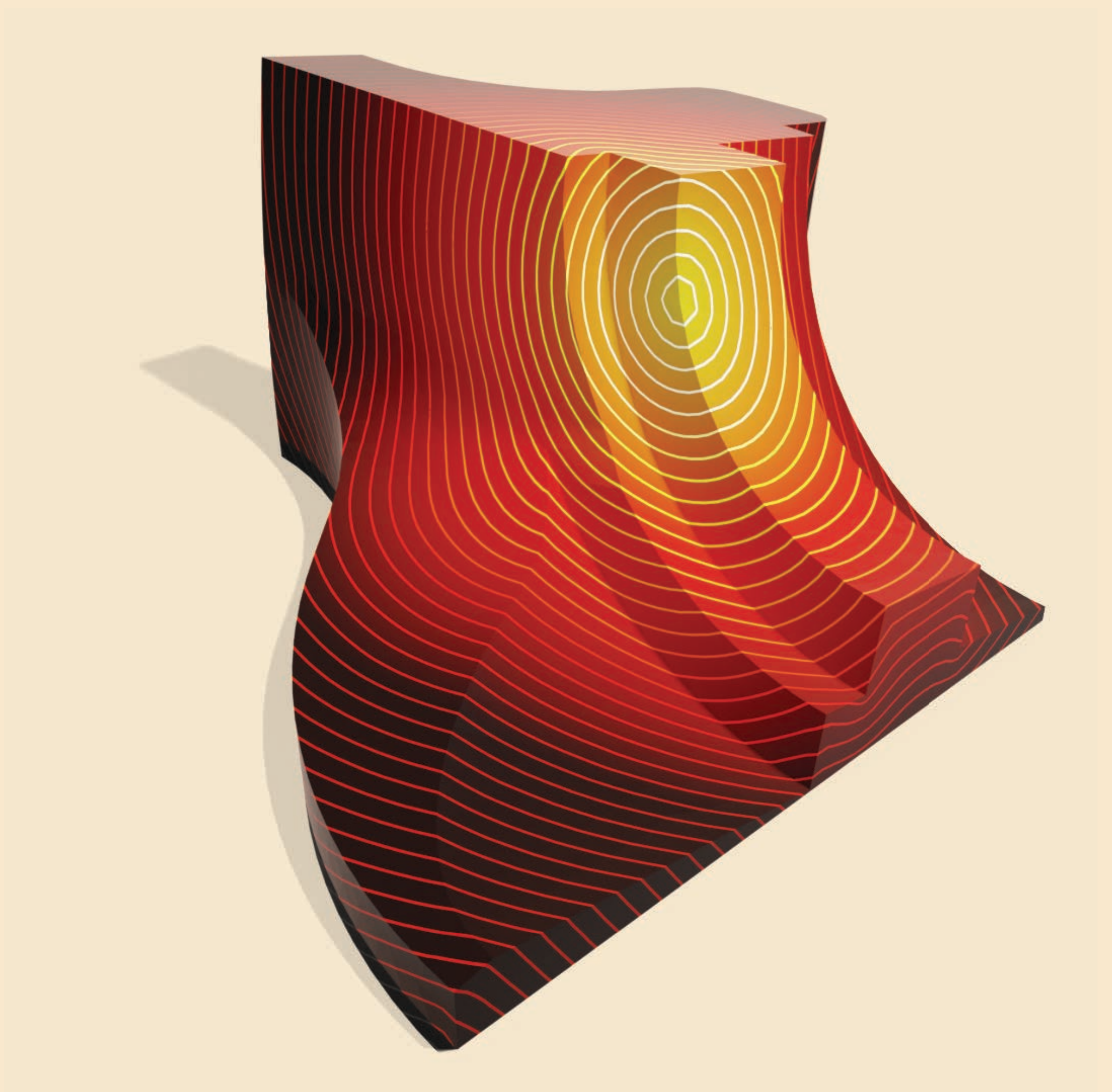
Integer coordinates for intrinsic triangulations
▸ *applications*

- Compute refinements & common subdivisions for Thingi10k dataset [Zhou & Jacobson 2016]
  - 7696 manifold meshes
- < 1s on most meshes; only took > 1m on 6 meshes
- 100% success rate for refinement & common subdivision
  - [Sharp, Soliman & Crane 2019] succeed on only 69.1% of meshes

Application: PDE-Based Geometry Processing



Integer coordinates for intrinsic triangulations
▸ *applications*



heat method for distance along surface
[Crane, Weischedel & Wardetzky 2013]

mean error: 28%

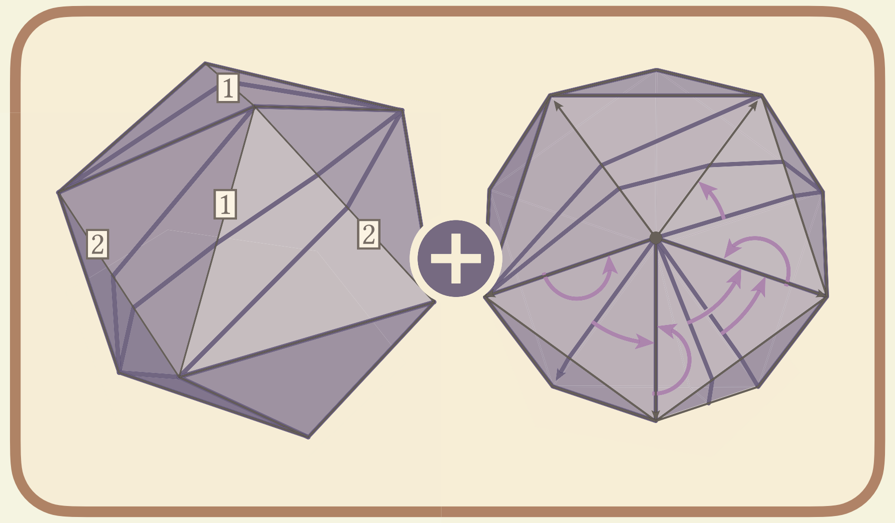
result on input mesh

mean error: 2%

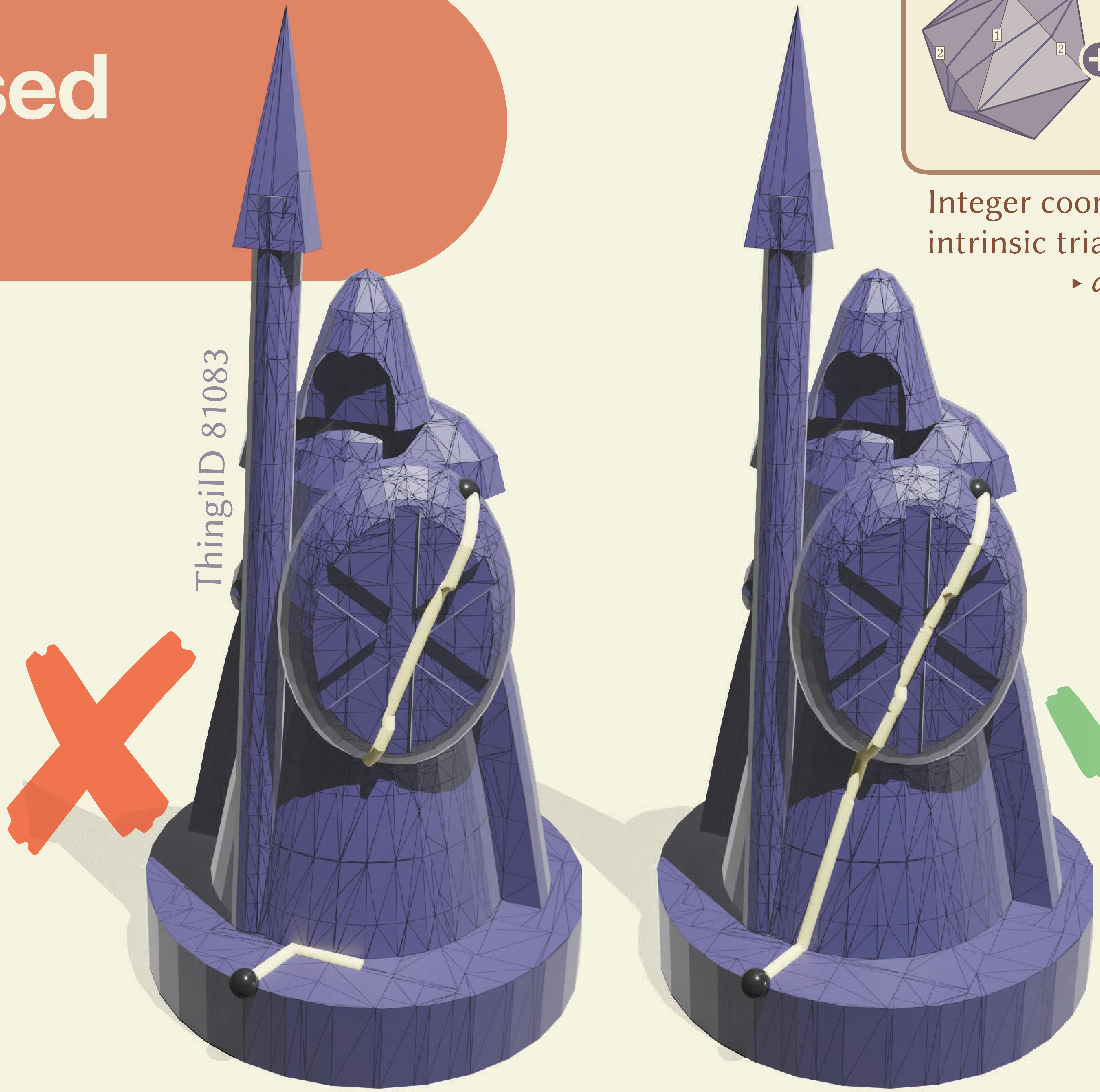
result on Delaunay refinement

Application: Flip-Based Straightest Paths

- FlipOut [Sharp & Crane 2020]:
  - computes straightest paths via edge flips



Integer coordinates for intrinsic triangulations
▸ *applications*



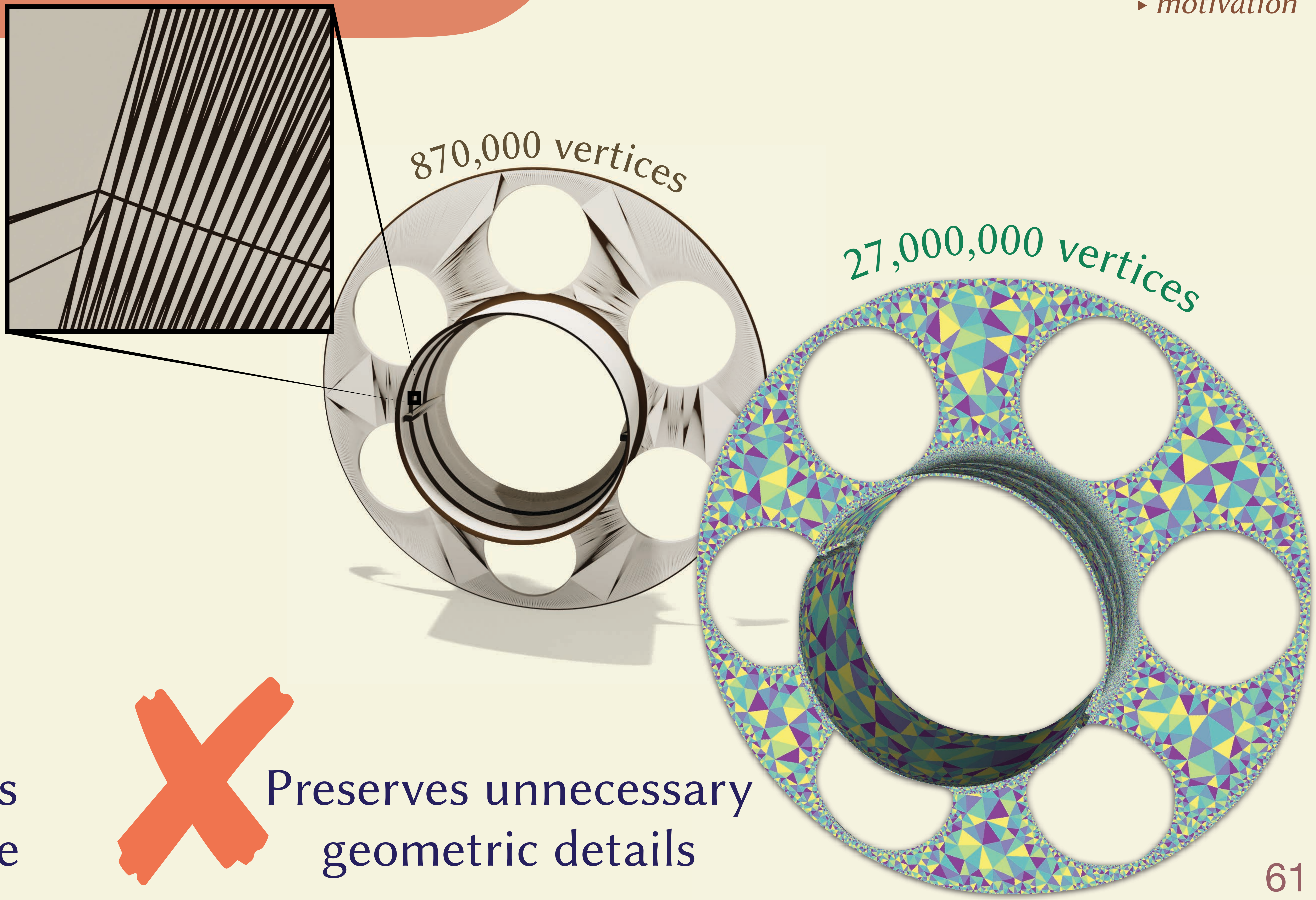
[Sharp, Soliman & Crane 2019]

[Integer coordinates]

III. 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

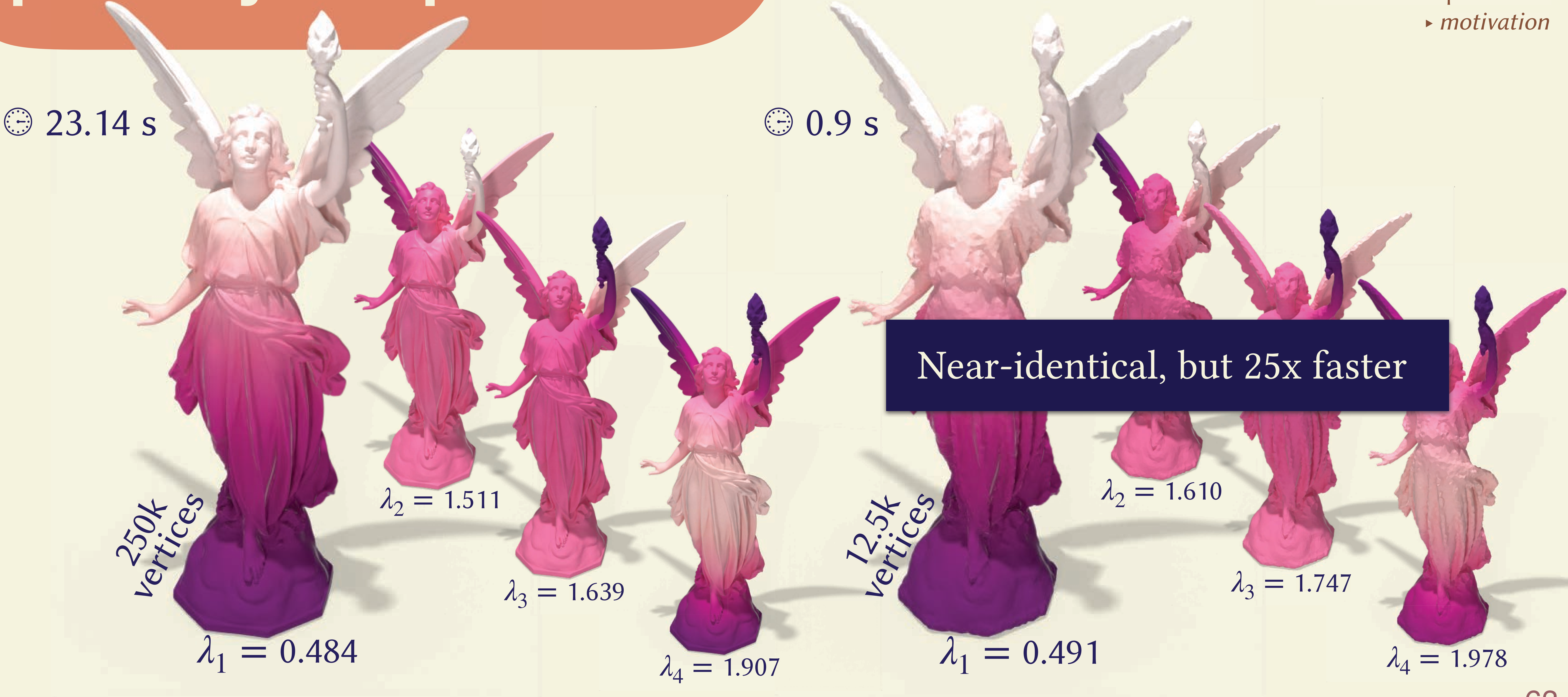


✓ Compute geometric quantities directly on the original surface

✗ Preserves unnecessary geometric details

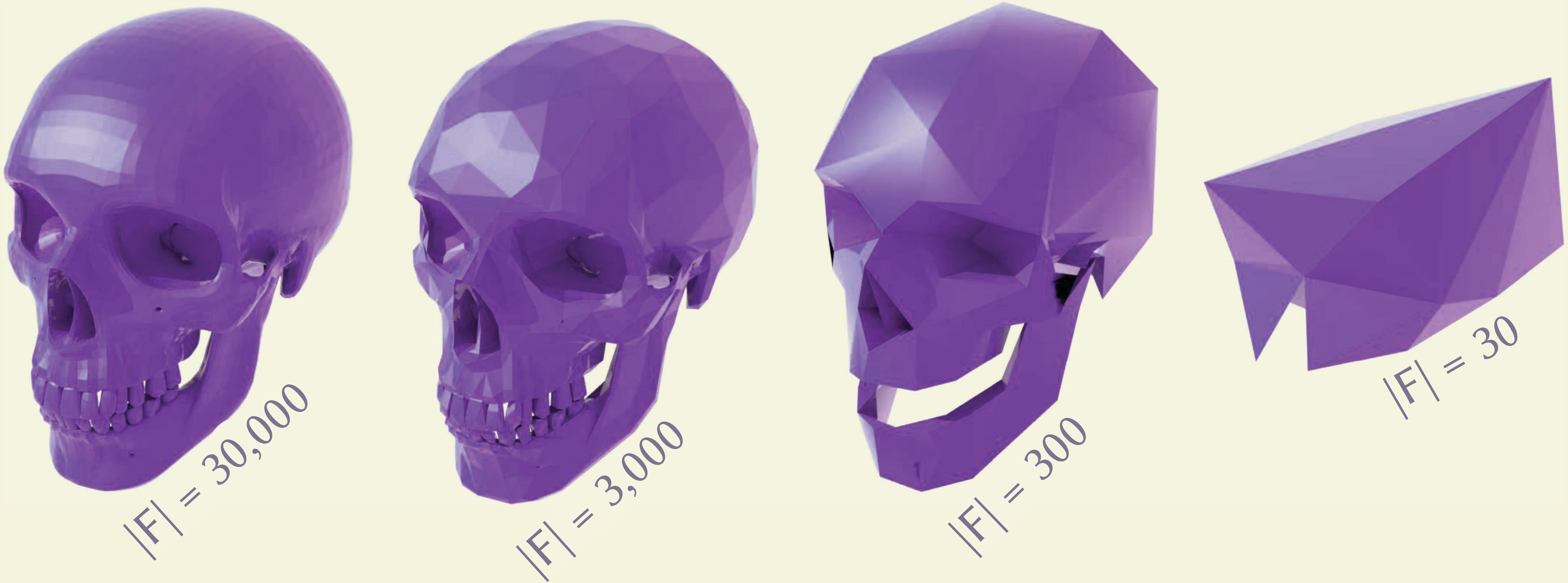
Coarse meshes can be perfectly adequate

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

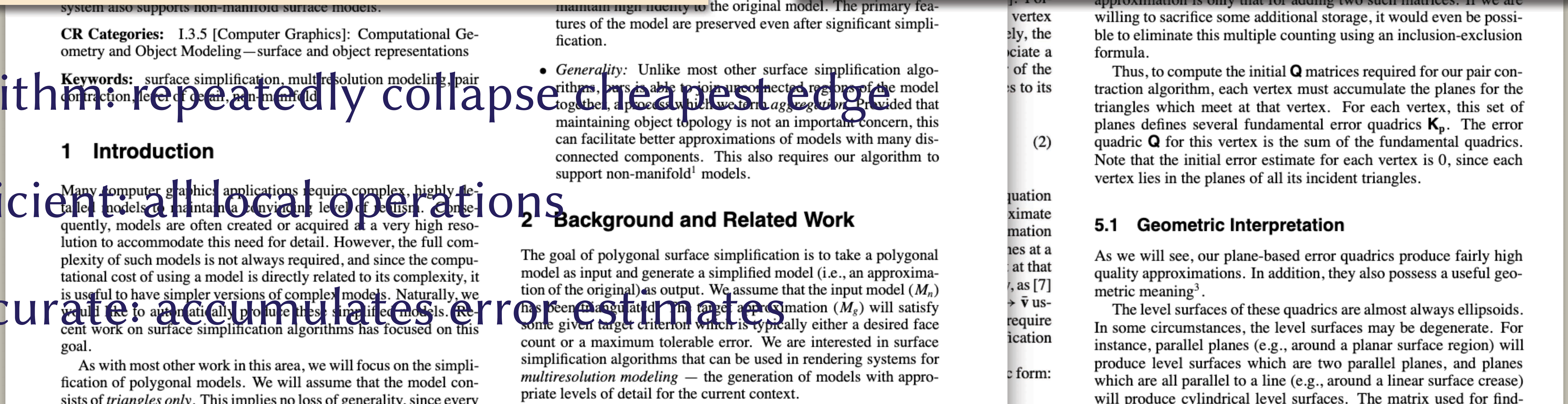


Intrinsic problems benefit from intrinsic simplification

- 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]



CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—surface and object representations

Keywords: surface simplification, multiresolution modeling, pair contraction, level of detail, non-manifold

1 Introduction

Many computer graphics applications require complex, highly detailed models to maintain a convincing level of realism. Consequently, models are often created or acquired at a very high resolution to accommodate this need for detail. However, the full complexity of such models is not always required, and since the computational cost of using a model is directly related to its complexity, it is useful to have simpler versions of complex models. Naturally, we would like to automatically produce these simplified models. Recent work on surface simplification algorithms has focused on this goal.

As with most other work in this area, we will focus on the simplification of polygonal models. We will assume that the model consists of *triangles only*. This implies no loss of generality, since every

maintain high fidelity to the original model. The primary features of the model are preserved even after significant simplification.

- *Generality:* Unlike most other surface simplification algorithms, ours is able to join unconnected regions of the model together, a process which we term *aggregation*. Provided that maintaining object topology is not an important concern, this can facilitate better approximations of models with many disconnected components. This also requires our algorithm to support non-manifold[1] models.

2 Background and Related Work

The goal of polygonal surface simplification is to take a polygonal model as input and generate a simplified model (i.e., an approximation of the original) as output. We assume that the input model (M_n) has been triangulated. The target approximation (M_g) will satisfy some given target criterion which is typically either a desired face count or a maximum tolerable error. We are interested in surface simplification algorithms that can be used in rendering systems for *multiresolution modeling* — the generation of models with appropriate levels of detail for the current context.

willing to sacrifice some additional storage, it would even be possible to eliminate this multiple counting using an inclusion-exclusion formula.

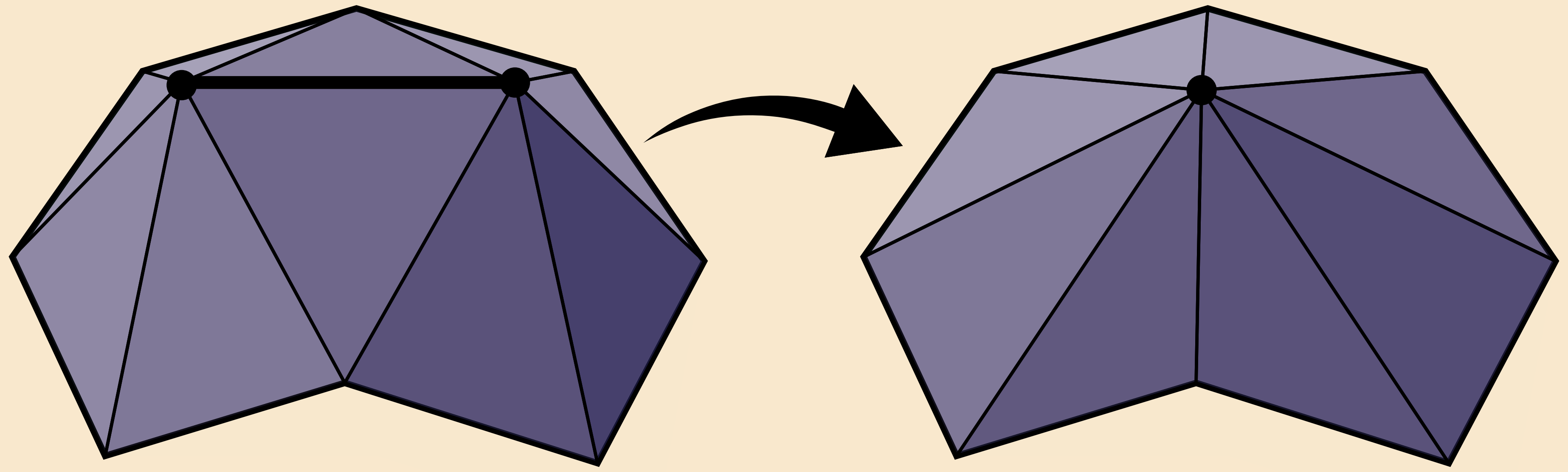
Thus, to compute the initial **Q** matrices required for our pair contraction algorithm, each vertex must accumulate the planes for the triangles which meet at that vertex. For each vertex, this set of planes defines several fundamental error quadrics $\mathbf{K_p}$. The error quadric **Q** for this vertex is the sum of the fundamental quadrics. Note that the initial error estimate for each vertex is 0, since each vertex lies in the planes of all its incident triangles.

5.1 Geometric Interpretation

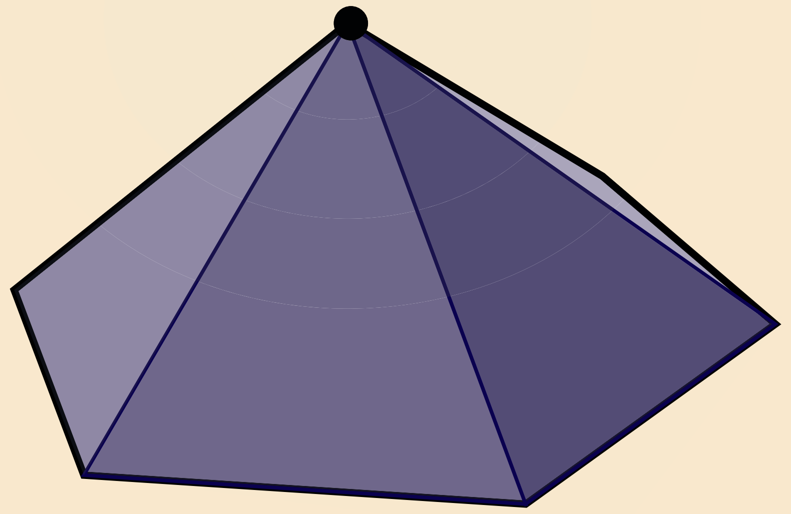
As we will see, our plane-based error quadrics produce fairly high quality approximations. In addition, they also possess a useful geometric meaning[3].

The level surfaces of these quadrics are almost always ellipsoids. In some circumstances, the level surfaces may be degenerate. For instance, parallel planes (e.g., around a planar surface region) will produce level surfaces which are two parallel planes, and planes which are all parallel to a line (e.g., around a linear surface crease) will produce cylindrical level surfaces. The matrix used for find-

1. Local simplification operation



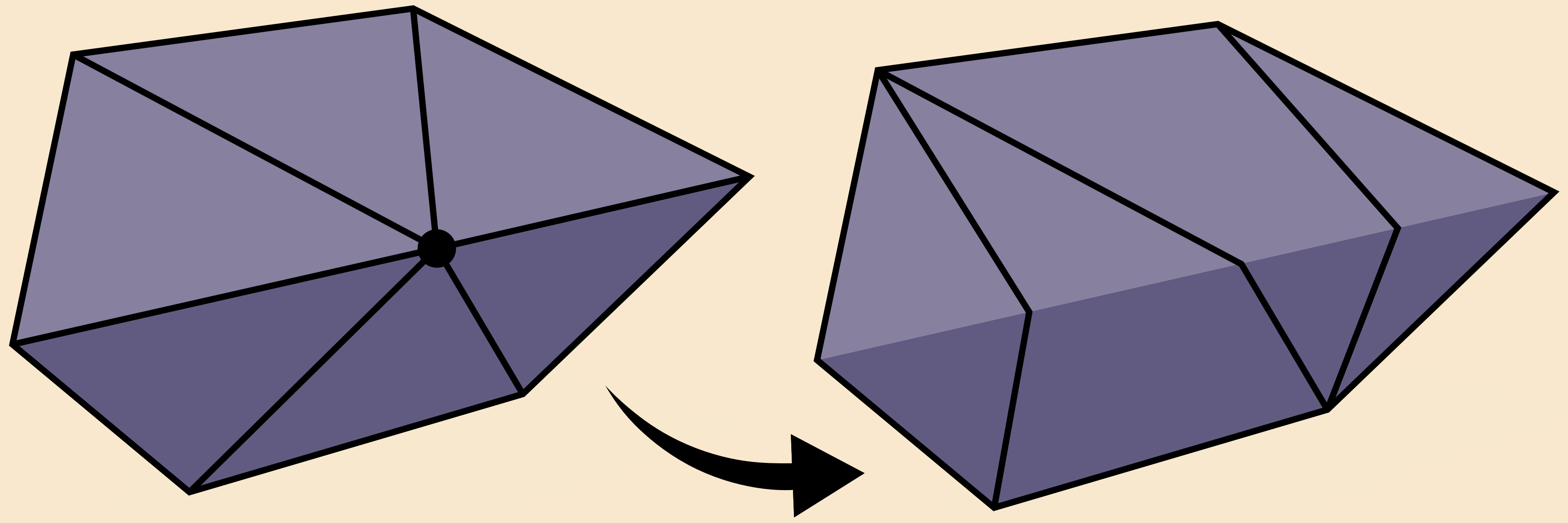
2. Accumulated distortion measurements



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

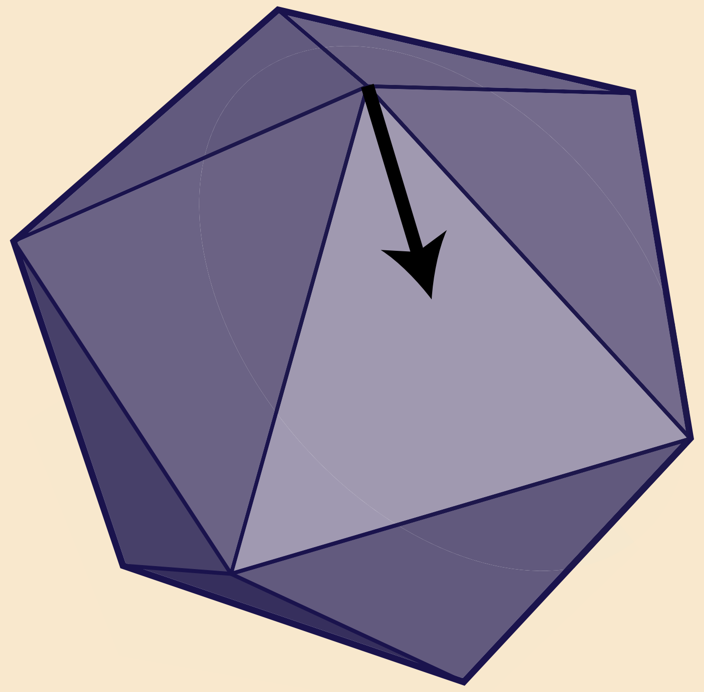
Intrinsic simplification

1. Local simplification operation



intrinsic vertex removal

2. Accumulated distortion measurements

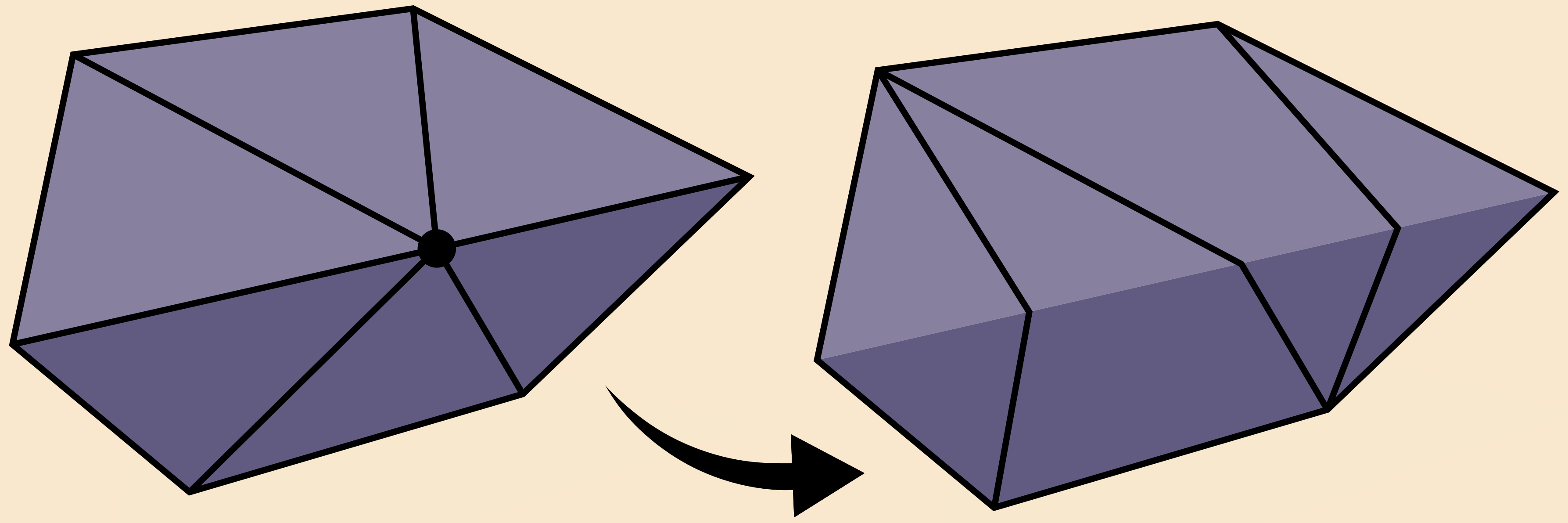


intrinsic curvature error

- Algorithm: repeatedly remove cheapest vertex

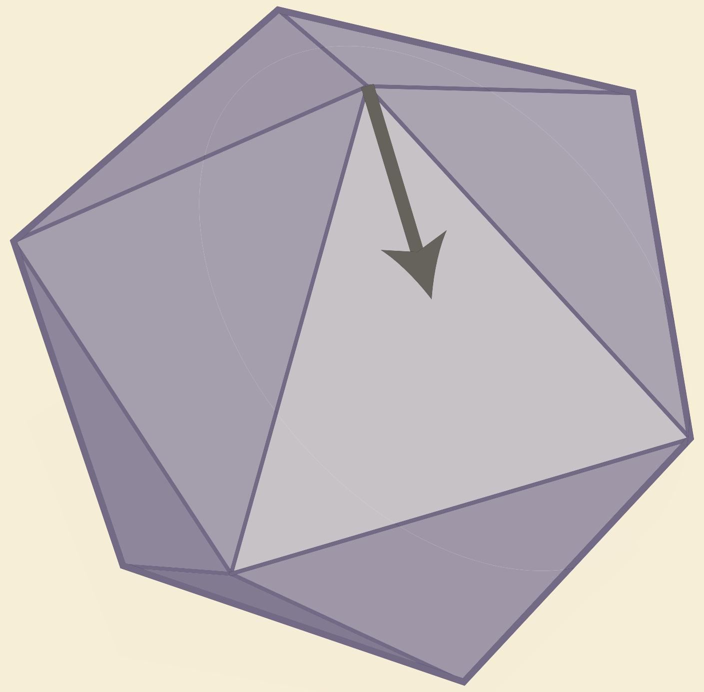
Intrinsic simplification

1. Local simplification operation



intrinsic vertex removal

2. Accumulated distortion measurements

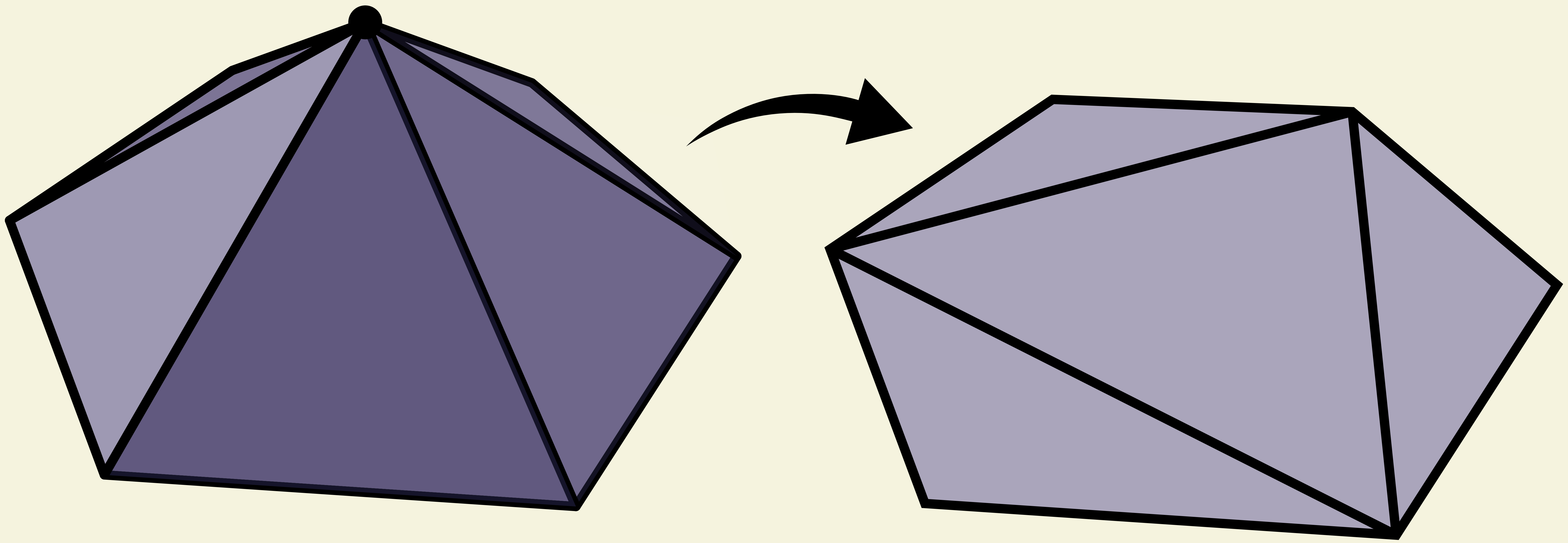


intrinsic curvature error

- Algorithm: repeatedly remove cheapest vertex

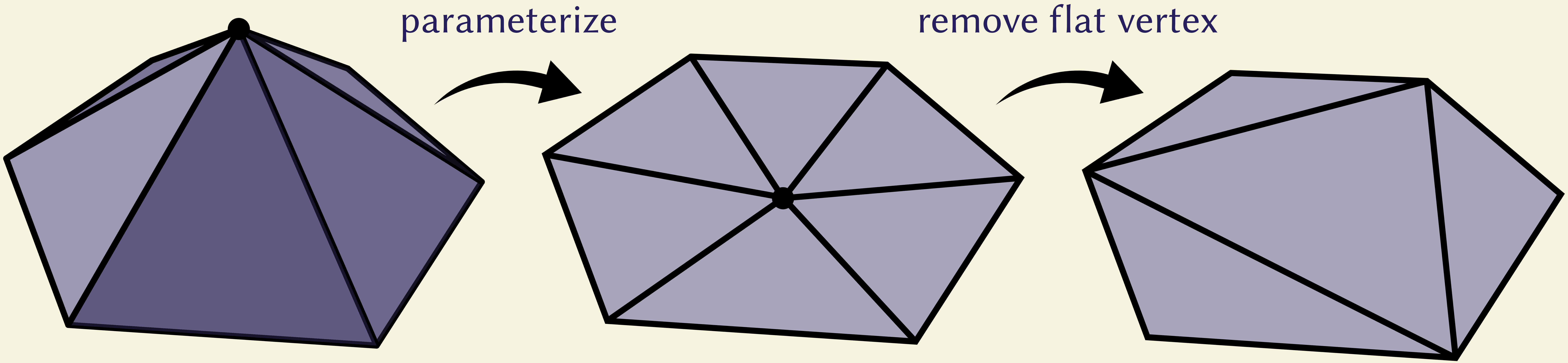
Intrinsic vertex removal

- Intrinsic view: replace curved vertex with flat patch

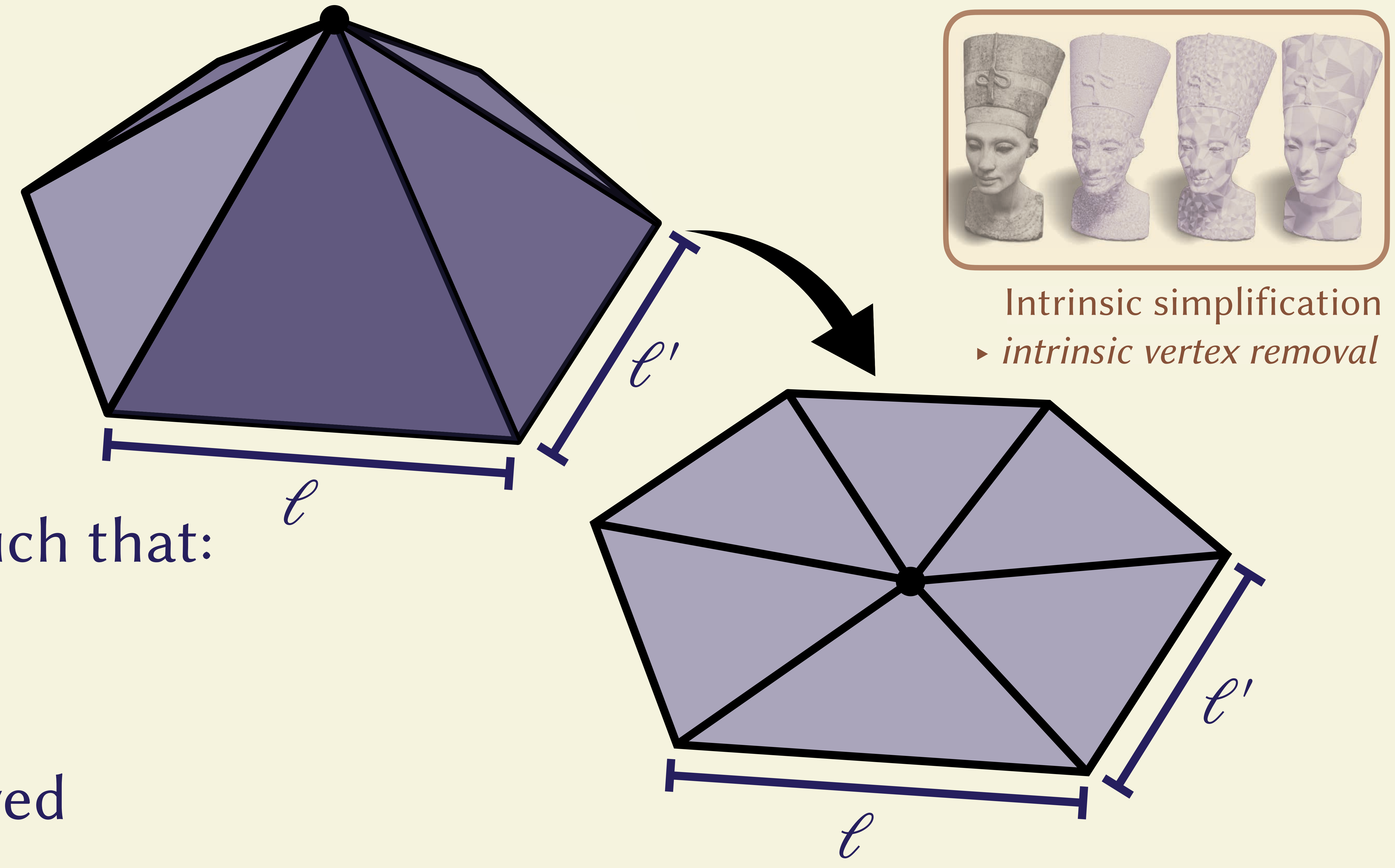


Intrinsic vertex removal

- Intrinsic view: replace curved vertex with flat patch



Vertex flattening



Intrinsic simplification
‣ *intrinsic vertex removal*

- Map neighboring triangles to plane such that:

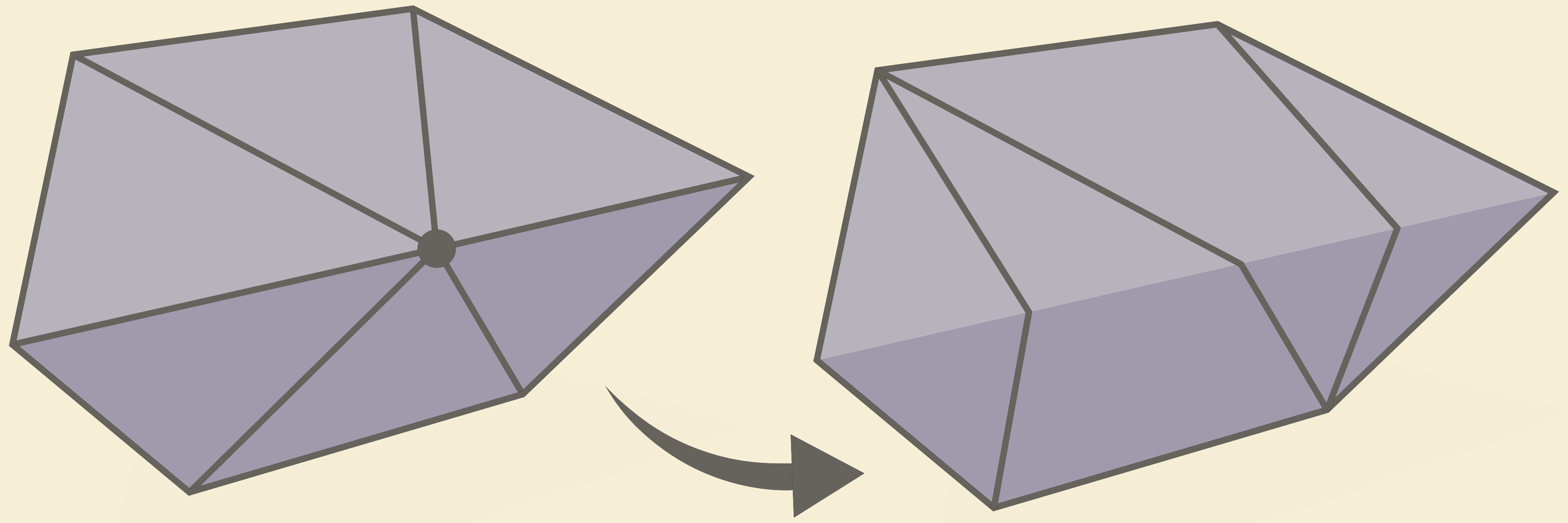
  (1) Distortion is low

  (2) Boundary edge lengths are preserved

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

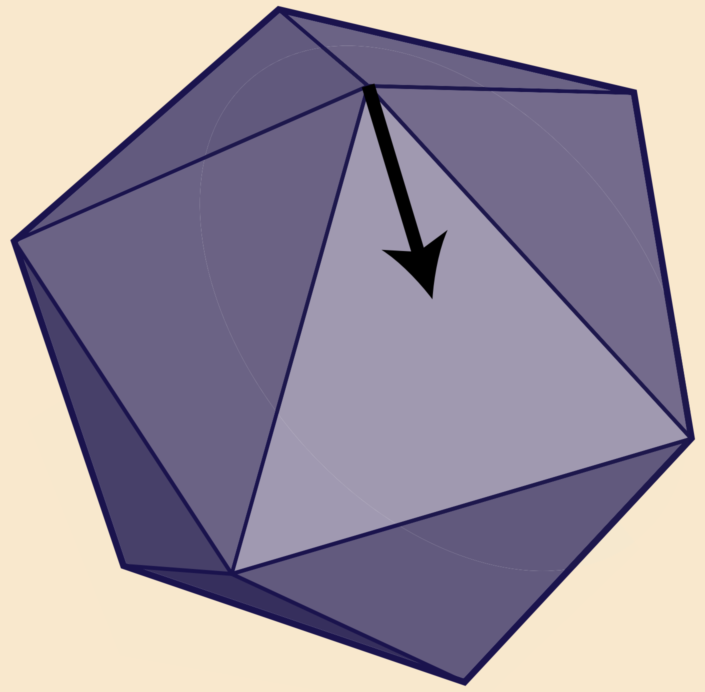
Intrinsic simplification

1. Local simplification operation



intrinsic vertex removal

2. Accumulated distortion measurements

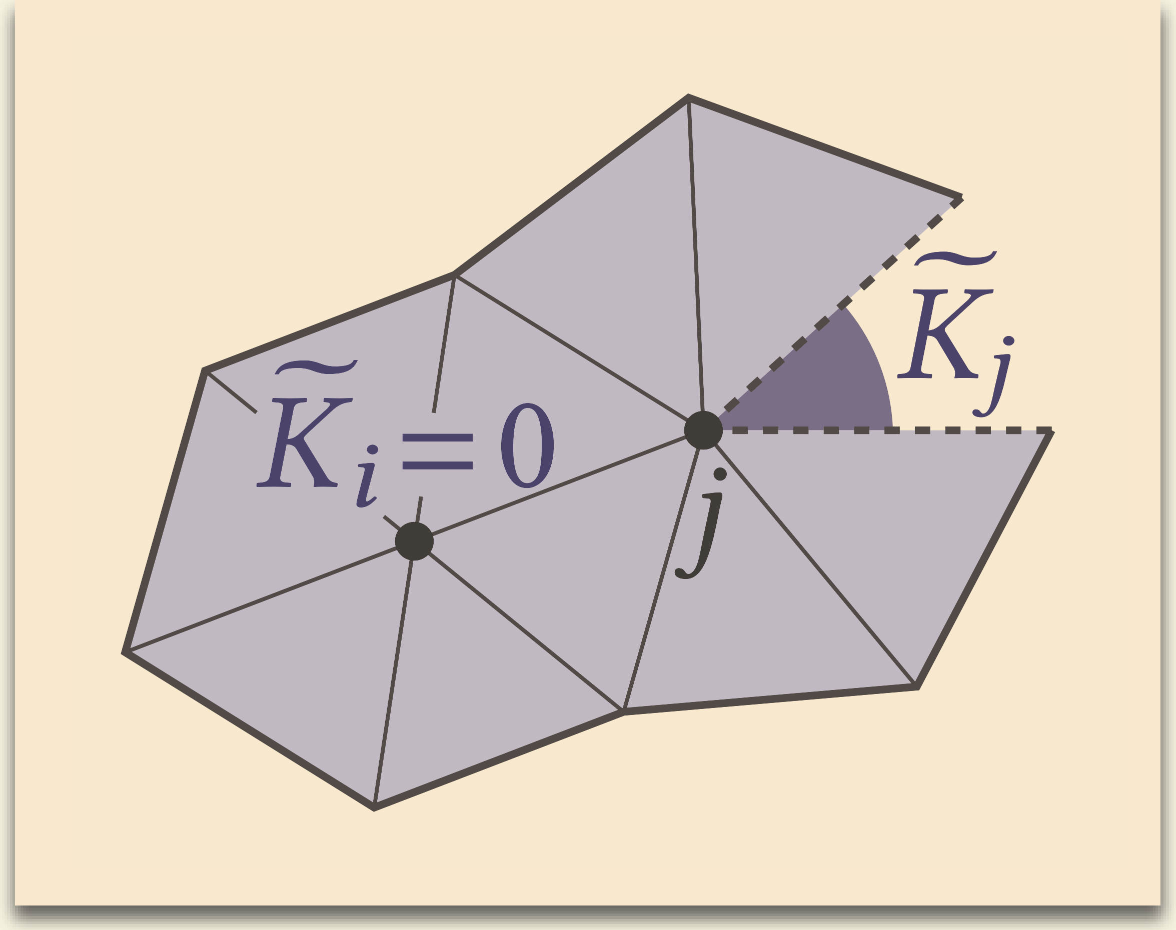


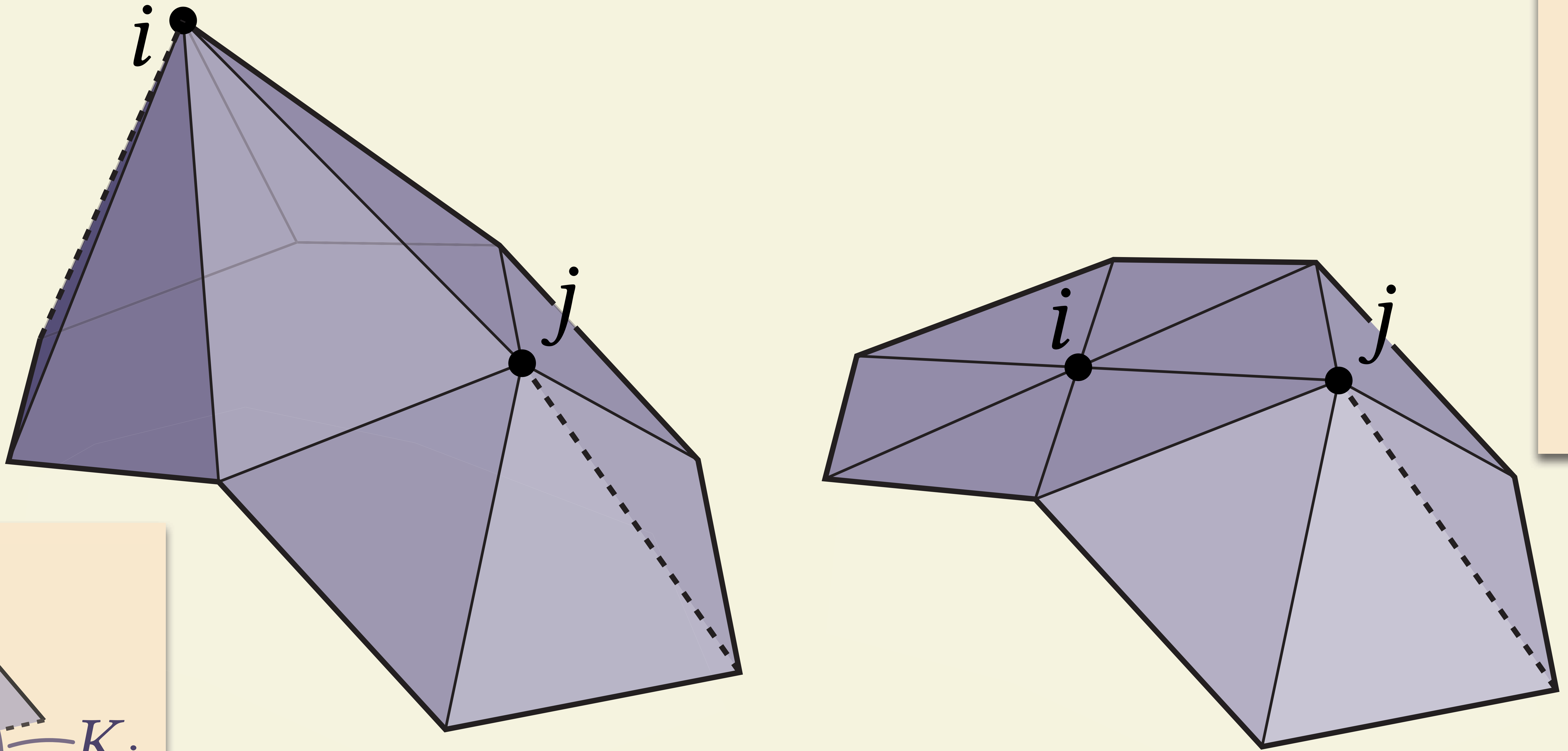
intrinsic curvature error

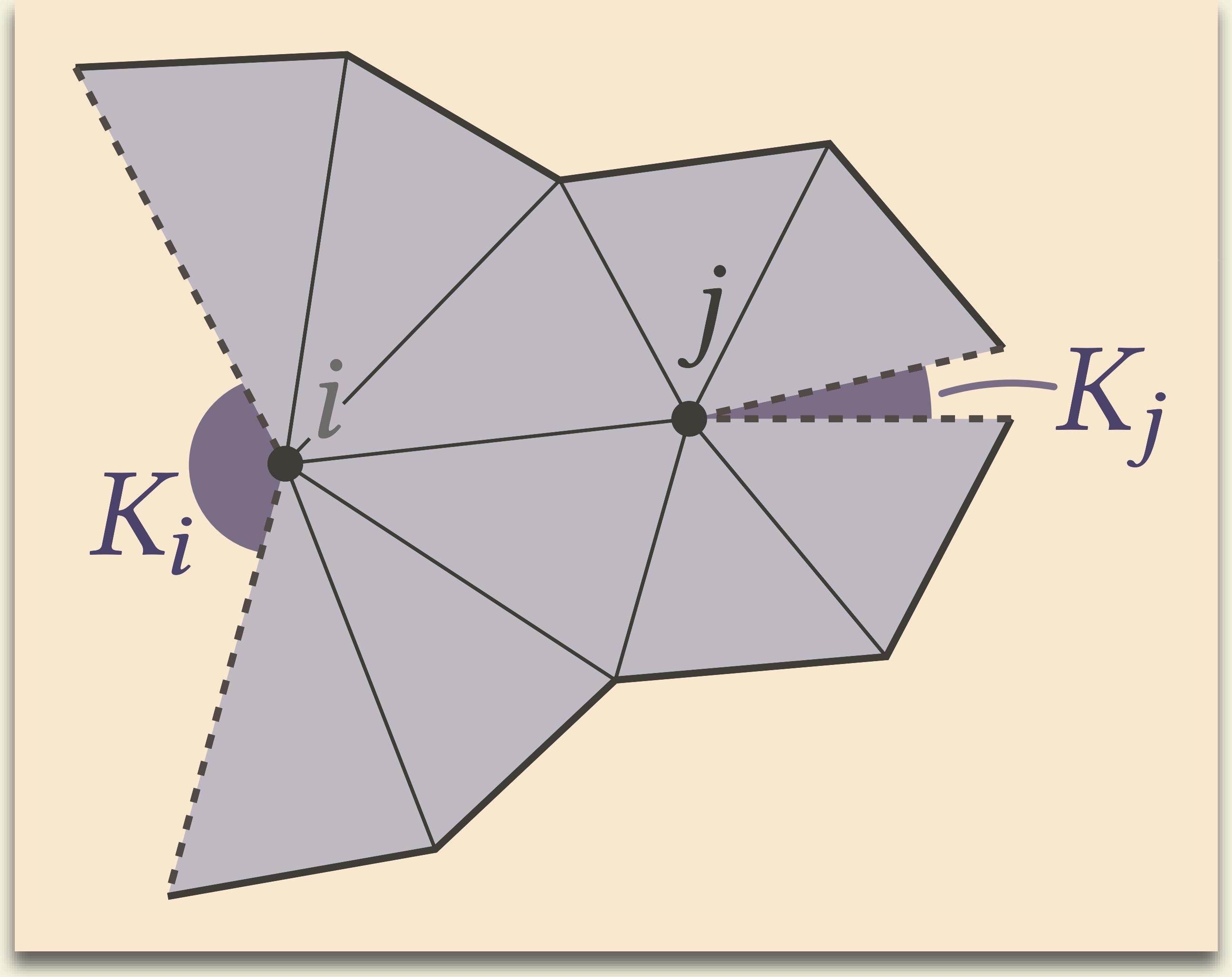
- Algorithm: repeatedly remove cheapest vertex

Distortion: curvature redistribution

Intrinsic simplification
▸ *intrinsic curvature error*

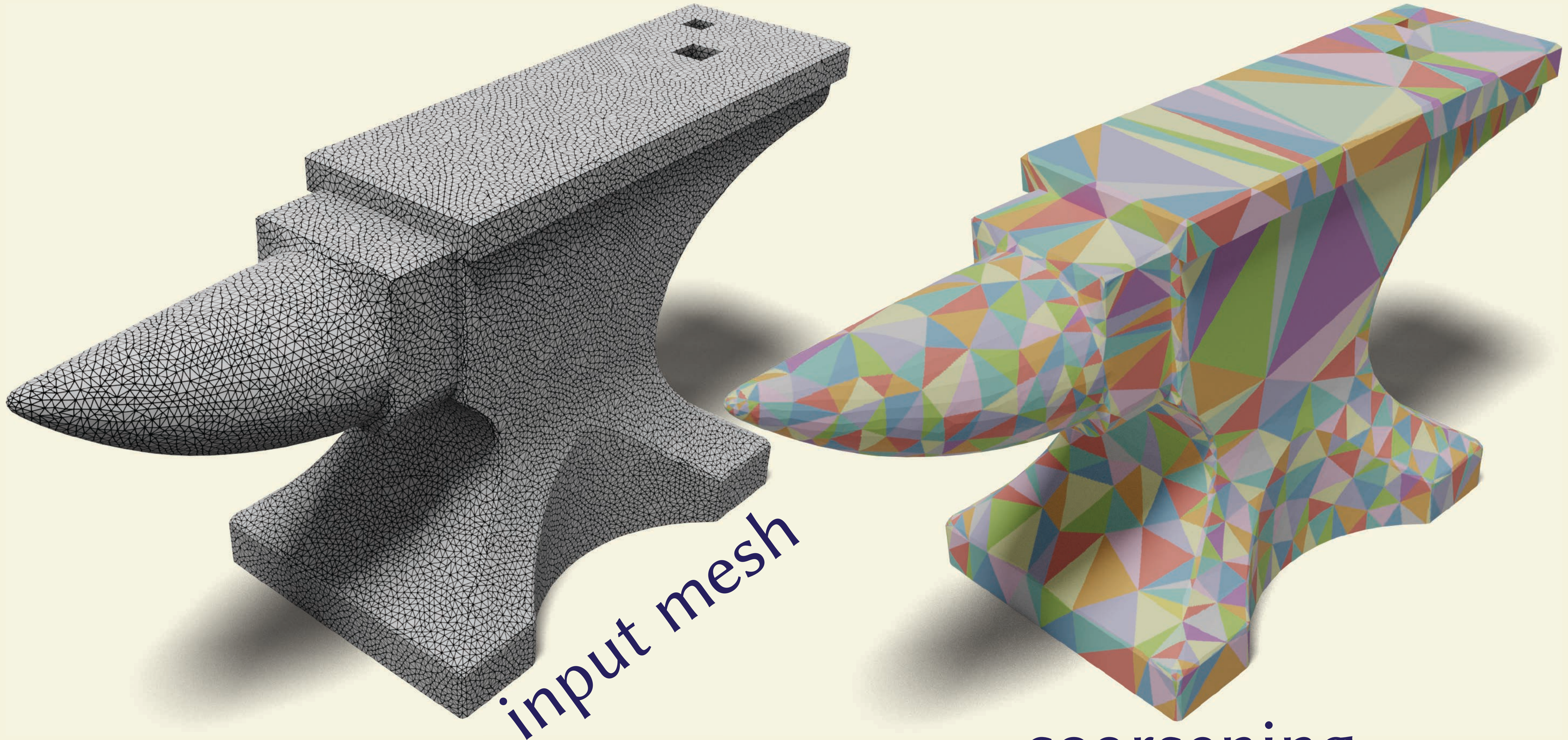






We approximate the *transport cost* of this curvature redistribution

Simplification with the curvature transport cost



Other transport costs

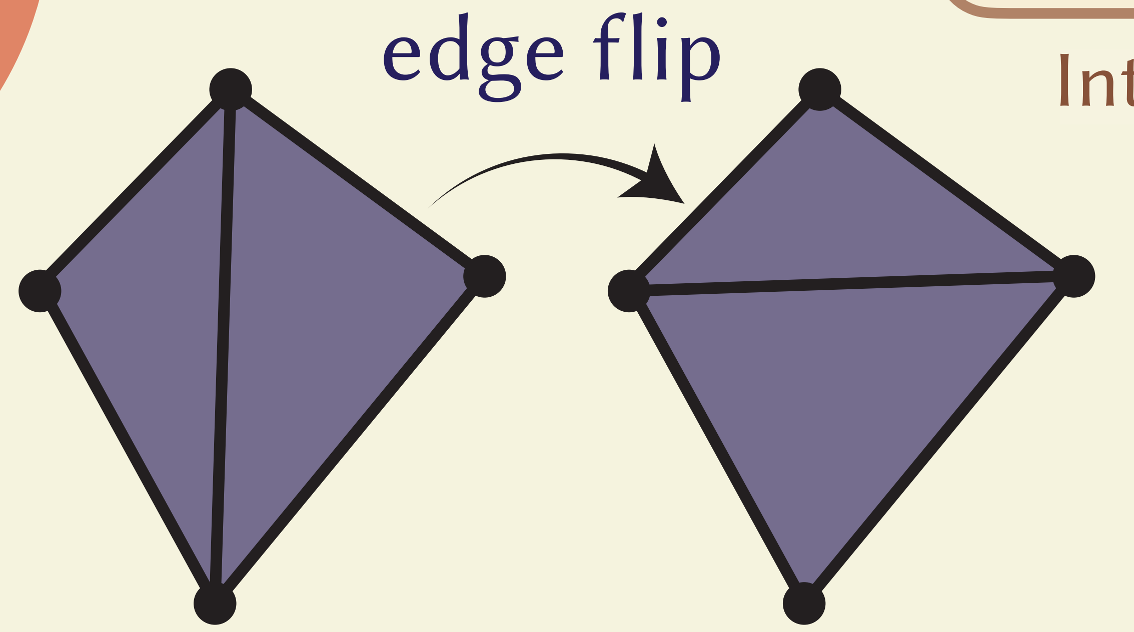

Intrinsic simplification
‣ *intrinsic curvature error*

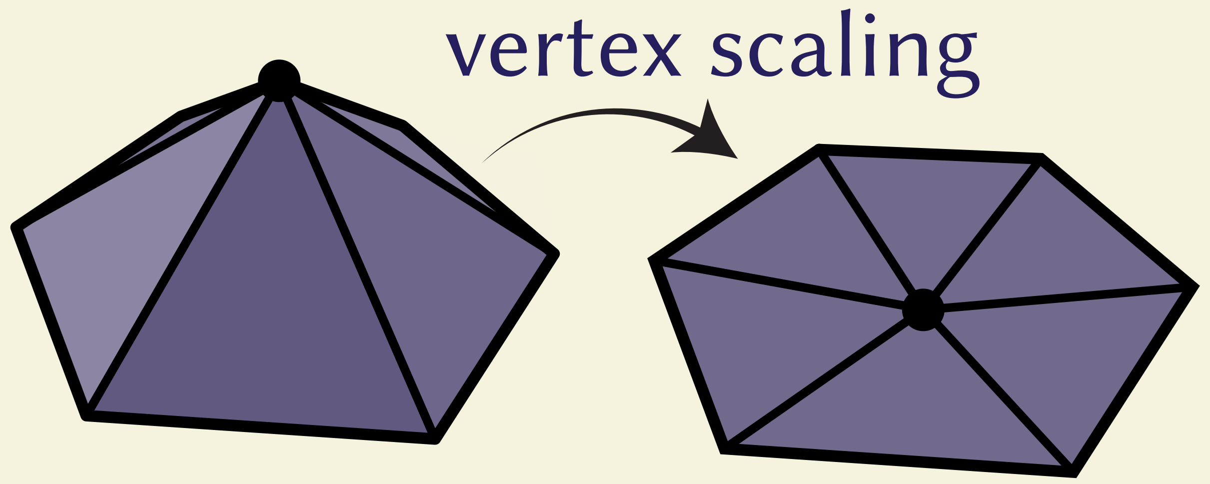
- Track transport cost of other data in same way
  - Can take weighted combinations of costs

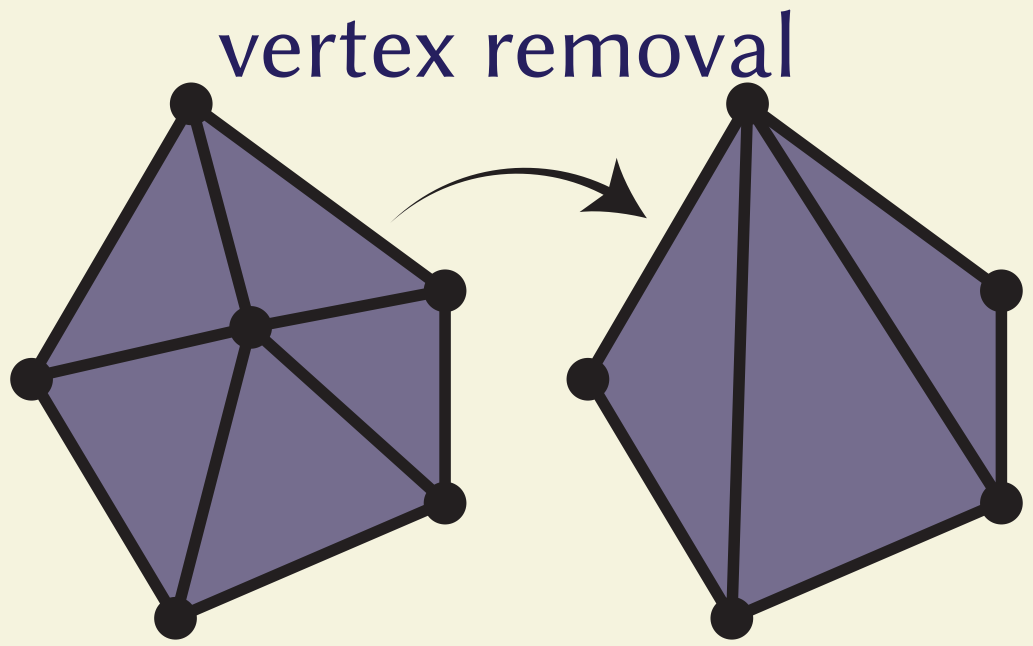
Surface correspondence

Intrinsic simplification
‣ *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



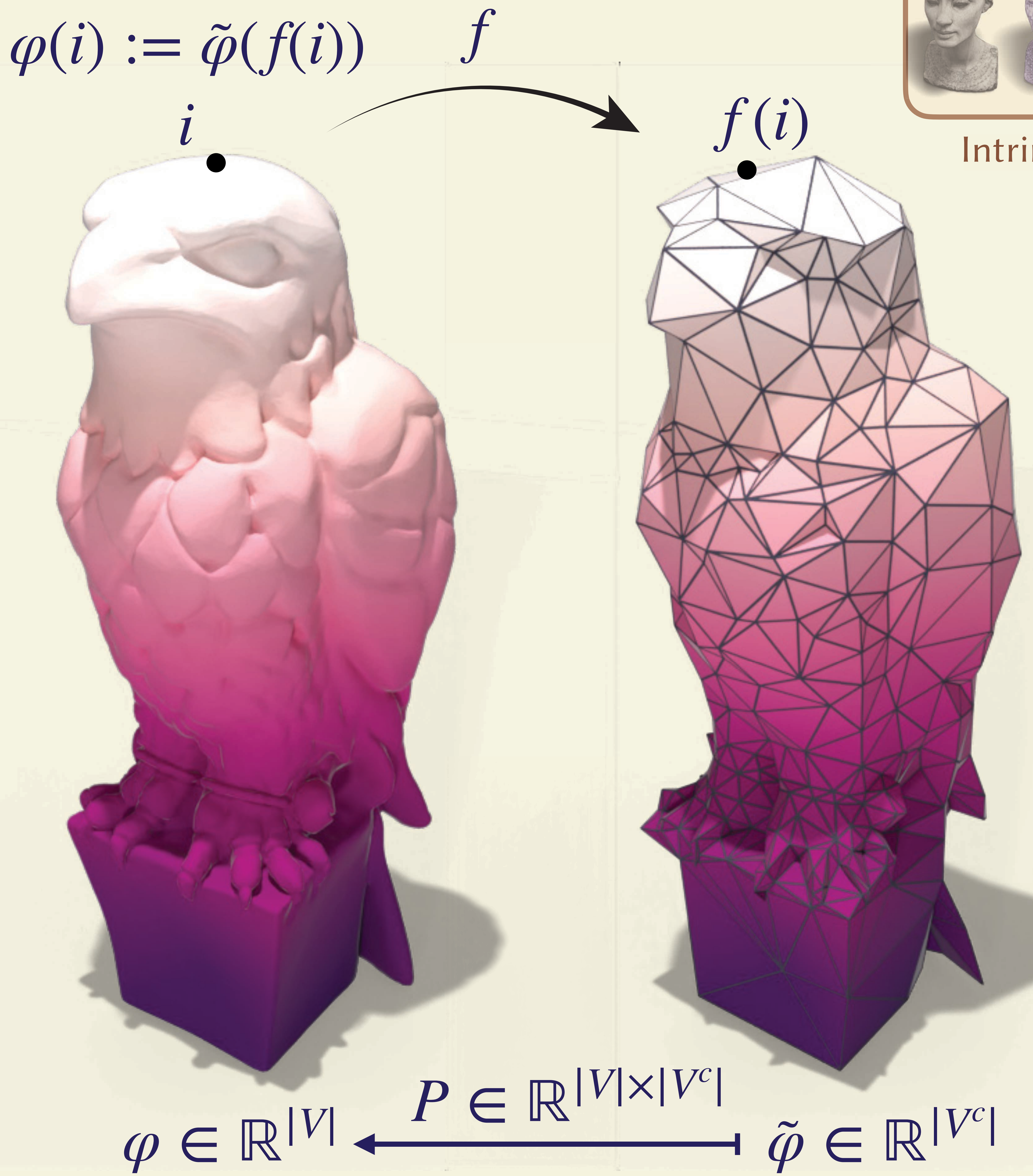




1. *Flip edge 1*
2. *Scale vertex 5*
3. *Remove vertex 5*
4. *Flip edge 8*
5. *Flip edge 12*
6. *Scale vertex 2*

Prolongation

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



Intrinsic simplification
‣ *correspondence*

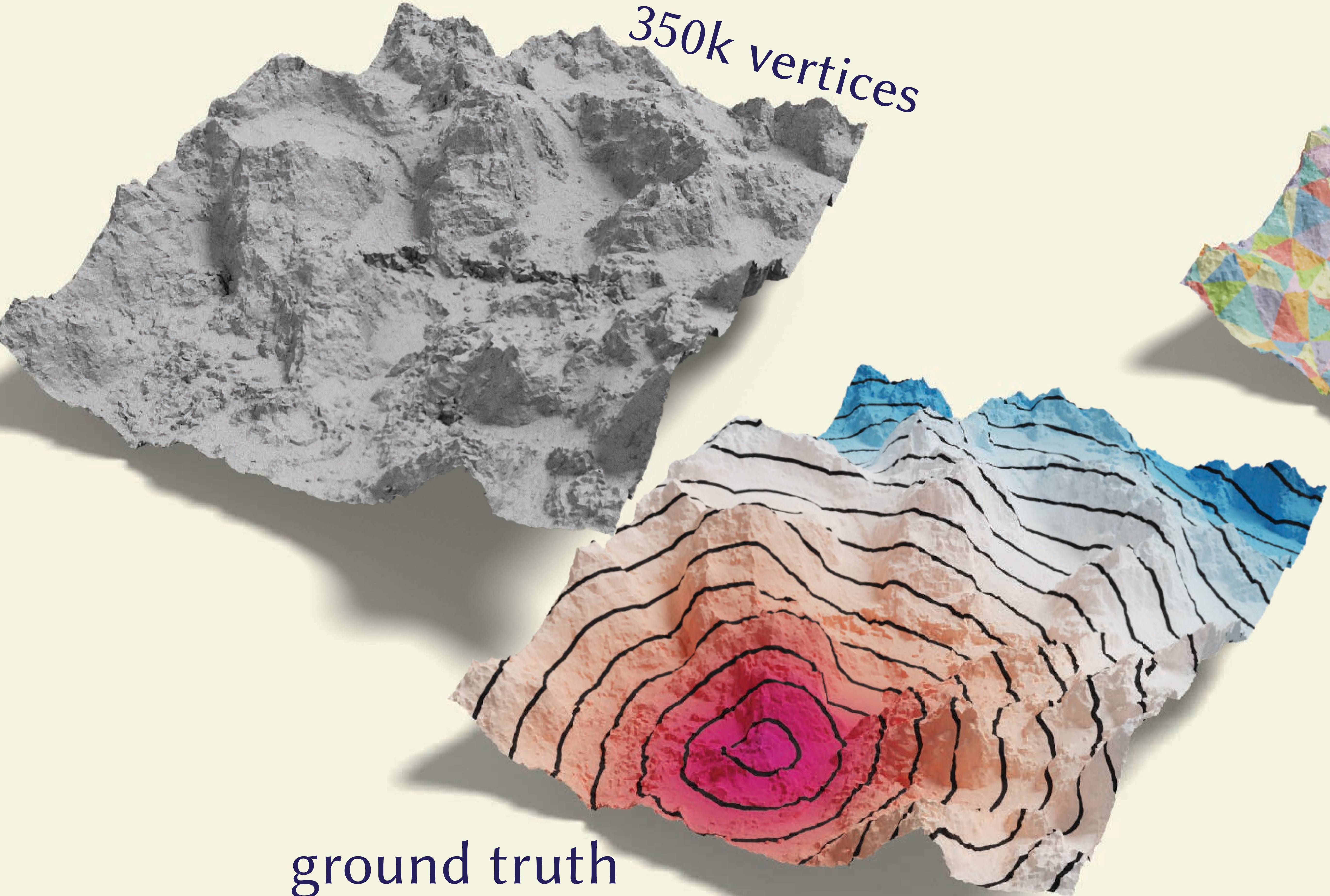
Intrinsic simplification

Examples

Computing geodesic distance

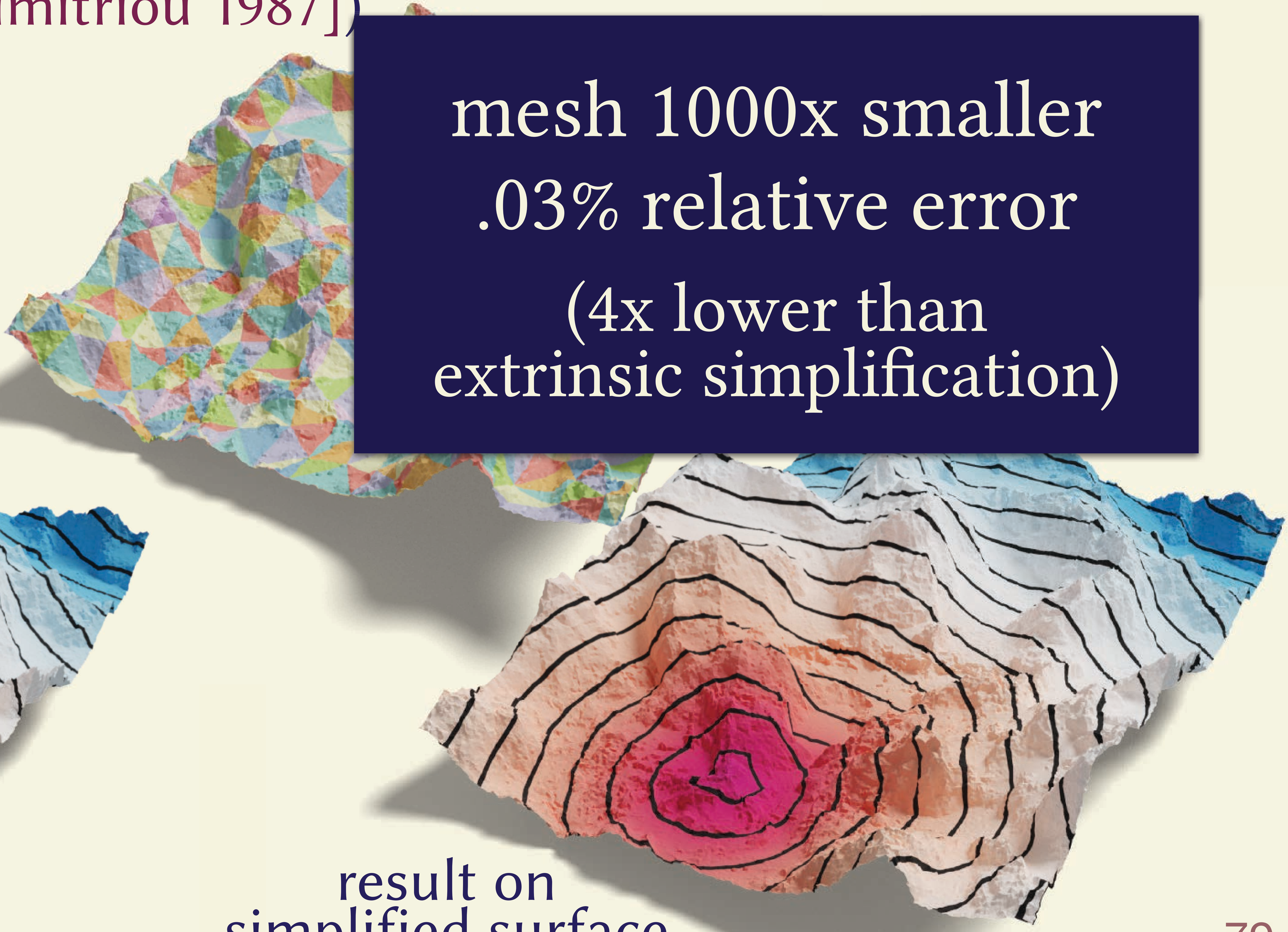
(Computed via [Mitchell, Mount & Papadimitriou 1987])

350k vertices



ground truth

mesh 1000x smaller
.03% relative error
(4x lower than extrinsic simplification)



result on simplified surface

Surface hierarchies

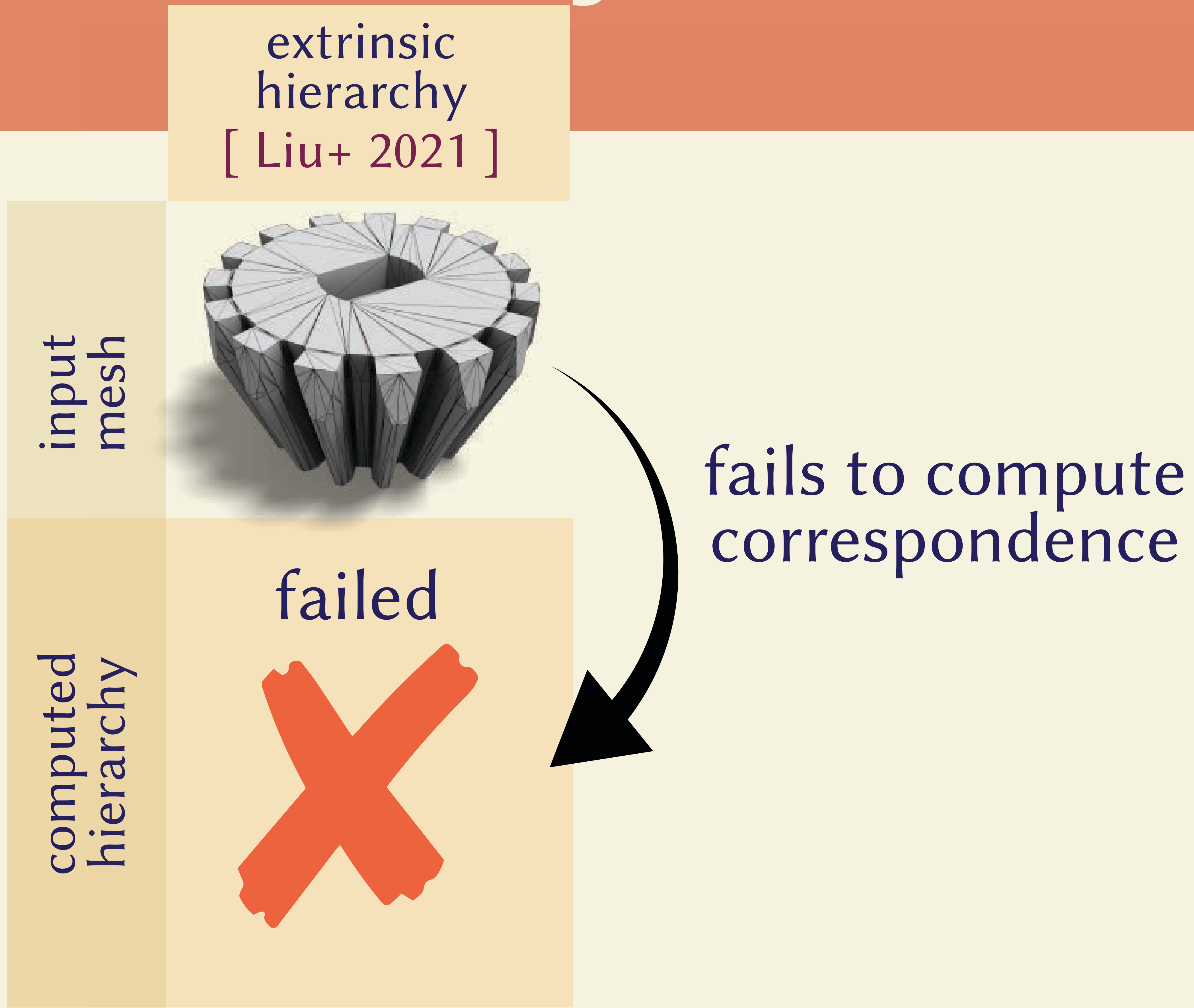
Intrinsic simplification
▸ *results*

Hierarchies accelerate computation

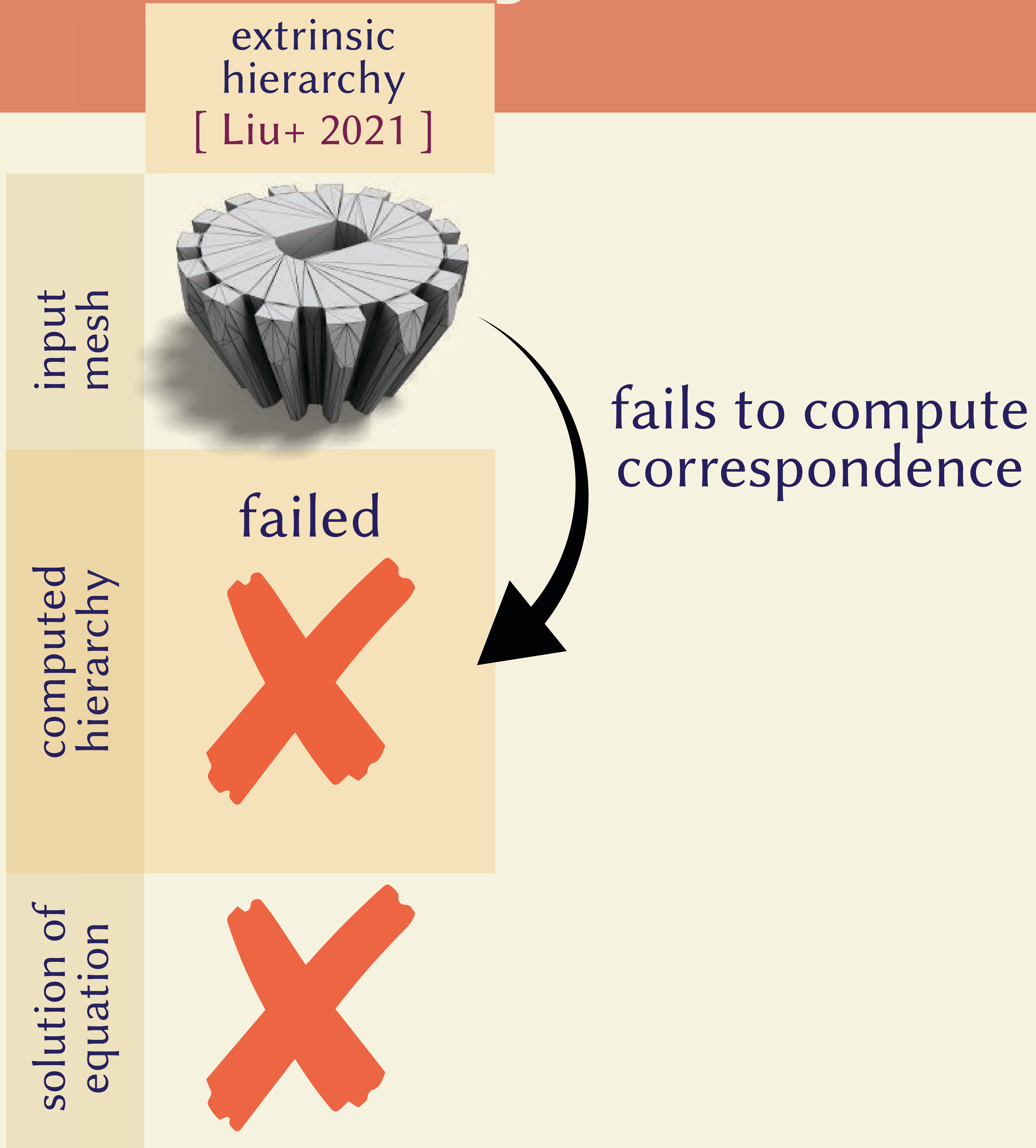
Intrinsic simplification
‣ *results*

- Accelerate geometric computations
  - Even helps with extrinsic problems

Robust hierarchy construction



Robust hierarchy construction



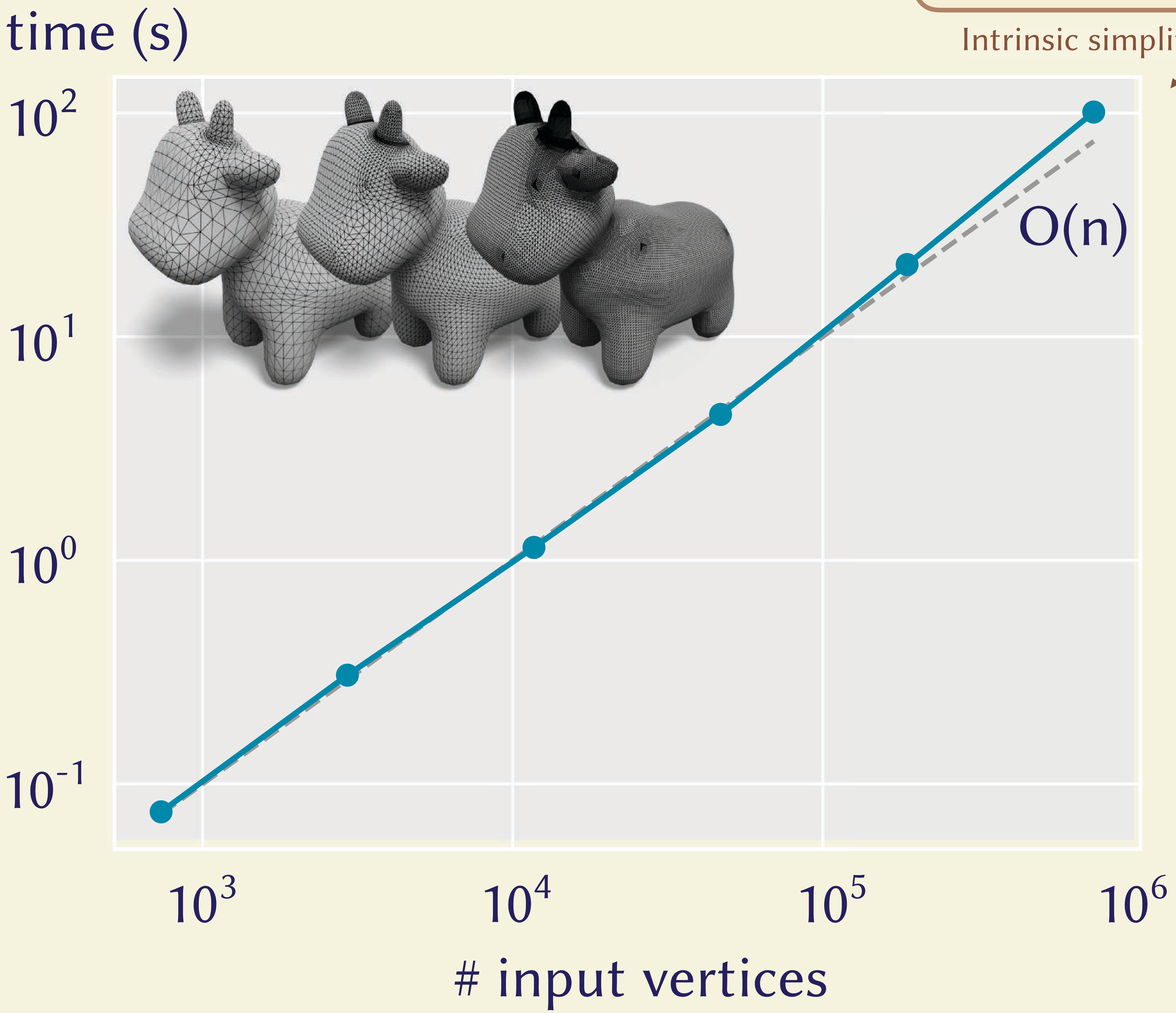
Robust hierarchy construction

Performance

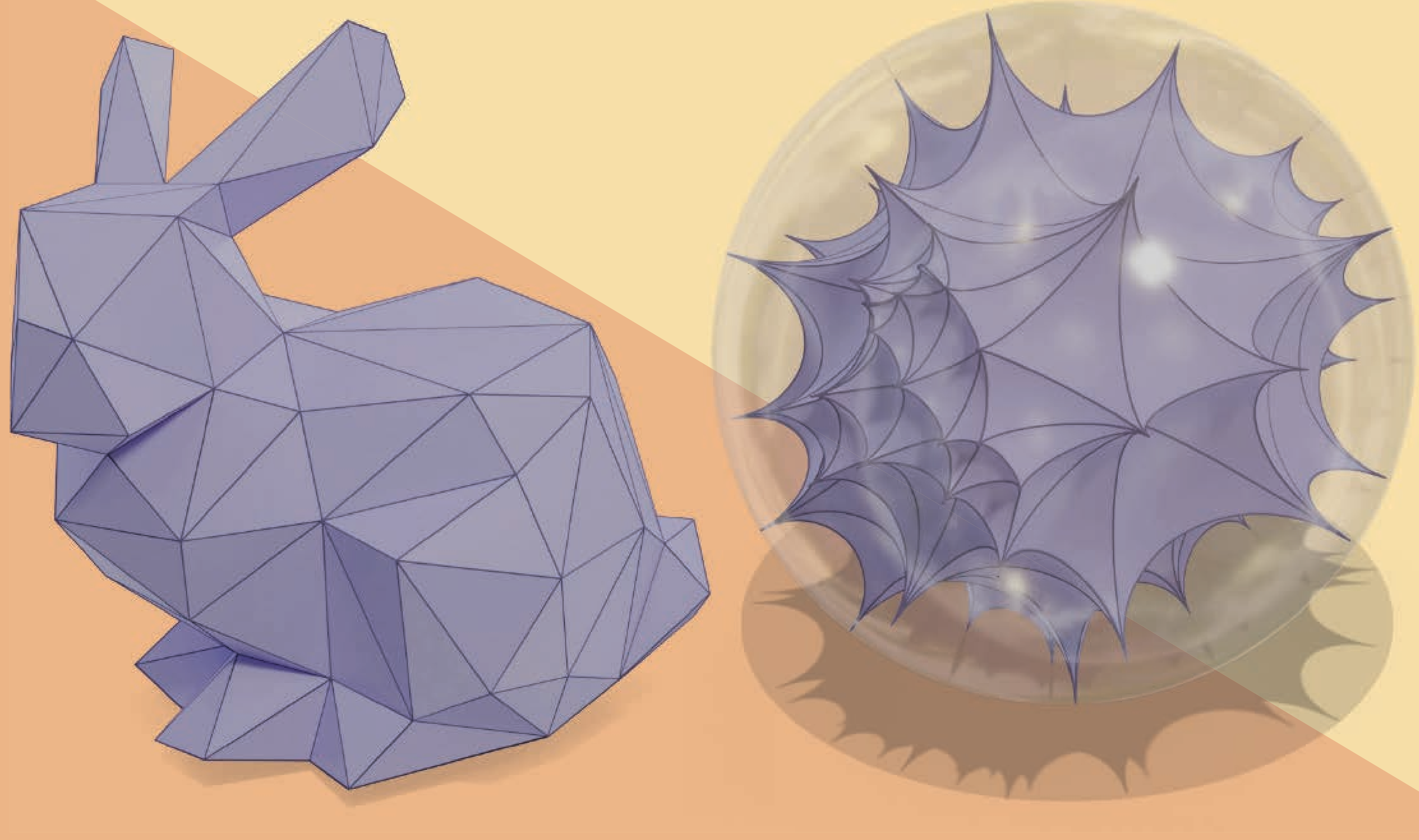
Intrinsic simplification
▸ *results*

- Linear scaling
  - Constant work per vertex

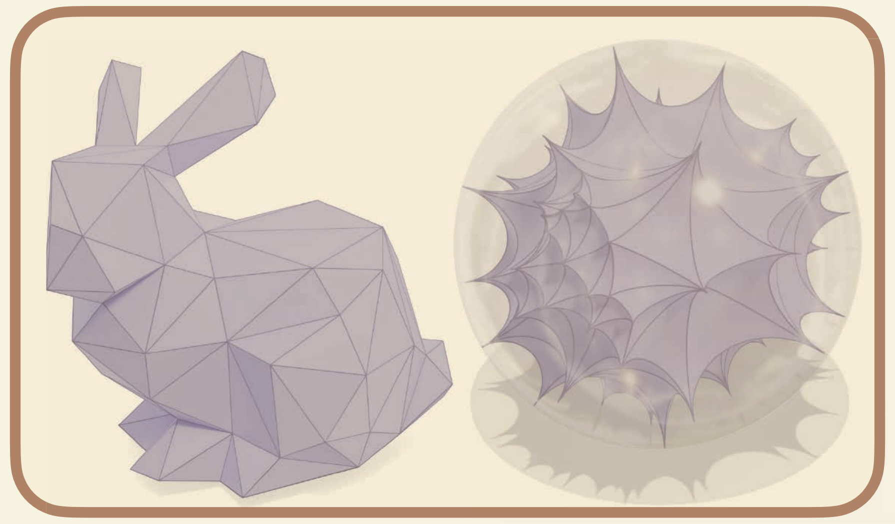
Removes ~13,000 vertices per second



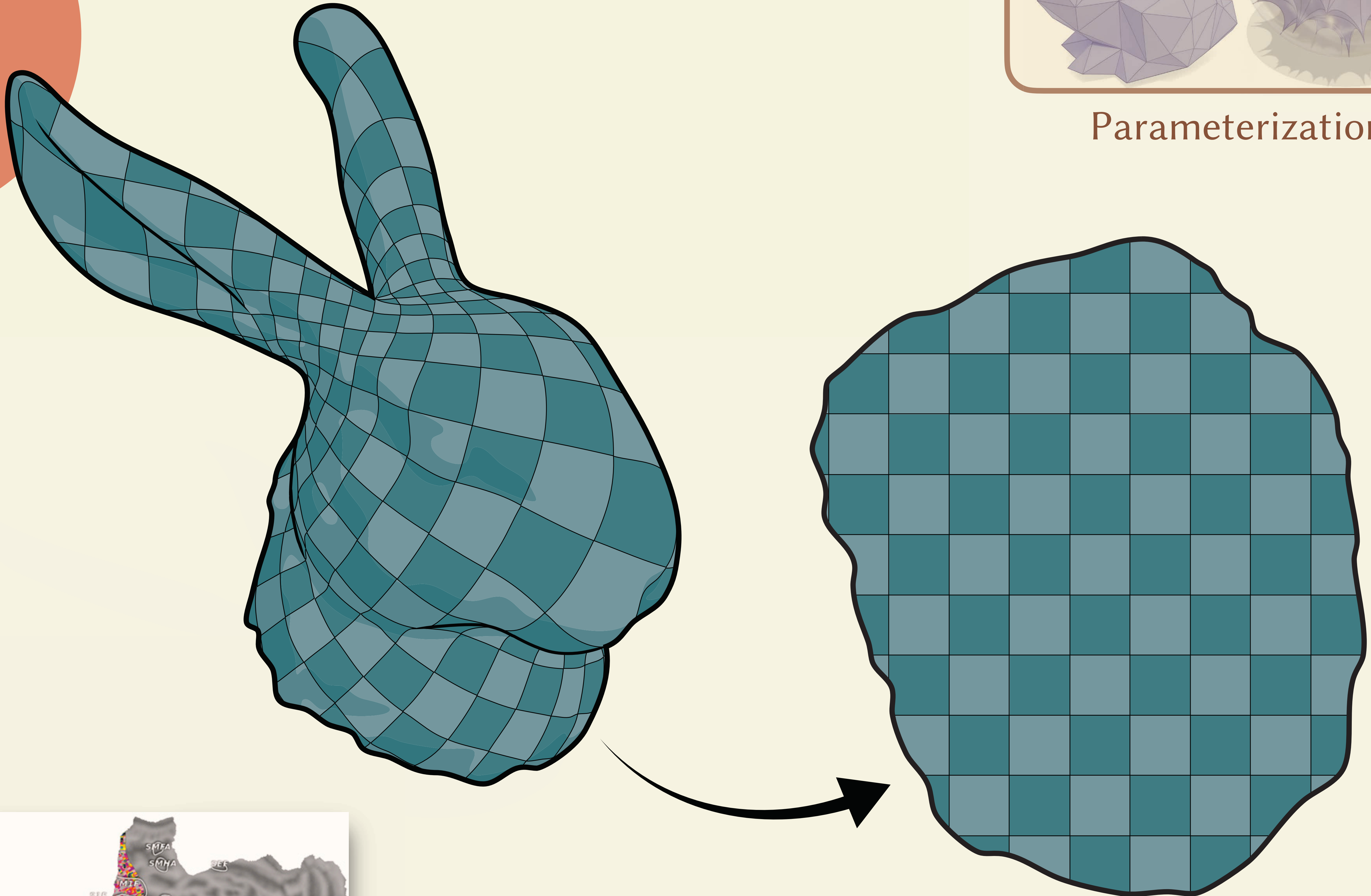
IV. Surface Parameterization



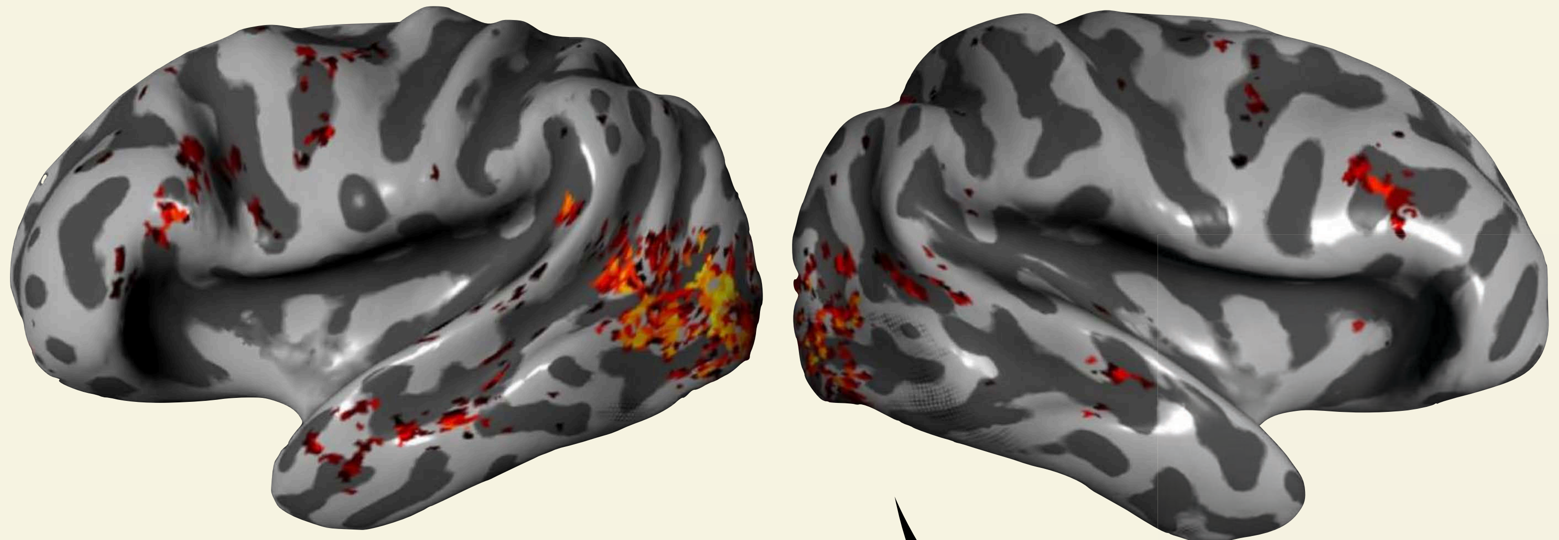
[**Gillespie**, Springborn, & Crane. 2021. Discrete conformal equivalence of polyhedral surfaces. *ACM Transactions on Graphics*]



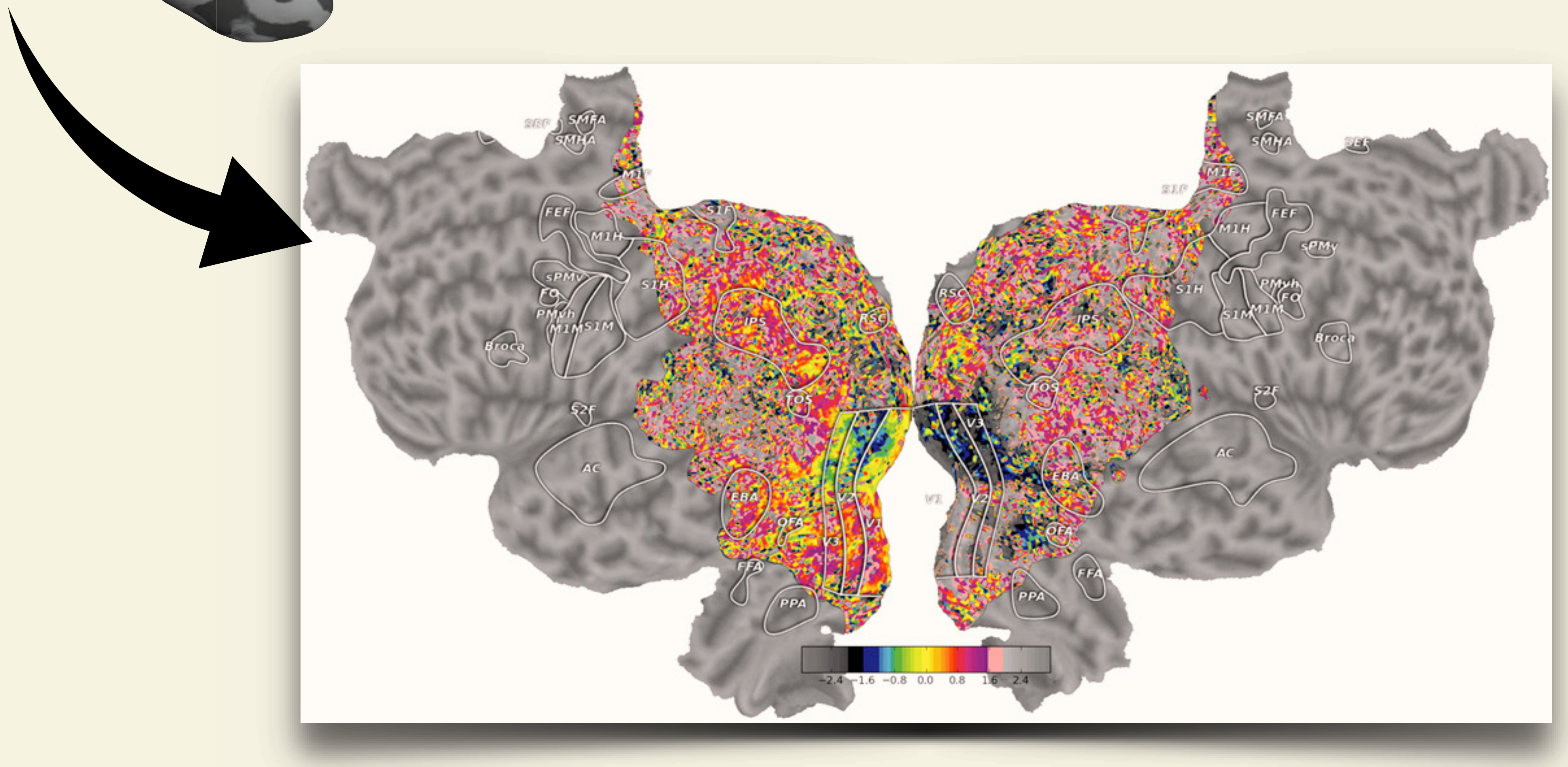
Parameterization

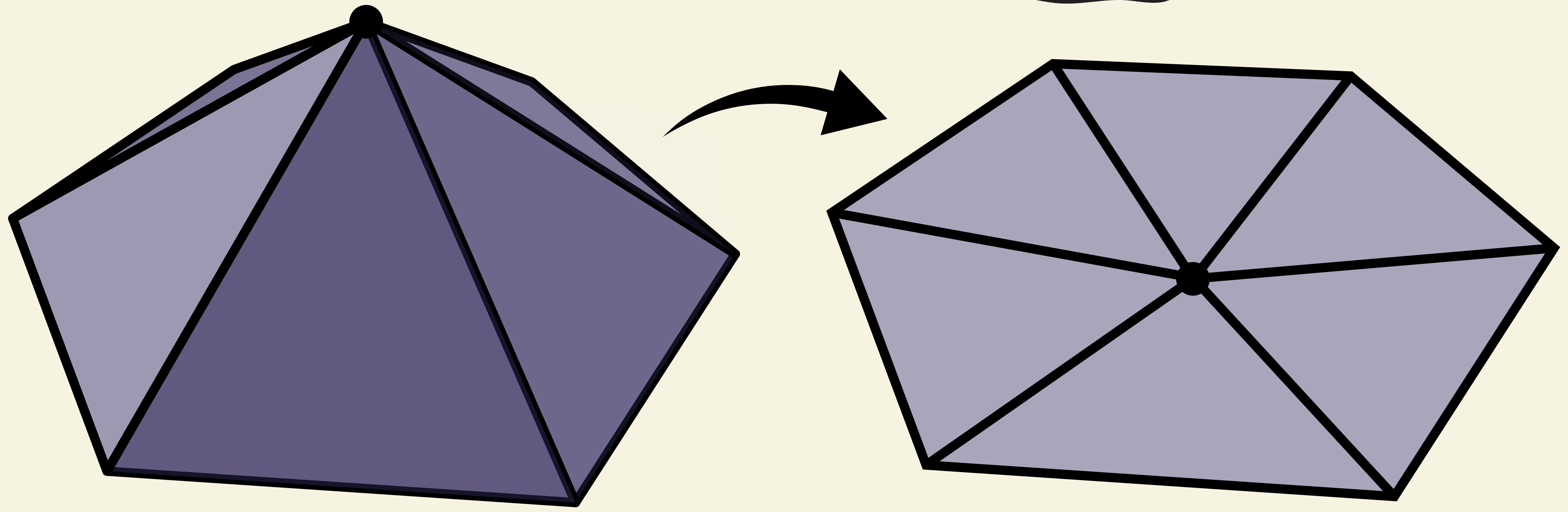


Mapping surfaces into the plane



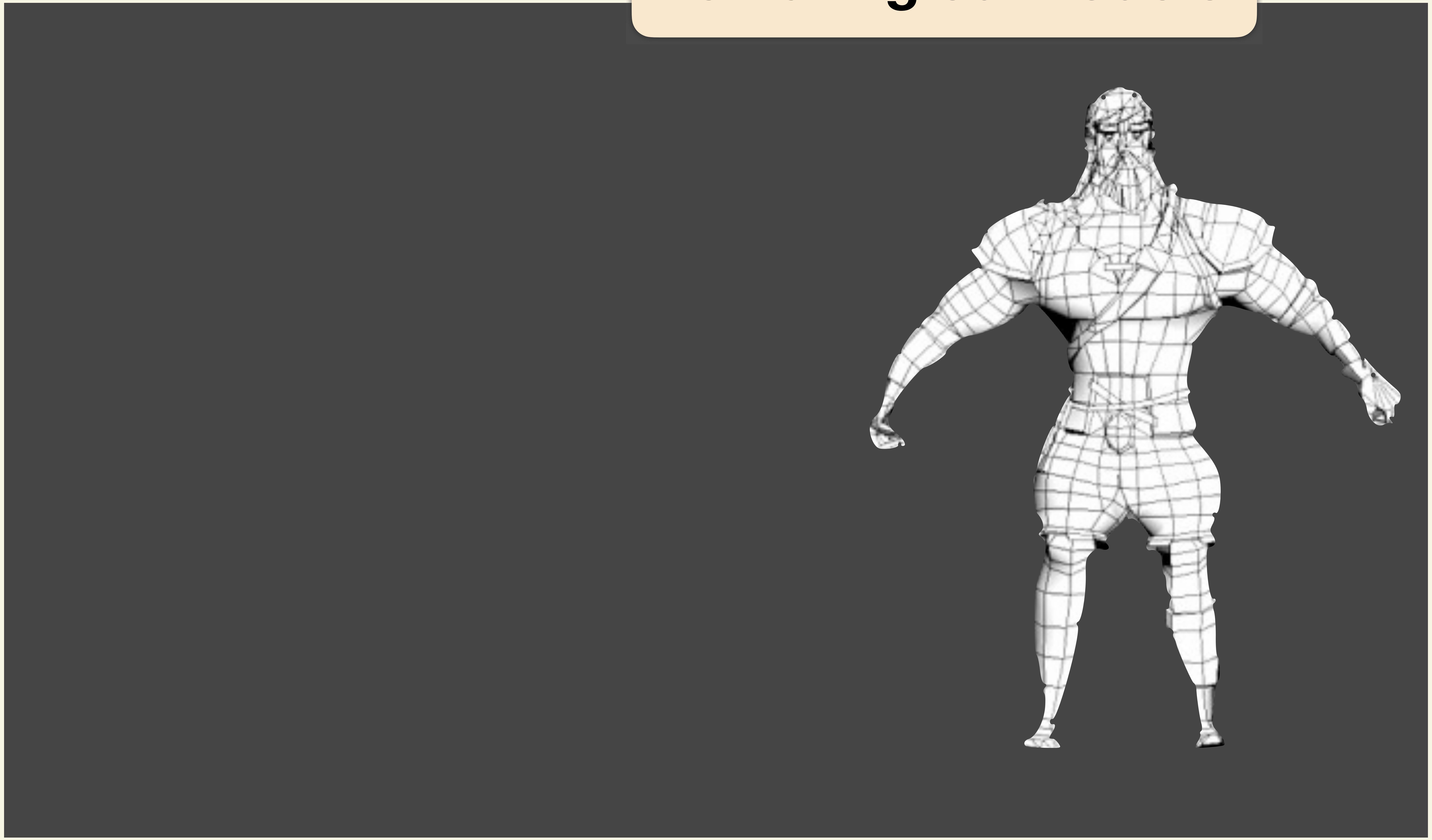
[Gao, Huth, Lescroart & Gallant 2015]





Applications of parameterization

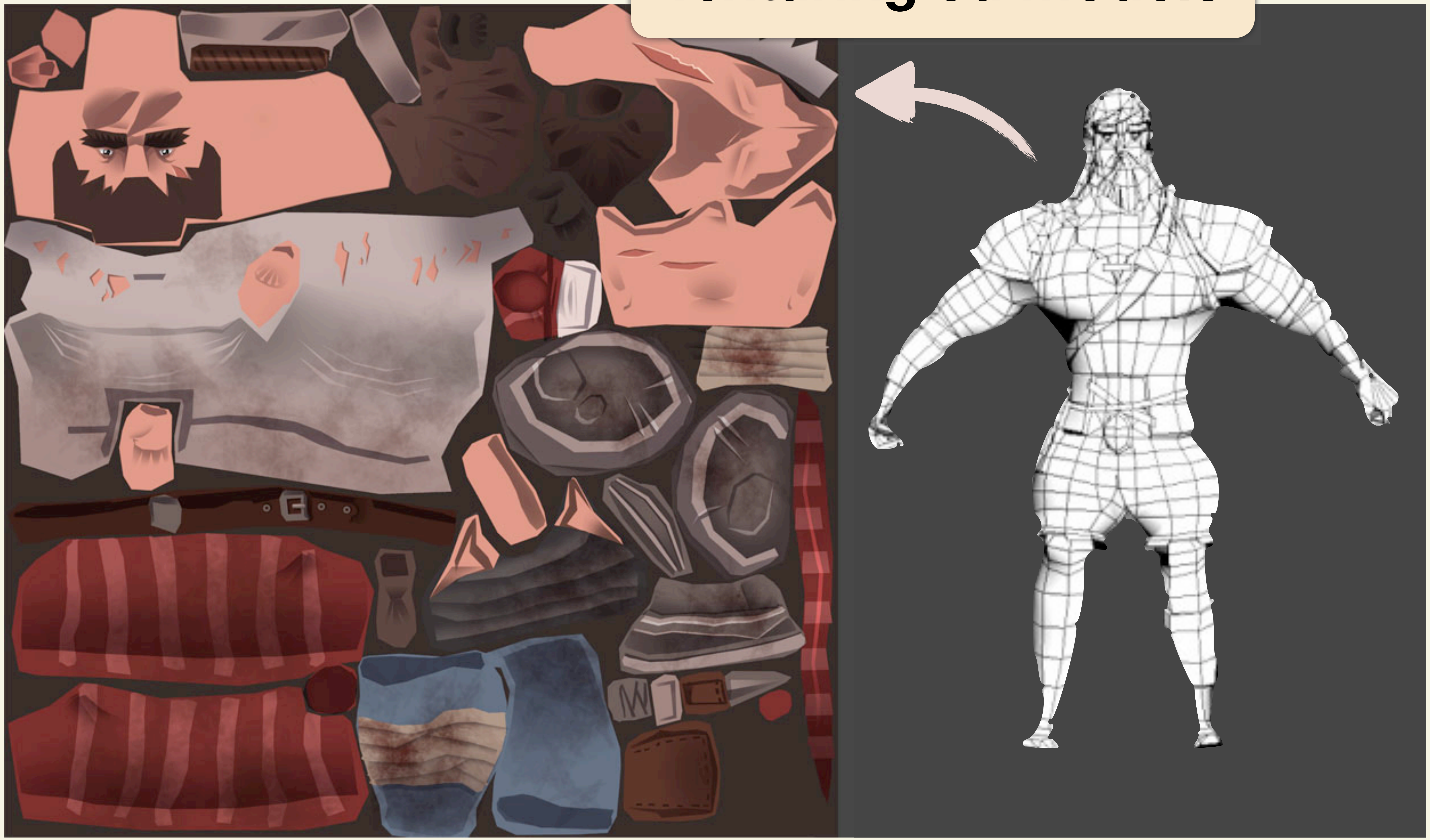
Texturing 3d models



[Timen 2012]

Applications of parameterization

Texturing 3d models



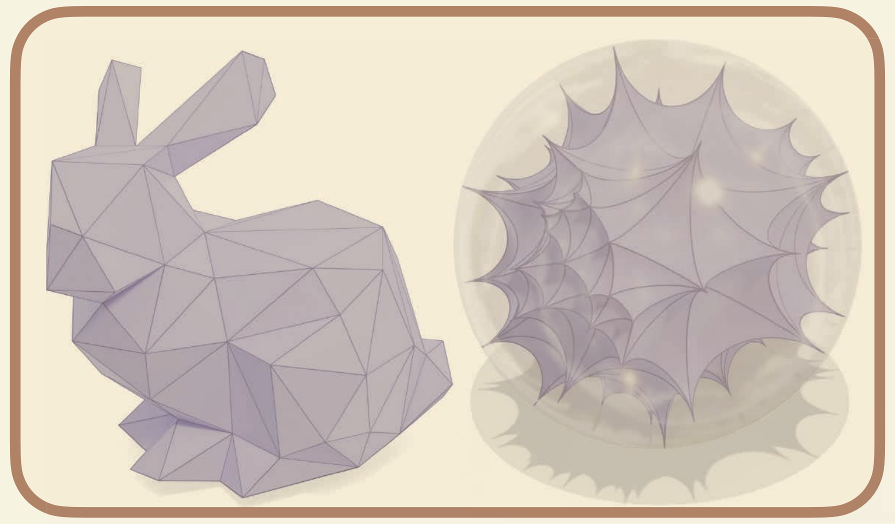
[Timen 2012]

Applications of parameterization

Texturing 3d models

[Timen 2012]

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


Parameterization

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

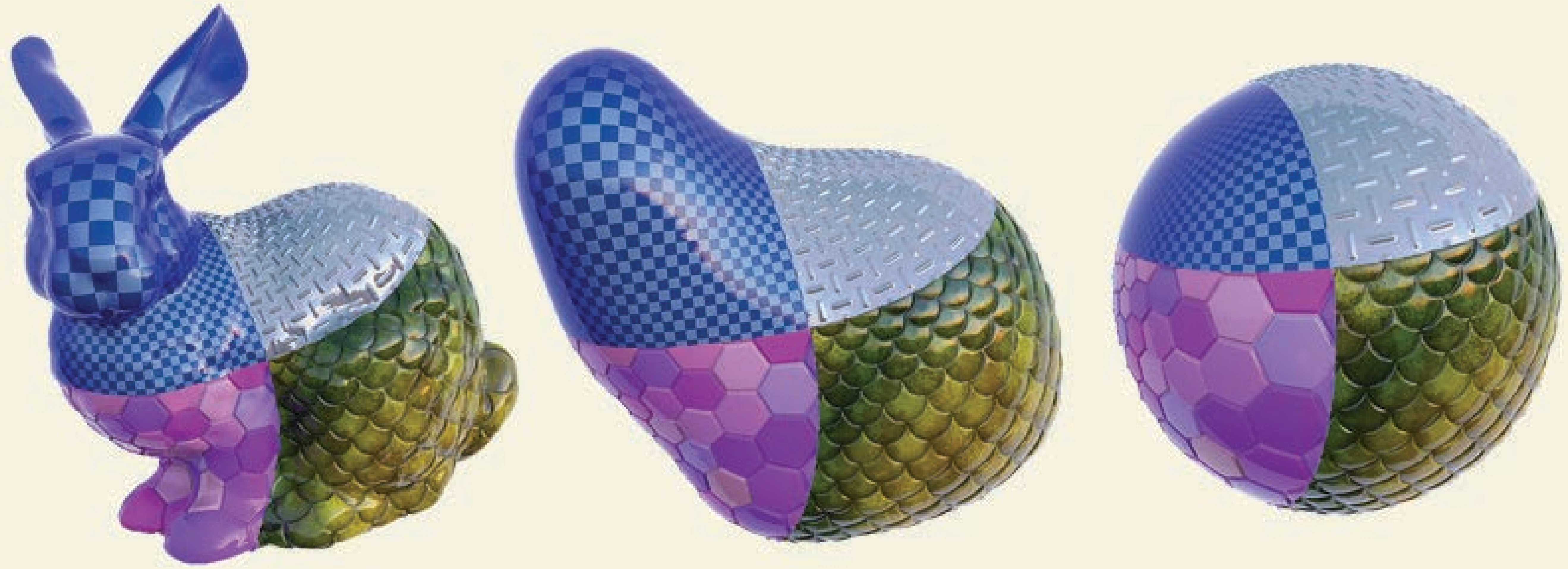
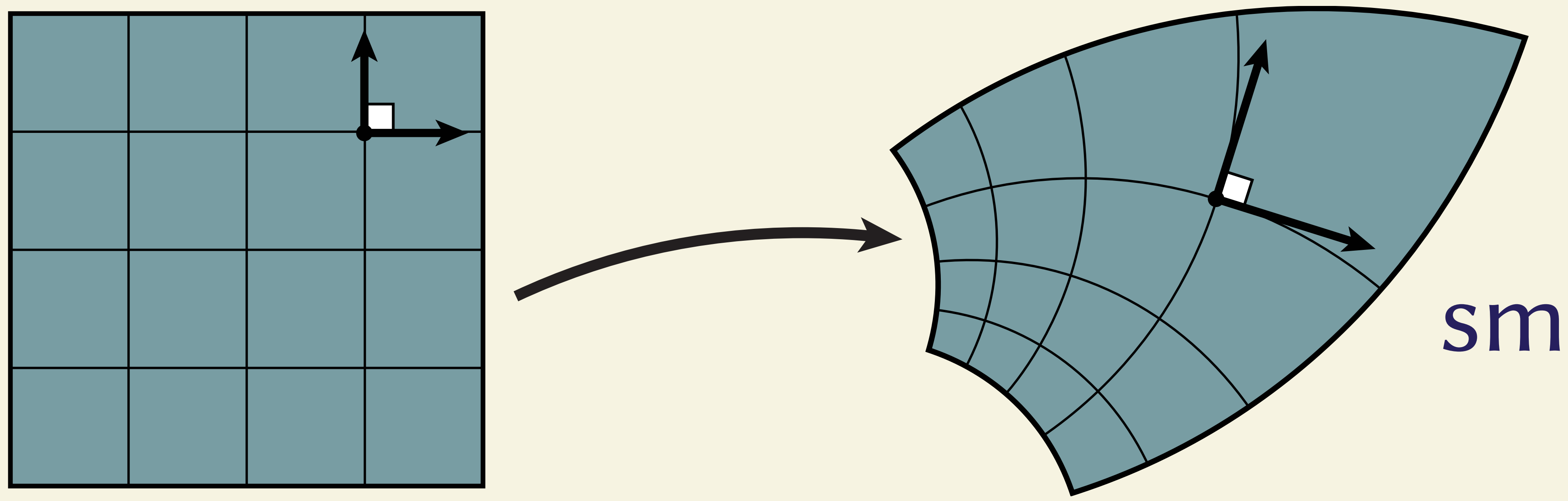


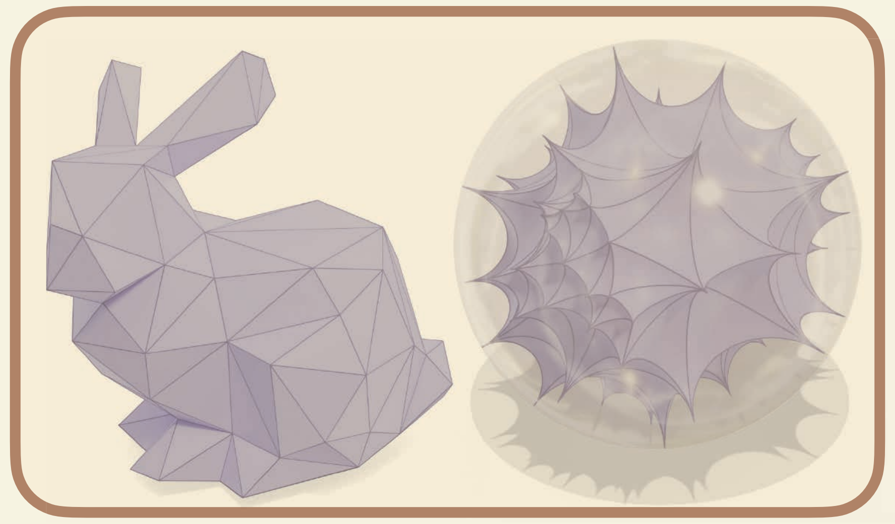
Image: [Crane, Pinkall & Schröder 2013]



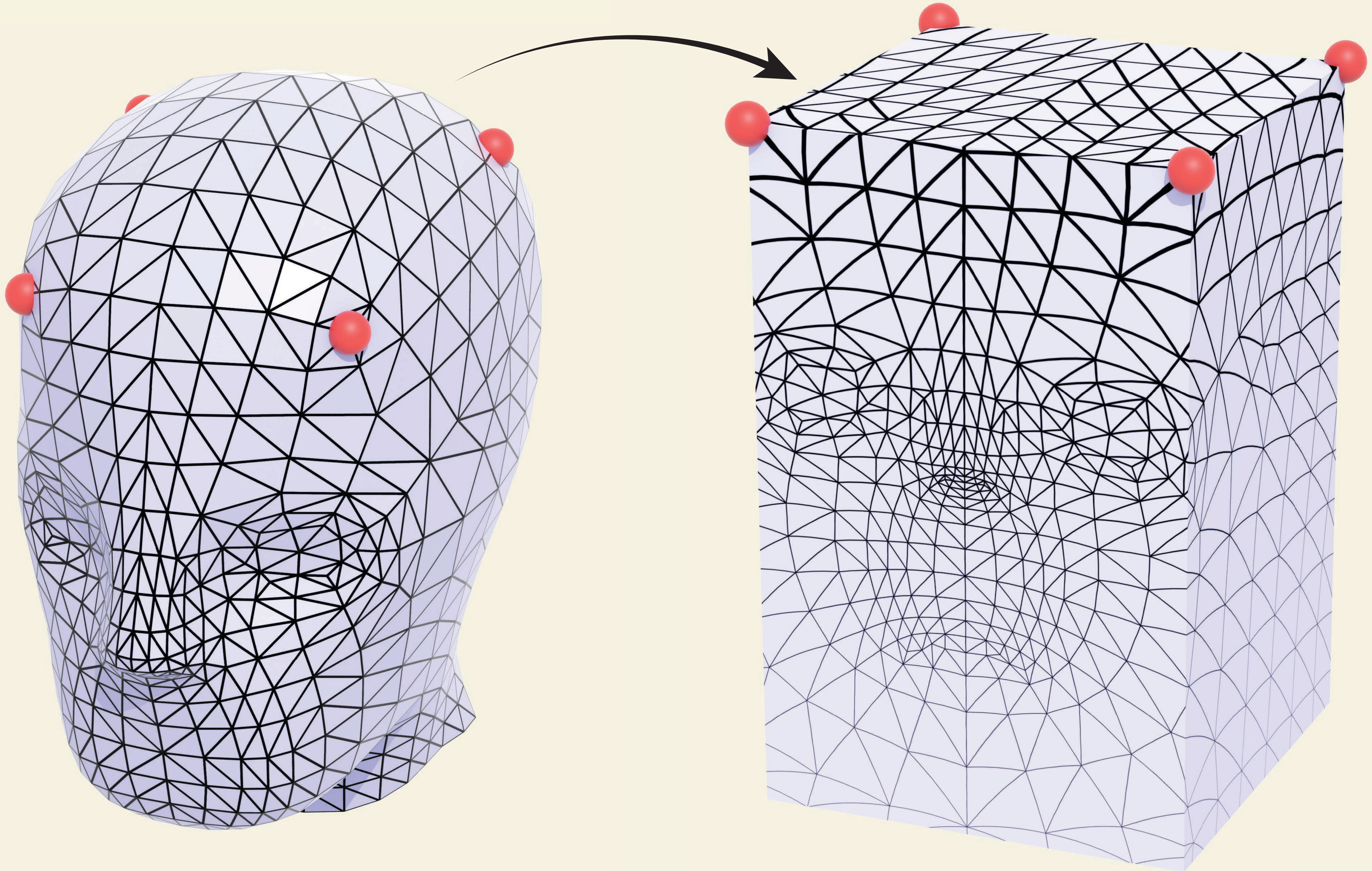
conformal map = angle-preserving

smooth maps with helpful properties

The discrete uniformization theorem
[Gu, Luo, Sun & Wu 2018; Springborn 2019]

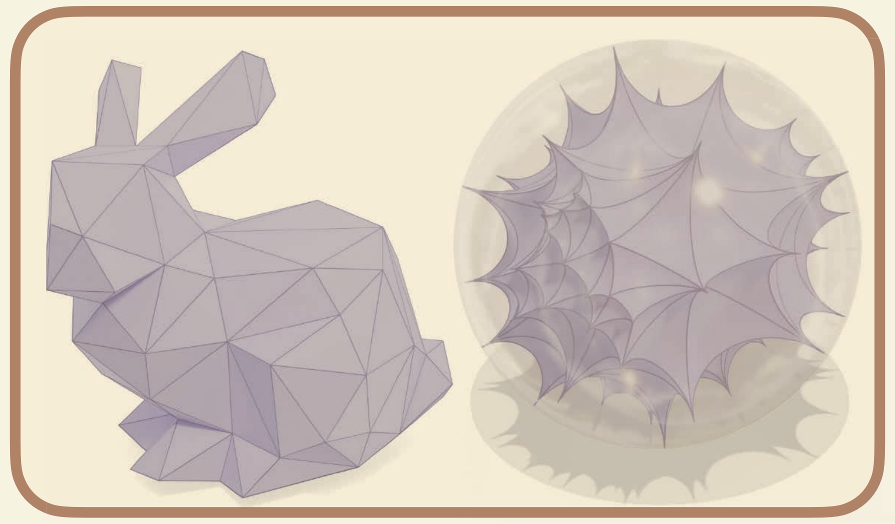


Any valid† vertex curvatures can be realized by some discrete conformal map.

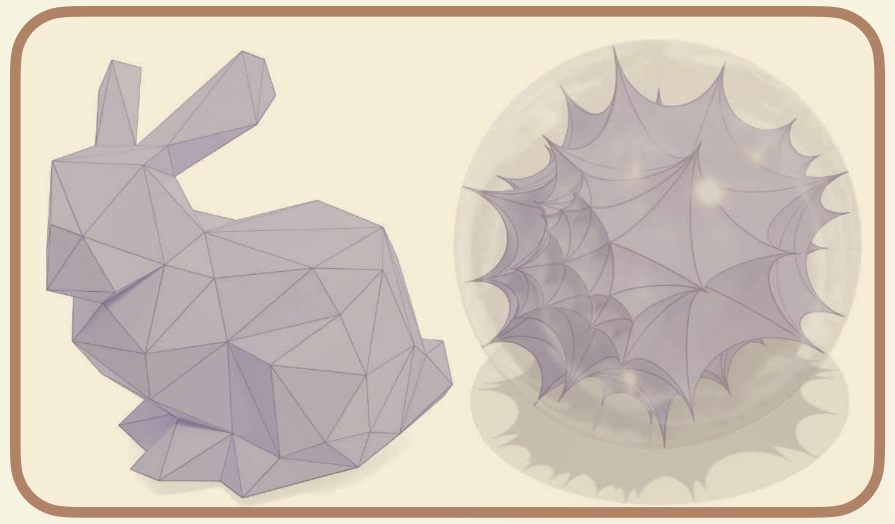


†*i.e.* $\leq 2\pi$ and satisfying Gauss-Bonnet

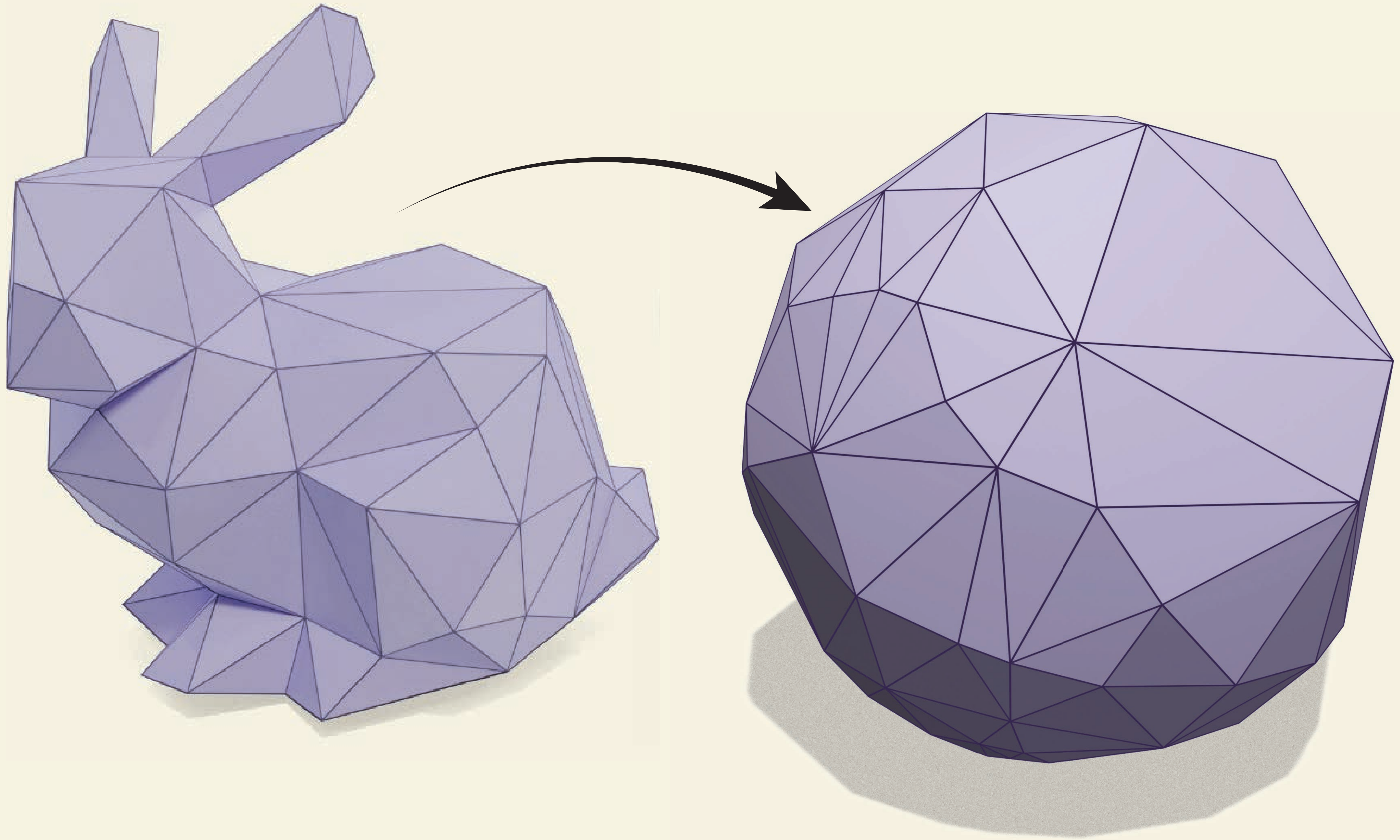
The discrete uniformization theorem
[Gu, Luo, Sun & Wu 2018; Springborn 2019]



The discrete spherical uniformization theorem
[Springborn 2019]

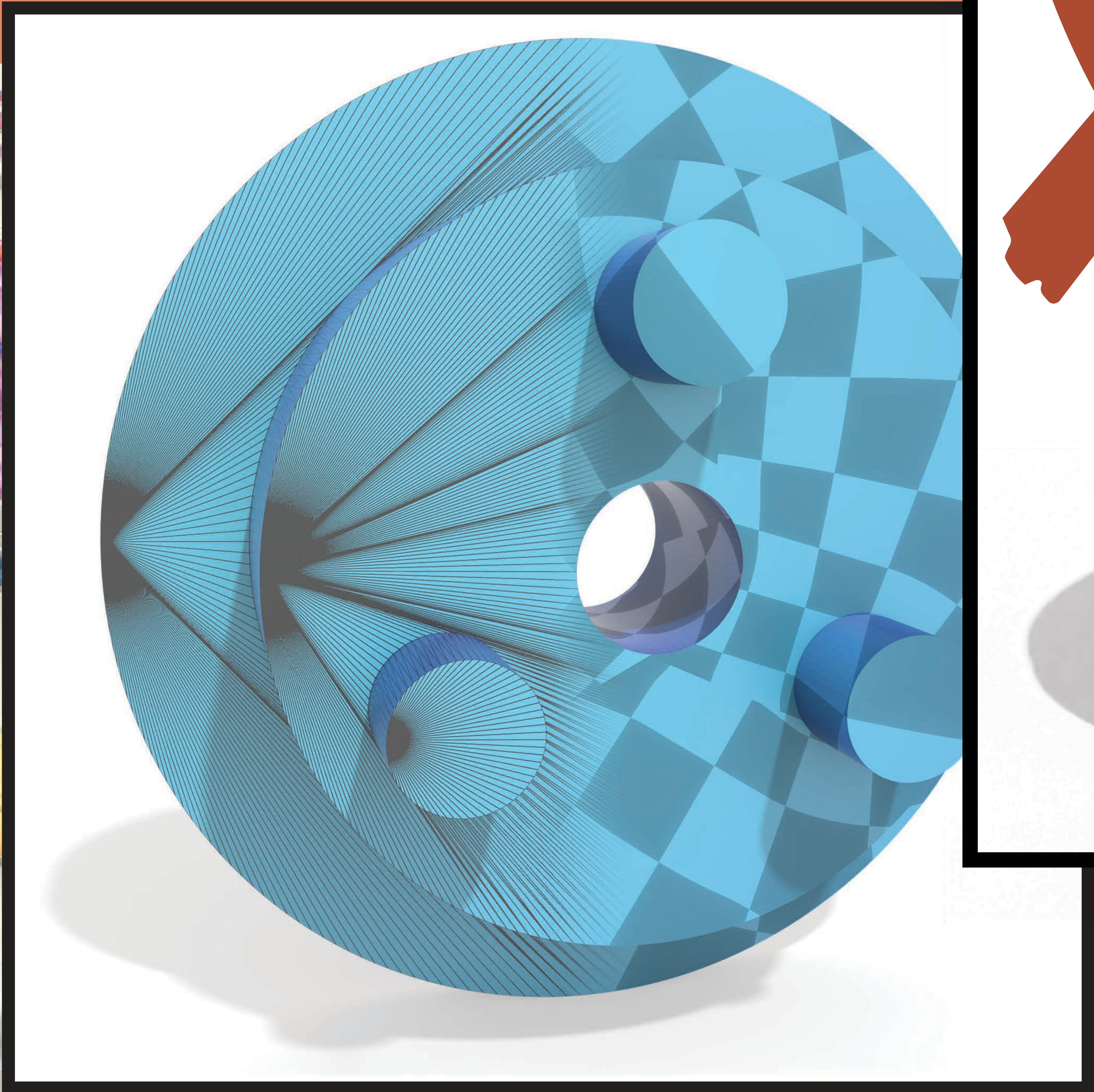


Any simply-connected triangle mesh is discretely conformally equivalent to a mesh whose vertices lie on the unit sphere

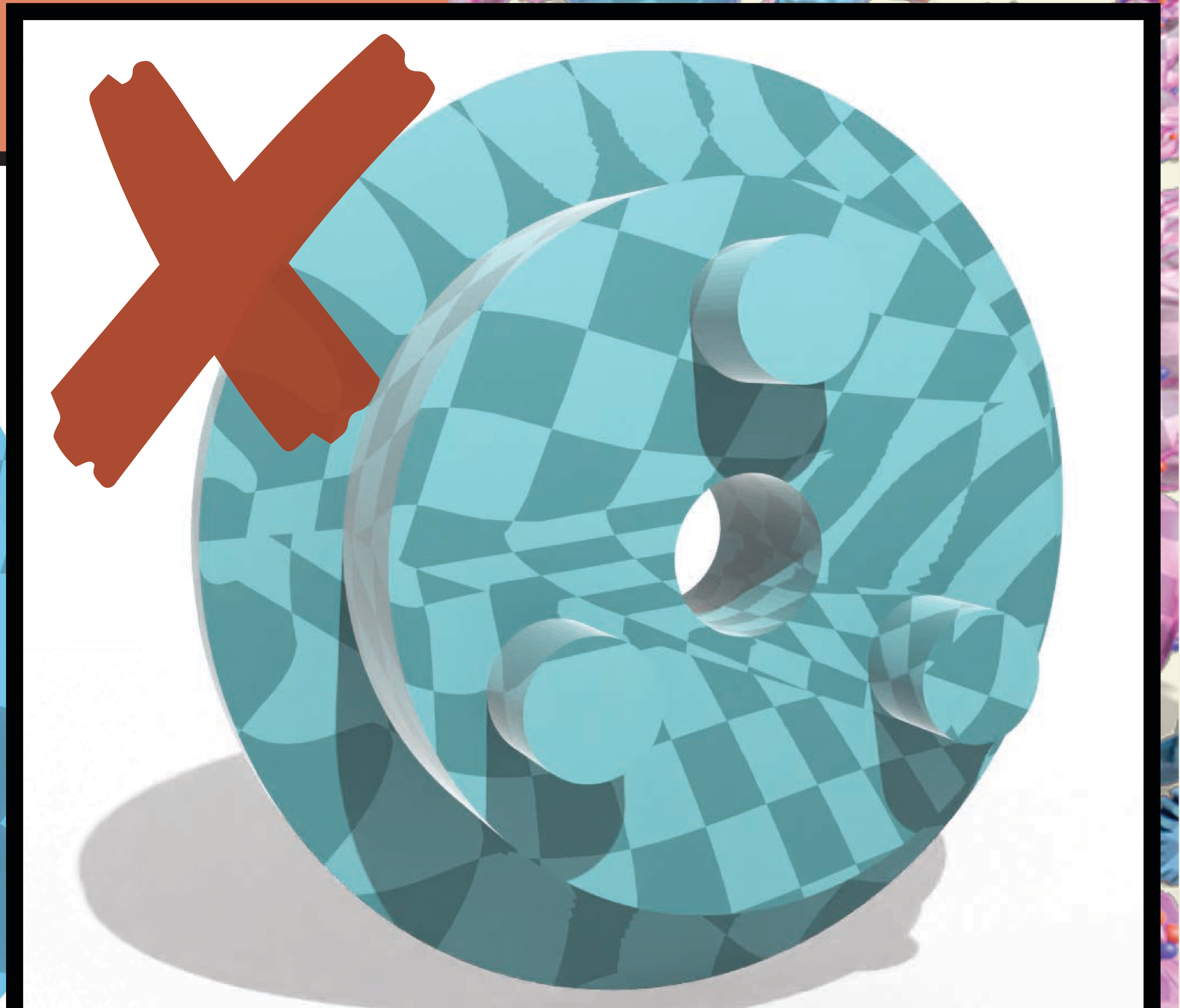


Discrete uniformization in action

[Gillespie, Springborn, & Crane. 2021]



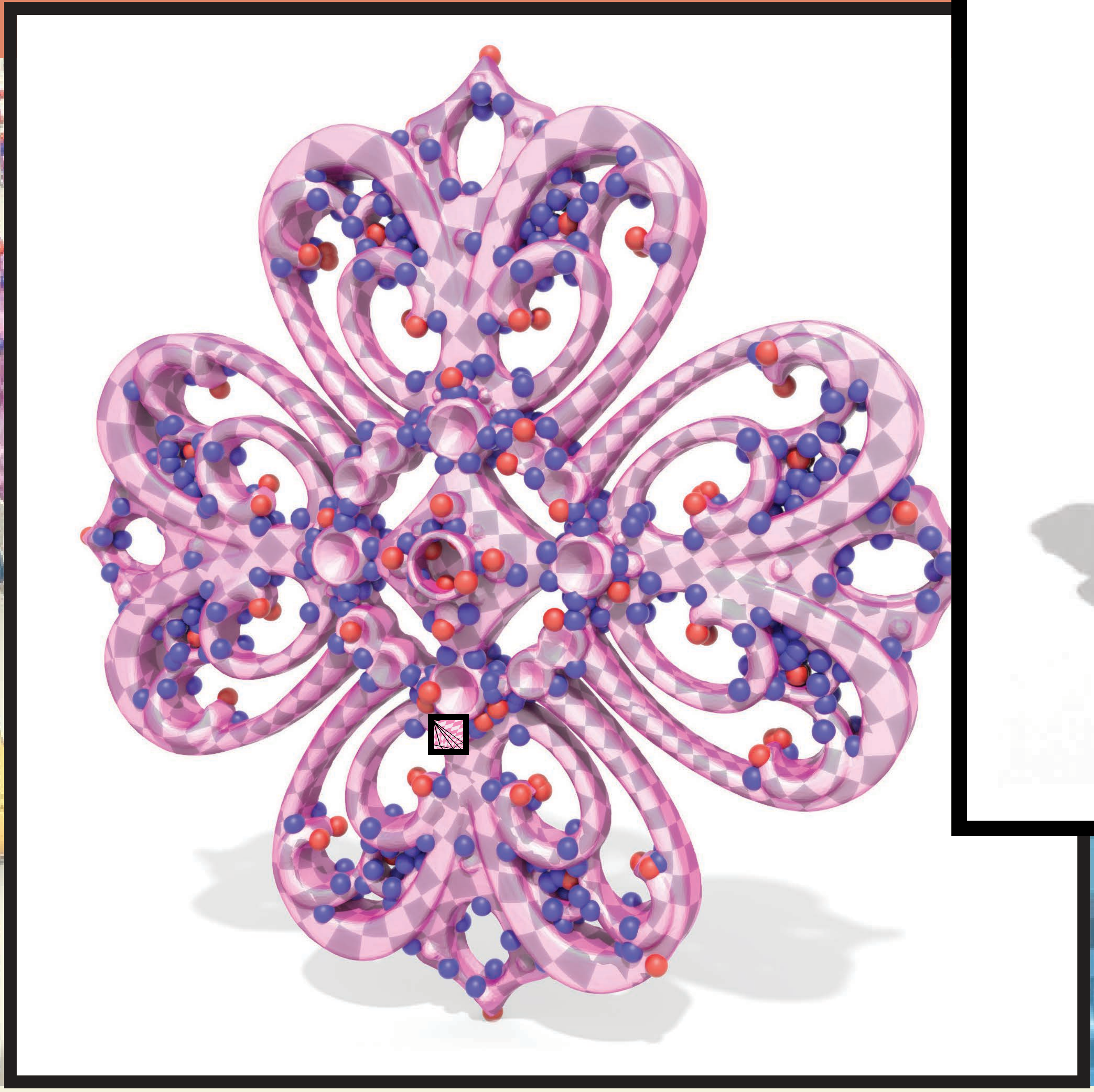
low-quality meshes



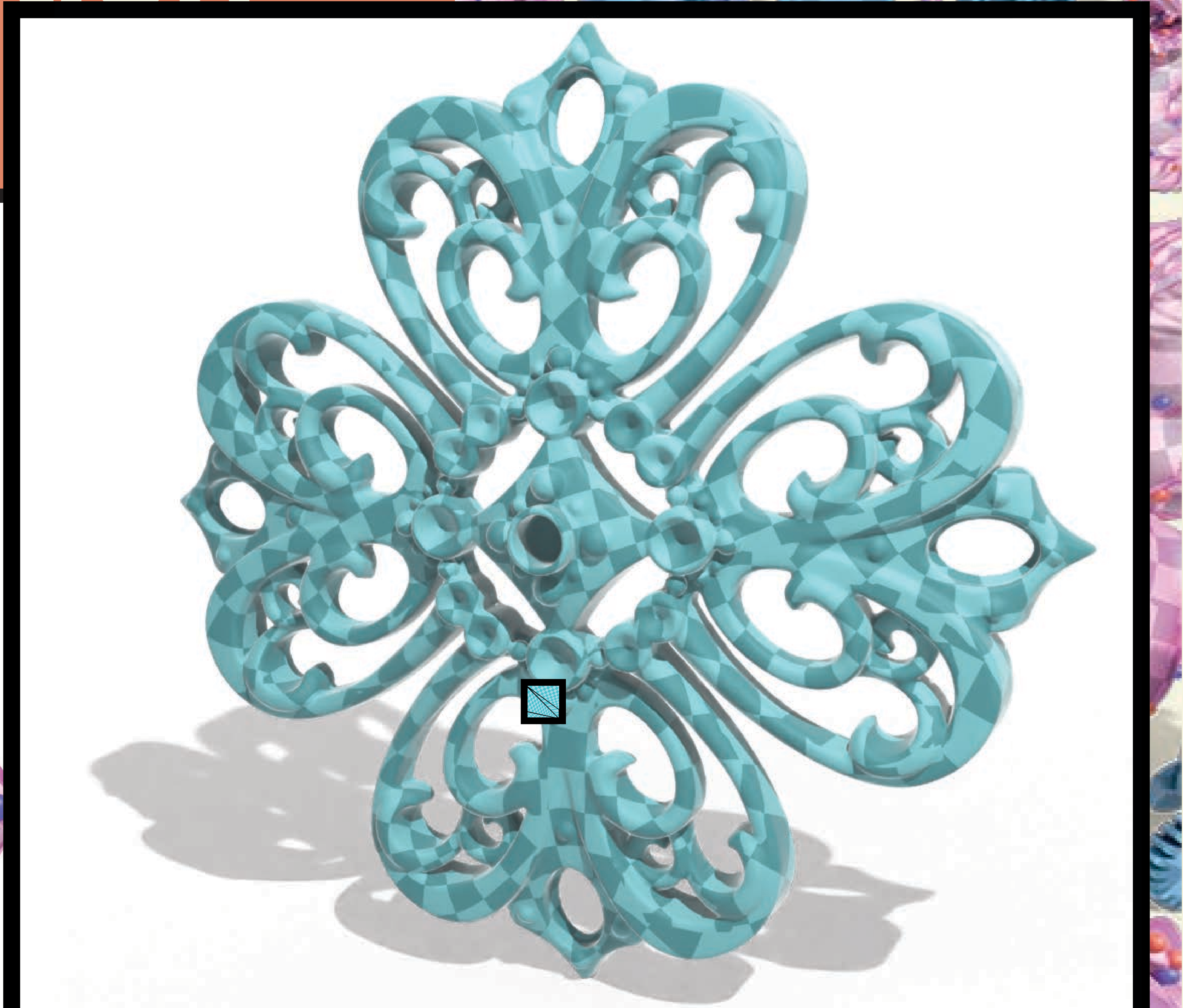
[Sawhney & Crane 2017]

Discrete uniformization in action

[Gillespie, Springborn, & Crane. 2021]



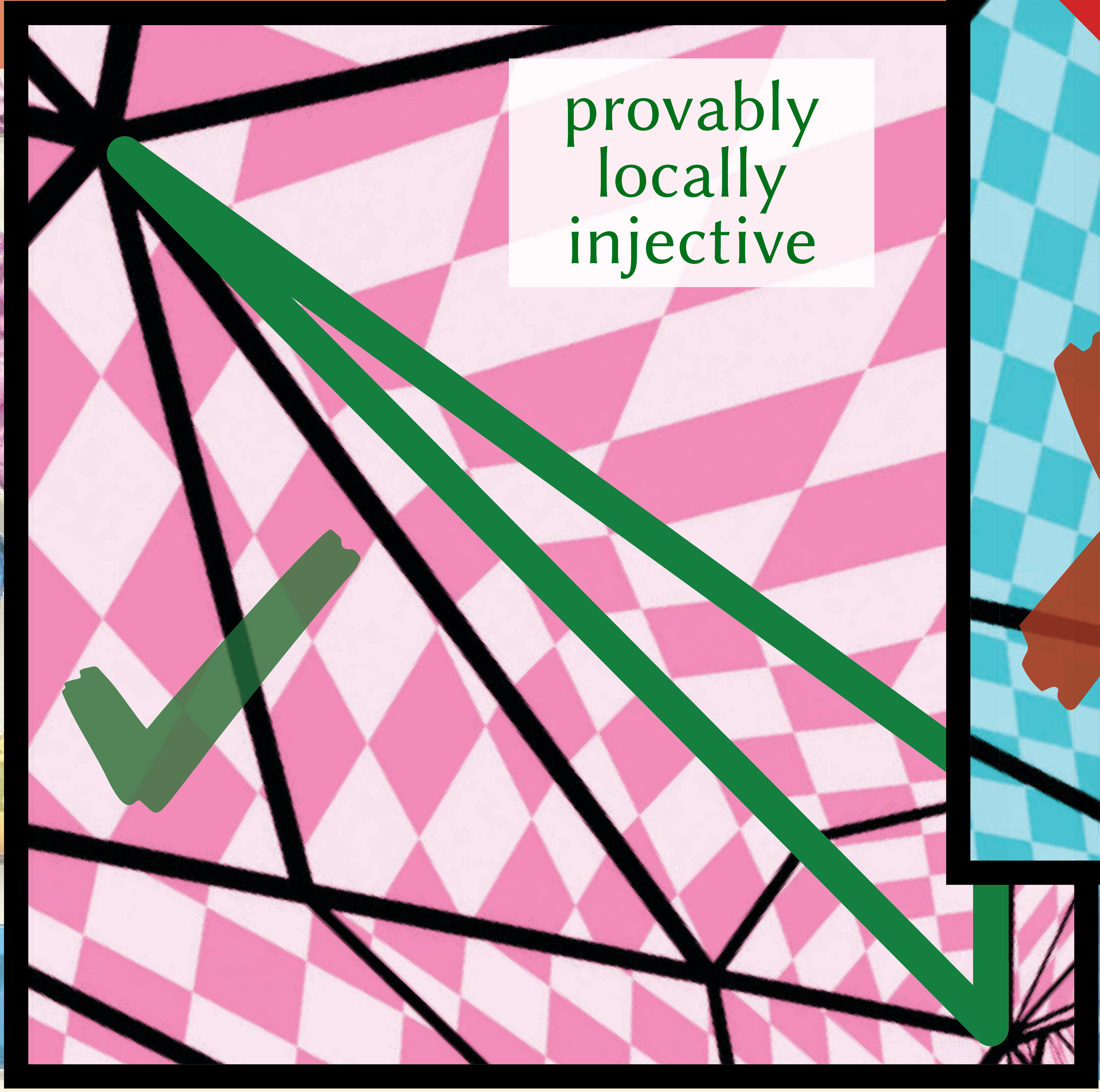
difficult cone constraints



[Sawhney & Crane 2017]

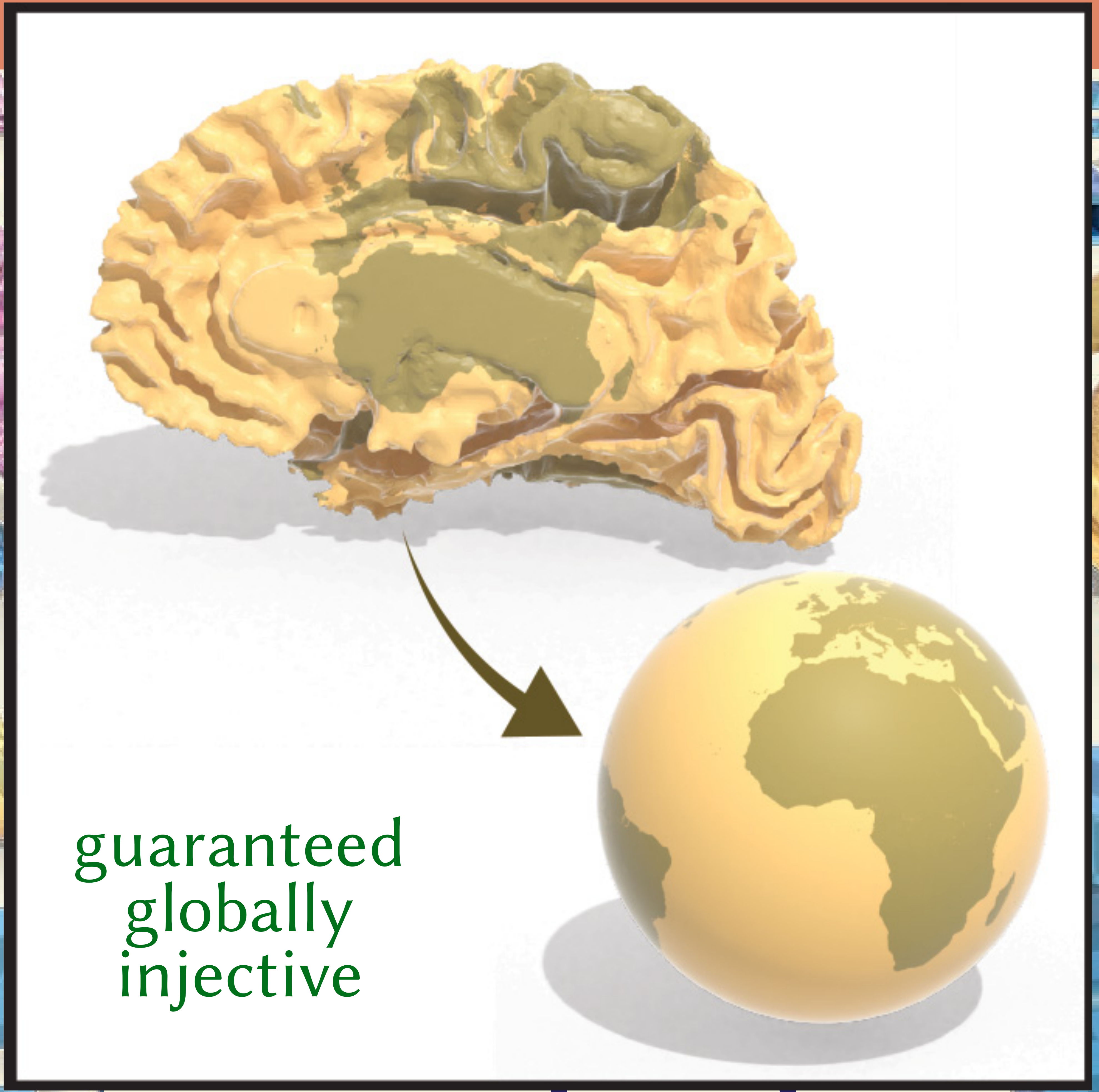
Discrete uniformization in action

[Gillespie, Springborn, & Crane. 2021]



Discrete uniformization in action

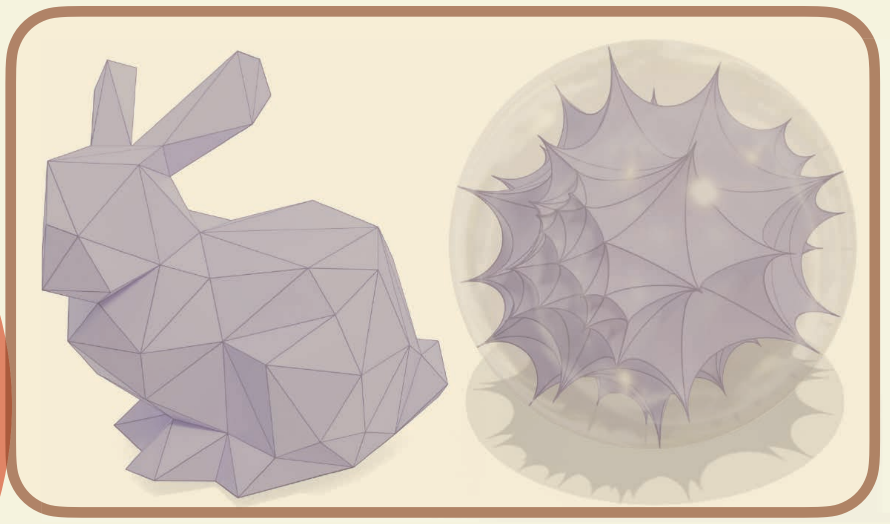
[Gillespie, Springborn, & Crane. 2021]



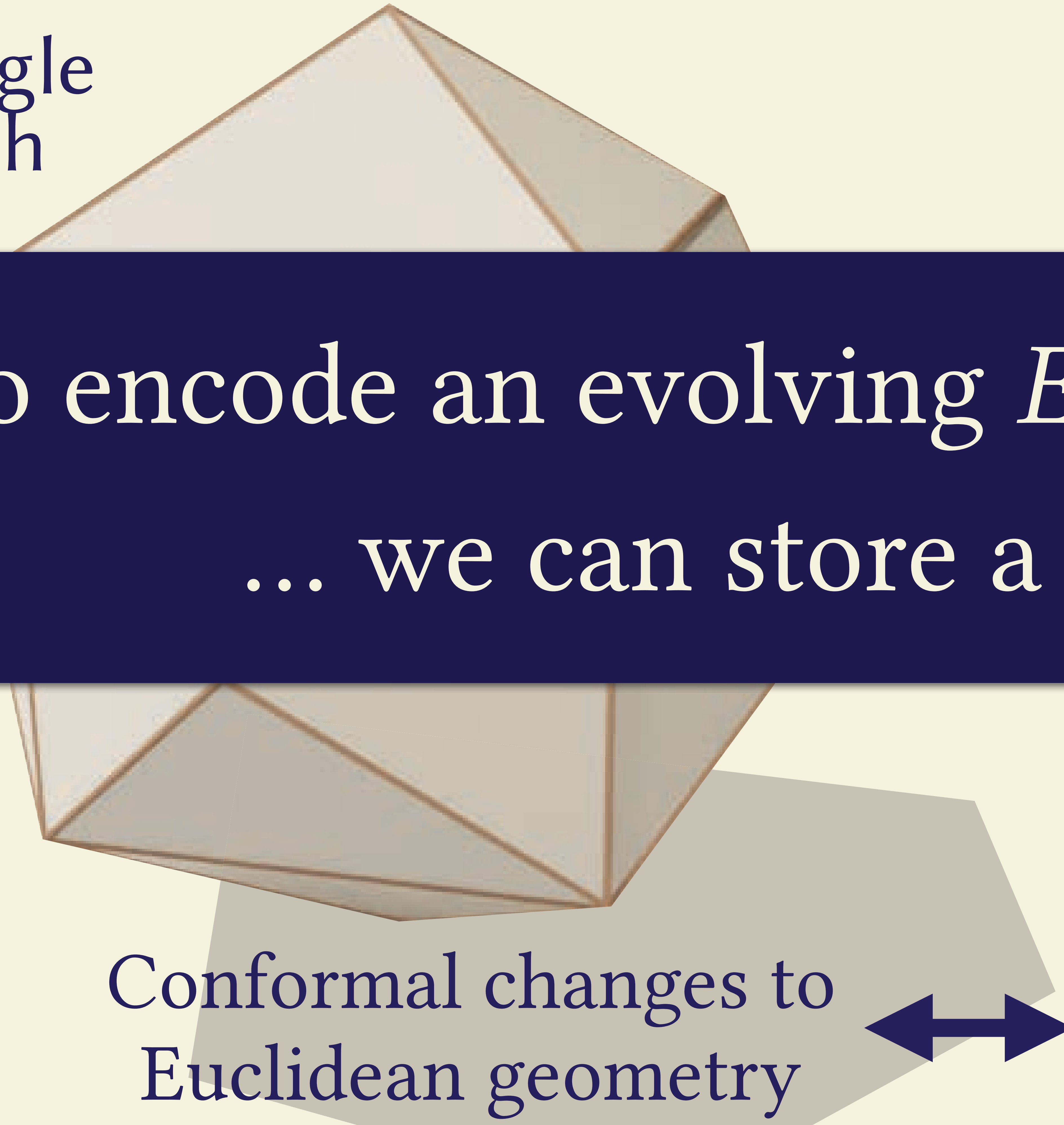
maps to the sphere

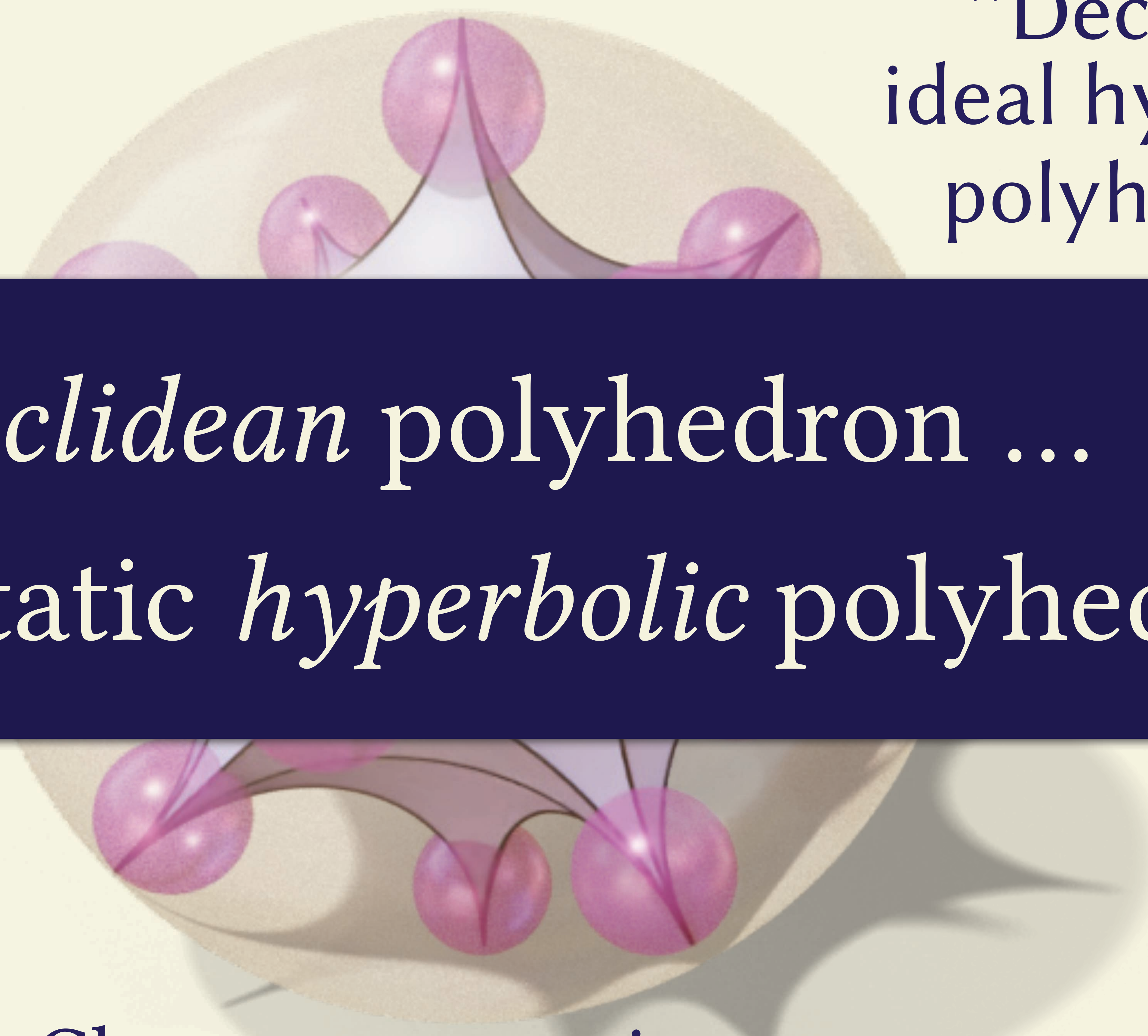
Triangle mesh ⟷ hyperbolic polyhedron

[Bobenko, Pinkall & Springborn 2010]



Parameterization



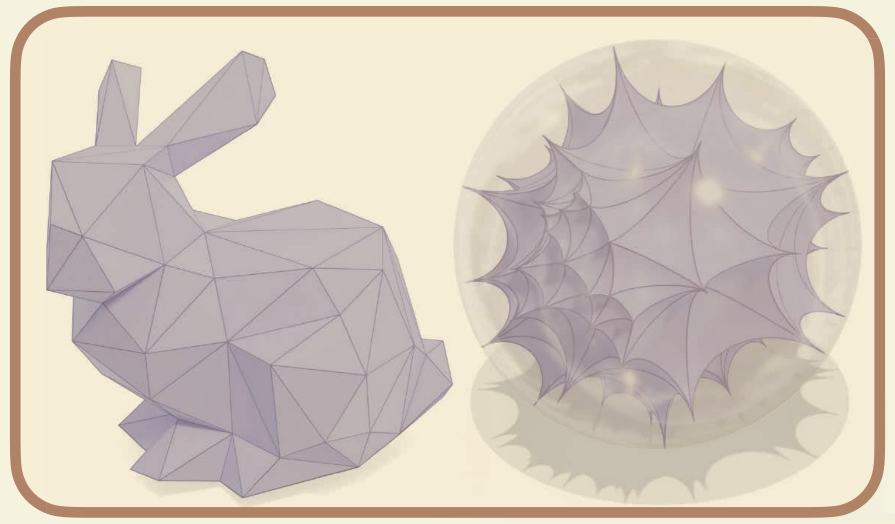


To encode an evolving *Euclidean* polyhedron …

… we can store a static *hyperbolic* polyhedron.

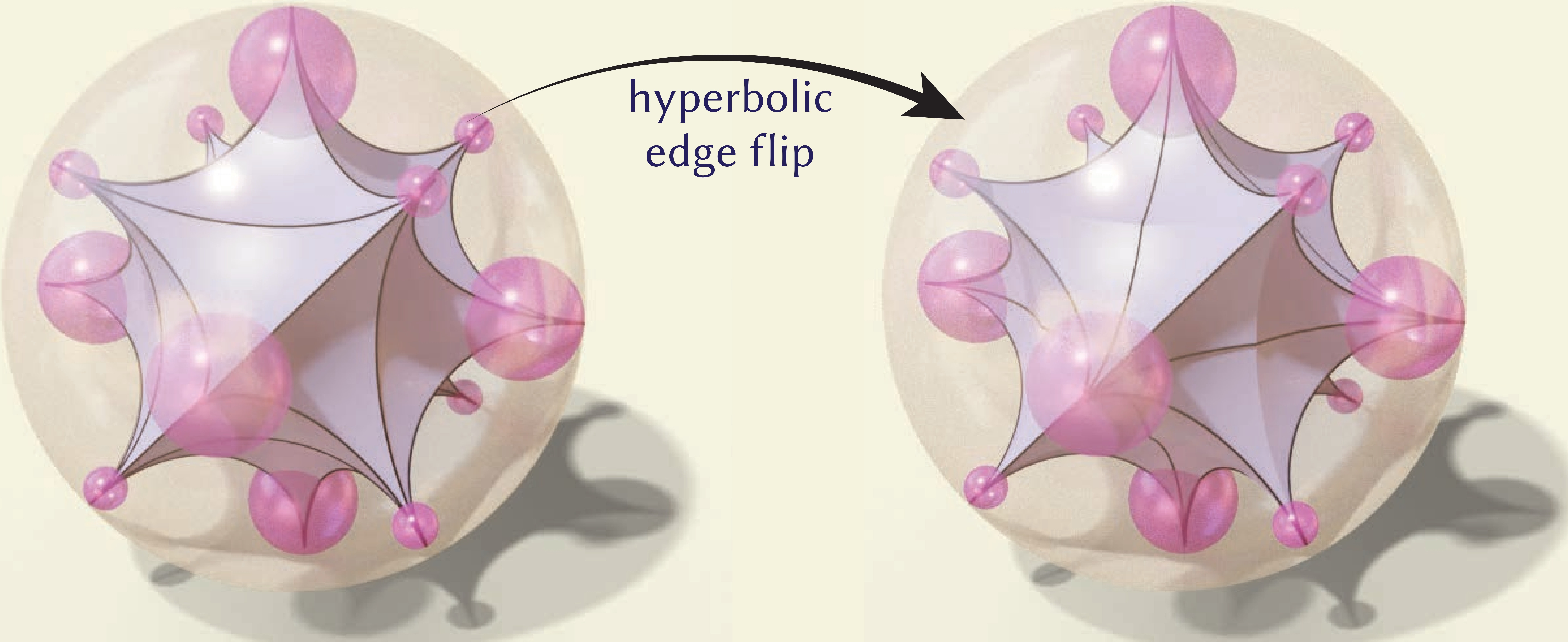
Conformal changes to Euclidean geometry ⟷ Changes preserving hyperbolic geometry

Intrinsic triangulations of hyperbolic polyhedra

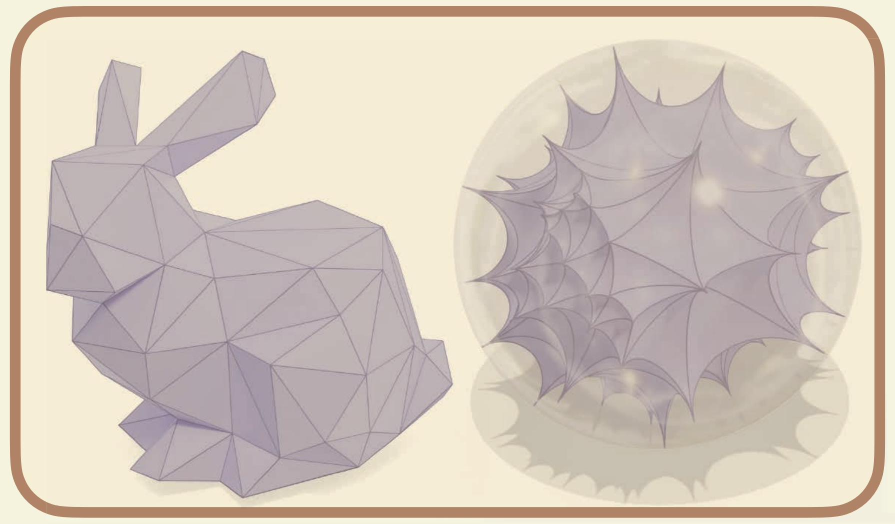


Parameterization

similar to ordinary intrinsic triangulations

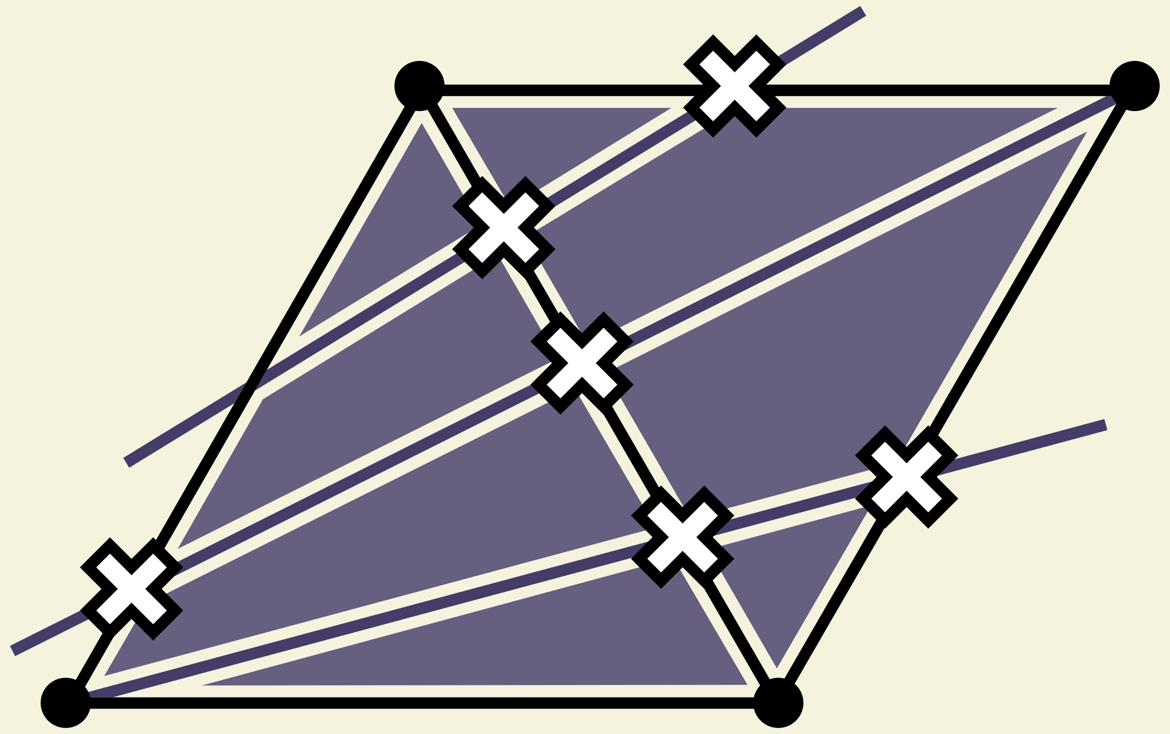


Data structures for hyperbolic intrinsic triangulations



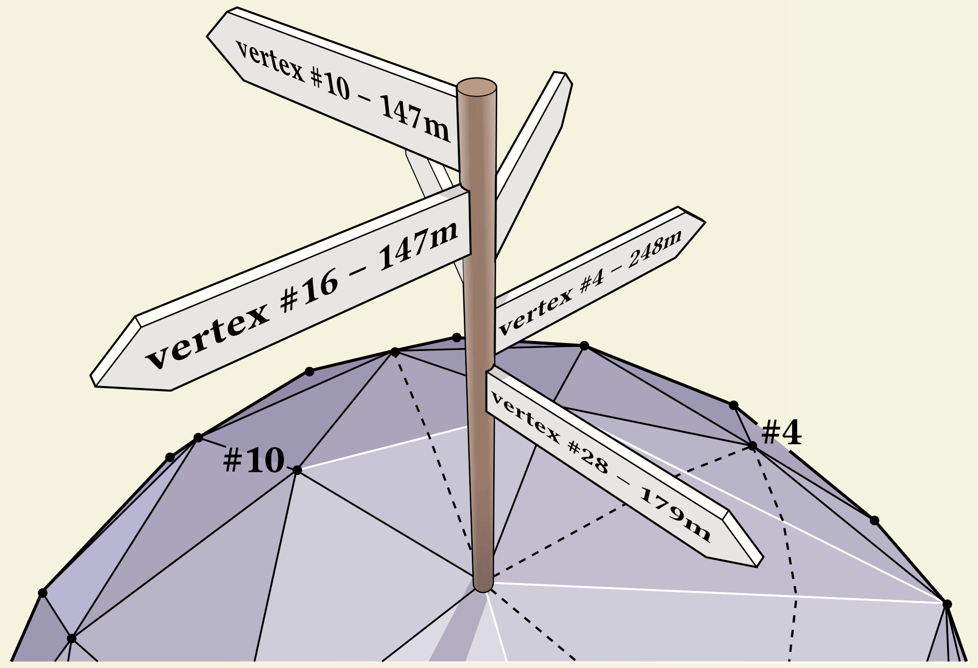
Parameterization

- Existing data structures naturally generalize



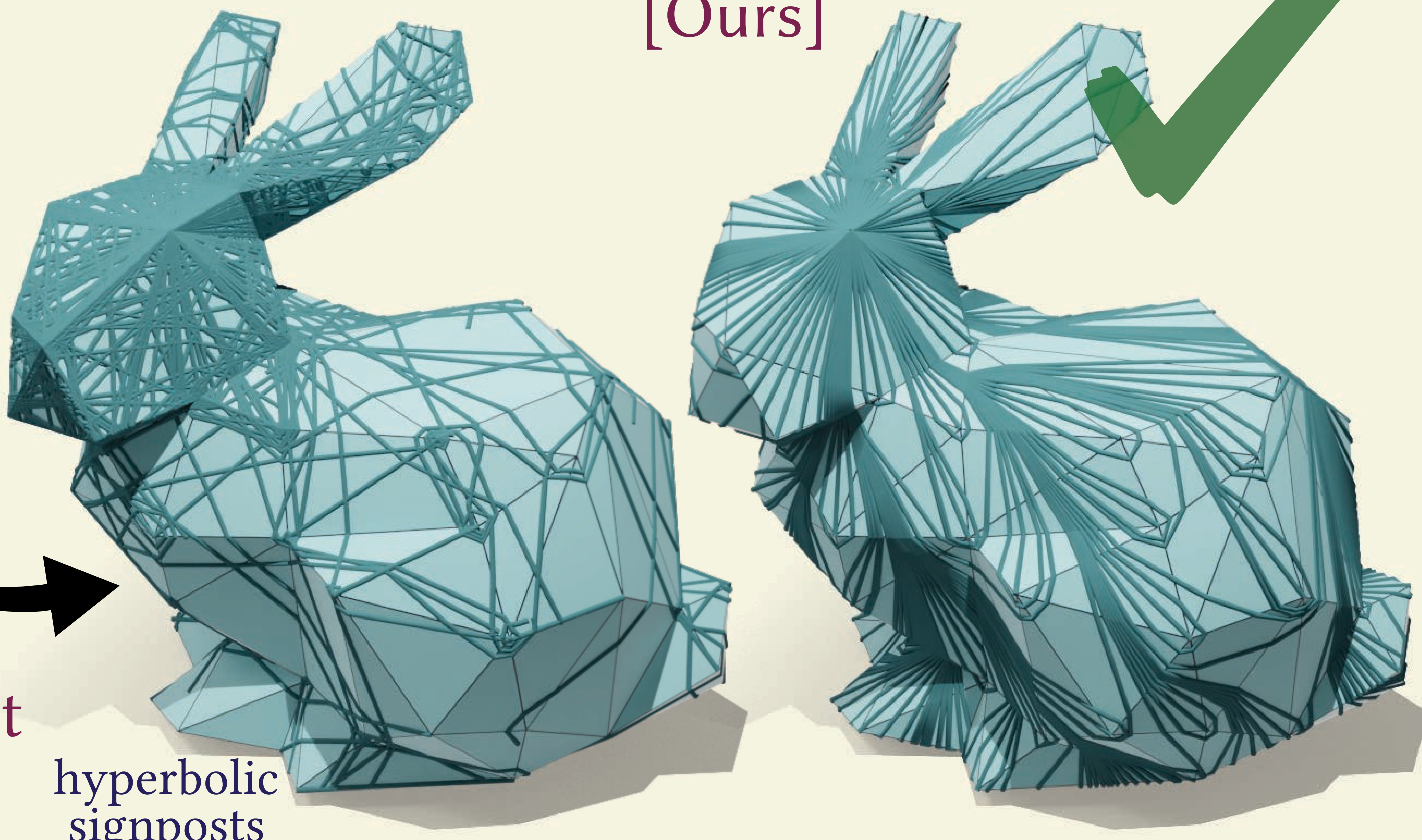
[Fisher, Springborn, Bobenko & Schröder 2006]

prohibitively complex



[Sharp, Soliman & Crane 2019]

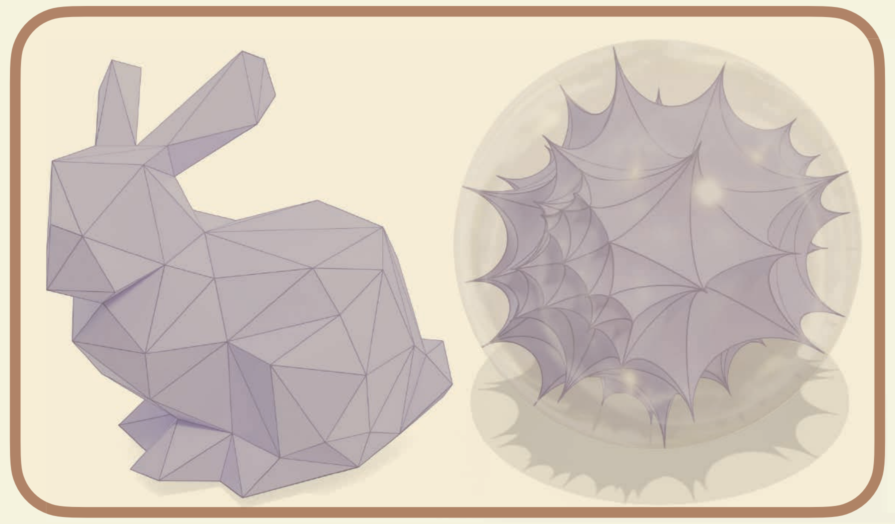
floating point errors



hyperbolic signposts

integer coordinates [Ours]

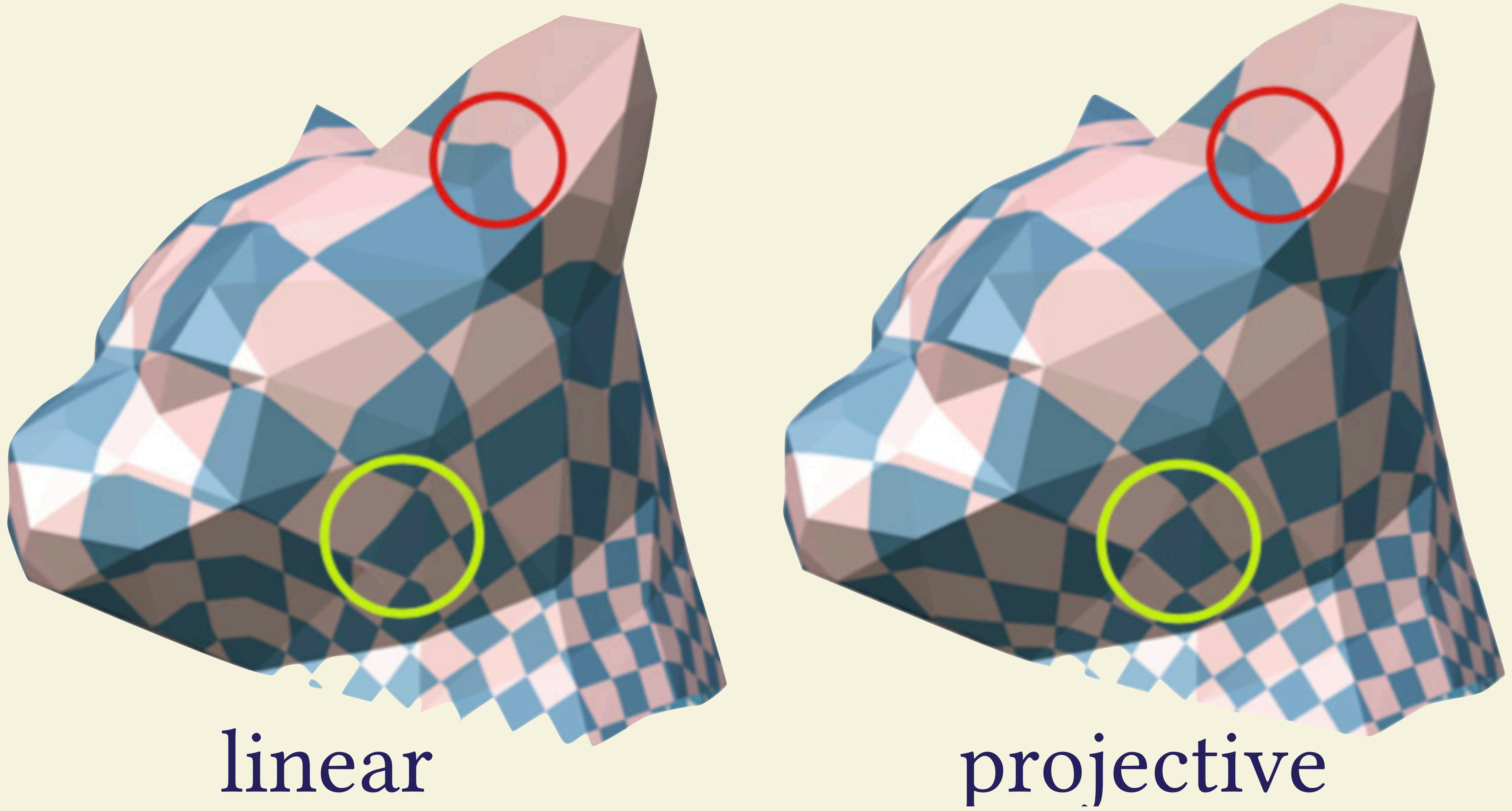
Projective interpolation

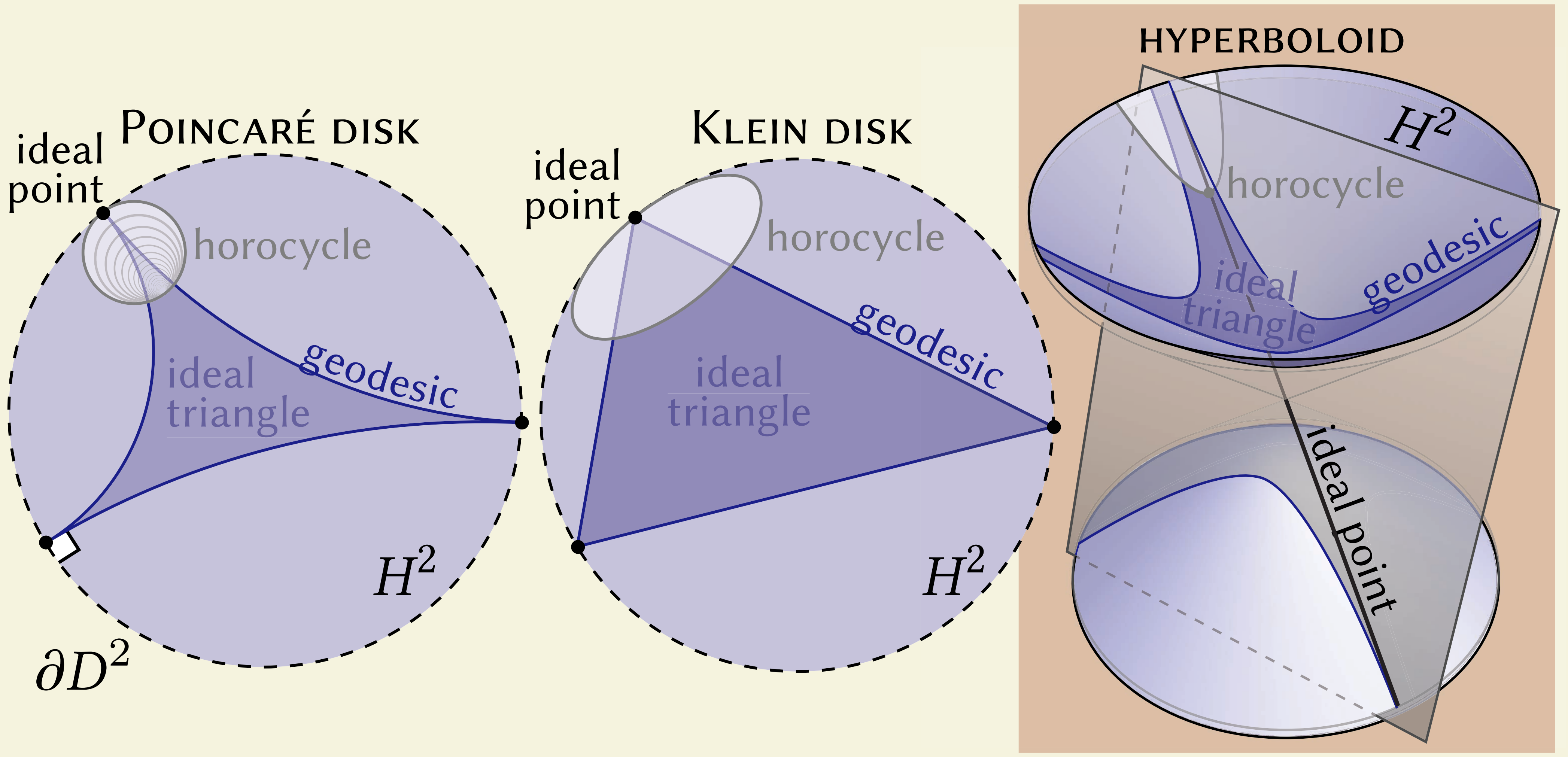


Parameterization

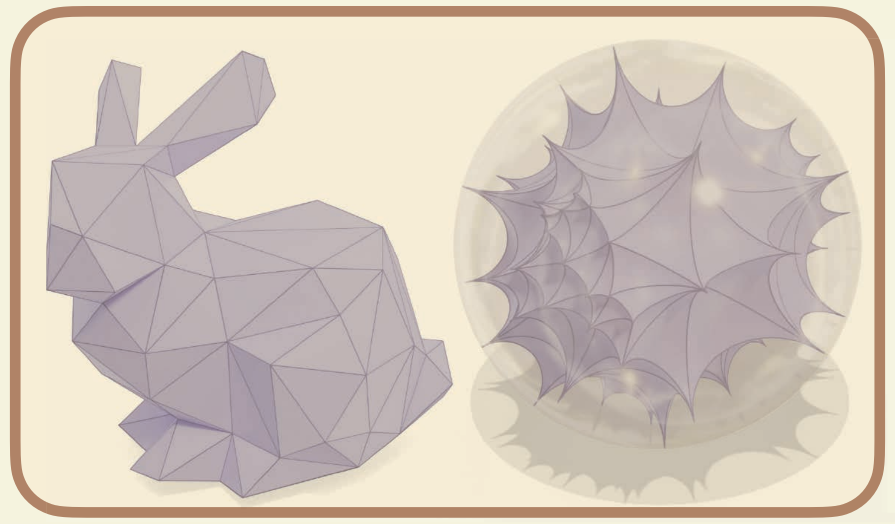
- [Springborn, Schröder & Pinkall 2008]: projective interpolation
  - Hyperbolic isometry
- [Ours]: novel projective interpolation using the hyperboloid model

Image: [Springborn, Schröder & Pinkall 2008]

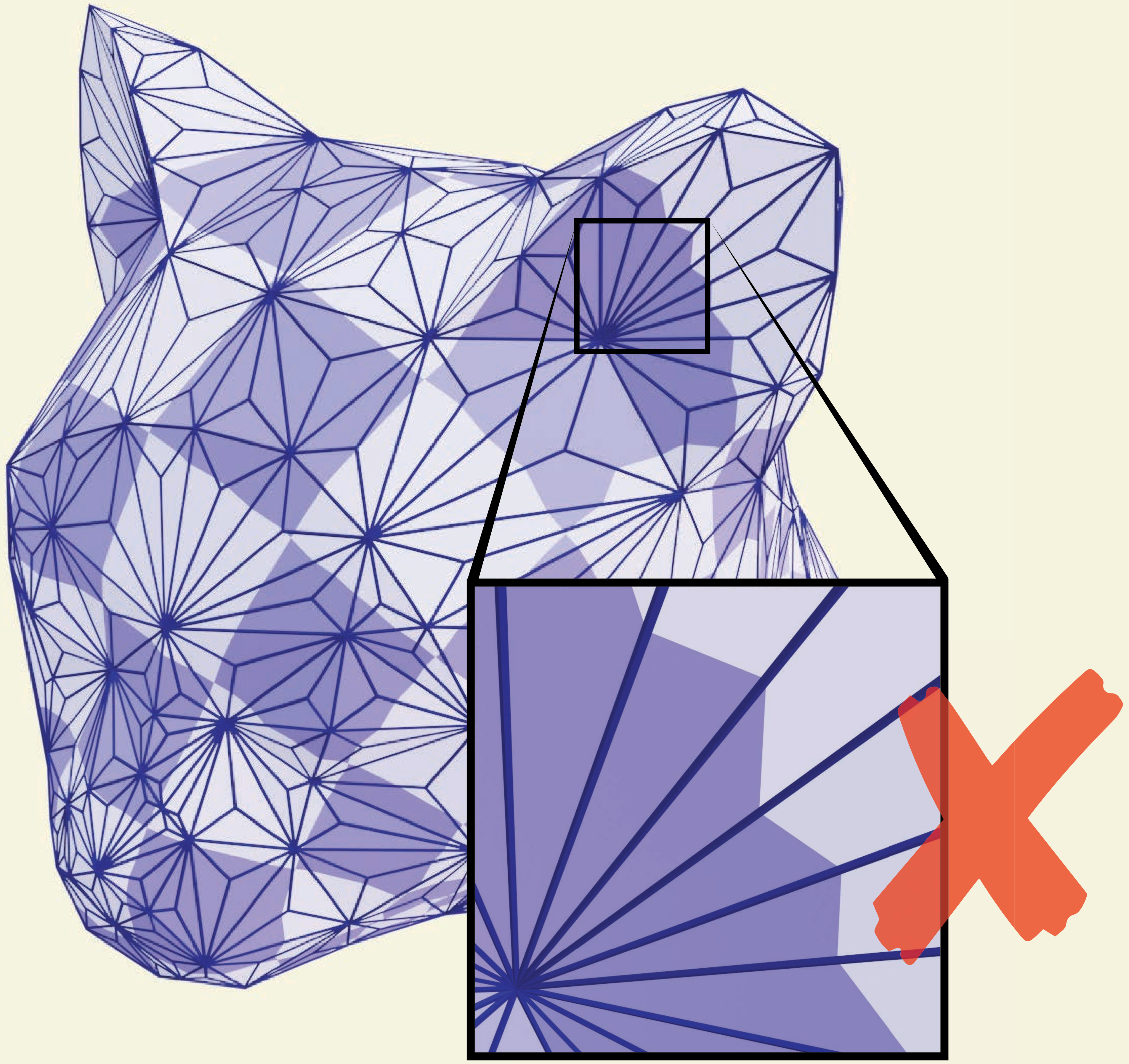




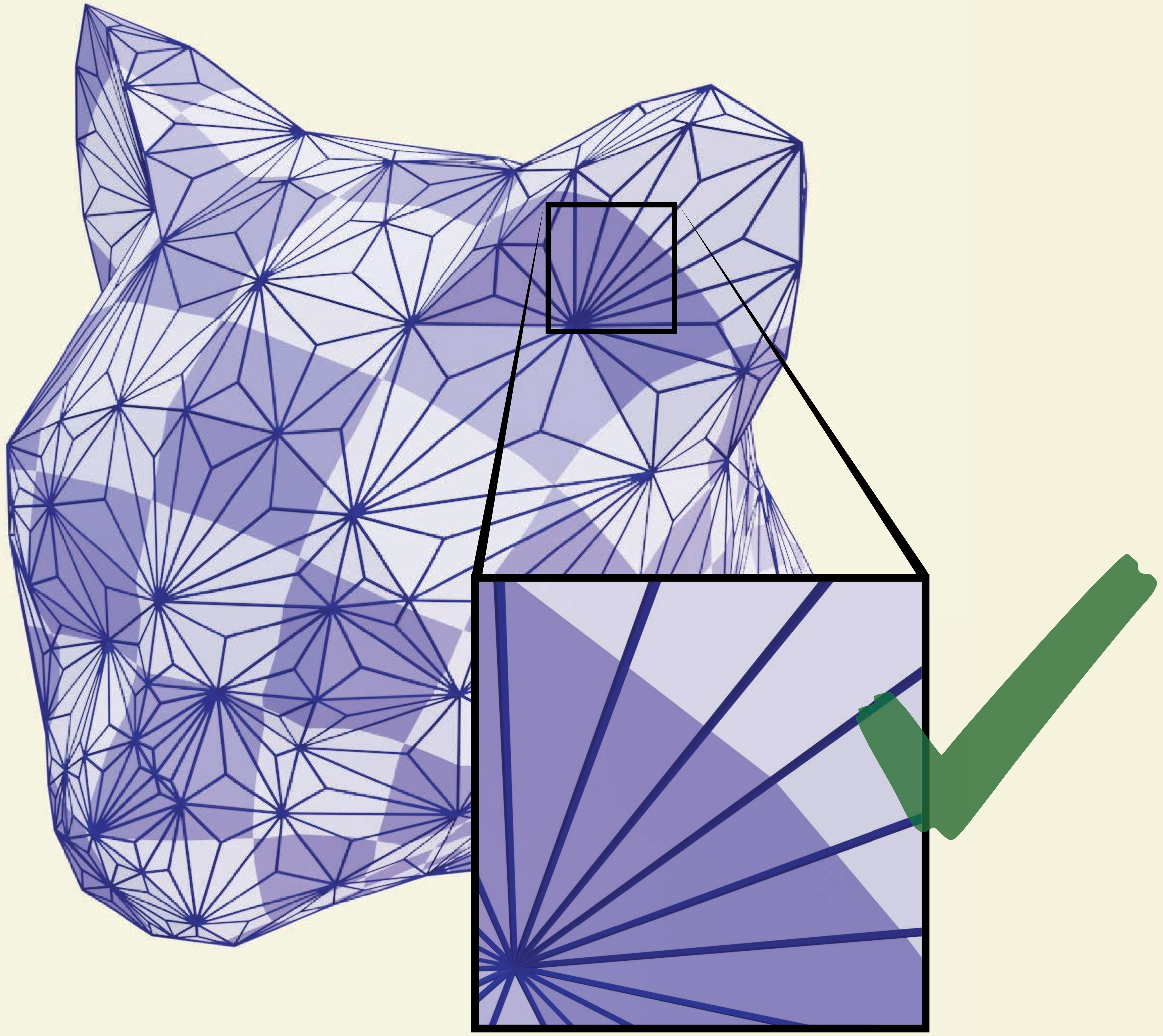
Variable triangulation > fixed triangulation



Parameterization



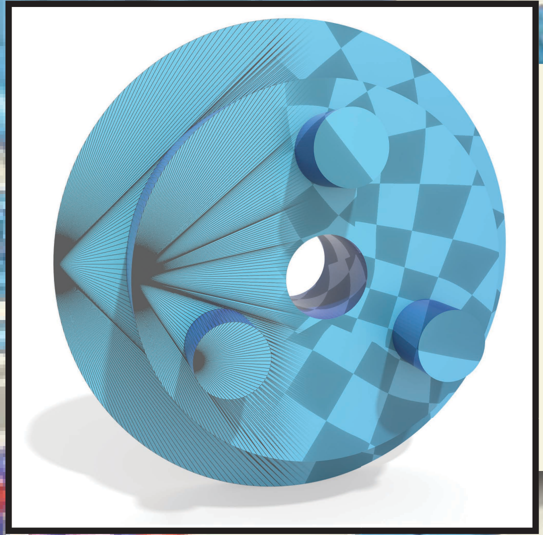
Fixed triangulation (CETM)



Variable triangulation (CEPS)

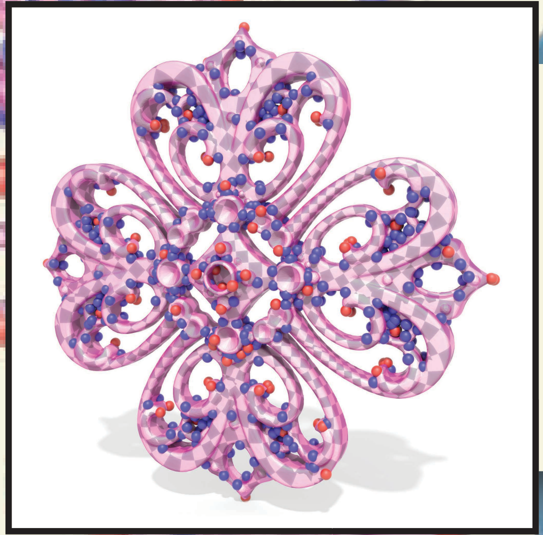
Validation

98% success



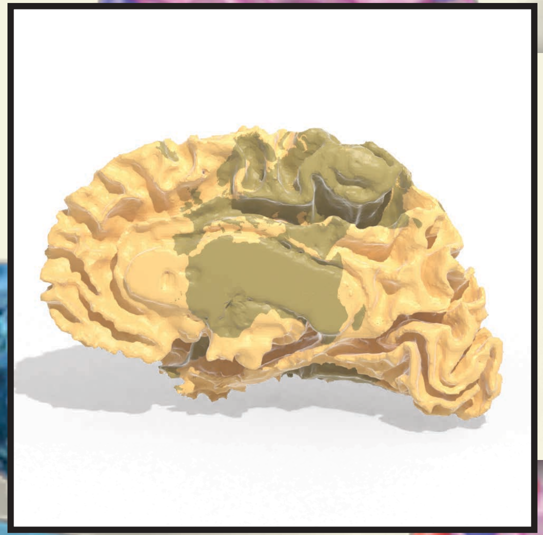
32,744 low-quality meshes [Zhou+ 2016]

100% success



114 difficult cone configurations [Myles+ 2014]

100% success



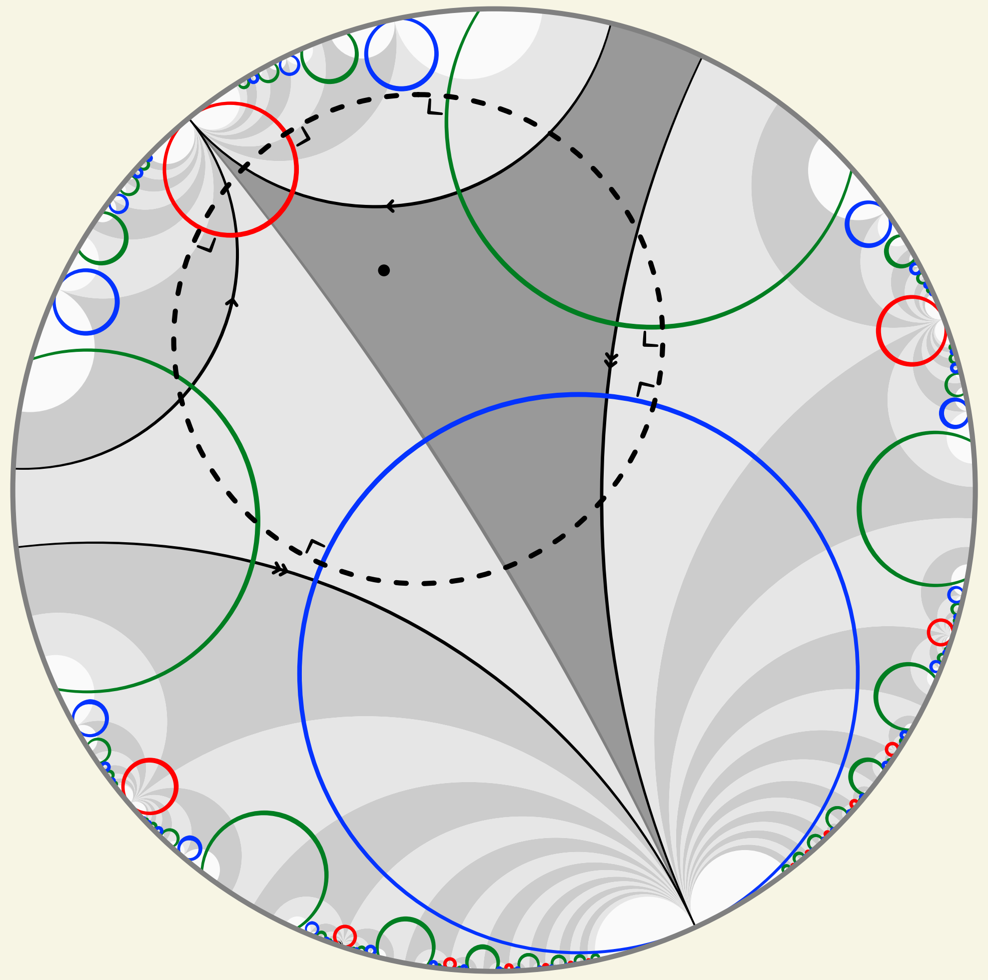
265 spherical parameterization problems [Yeo+ 2009; Boyer+ 2011]

Uptake

useful to people in a variety of areas

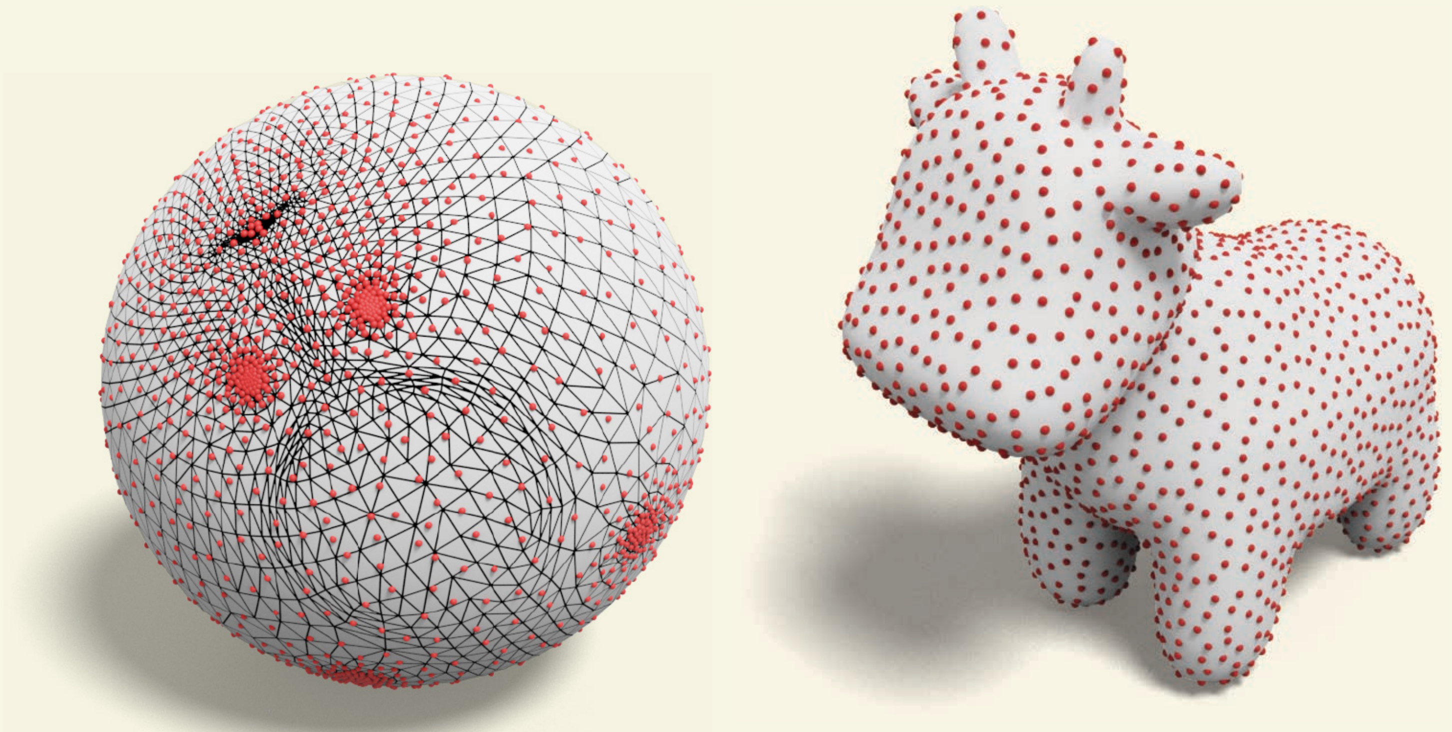
3D printing

[Lenihan *et al.* 2023]



geometric topology

[Bobenko & Lutz 2023]



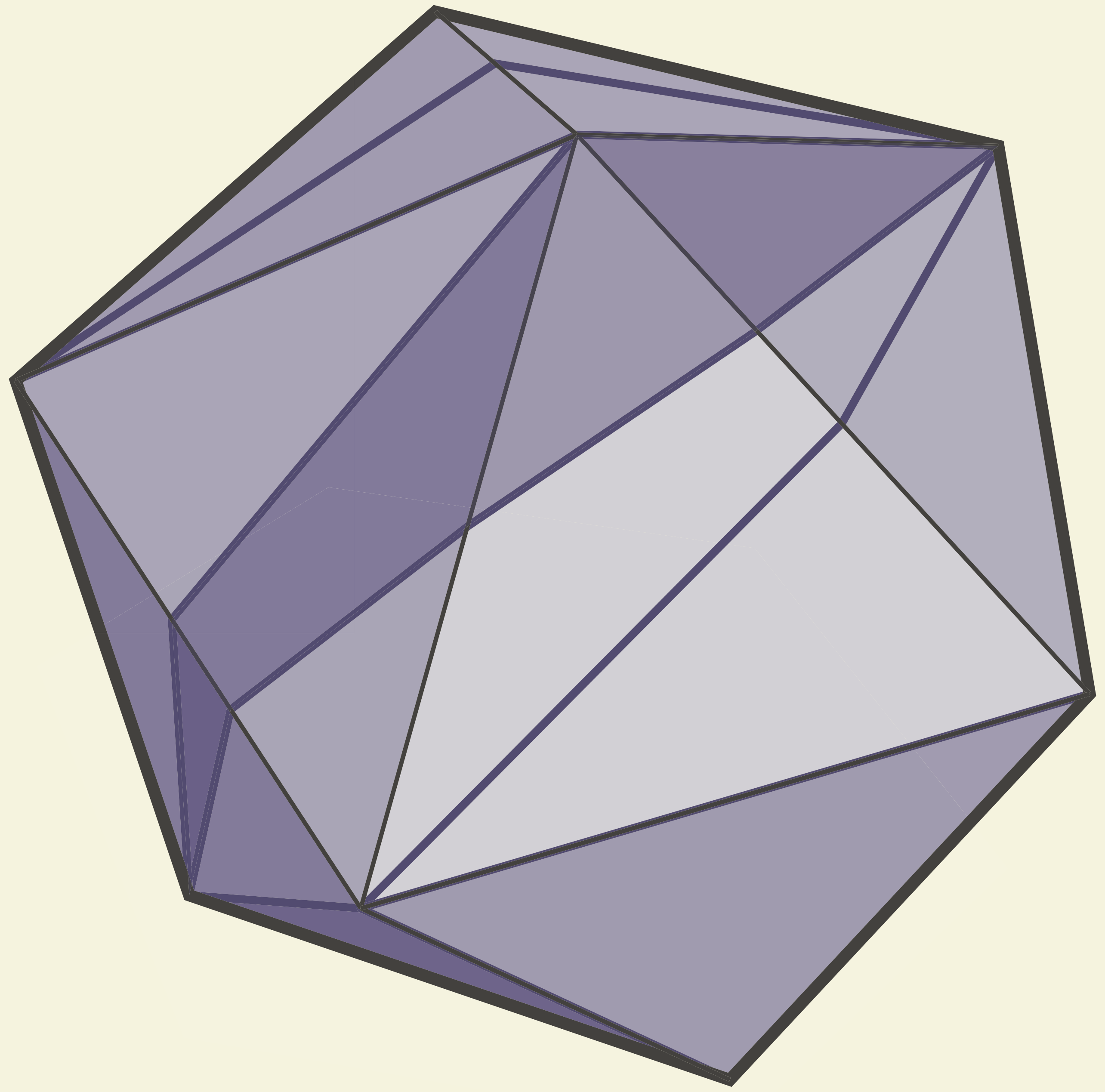
optimal transport

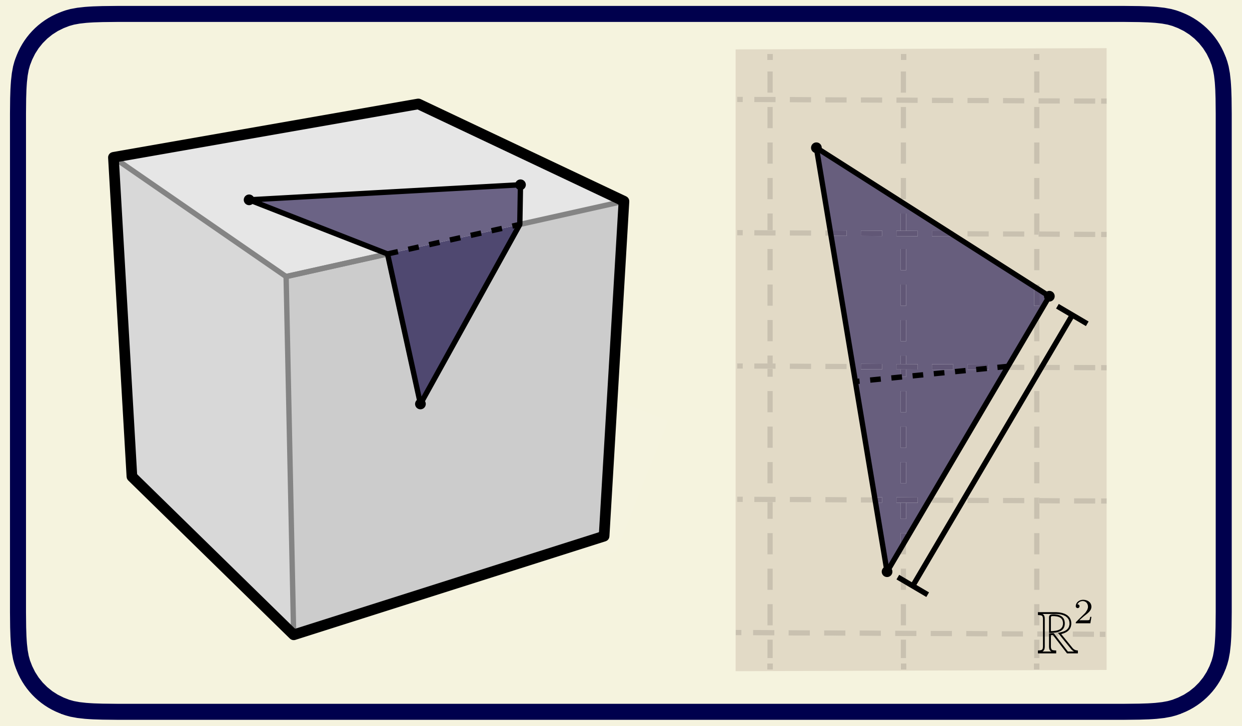
[Genest *et al.* 2024]

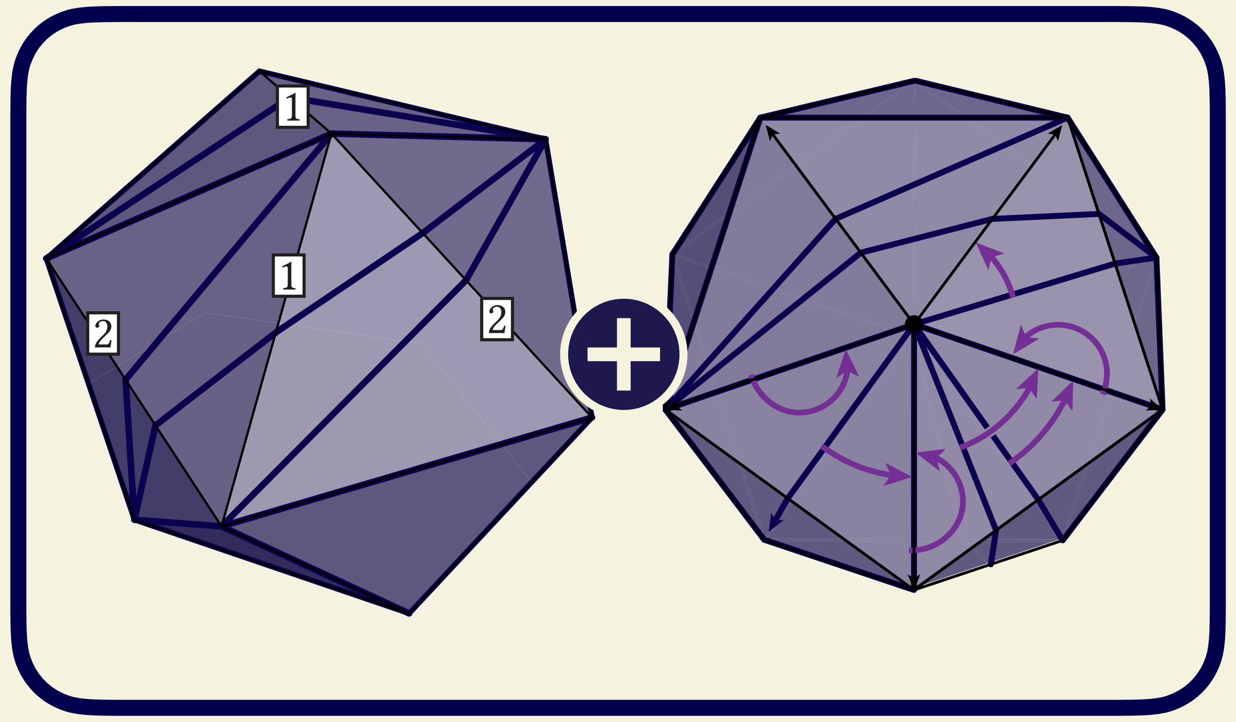
generative models

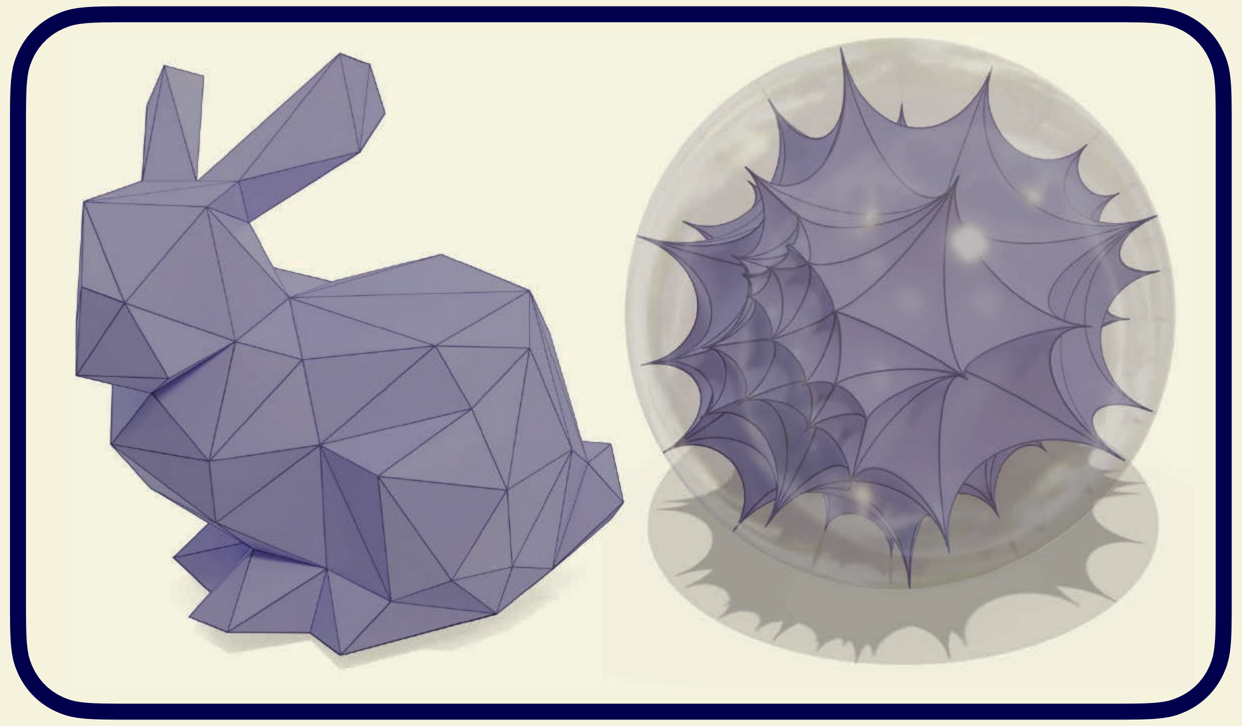
(ongoing)

Thanks for listening









Supplemental Slides

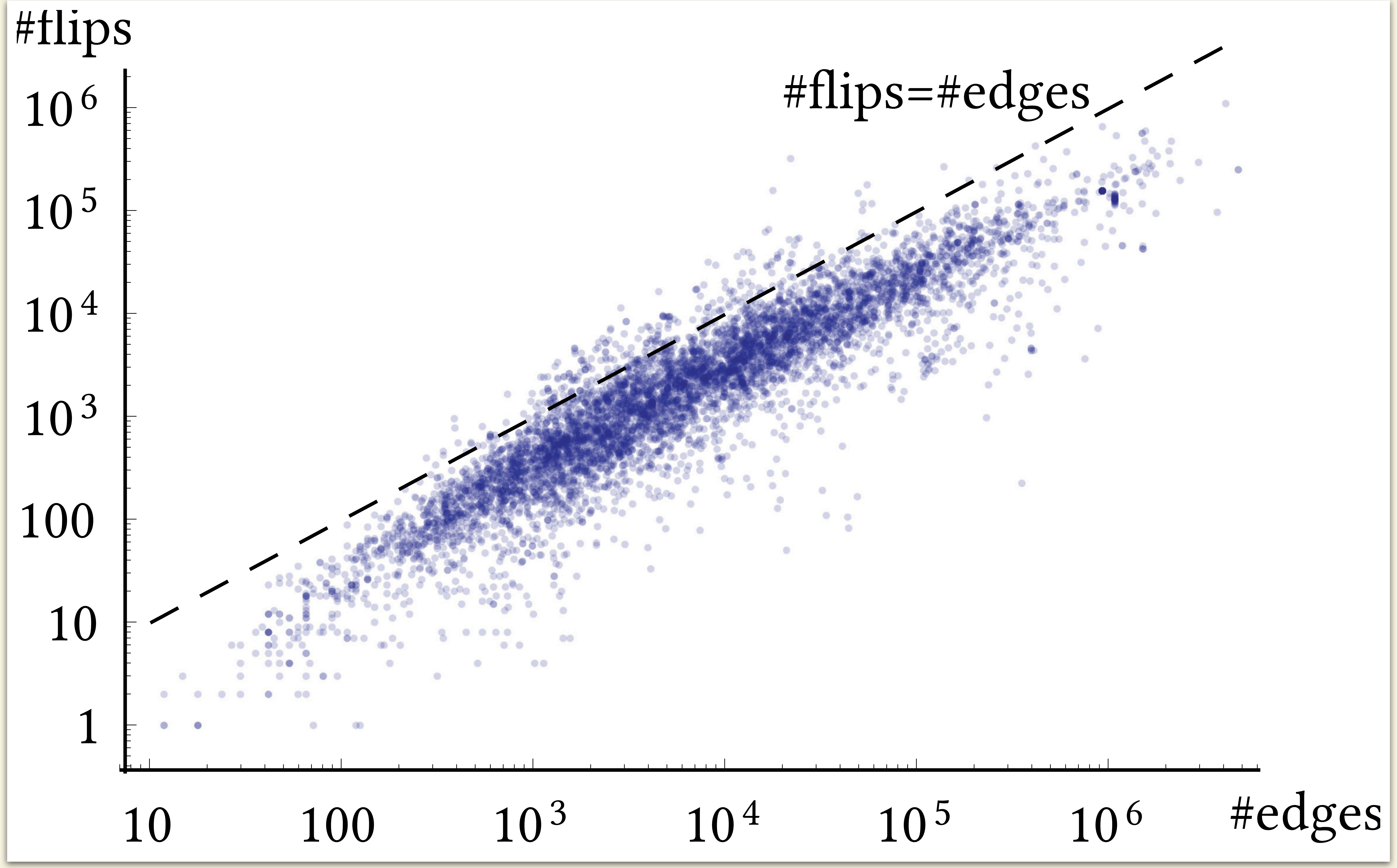
Bad basis functions

Input basis function

Intrinsic basis function

[Sharp, Soliman & Crane 2019]

Delaunay flip complexity



[Sharp, Soliman & Crane 2019]

Units for transport cost

integrated curvature is dimensionless (angle => units of radians)

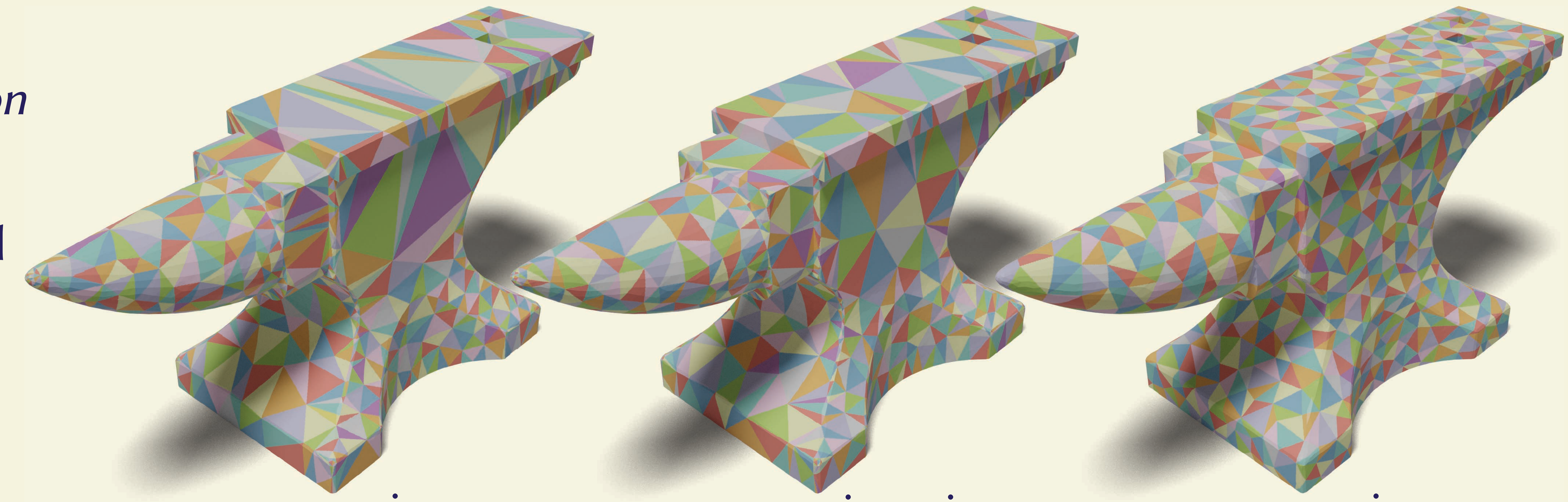
what units make sense here?

area has units of area

Easy resolution: measure *mass fraction* rather than mass

e.g. fraction of total area or curvature present at a vertex

then everything is unitless



coarsening via curvature transport cost

coarsening via blended cost

coarsening via area transport cost

Exact isometric embeddings

(intrinsically) convex polyhedra

unique convex embedding into $\mathbb{R}^3$
[Alexandrov 1942]

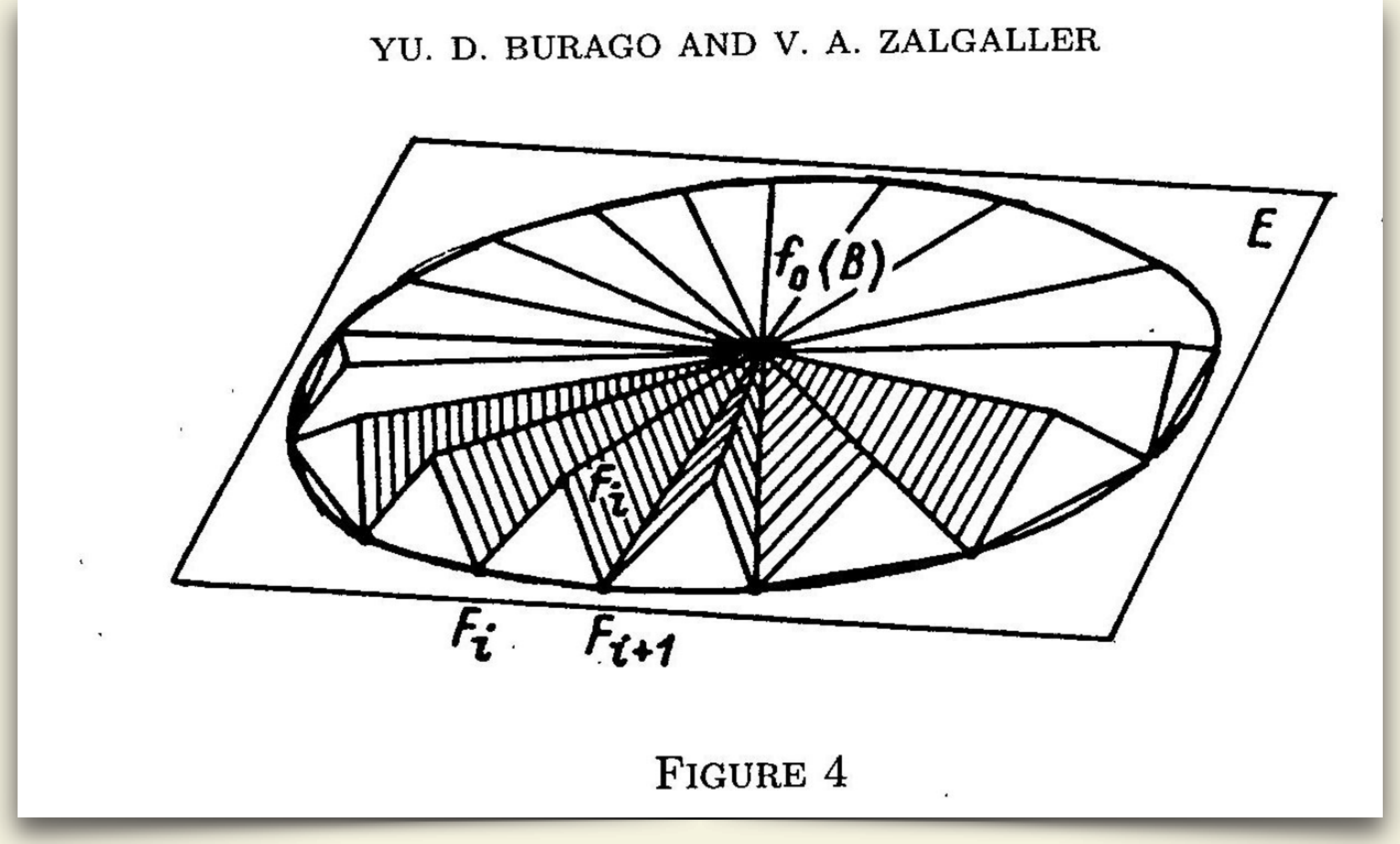
(may need to flip edges)

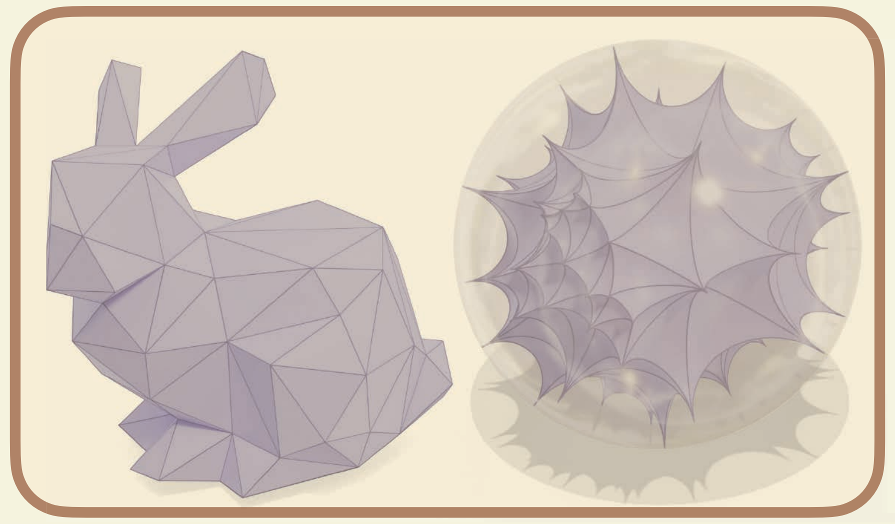
constructive proof/algorithm
[Bobenko & Izmestiev 2008]

general polyhedra

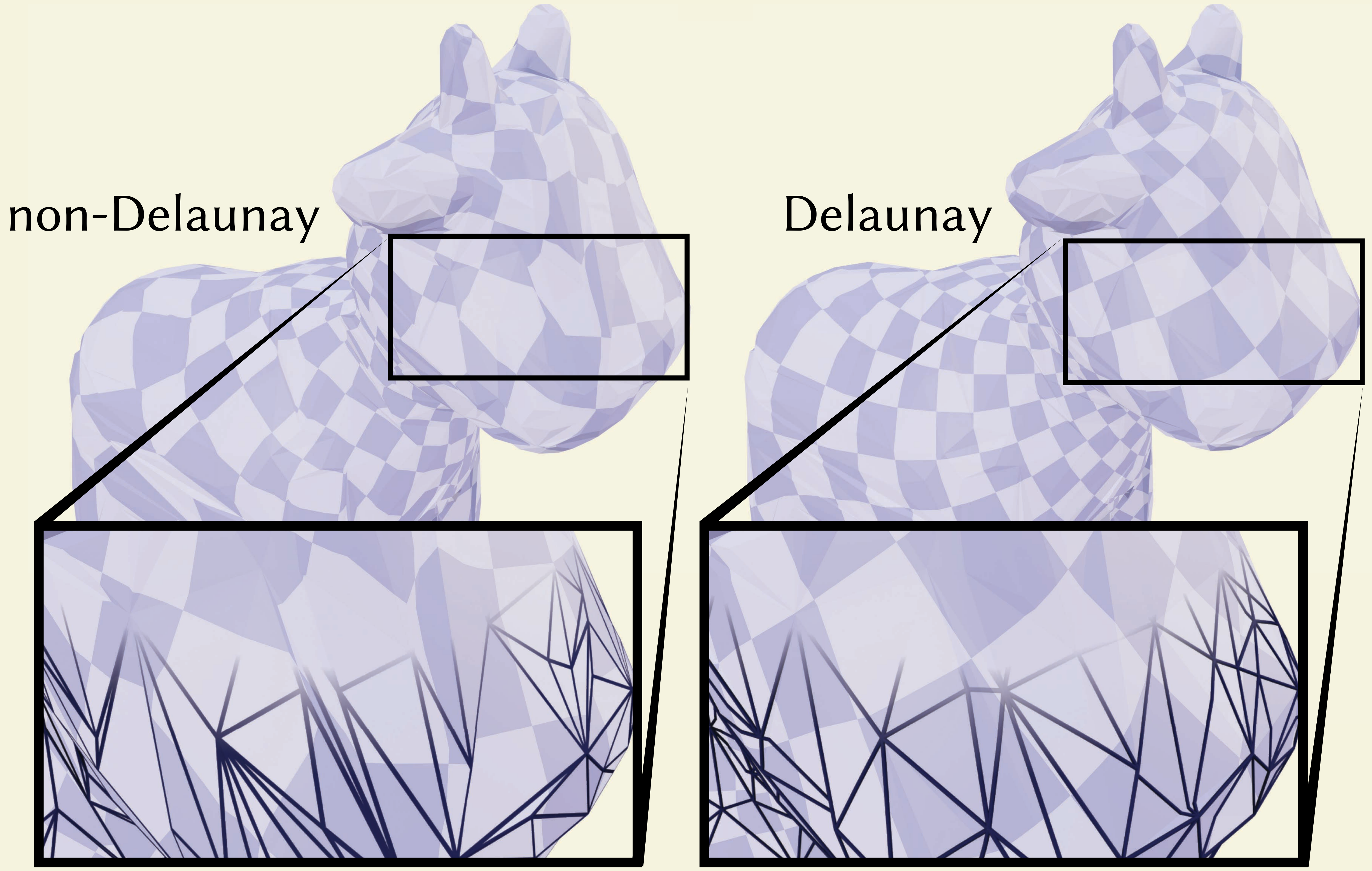
many embeddings into $\mathbb{R}^3$
[Burago & Zalgaller 1960, 1995]

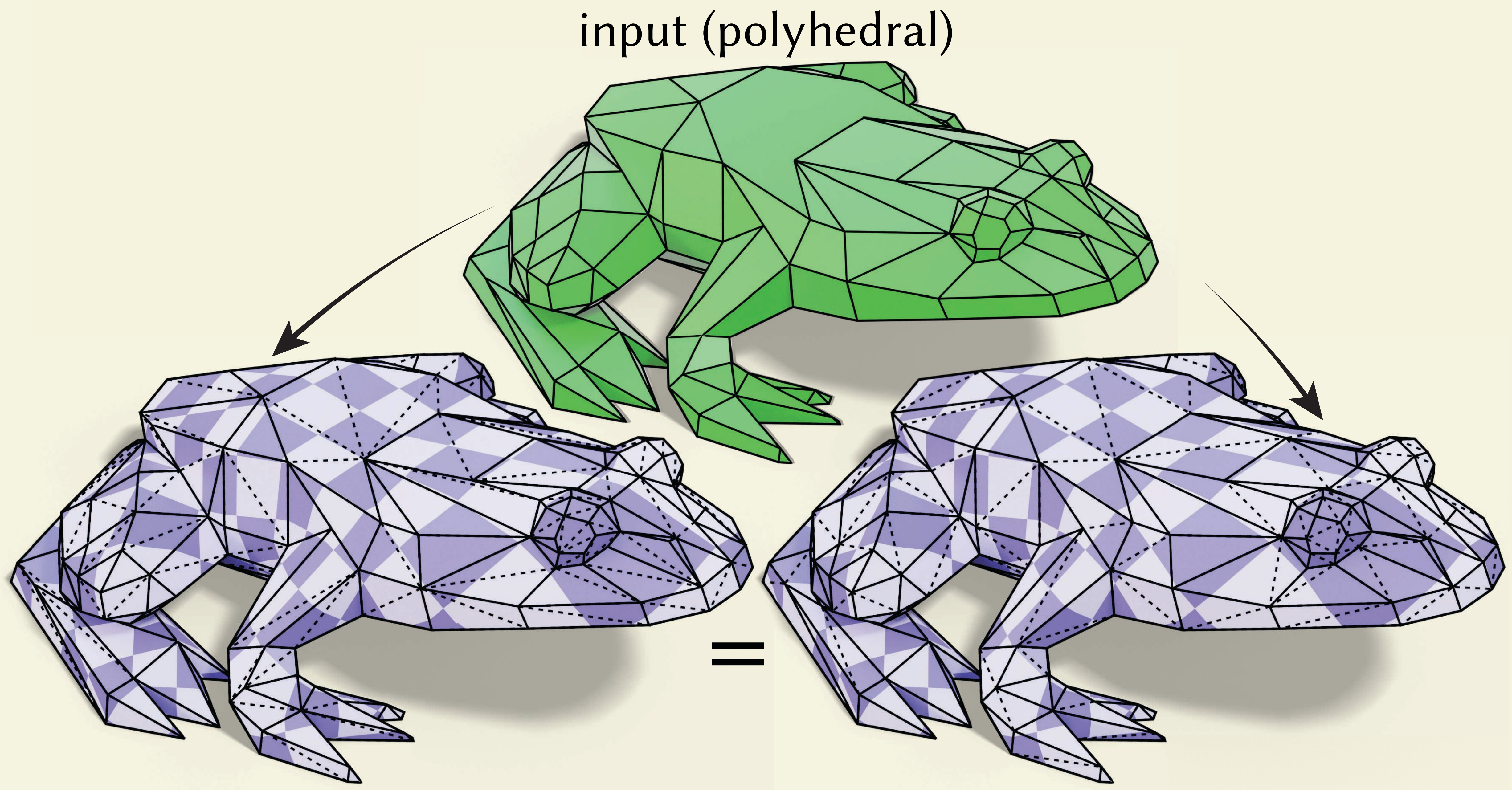
(may need to subdivide mesh many times)



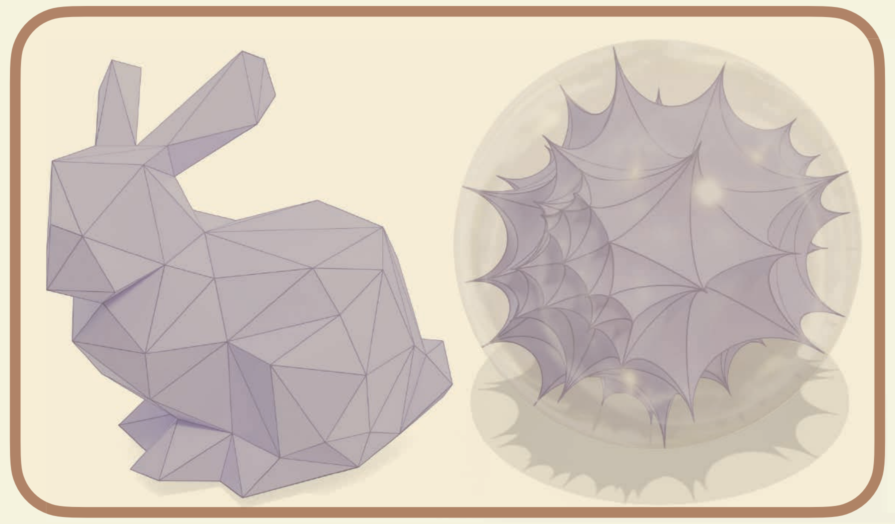


Starting from Delaunay





Final algorithm


Parameterization

flip to (Euclidean) Delaunay

solve for discrete conformal map

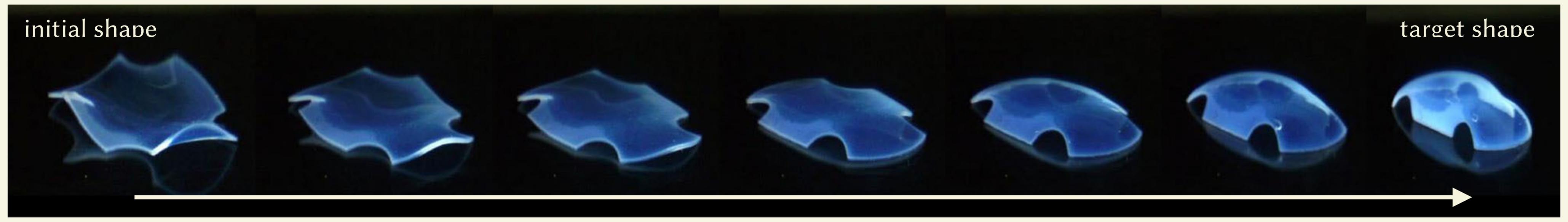
lay out in plane

extract correspondence

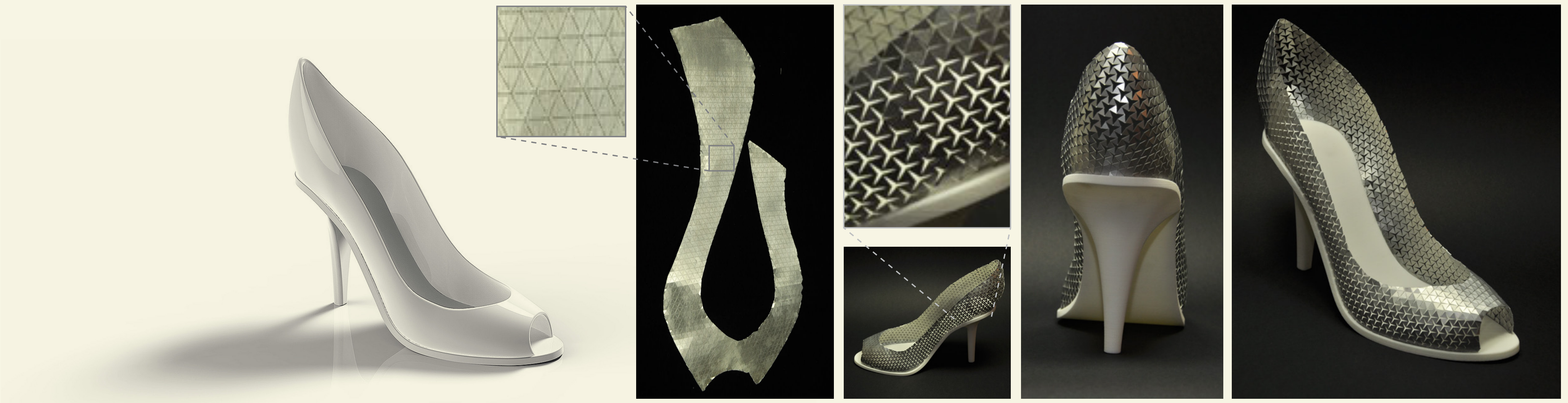
interpolate via hyperboloid

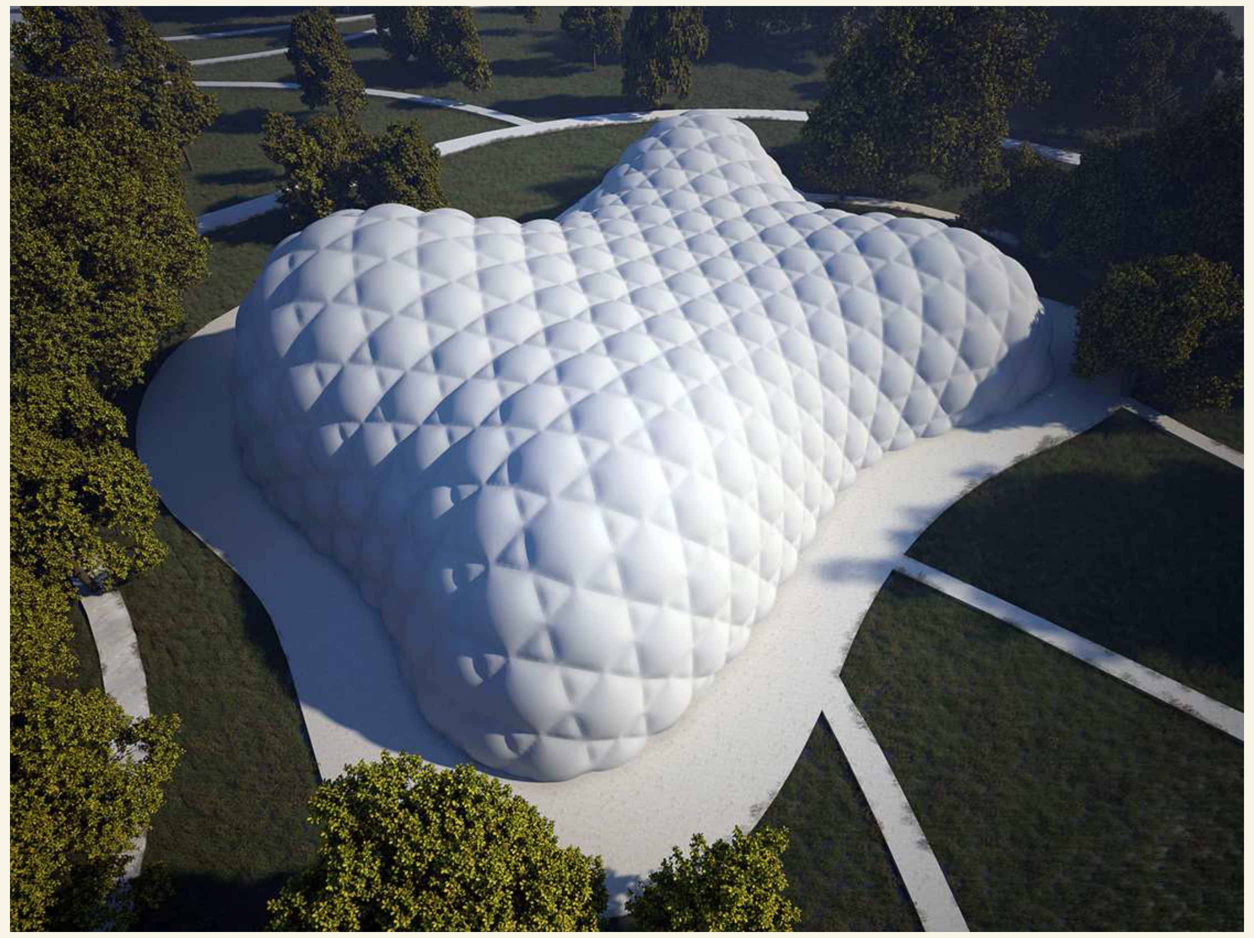
Applications of parameterization

Fabrication

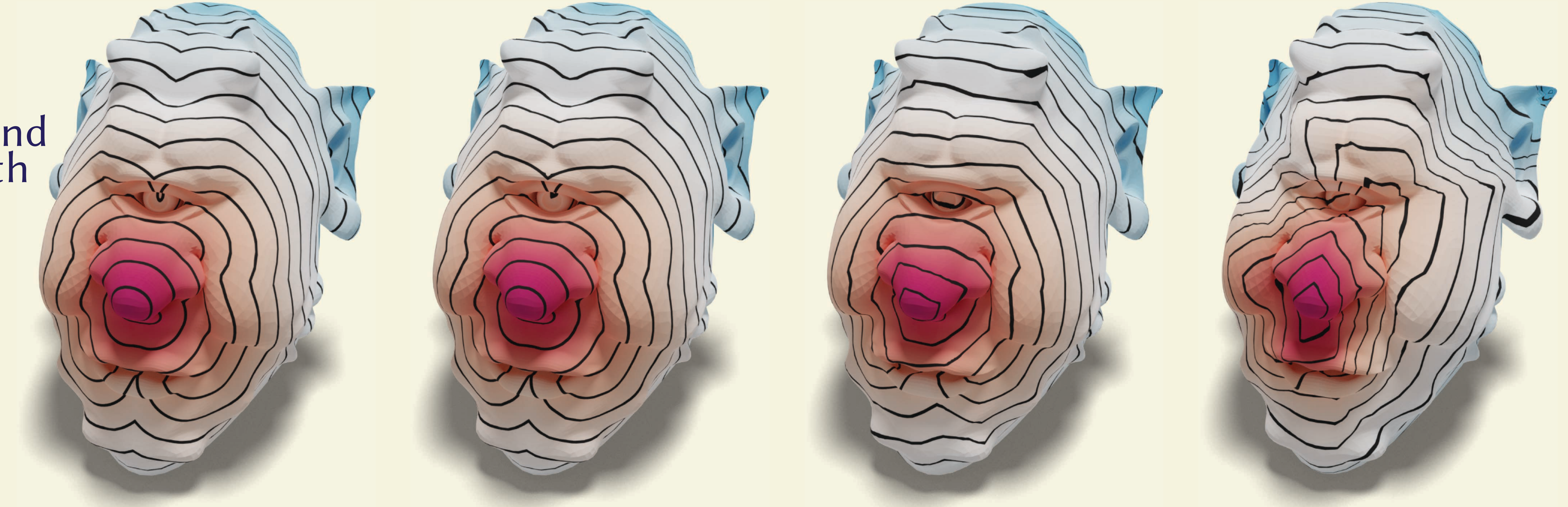


[Nojoomi *et al.* 2021]

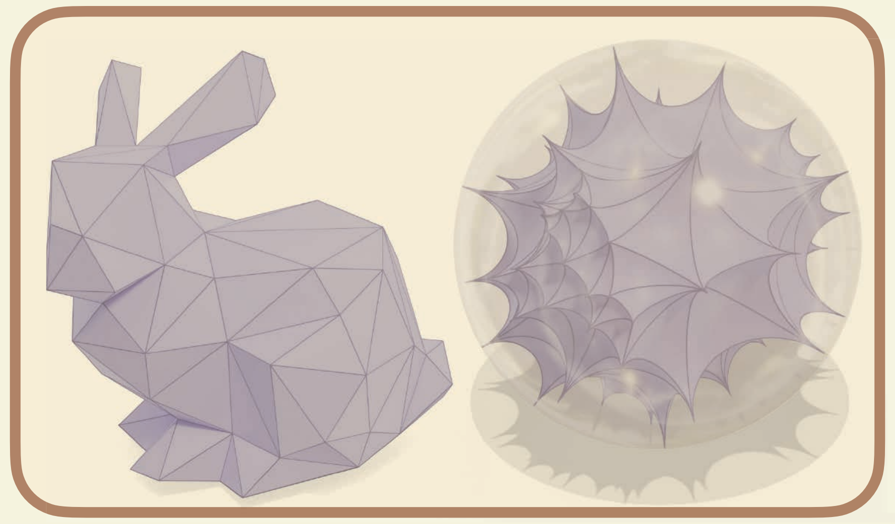

[Konaković *et al.* 2016]

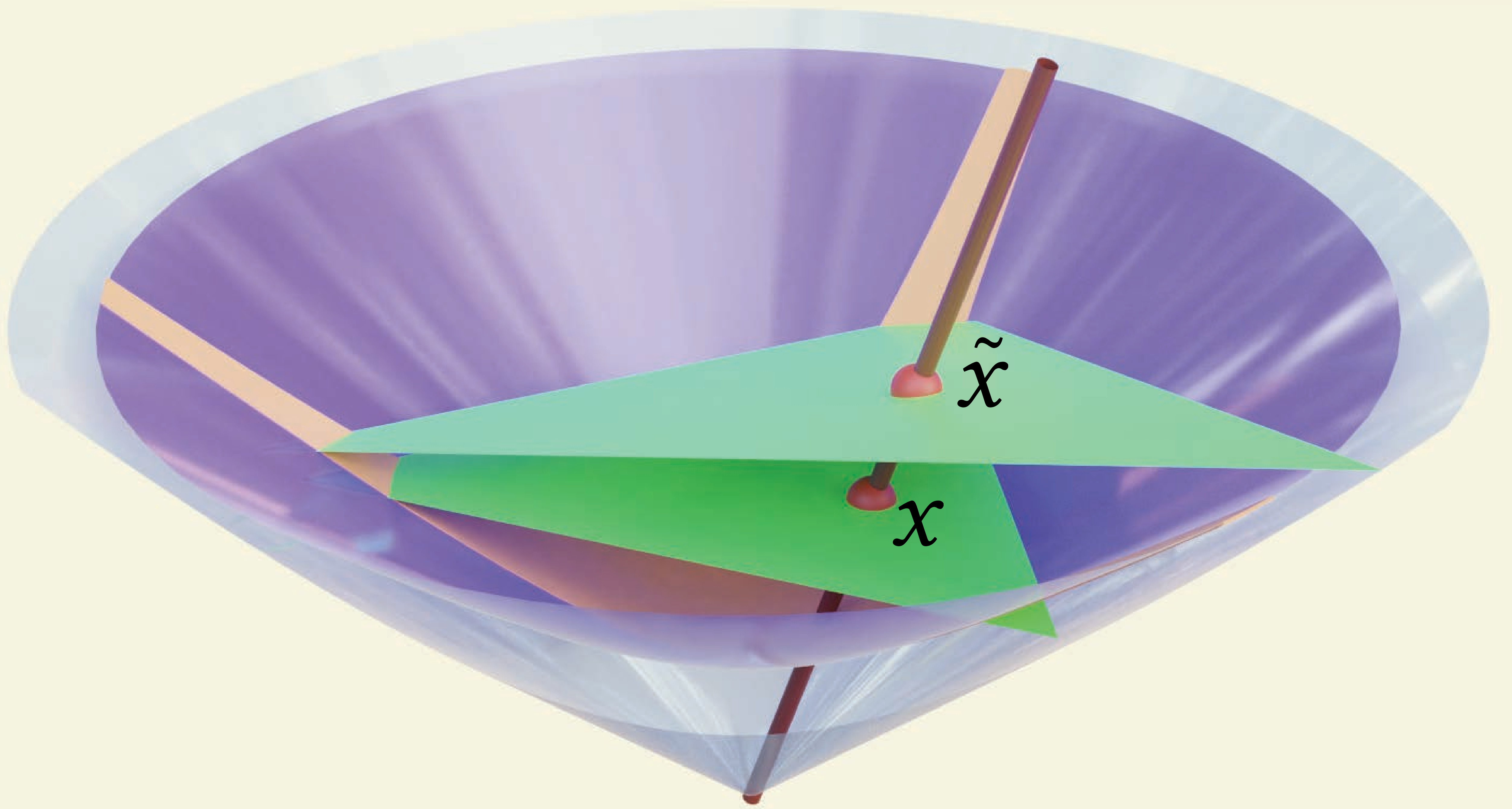

[Konaković-Luković *et al.* 2018]

Speedup vs error in geodesic distance



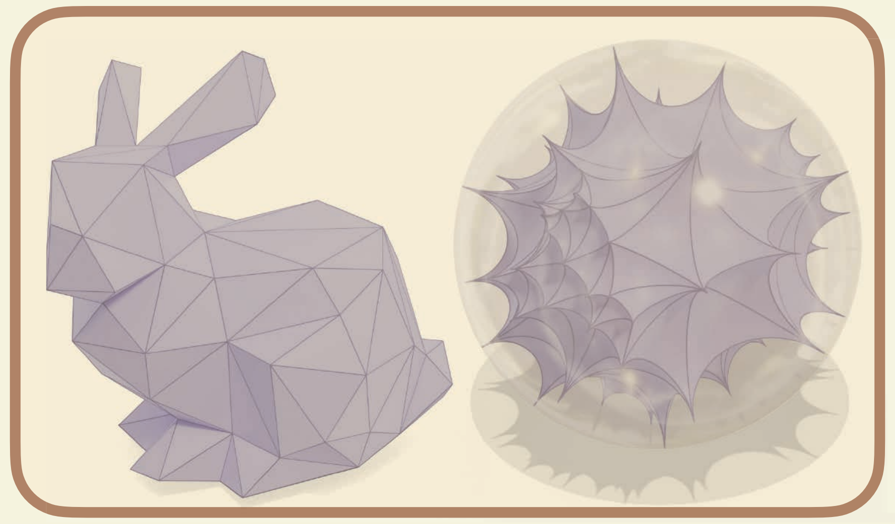
Interpolation in the hyperboloid model

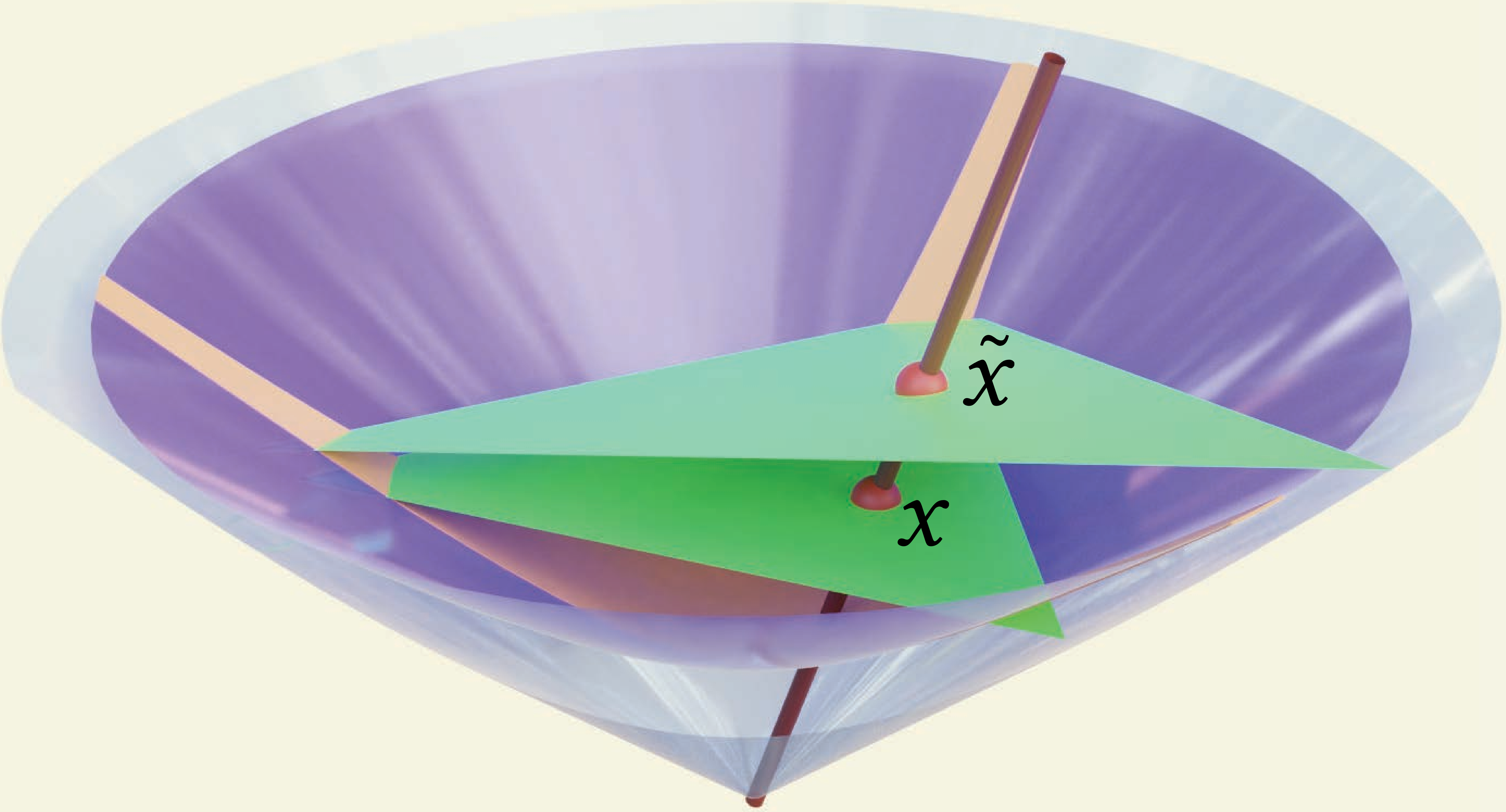




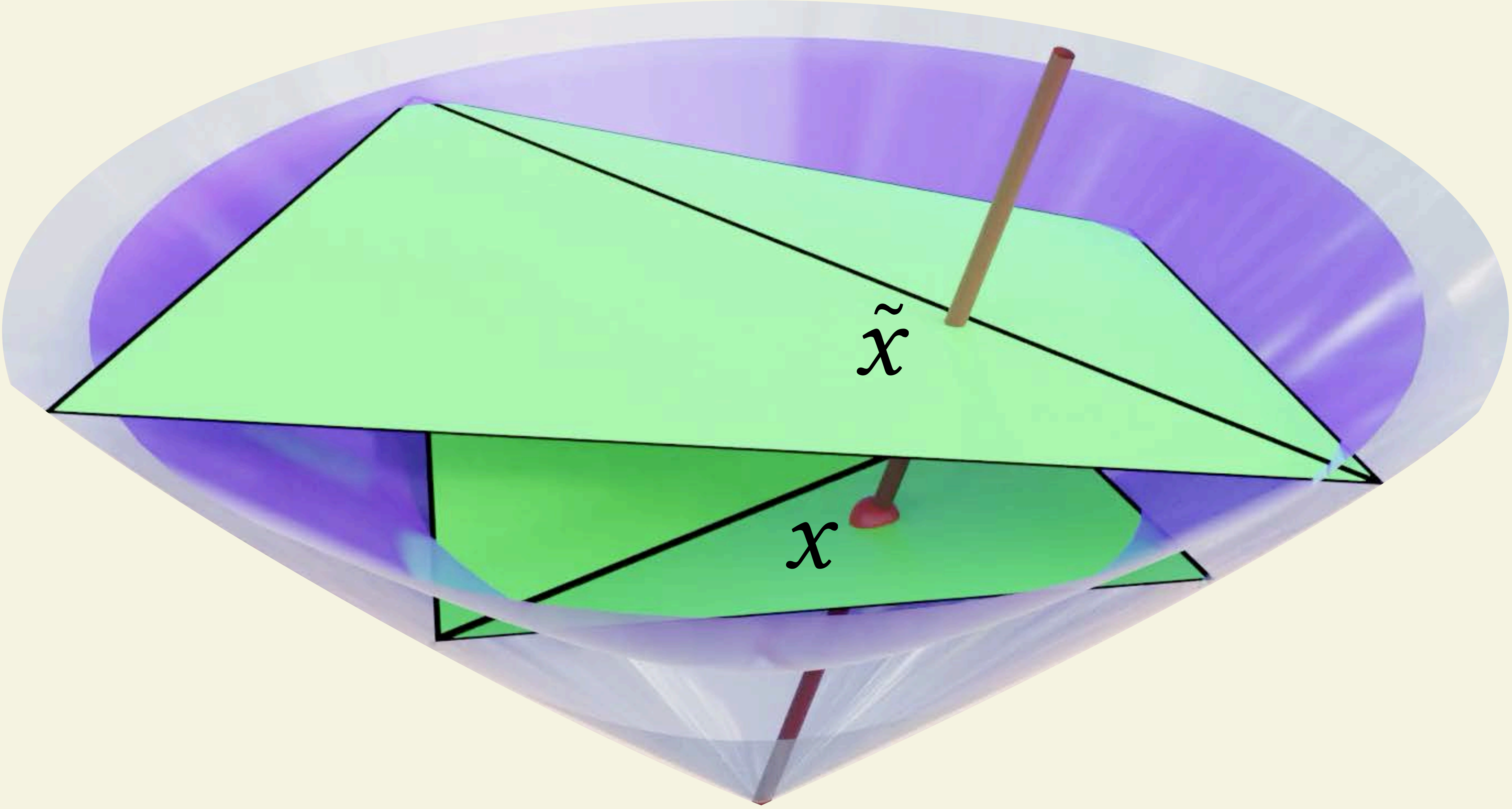
fixed triangulation

Interpolation in the hyperboloid model





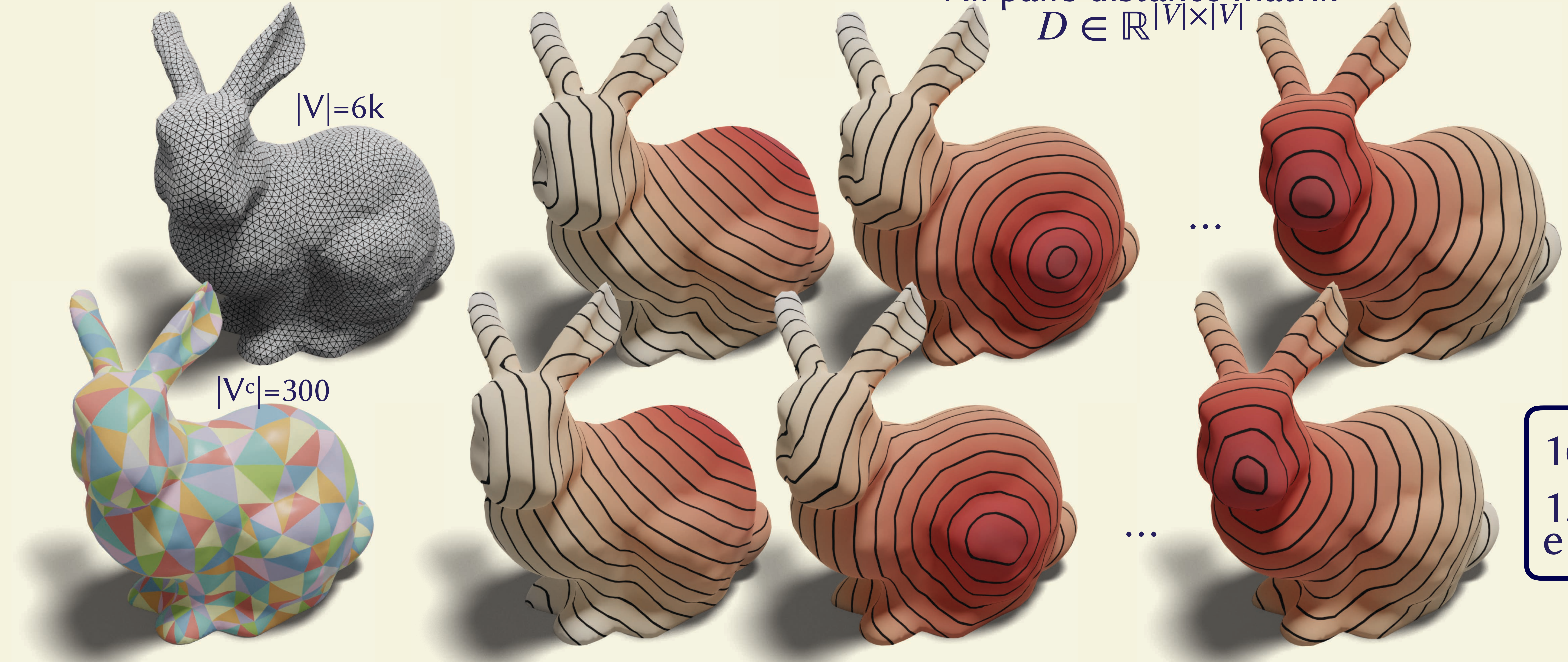
fixed triangulation



variable triangulation

Low rank all-pairs distance matrix approximation

All pairs distance matrix
$D \in \mathbb{R}^{|V| \times |V|}$

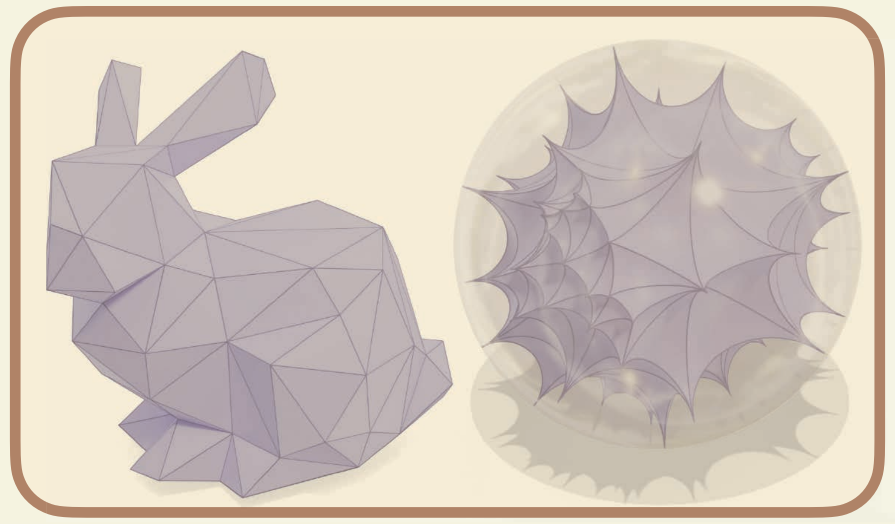


1650x faster

1.4% relative error

Distance matrix of simplified mesh ➕ Prolongation operator $P : \mathbb{R}^{|V^c|} \rightarrow \mathbb{R}^{|V|}$: Approximate distance matrix $\hat{D} = P\tilde{D}P^\top$

Application: Shape Correspondence



- Uniformization can been used to find correspondences between shapes

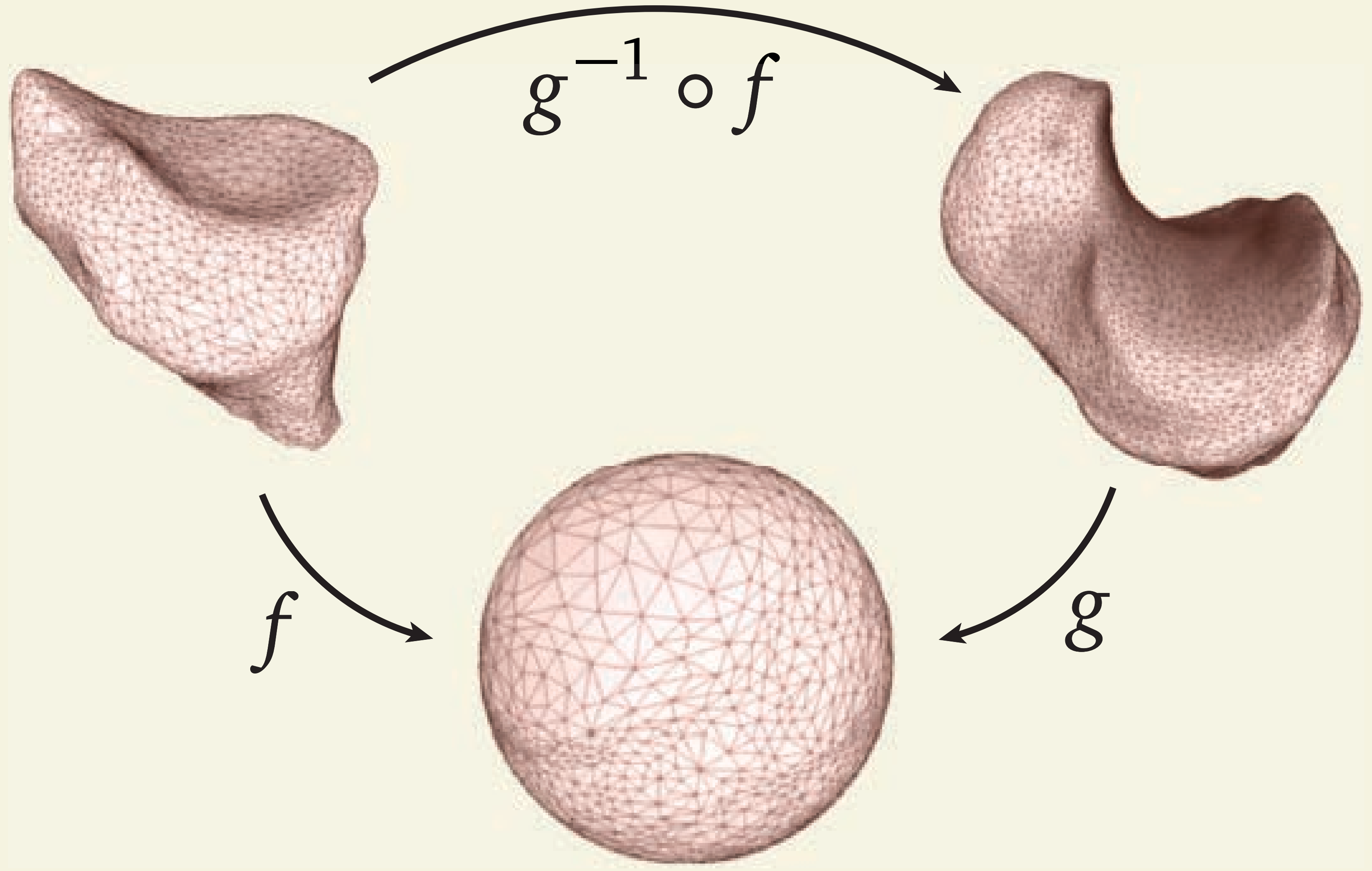
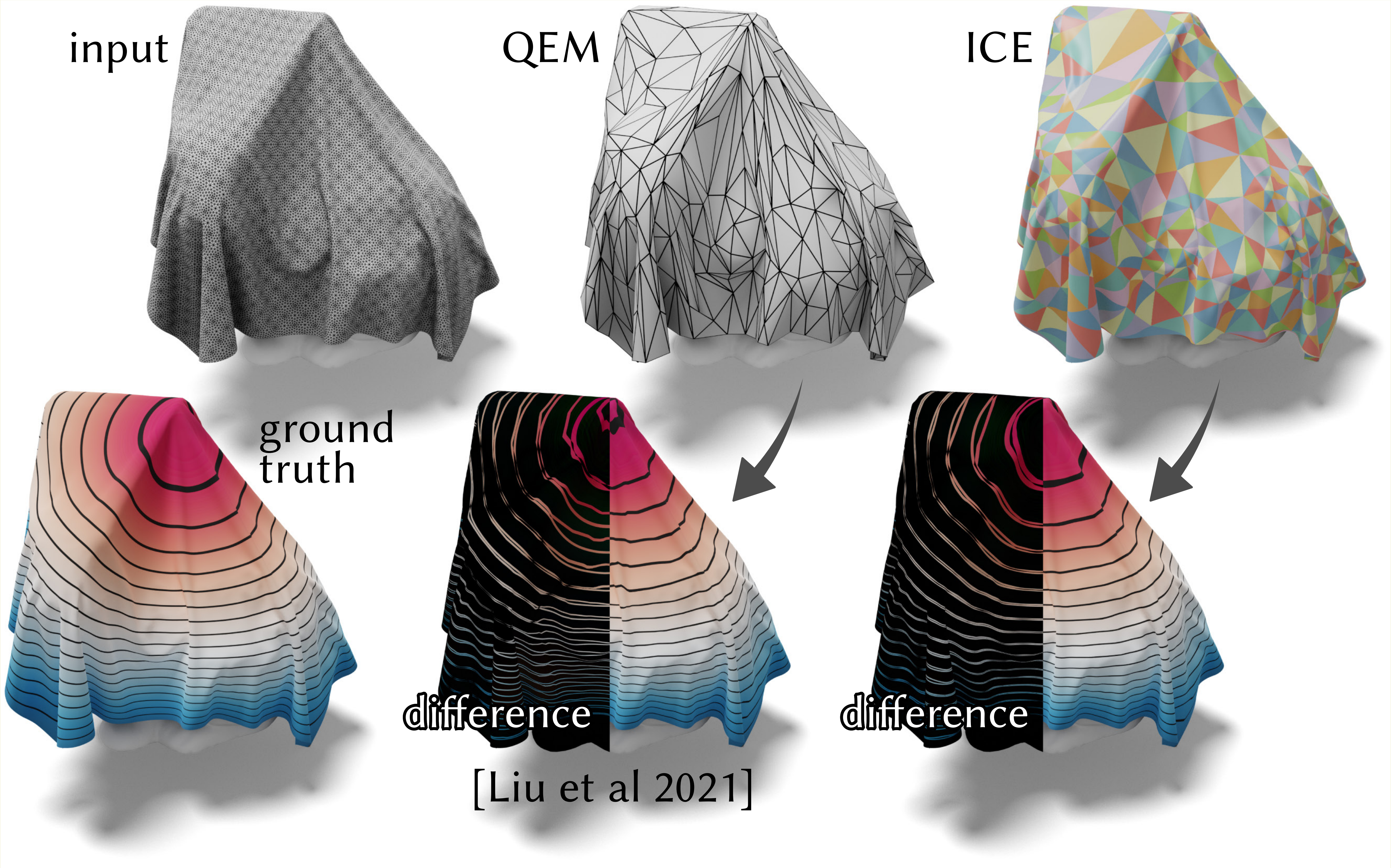


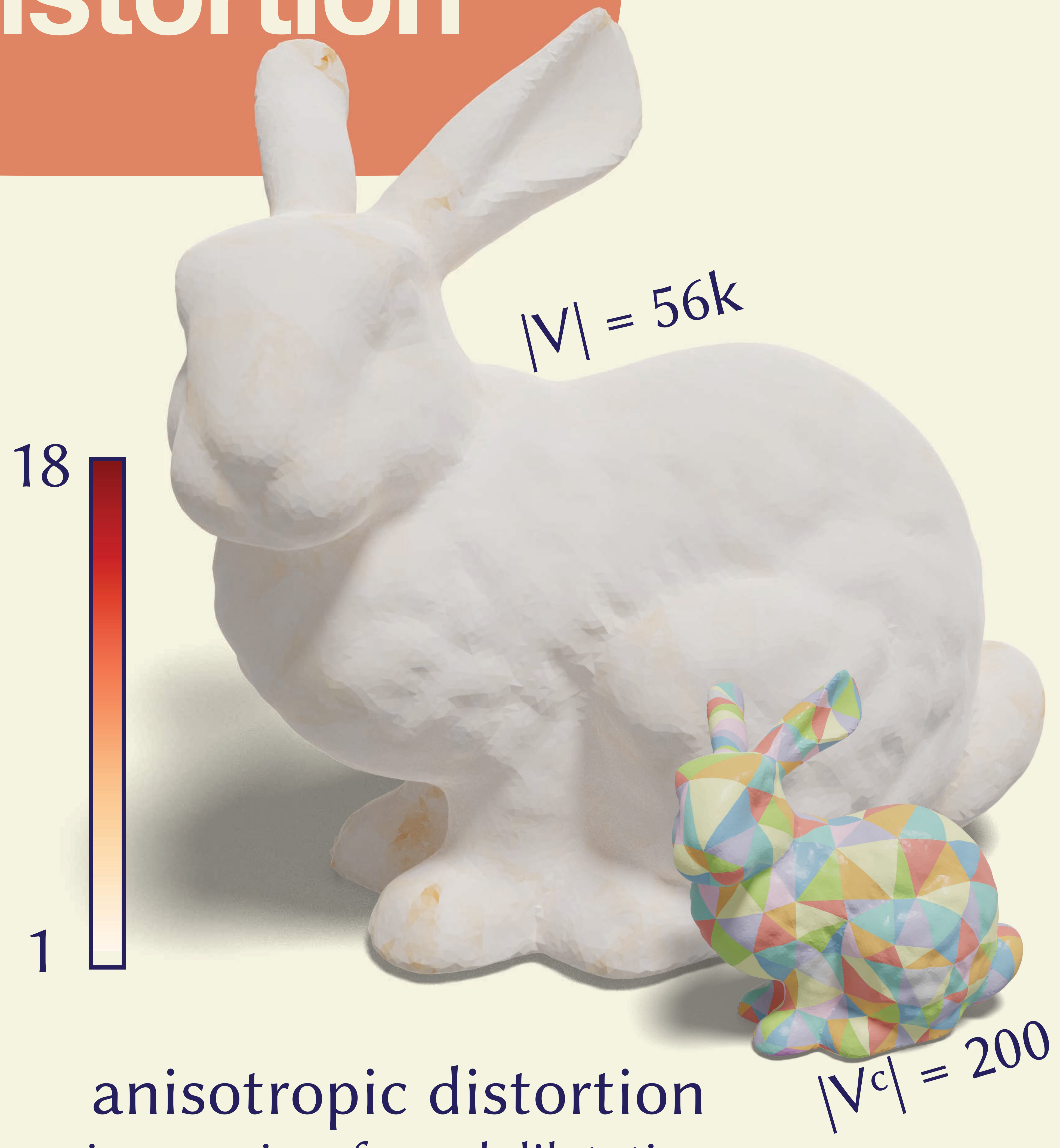
Image: [Koehl & Hasse 2015]

Image: [Schmidt, Campen, Born & Kobbelt 2020]

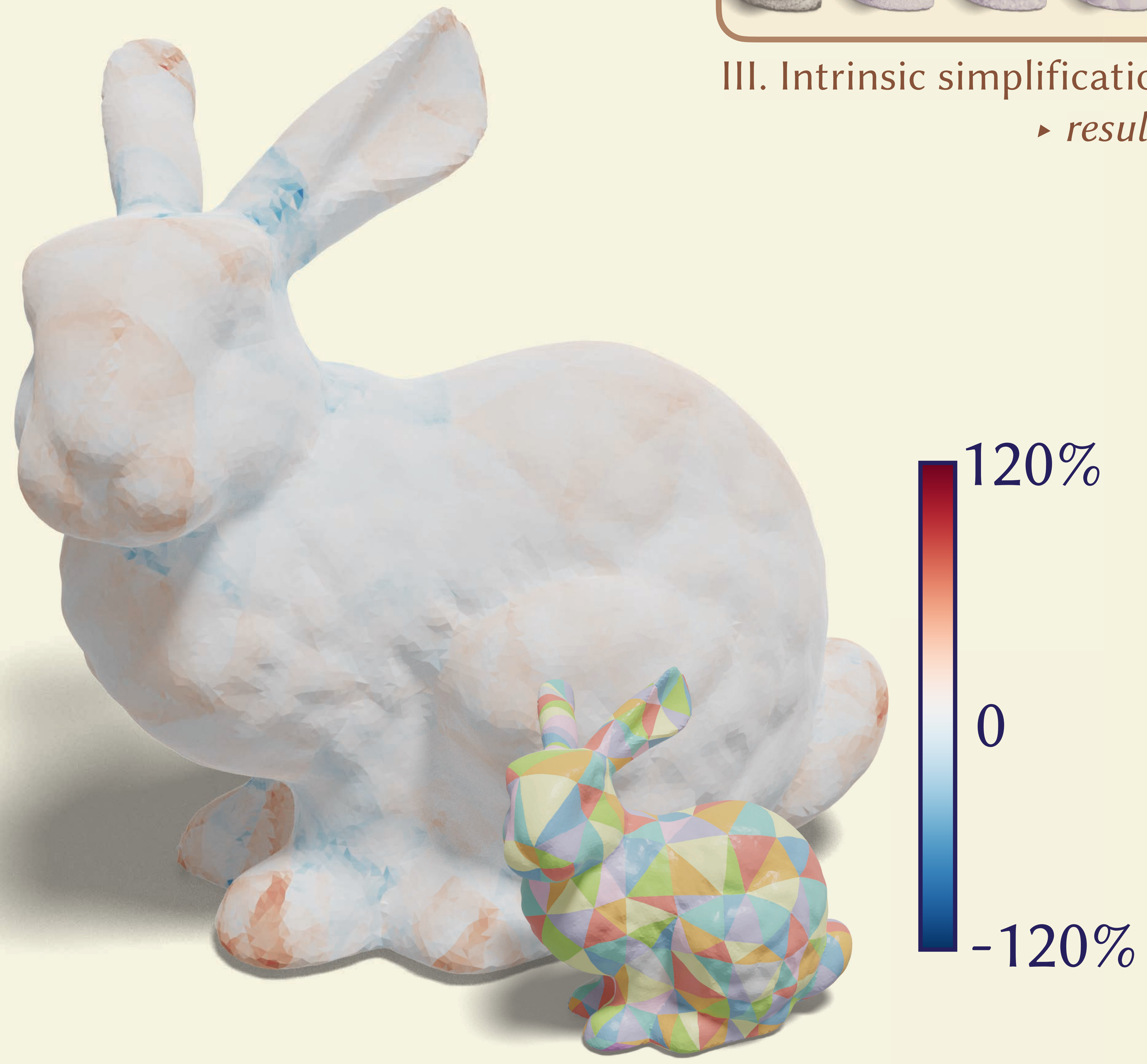
Solution accuracy



Distortion

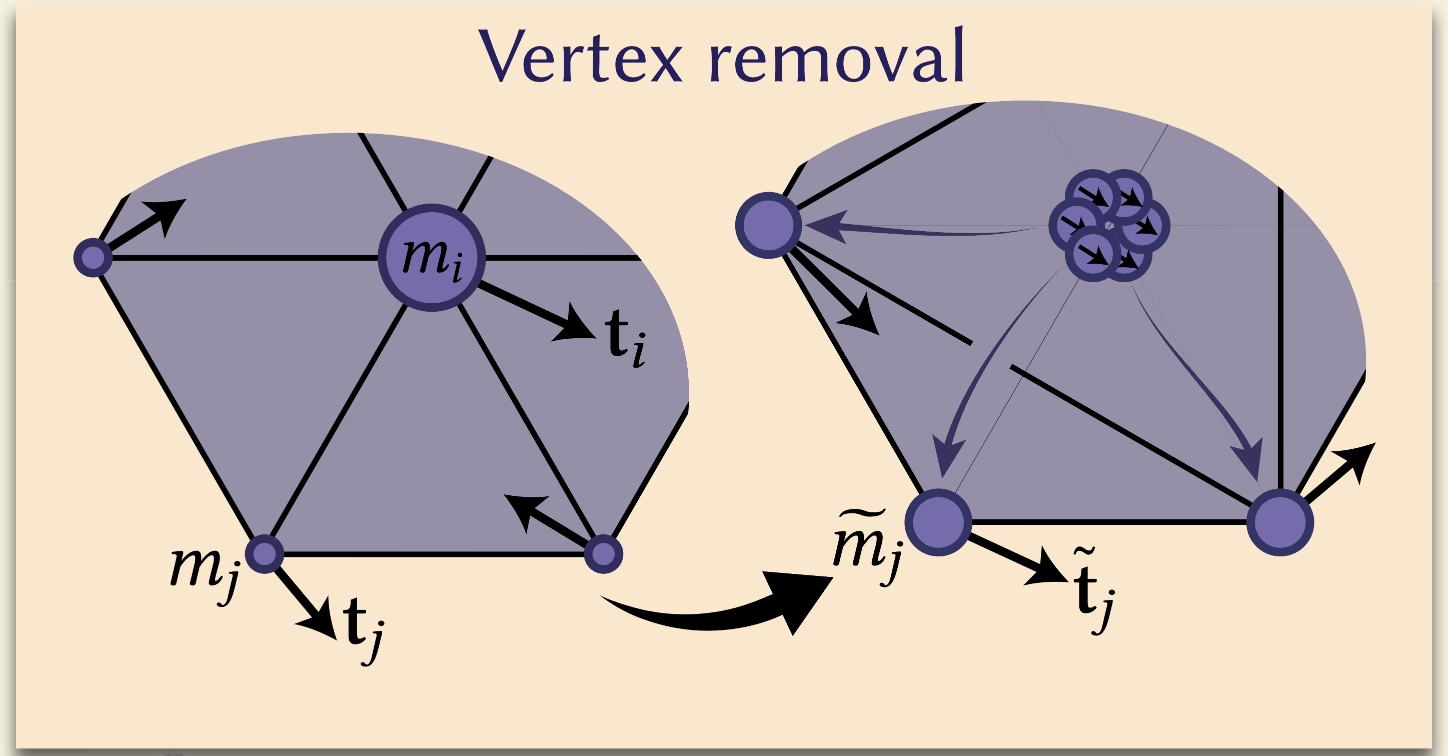


anisotropic distortion
i.e. quasiconformal dilatation
(mean 1.115)

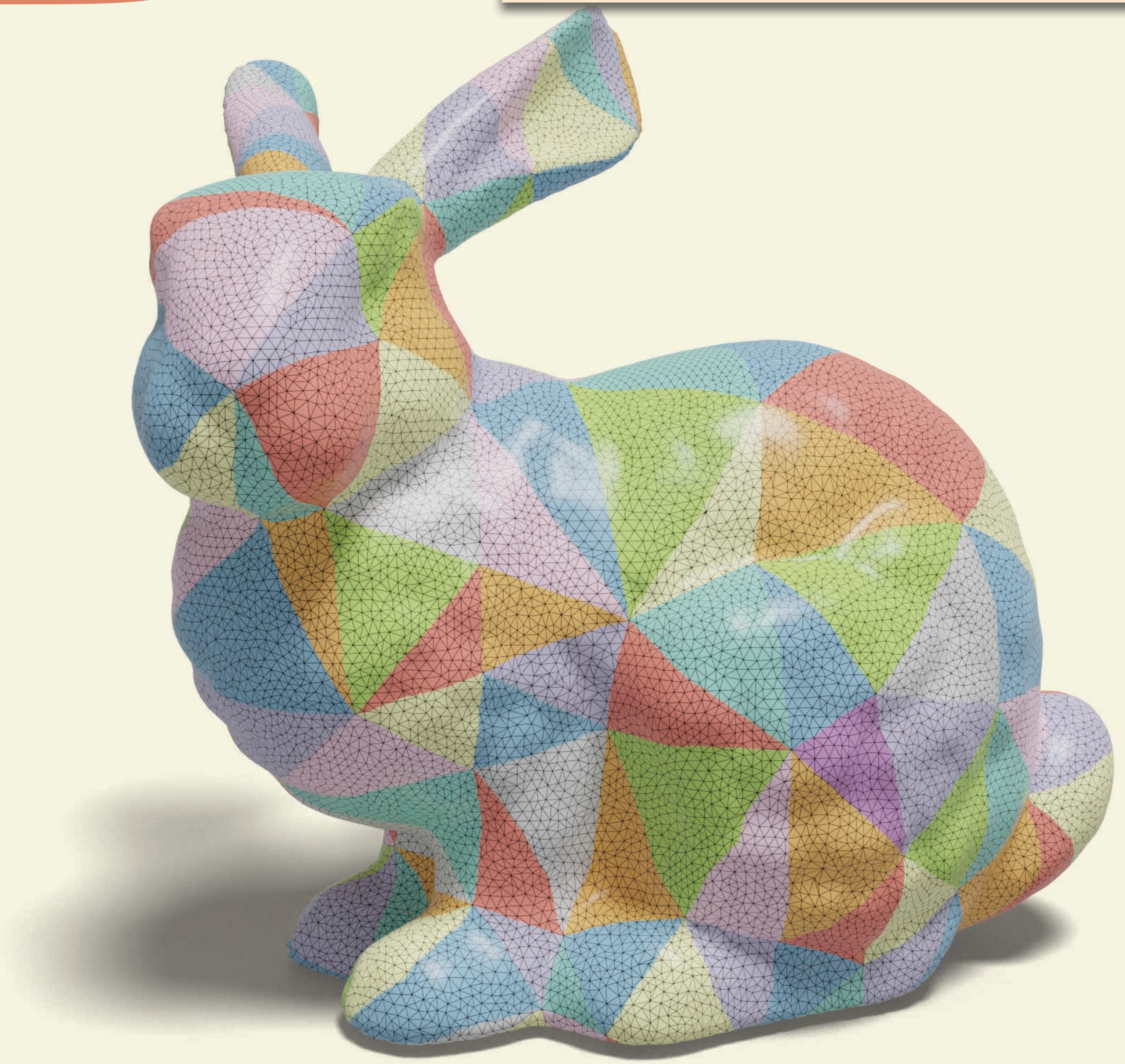


area distortion
(mean 8.1%)

The importance of memory

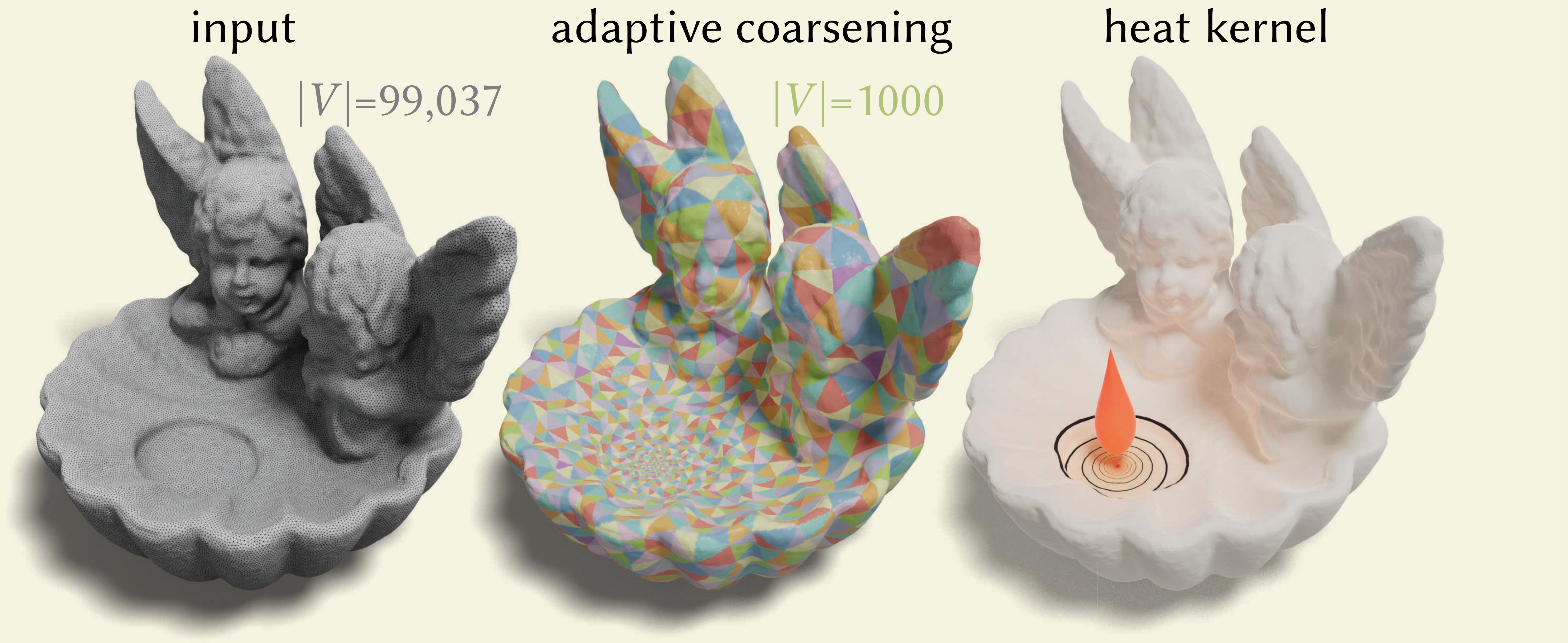


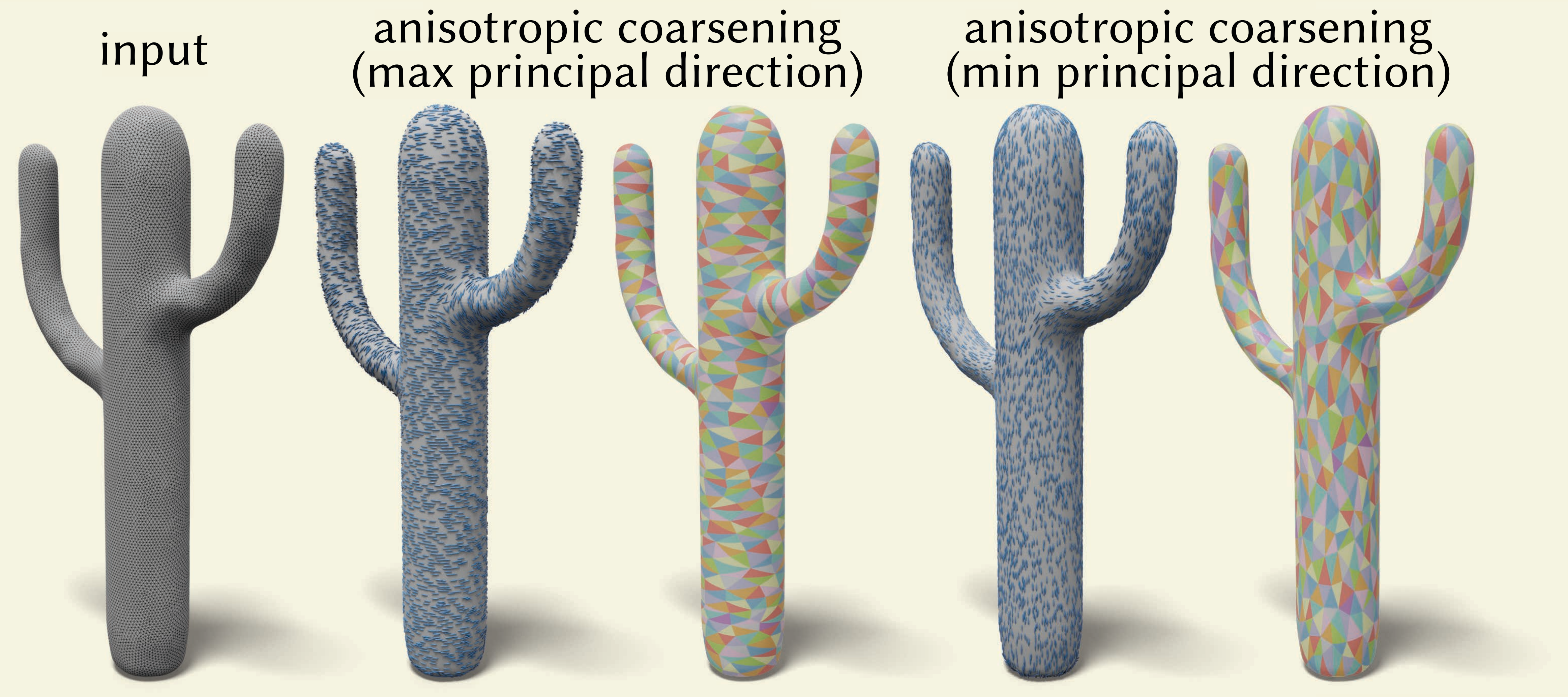
Memoryless transport cost

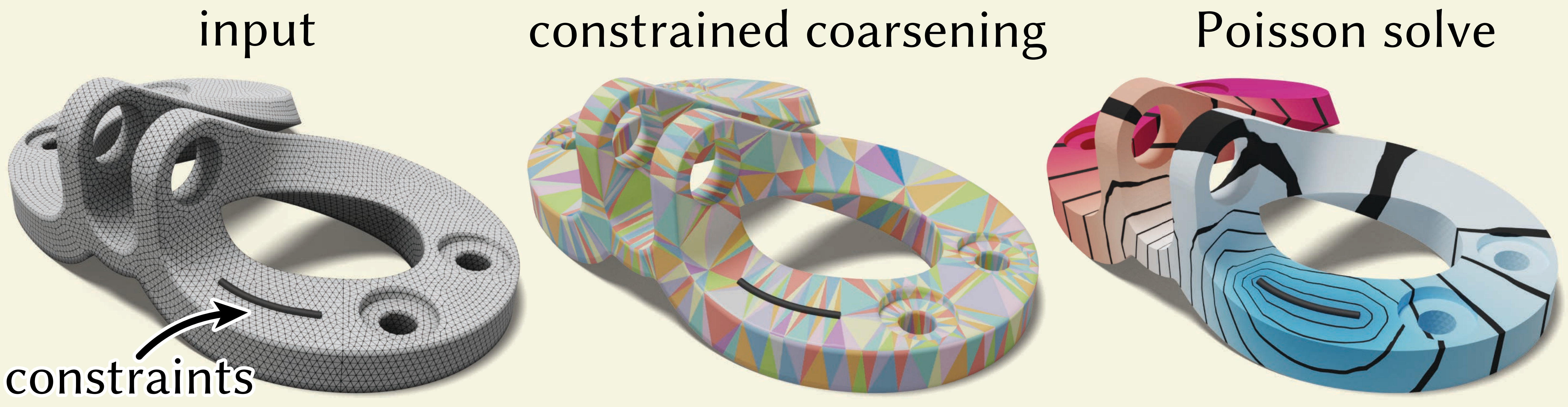


Full transport cost

Adaptive simplification



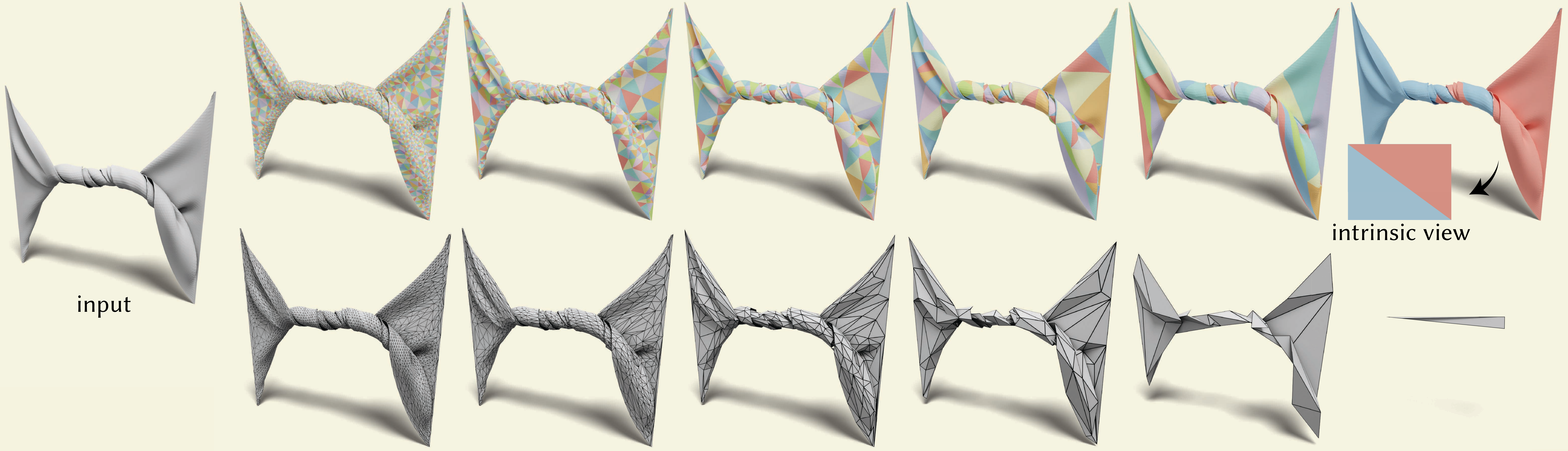




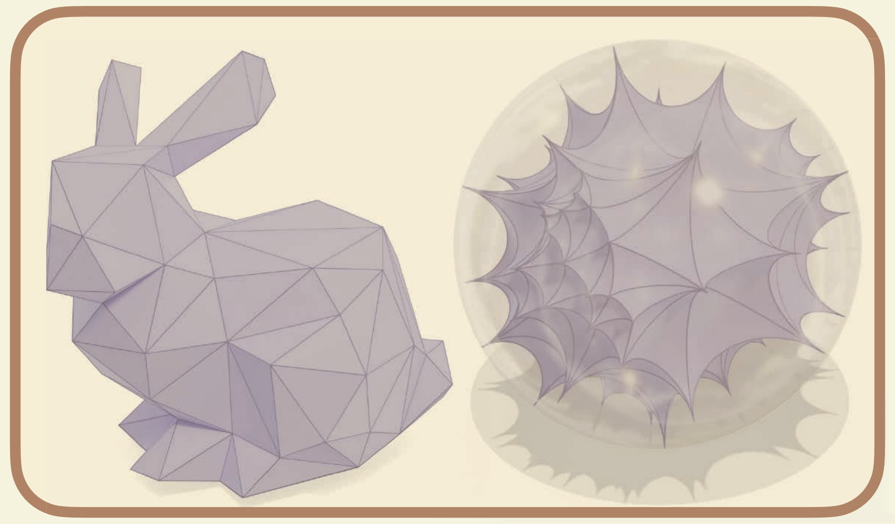
Geodesic Voronoi diagrams

Intrinsic Delaunay refinement

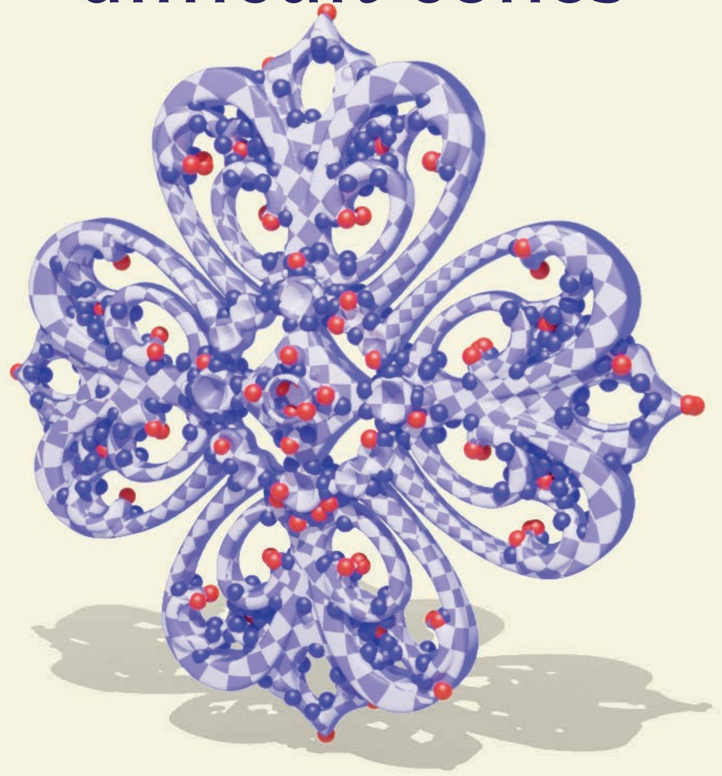
Near-developable surfaces



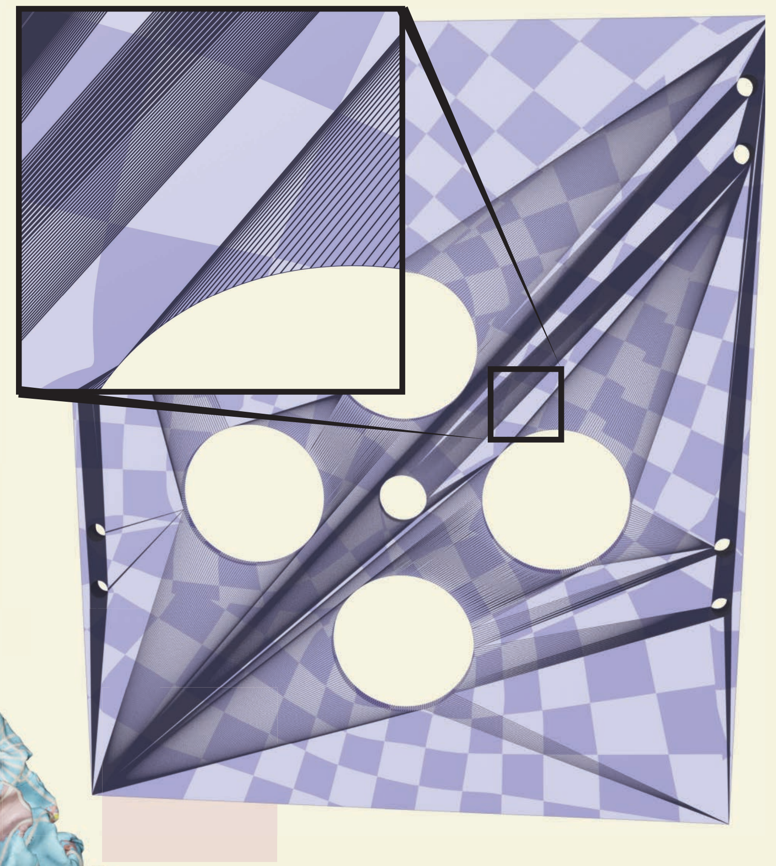
Challenging datasets

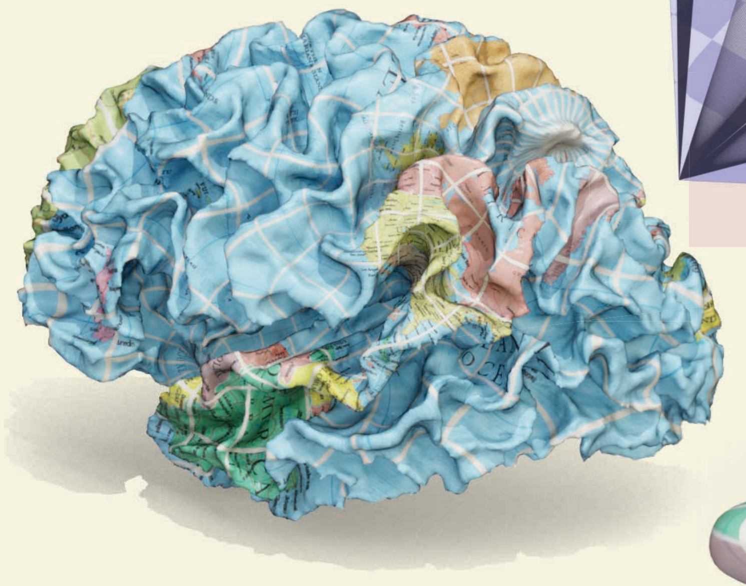


difficult cones



bad meshes





	Dataset	# Models	Success rate	Average time
cone flattening	**MPZ** [Myles+ 2014]	114	100%	8s
cone flattening	**Thingi10k** [Zhou+ 2016]	32,744*	97.7%	57s†
spherical uniformization	**brain scans** [Yeo+ 2009]	78	100%	493s
spherical uniformization	**anatomical surfaces** [Boyer+ 2011]	187	100%	15s

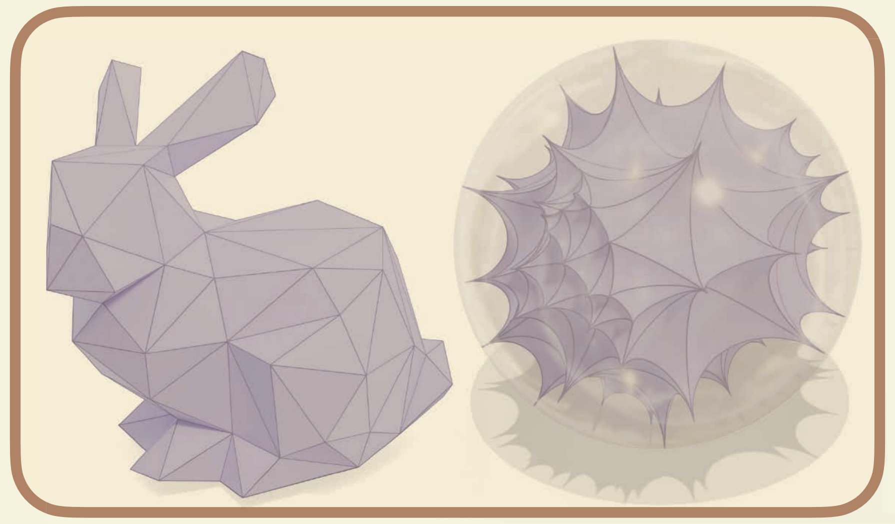
* connected components of models from Thingi10k

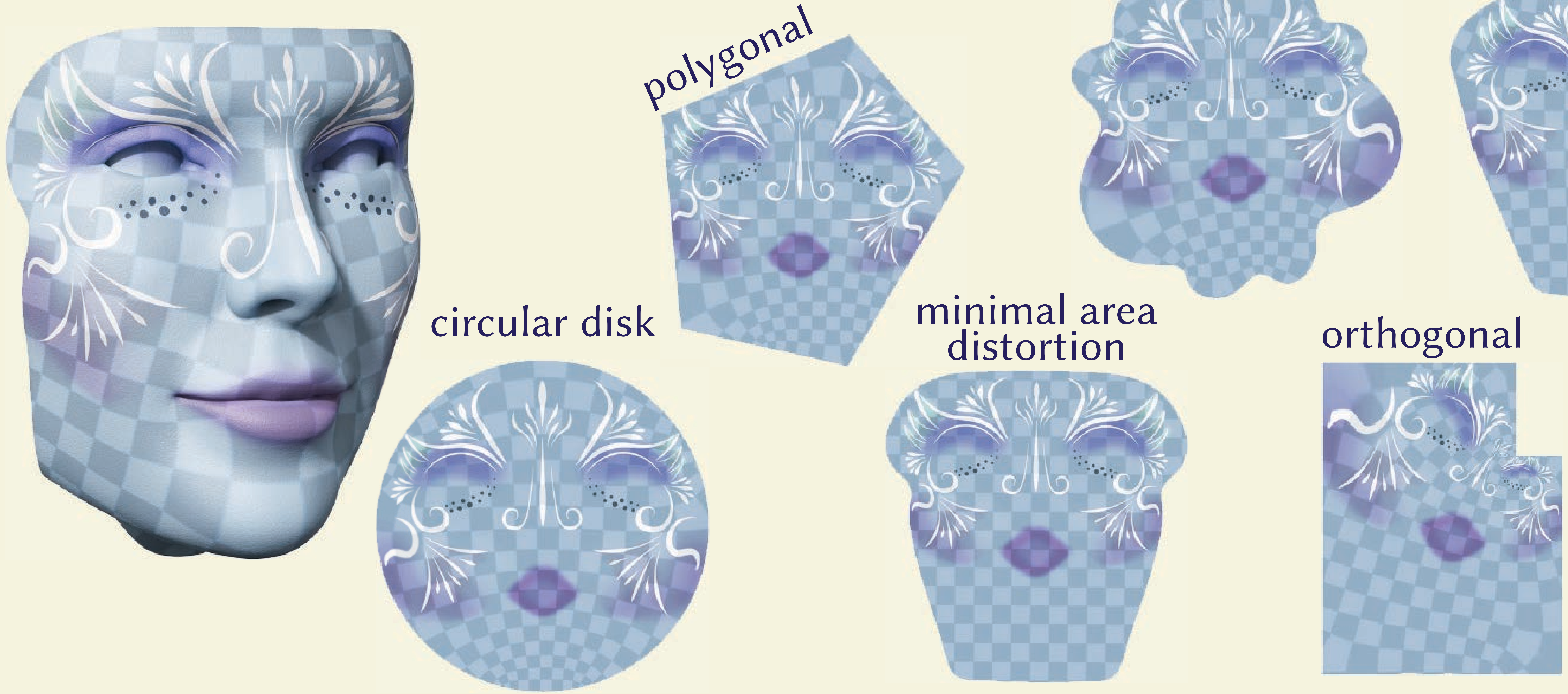
† average time on models with > 1000 vertices

Ptolemy flips improve performance

MPZ

Boundary conditions

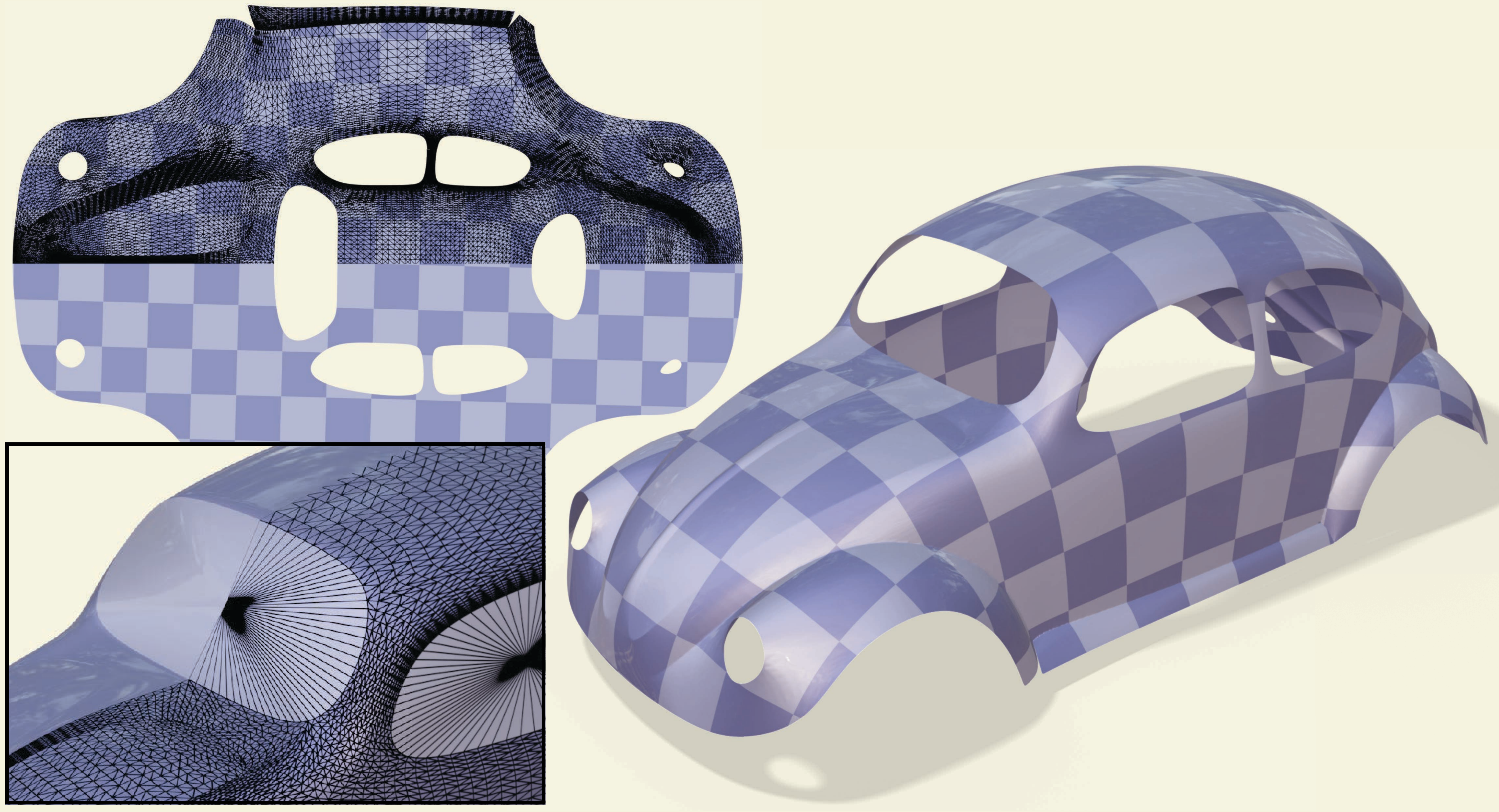




Multiply-connected domains

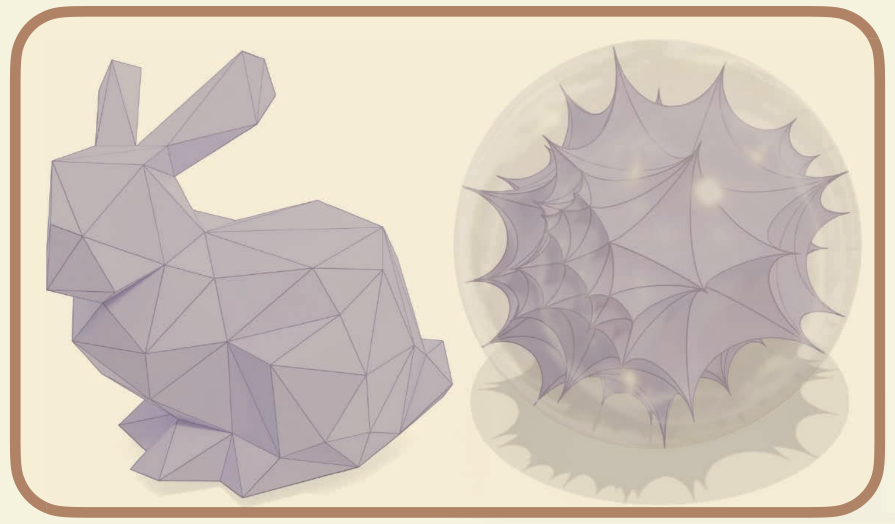
[Hefetz+2019]

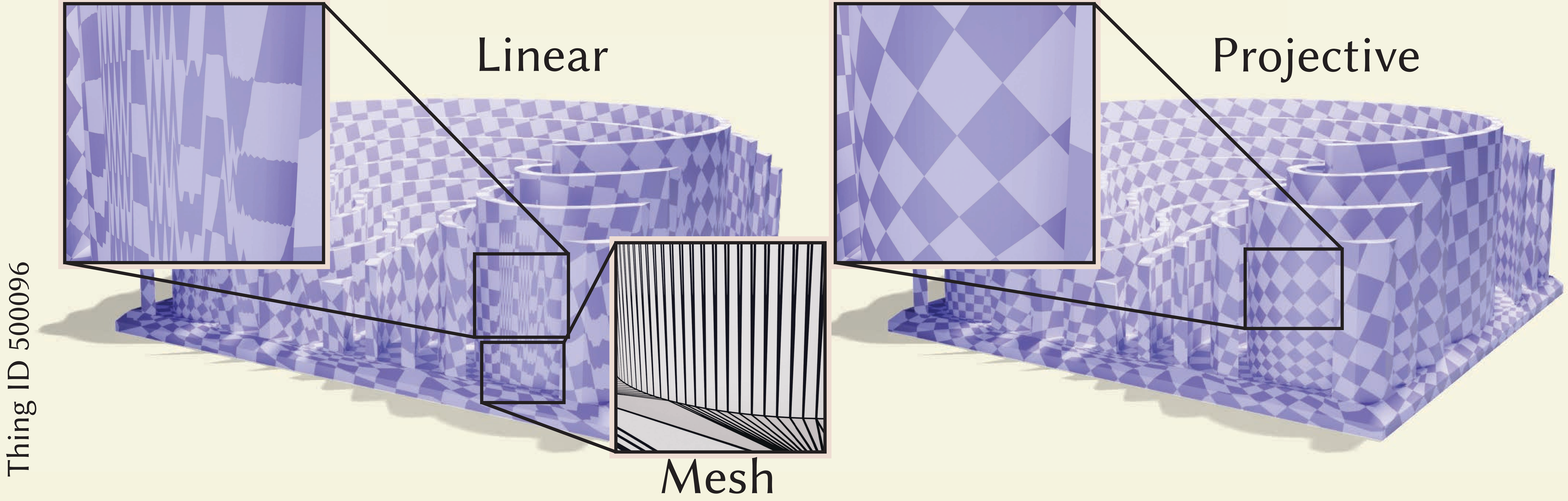
No hole filling



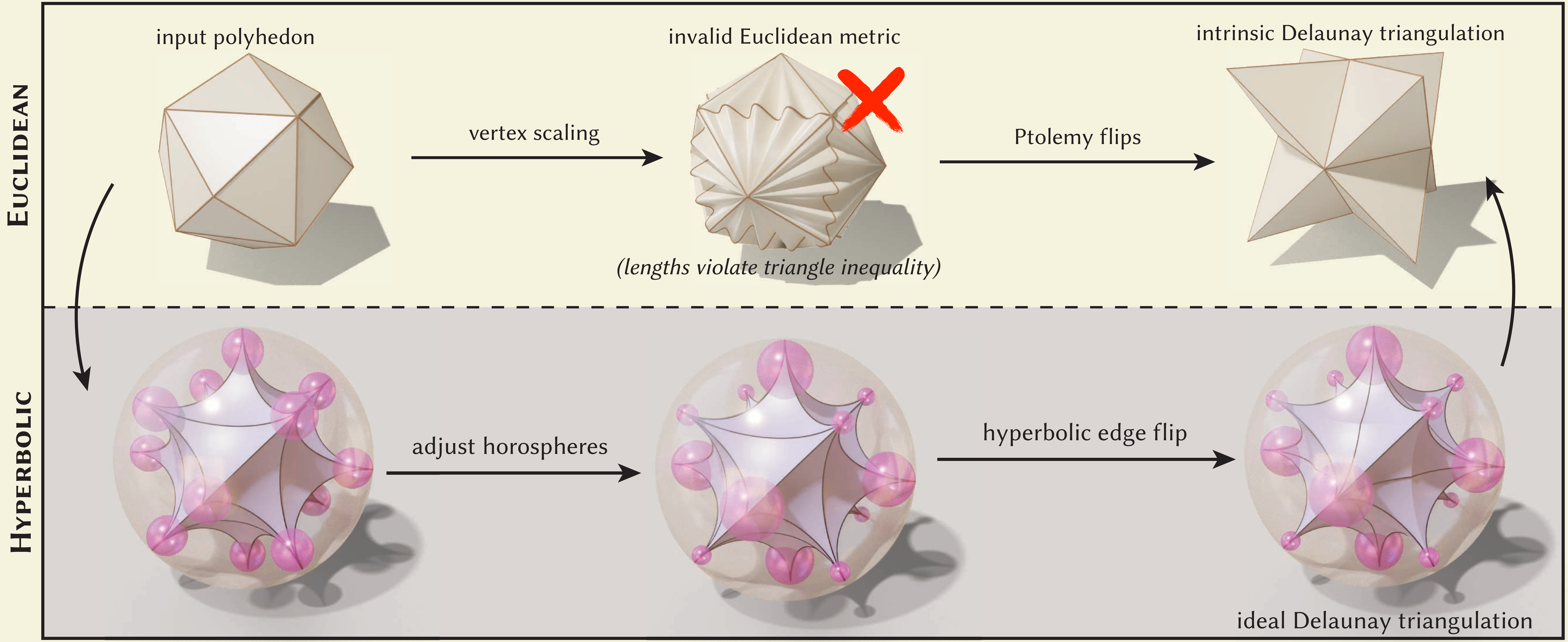
Hole filling

Projective interpolation improves quality



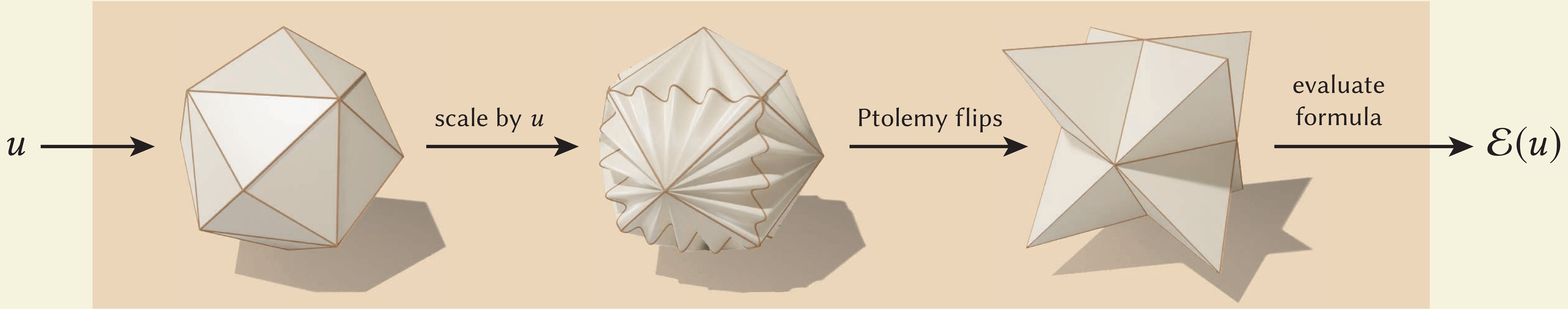


Discrete conformal equivalence across triangulations



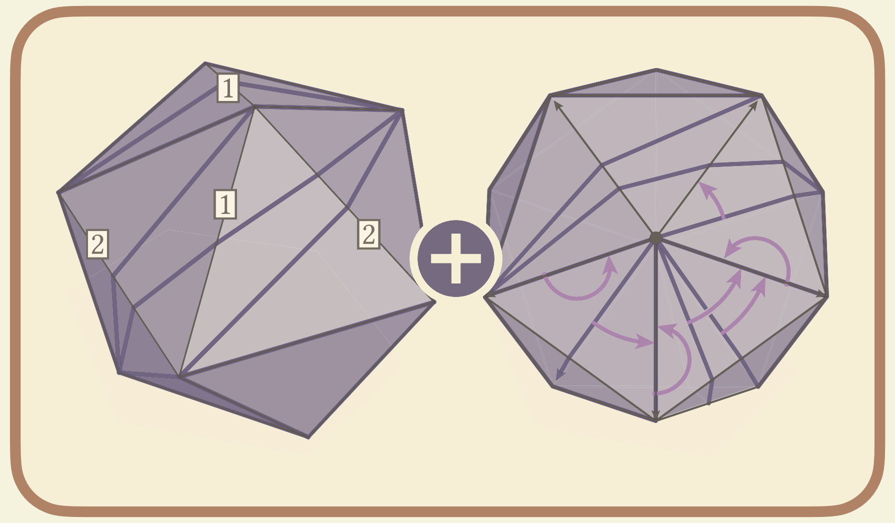
Optimization with Ptolemy Flips

- Express energy and derivatives in terms of edge lengths [Springborn 2019]

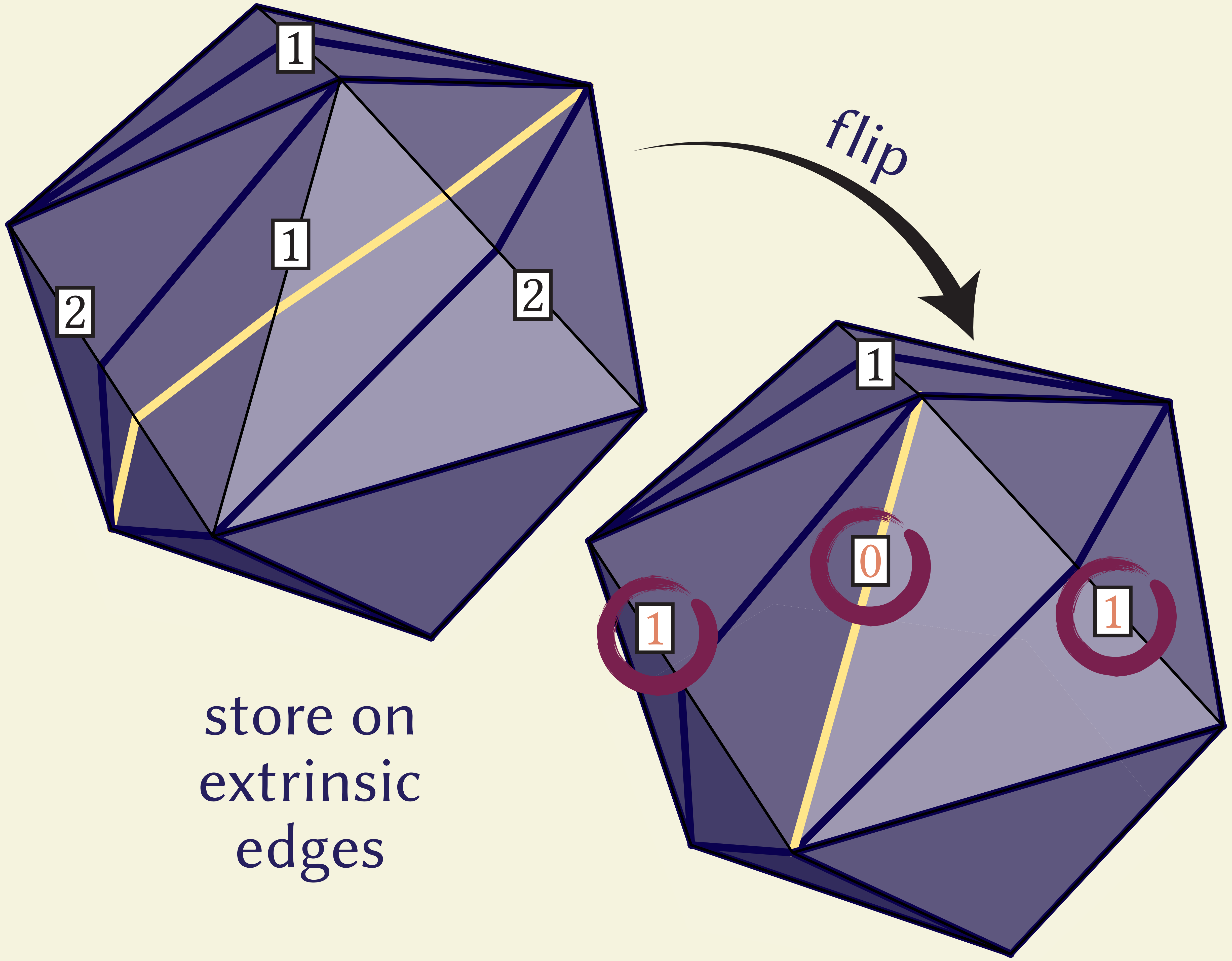


- Hand to any optimization algorithm

Where to store integer coordinates

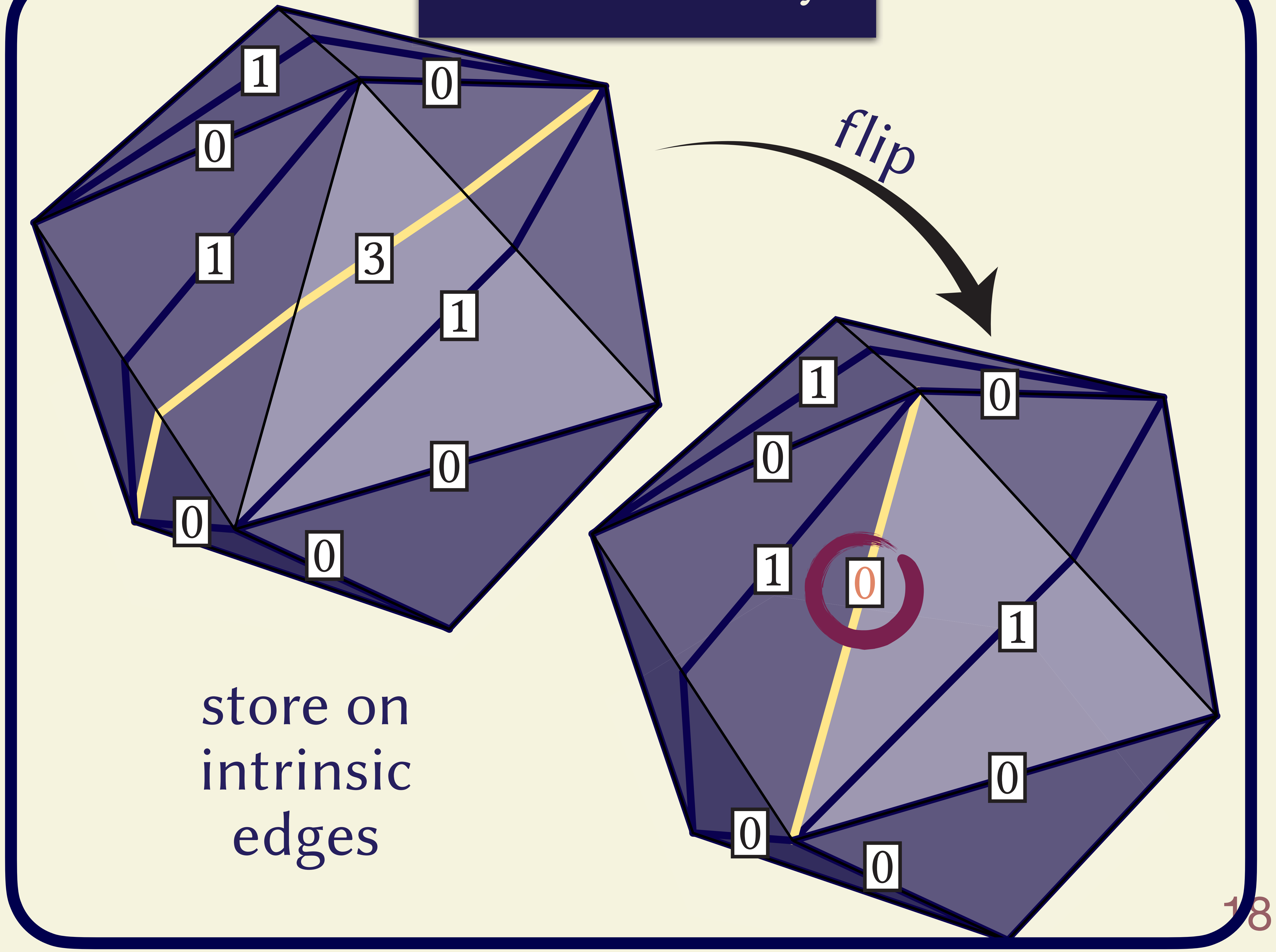


Integer coordinates for intrinsic triangulations



store on extrinsic edges

Better locality



store on intrinsic edges