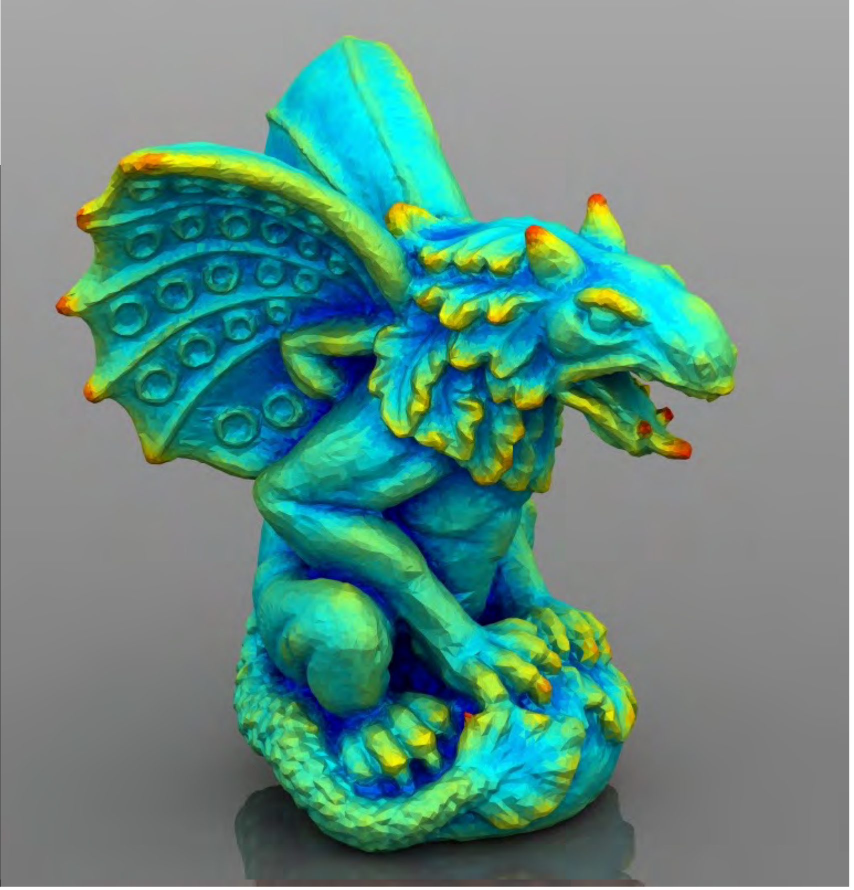
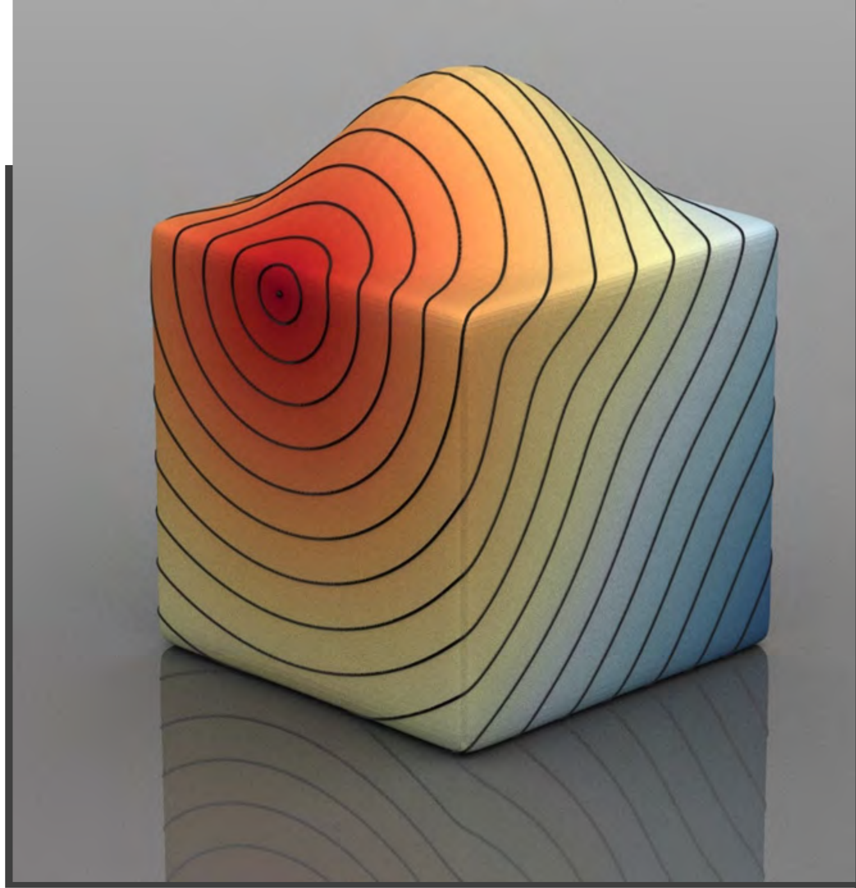




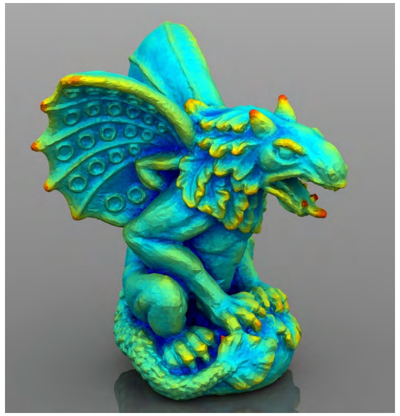
Mathematical Geometry Processing: Laplacian and Beyond

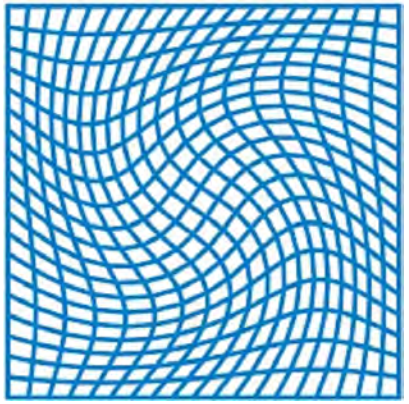
Yu Wang

Harvard Medical School

MIT Computer Science & Artificial Intelligence Laboratory

Geometry, Diffeomorphism, Inverse PDE & Operator Learning






Joint work with

- Prof. Mirela Ben-Chen, Technion
- Prof. Iosif Polterovich, U. of Montreal
- Dr. Vladimir Kim, Adobe Research
- Prof. Michael Bronstein, Oxford
- Minghao Guo, MIT
- Prof. Justin Solomon, MIT



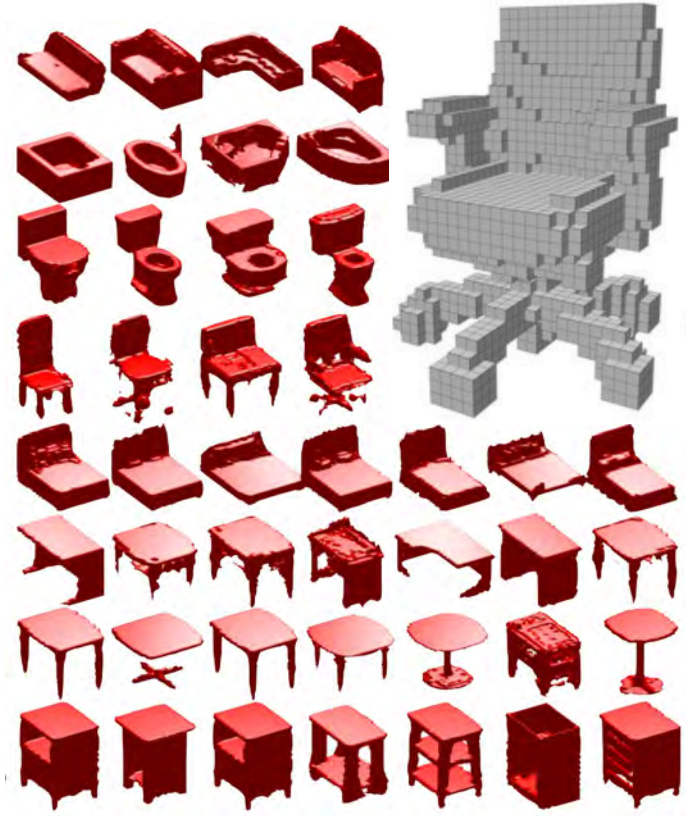



Review:
Operators for Geometric Computing

"Intrinsic and Extrinsic Operators for Shape Analysis"
"Variational Quasi-Harmonic Maps for Computing Diffeomorphisms"

Machine Learning on 3D Shapes?

- Voxel grid



3D ShapeNets [Wu et al. 2015]

- Point cloud

PointNet [Qi et al. 2016]

- Mesh

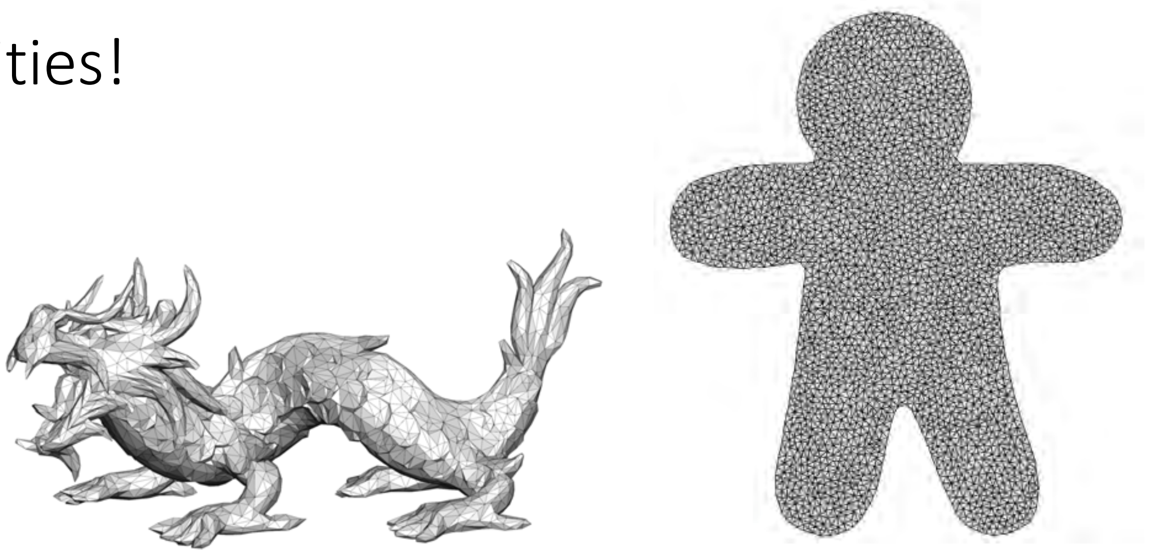
[Maron et al. 2017]

Geodesic CNN
[Masci et al. 2015]

The Operator/Matrix for Geometric Computing

- Geometry ***representation***: many possibilities!

distance on point clouds [Crane et al. 2013]

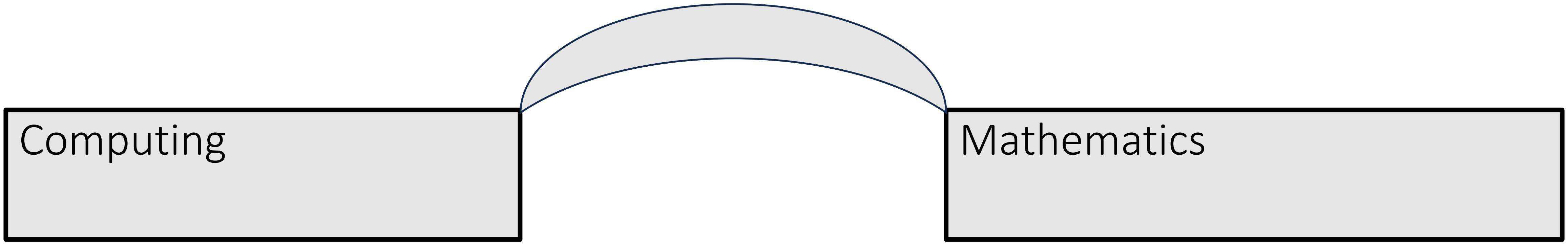


triangle meshes, polygonal meshes,

triangle soups, implicit representations…

- Q: algorithms behave ***consistently*** across representations?
- A: work with *continuous operator* and *discretize* it as a *matrix*.
  - geometric computing: identify the right computational model.

Bridging Geometric Computing and Applied Mathematics

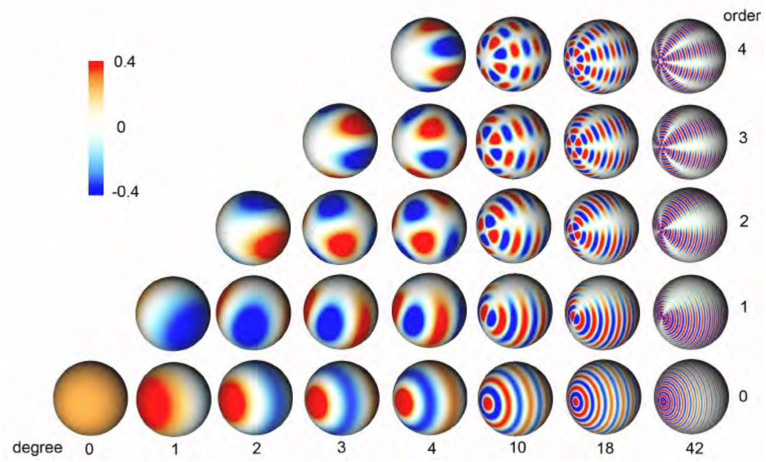


- identify / discover mathematics most relevant in
- designing geometric algorithms and numerical optimization
- that perform best on applications with empirical evaluation metrics

- math/geometry ideas → discretized PDEs/discrete representation → optimization & numerical algorithms → geometric algorithms → applications

Non-Euclidean Signals: Laplacian Spectral Basis

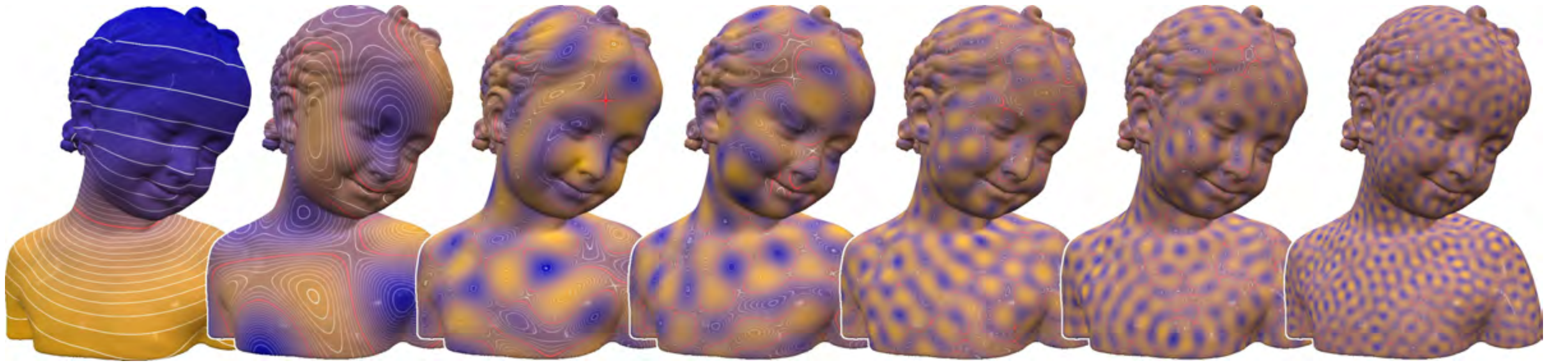
- Euclidean domain $\mathbb{T}^d$: Fourier bases, 1D: $\sin(nx), \cos(nx)$
  - $\Delta = \nabla \cdot \nabla = \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}$ $\Delta(\sin(nx)) = -n^2 \sin(nx)$
- Spherical domain: spherical harmonics
  - $\Delta = \dots$



[Chung]

- Any non-Euclidean domain: *eigenfunctions of Laplace-Beltrami (Laplacian)*
  - $\Delta = \nabla \cdot \nabla = \frac{1}{\sqrt{|\det g|}} \partial_i (\sqrt{|\det g|} g^{ij} \partial_j)$

g: metric tensor



$\Delta \phi_i = \lambda \phi_i$

Laplacian eigenfunctions [Levy 2006]

Laplacian Operator on Meshes?

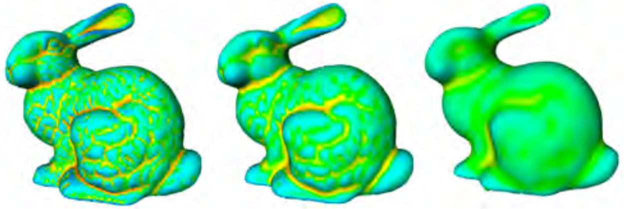
$$\Delta = \nabla \cdot \nabla = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2}$$

Image Processing.

- Laplacian on images
  - finite difference as approximation
  - $\frac{df(x)}{dx} \approx \frac{f(x+1)-f(x-1)}{2}$
  - $\Delta f(x,y) \approx \frac{f(x+1,y)+f(x-1,y)+f(x,y+1)+f(x,y-1)-4\times f(x,y)}{4}$
  - via the stencil:

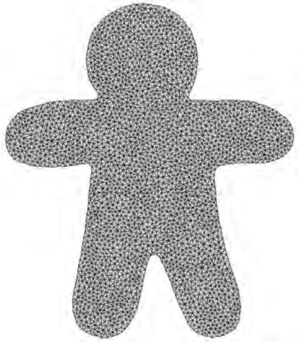
$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

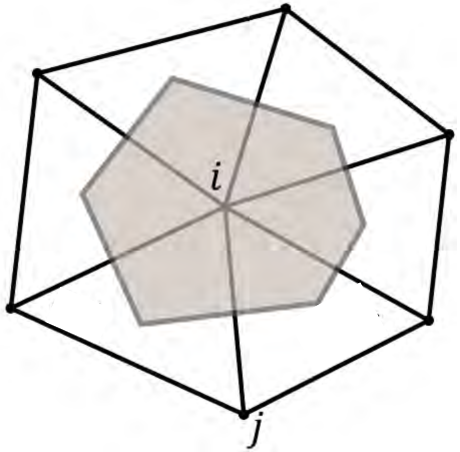
Mesh Processing.



- Laplacian on meshes?
  - similar idea, $\Delta u \approx \sum_{j \in \mathcal{N}(i)} L_{ij}(u_j - u_i)$

  but
  - *irregular connectivity*
  - *entries L_{ij} depends on the shape of triangles*





Triangle Mesh as Graph

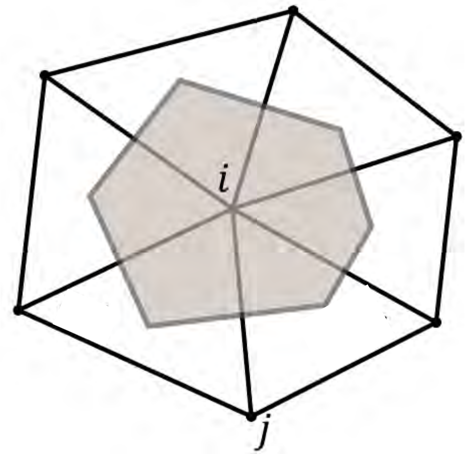
- Graph: nodes connected by edges ($\mathcal{E}$)
  - Dirichlet energy for $\mathrm{u} \in \mathbb{R}^{\mathrm{n}\times 1}$ (#nodes by 1)

$$E(\mathrm{u}) = \frac{1}{2} \sum_{\{i,j\}\in\mathcal{E}} w_{ij}\left(\mathrm{u}_i - \mathrm{u}_j\right)^2 = \frac{1}{2}\mathrm{u}^{\mathrm{T}}\mathrm{L}\mathrm{u}$$

  - adjacency matrix $\mathrm{J} \in \mathbb{R}^{\mathrm{e}\times\mathrm{n}}$ (#edges by #nodes)
  - graph Laplacian $\mathrm{L} \in \mathbb{R}^{\mathrm{n}\times\mathrm{n}}$ (#nodes by #nodes)

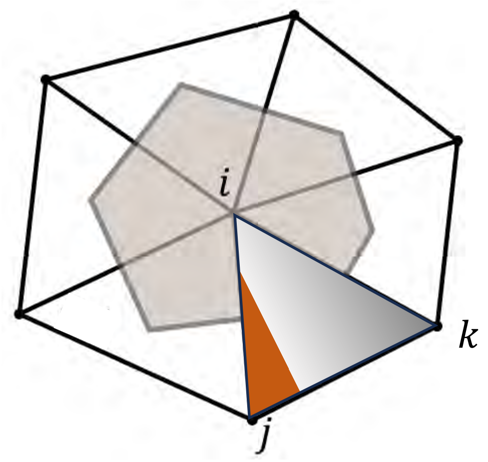
$$\mathrm{L}_{ij} = \begin{cases} w_{ij} & \text{if } \{i,j\} \text{ is an edge} \\ -\sum_{j\neq i} \mathrm{L}_{ij} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \qquad \mathrm{J}_{ei} = \begin{cases} 1 & \text{if } e \text{ is an edge starts at } i \\ -1 & \text{if } e \text{ is an edge ends at } i \\ 0 & \text{otherwise} \end{cases}$$

$$\mathrm{L} = \mathrm{J}^{\mathrm{T}}\mathrm{diag}(\mathrm{w})\,\mathrm{J}$$



- Mesh: still a graph, except the edges comes from triangles
  - graph has to be embeddable in 3D/2D
  - edges are associated with some lengths (with triangle inequality)

Triangle Mesh as Graph



- Mesh: vertices connected by triangles ($\mathcal{T}$)
  - Dirichlet energy for $\mathbf{u} \in \mathbb{R}^{n\times 1}$ (#nodes by 1)

$$E(\mathrm{u}) = \frac{1}{2} \sum_{t=\{i,j,k\}\in\mathcal{T}} ||\mathrm{G}_{ti}\mathrm{u}_i + \mathrm{G}_{tj}\mathrm{u}_j + \mathrm{G}_{tk}\mathrm{u}_k||^2 = \frac{1}{2}\mathrm{u}^{\mathrm{T}}\mathrm{L}\mathrm{u}$$

  - grad matrix $\mathbf{G} \in \mathbb{R}^{2f\times n}$ (2#faces by #vertices)
    - assume u is a piece-wise linear function in the triangle $t = \{i, j, k\}$
    - $\nabla u = \left(\mathrm{G}_{ti}u_i + \mathrm{G}_{tj}u_j + \mathrm{G}_{tk}u_k\right) \in \mathbb{R}^2$ is the grad of u in $t = \{i, j, k\}$
  - “mesh” Laplacian $\mathbf{L}$: still a graph Laplacian

$$\mathrm{L} = \mathrm{G}^{\mathrm{T}}\mathrm{G}$$

    - with weights depends on local geometry, i.e., G

$$\mathrm{L}_{ij} = \begin{cases} w_{ij} & \text{if } \{i,j\} \text{ is an edge} \\ -\sum_{j\neq i} \mathrm{L}_{ij} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

Graph Laplacian vs. Mesh Laplacian

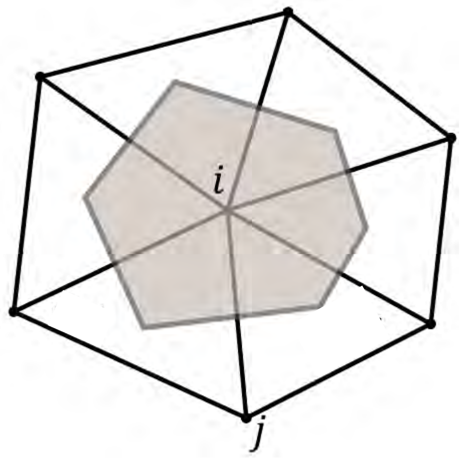
- Graph Laplacian $\mathbf{L} = \mathrm{J}^{\mathrm{T}}\mathrm{J}$
  - J: edge-node adjacency matrix
  - $\mathrm{J} \in \mathbb{R}^{\mathrm{e}\times\mathrm{n}}$ (#edges by #nodes)
  - $\mathrm{J}\mathbf{u}$: the difference per edge for $\mathbf{u} \in \mathbb{R}^{\mathrm{n}\times 1}$

- Mesh Laplacian $\mathbf{L} = \mathrm{G}^{\mathrm{T}}\mathrm{G}$
  - G: face-vertex gradient matrix
  - $\mathrm{G} \in \mathbb{R}^{2\mathrm{f}\times\mathrm{n}}$ (2#faces by #vertices)
  - $\mathrm{G}\mathbf{u}$: the gradient per triangle for $\mathbf{u} \in \mathbb{R}^{\mathrm{n}\times 1}$

Harmonic Functions: Mean Value Property

- Optimization: $\min \frac{1}{2}\mathrm{u}^{\mathrm{T}}\mathrm{Lu} \ \mathrm{s.t.}\ \mathrm{R}^{\mathrm{T}}\mathrm{u} = \mathrm{b}$
  - $\mathrm{R}^{\mathrm{T}}\mathbf{u} = \mathrm{b}$: the constraint that $\mathbf{u}$ is known at some nodes
  - R is the binary selection matrix choosing *known* rows in $\mathbf{u}$
  - S is the binary selection matrix choosing *unknown* rows in $\mathbf{u}$
  - $\mathrm{S}^{\mathrm{T}}\mathrm{R} = \mathbf{0}$
- Lagrangian multiplier method:
  - $\min \frac{1}{2}\mathrm{u}^{\mathrm{T}}\mathrm{Lu} + \lambda^{\mathrm{T}}(\mathrm{R}^{\mathrm{T}}\mathrm{u} - \mathrm{b})$
  - $\mathrm{Lu} + \mathrm{R}\lambda = 0 \rightarrow \mathrm{S}^{\mathrm{T}}\mathrm{Lu} = 0$
- This implies the unknown $\mathbf{u}_i$ is

harmonic: $\mathbf{u}_i$ equals to the weighted average of its neighbors' values



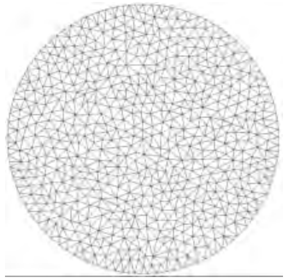
$$\mathrm{u}_i = \frac{1}{\sum_{j\in\mathcal{N}(i)} \mathrm{L}_{ij}} \sum_{j\in\mathcal{N}(i)} \mathrm{L}_{ij}\mathrm{u}_j$$

Discrete v.s. Continuous: Harmonic Interpolation

- A quadratic optimization with linear constraints

minimize the derivation of u_i or $u(\mathrm{x})$ locally

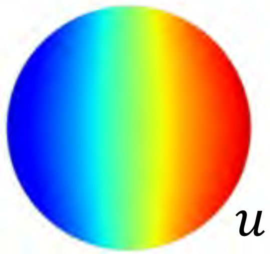
- $\min_{\mathrm{u}\in\mathbb{R}^n} \frac{1}{2}\mathrm{u}^{\mathrm{T}}\mathrm{L}\mathrm{u} \;\; \mathrm{s.t.} \;\; \mathrm{R}^{\mathrm{T}}\mathrm{u} = \mathrm{b}$



- $\mathrm{S}^{\mathrm{T}}\mathrm{G}^{\mathrm{T}}\mathrm{G}\,\mathrm{u} = 0$
  - S selects *unknown* rows in u
  - solve a linear system

- $\min_{u(\mathrm{x})} \frac{1}{2}\int_{\Omega} |\nabla u(\mathrm{x})|^2 d\mathrm{x} \;\; \mathrm{s.t.} \;\; u|_{\partial\Omega} = b(\mathrm{x})$

$b(\partial\Omega)$



$u(\Omega)$

- $\nabla \cdot [\nabla u(\mathrm{x})] = 0$
  - for $\mathrm{x} \in \Omega \backslash \partial\Omega$: where $u(\mathrm{x})$ is unknown
  - solve a linear PDE

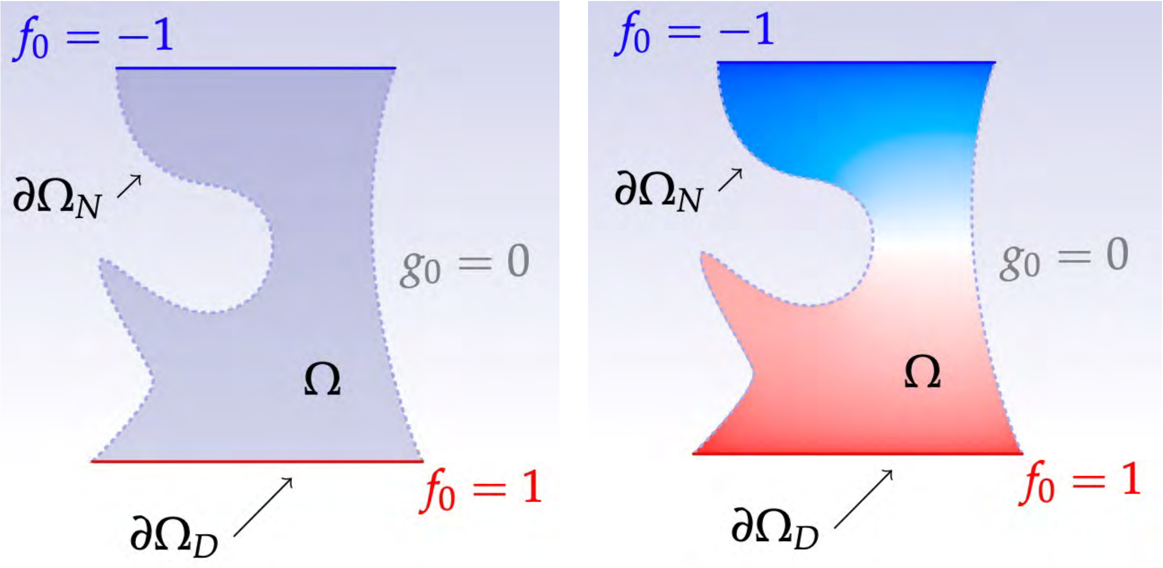
- Mean Value Property: u_i or $u(\mathrm{x})$ equals to the weighted average of its neighbors' values

Laplace Equation: PDE 101

Definition (Laplace Equation: Mixed boundary condition)

Consider a partition of the boundary: $\partial\Omega = \partial\Omega_N \cup \partial\Omega_D$ such that $\partial\Omega_N \cap \partial\Omega_D = \emptyset$.

$$
\begin{aligned}
\Delta f(\mathrm{x}) &= 0, & \forall \mathrm{x} \in \Omega \backslash \partial\Omega \quad & \text{(interior)} \\
f(\mathrm{x}) &= f_0(\mathrm{x}), & \forall \mathrm{x} \in \partial\Omega_D \quad & \text{(Dirichlet condition)} \\
\mathrm{n} \cdot \nabla f(\mathrm{x}) &= g_0(\mathrm{x}), & \forall \mathrm{x} \in \partial\Omega_N \quad & \text{(Neumann condition)}
\end{aligned}
$$



Images from [Etienne et al. 2014]

$$\Delta = \nabla \cdot \nabla = \frac{\partial^2}{\partial \mathrm{x}_1^2} + \frac{\partial^2}{\partial \mathrm{x}_2^2}$$

- Fact: forward PDE allows only one boundary condition

A Closely Related Concept: Dirichlet-to-Neumann (DtN) Operator $\mathcal{S}$

Consider a shape Ω bounded by the surface $\Gamma = \partial\Omega$.

$$\begin{cases} \Delta u(\mathbf{x}) = 0 & \mathbf{x} \in \Omega \\ u(\mathbf{x}) = g(\mathbf{x}) & \mathbf{x} \in \partial\Omega \end{cases}$$

where $g(\Gamma)$ is Dirichlet data

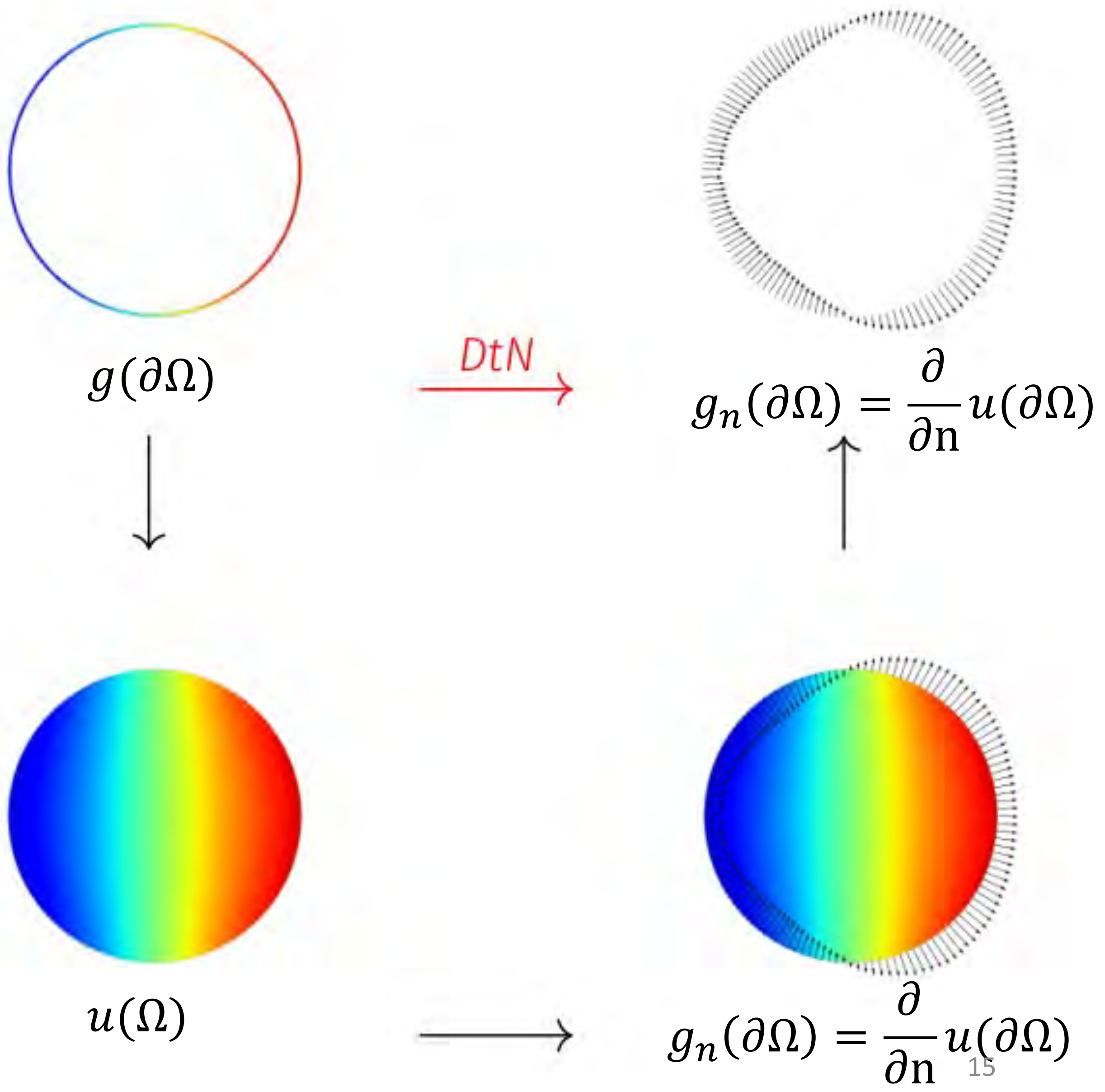
Neumann data $g_n = \frac{\partial}{\partial n} u(\Gamma)$

Dirichlet-to-Neumann (DtN) operator:

$$\mathcal{S} := g \mapsto g_n$$

a.k.a the Steklov-Poincaré operator.

(temperature-to-flux, voltage-to-current)

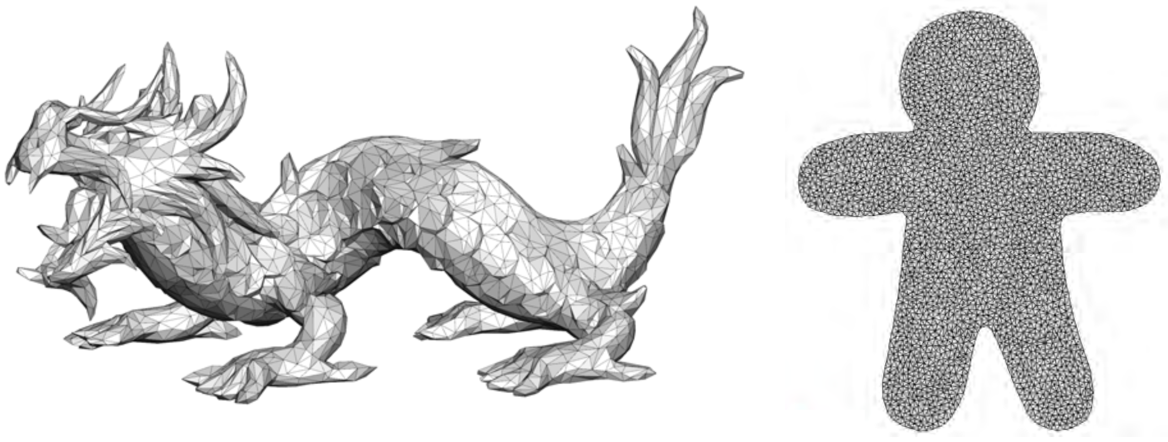


Discrete Laplacian: A Sparse Matrix

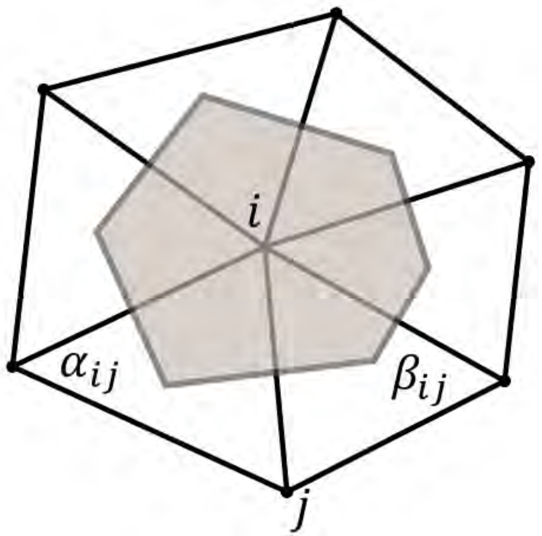
continuous $\Delta = \nabla \cdot \nabla$

discrete

$$\mathrm{L} = \mathrm{G}^{\mathsf{T}}\mathrm{G} \in \mathbb{R}^{n \times n}$$



$\mathrm{G} \in \mathbb{R}^{2f \times n}$: gradient operator (weighted)
n: #vertices
f: #faces



$$\mathrm{L}_{ij} = \begin{cases} {}^{1}\!/_{2}\,(\cot \alpha_{ij} + \cot \beta_{ij}) & \text{if } \{i, j\} \text{ is an edge} \\ -\sum_{j \neq i} \mathrm{L}_{ij} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

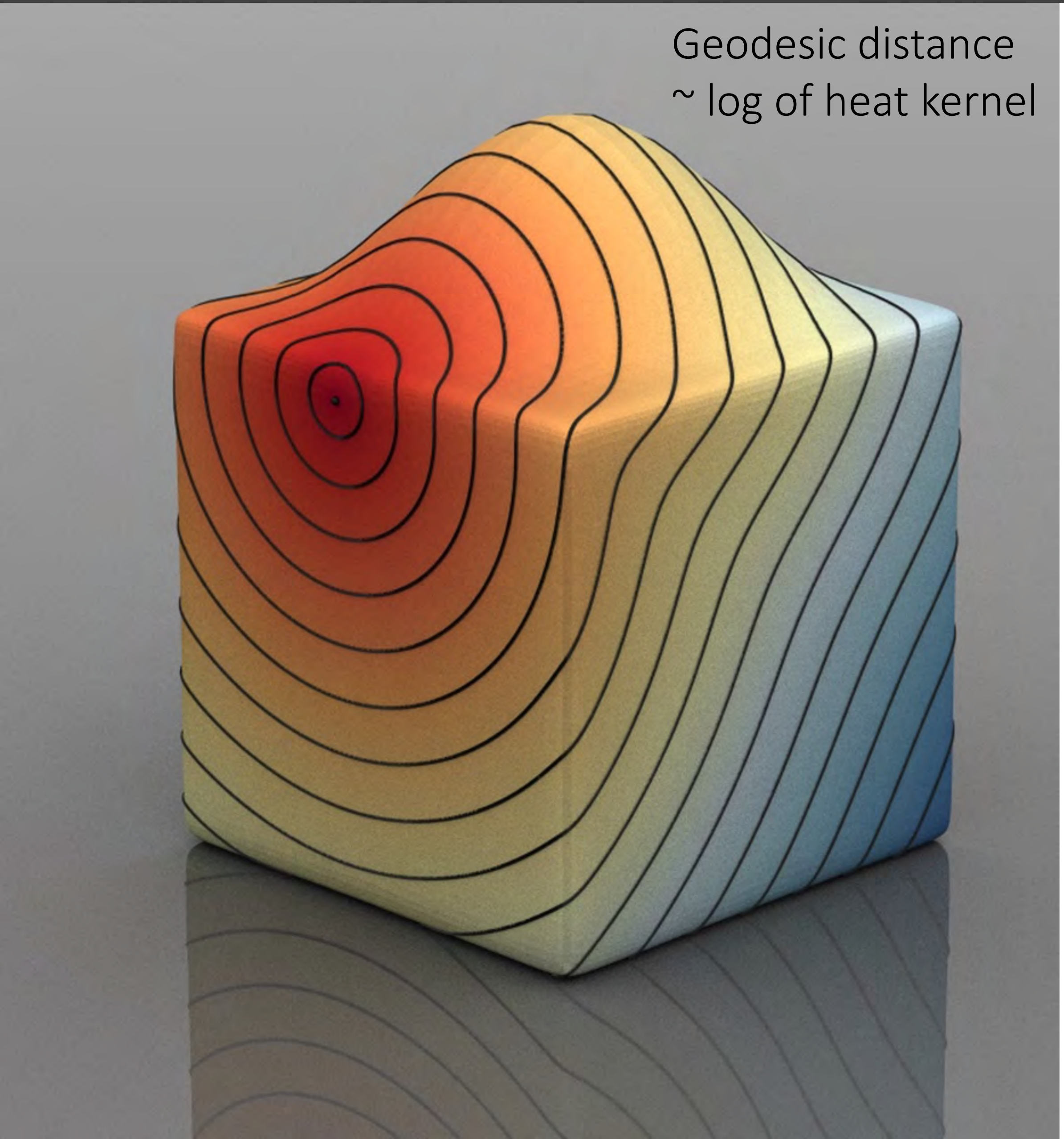
L is a graph Laplacian with geometry-determined edge weights.

Same formula for curved and flat surfaces

Finite Element Method (FEM) [Steinbach 2007]

Discrete Exterior Calculus (DEC) [Desbrun et al., 2005]

Tasks in Geometric Computing



Geodesic distance
~ log of heat kernel

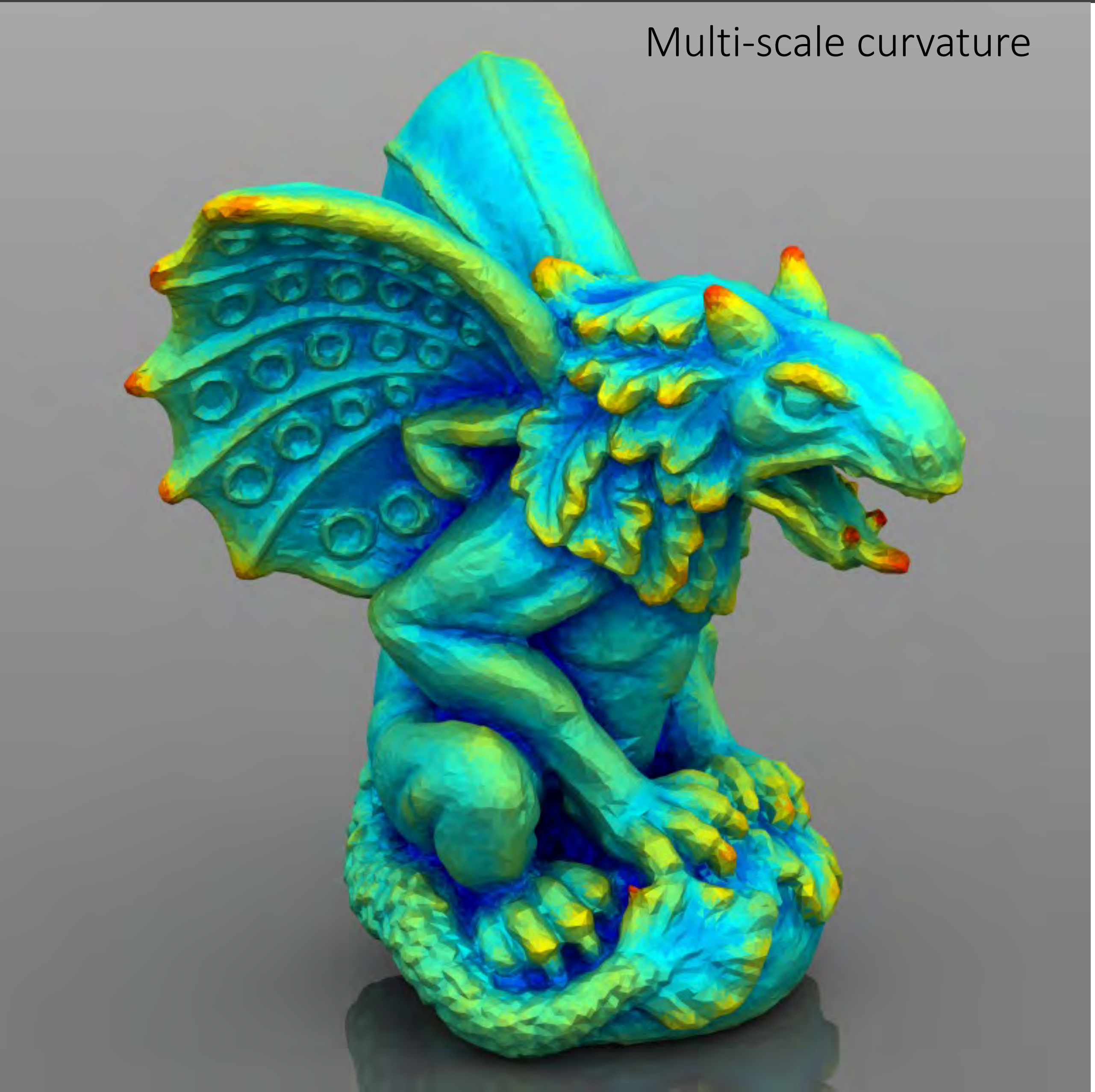
- Distance

Tasks in Geometric Computing

Conformal parameterization or diffeomorphism

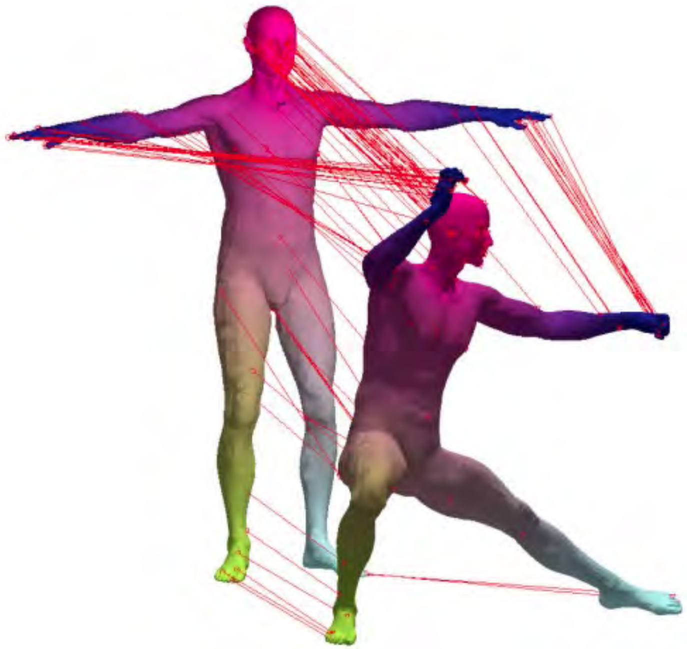
- Distance
- Parameterization

Tasks in Geometric Computing



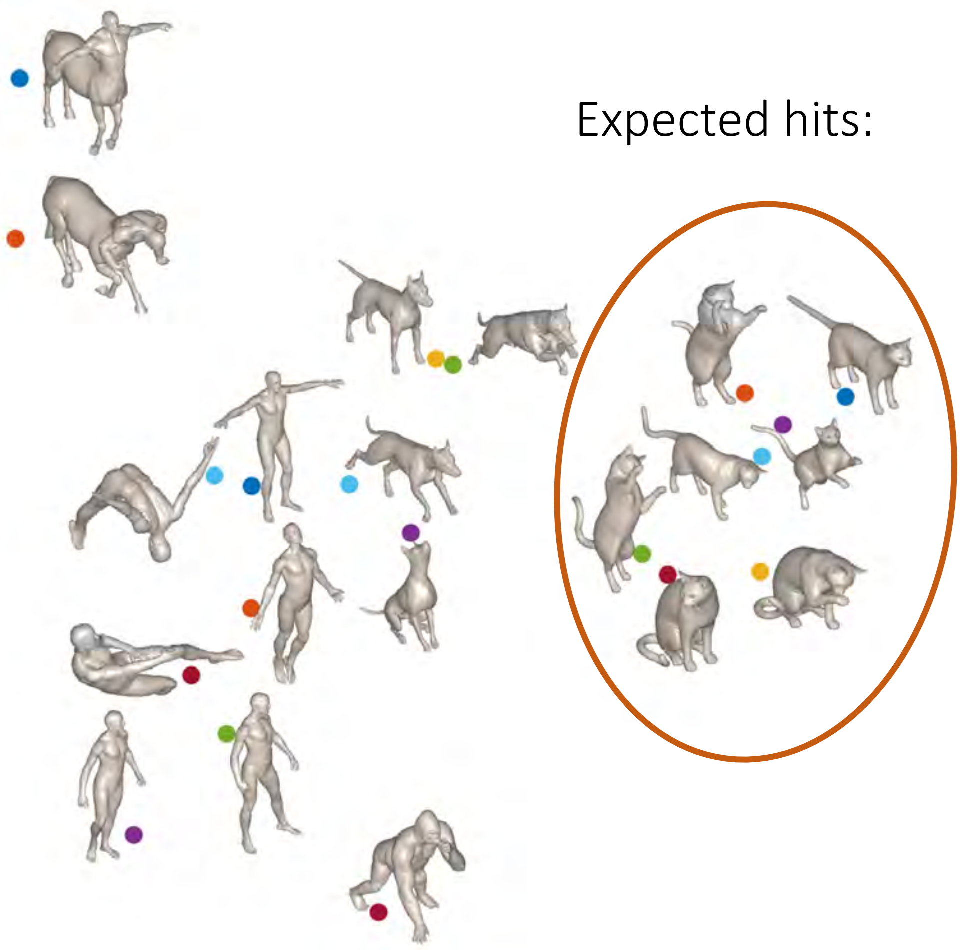
Multi-scale curvature

- Distance
- Parameterization
- **Shape descriptor**
- **Correspondence**



[Lombaert et al. 2013]

Tasks in Geometric Computing



Query: Cat

Expected hits:

- Distance
- Parameterization
- Shape descriptor
- Correspondence
- **Shape classification**

Laplacian: Ubiquitous in Shape Analysis & Geometry Processing

```
def Geometry_Processing(mesh):
    L = Laplacian_Operator(mesh)
    del mesh # delete the mesh
    results = Linear_Solve(L)
    ...
    return results
```

- Distance [Crane et al. 2013]
- Parameterization [Mullen et al. 2008]
- Shape description [Sun et al. 2009]
- Correspondence [Ovsjanikov et al. 2012]
- Shape classification [Bronstein et al. 2011]
- Shape exploration [Rustamov et al. 2013]
- Deformation [Boscaini et al. 2015]
- Shape optimization...
- Mesh generation...

$\mathrm{L} \rightarrow \mathrm{L}^{(\mathrm{A})}, \tilde{\mathrm{L}}$, or S

This talk covers:

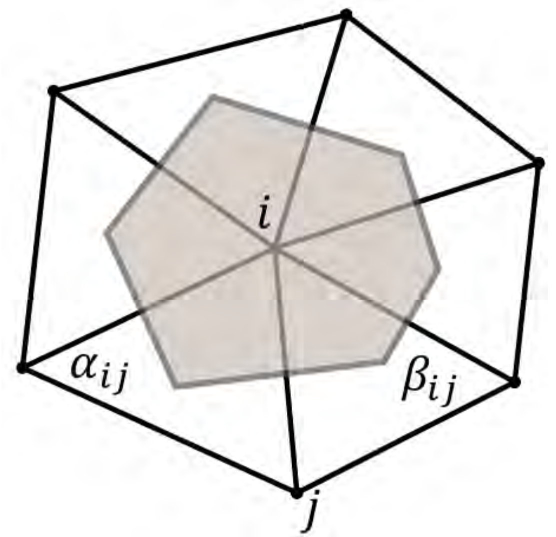
- Search for a matrix $\mathrm{L}^{(\mathrm{A})} = \mathrm{G}^{\intercal}\mathrm{A}\mathrm{G}$: same sparsity pattern to $\mathrm{L} = \mathrm{G}^{\intercal}\mathrm{G}$ (major focus)
- Learn from data FEM kernels to assemble entries $\tilde{\mathrm{L}}_{ij}$
- Design explicitly a different matrix S: more informative/robust

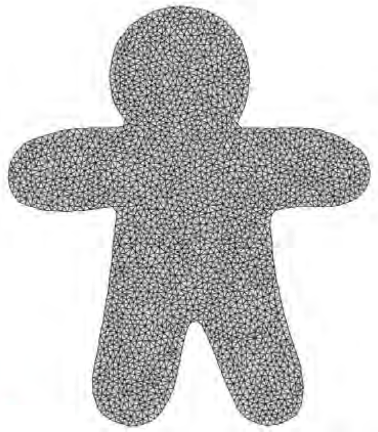
Why?

- New operators → (much) more expressive computational models
- Systematically improve potentially every task in geometric computing

Optimization in the Space of Laplacians

"Harmonicity": The Key Notion of "Smoothness"



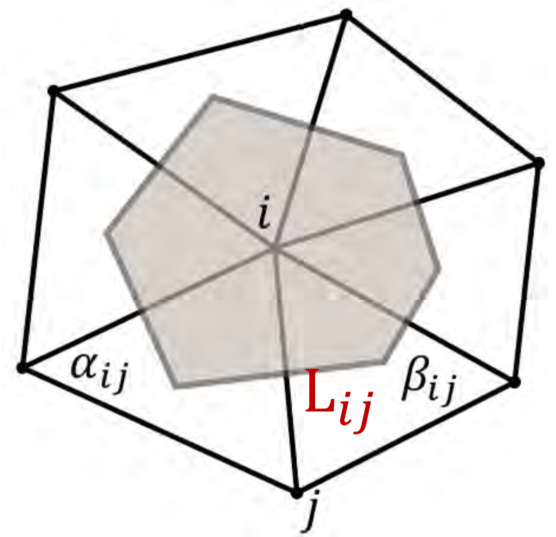


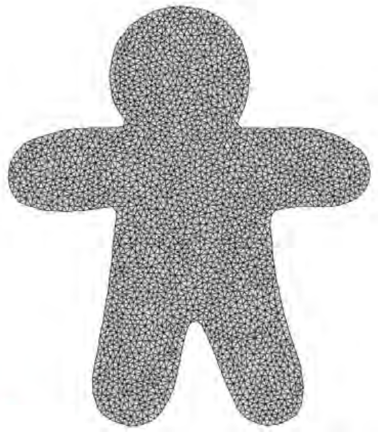
- Key: what is a smooth function on the mesh/graph?
- Harmonic function: a function $\mathbf{u}$ whose value at each node/vertex i equals to the average over $\mathcal{N}(i)$, the neighbors of i

$$\mathrm{u}_i = \frac{1}{\sum_{j\in\mathcal{N}(i)} 1} \sum_{j\in\mathcal{N}(i)} \mathrm{u}_j$$

$$(\Delta \mathrm{u})_i := \frac{1}{\sum_{j\in\mathcal{N}(i)} 1} \sum_{j\in\mathcal{N}(i)} (\mathrm{u}_i - \mathrm{u}_j)$$

“Harmonicity”: The Key Notion of “Smoothness”





- Key: what is a smooth function on the mesh/graph?
- Quasi-harmonic function: a function $\mathbf{u}$ whose value at each node/vertex i equals to the weighted average over $\mathcal{N}(i)$, the neighbors of i

$$\mathrm{u}_i = \frac{1}{\sum_{j \in \mathcal{N}(i)} \mathrm{L}_{ij}} \sum_{j \in \mathcal{N}(i)} \mathrm{L}_{ij} \mathrm{u}_j$$

$$\left(\Delta^{(w)} \mathrm{u}\right)_i = \frac{1}{\sum_{j \in \mathcal{N}(i)} \mathrm{L}_{ij}} \sum_{j \in \mathcal{N}(i)} \mathrm{L}_{ij} (\mathrm{u}_i - \mathrm{u}_j)$$

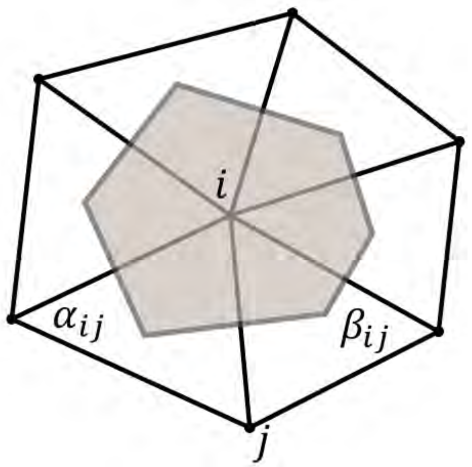
Quasi-Harmonic Function and Generalized Laplacians

- Smooth $\Delta = \nabla \cdot \nabla$
- Harmonic: $\Delta u = 0$
  - u_i equals to the *average* over i's neighbors

$$0 = \sum_{j \in N(i)} (u_j - u_i)$$

- Smooth $\Delta^{(\mathrm{A})} = \nabla \cdot [\mathrm{A}(\mathrm{x})\nabla]$
- Quasi-harmonic: $\Delta^{(\mathrm{A})} u = 0$
  - u_i equals to the *weighted-average* over i's neighbors

$$0 = \sum_{j \in N(i)} \mathrm{L}_{ij}(u_j - u_i)$$

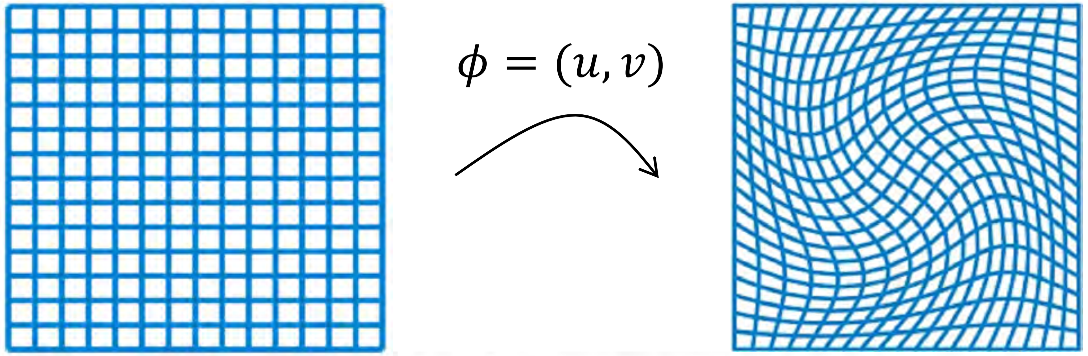


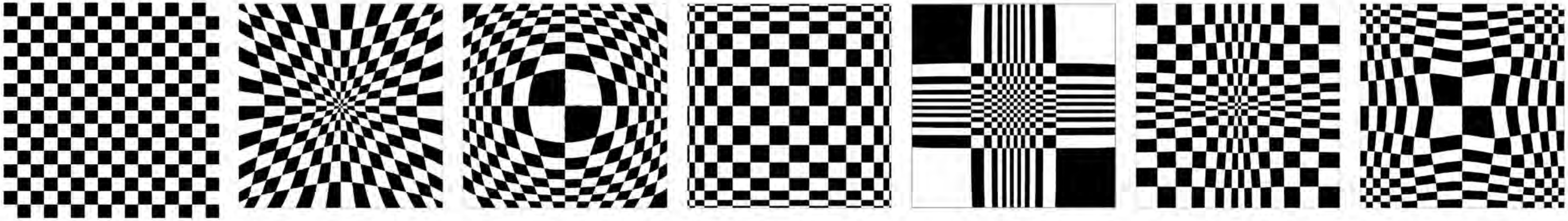
Inverse Problems of PDEs for Computing Diffeomorphisms

"Variational Quasi-Harmonic Maps for Computing Diffeomorphisms."
Yu Wang, Minghao Guo, and Justin Solomon.
ACM Transactions on Graphics (TOG) 42(4). ACM SIGGRAPH 2023 Journal Track.

Diffeomorphisms

- Diffeomorphism ϕ: a *smooth* map with *smooth* inverse (ϕ^{-1} must exist)
  - diffeomorphisms: all physically possible deformation (no negative volume)
- Homeomorphism ϕ: *smooth* → *continuous*
  - injective: $\phi(\mathrm{x}) \neq \phi(\mathrm{y})$ for $\mathrm{x} \neq \mathrm{y}$
  - inversion-free: $\det D\phi(\mathrm{x}) > 0, \forall \mathrm{x}$, positive Jacobian $D\phi(\mathrm{x}) \in \mathrm{R}^{2\times 2}$

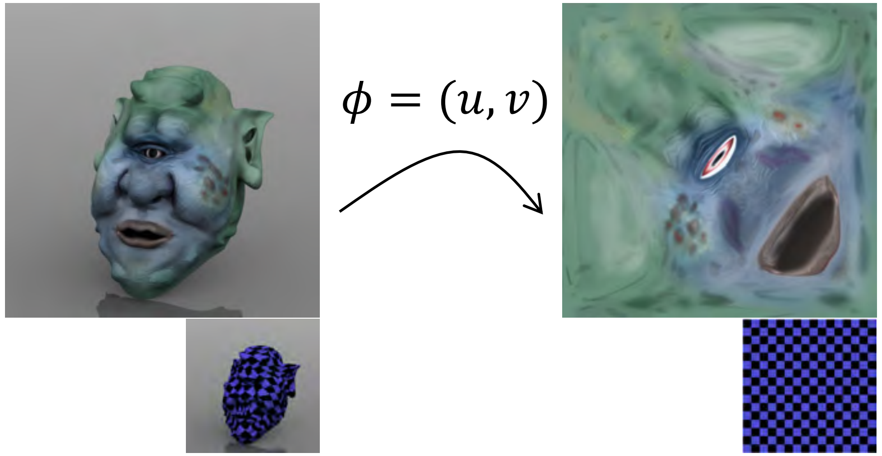


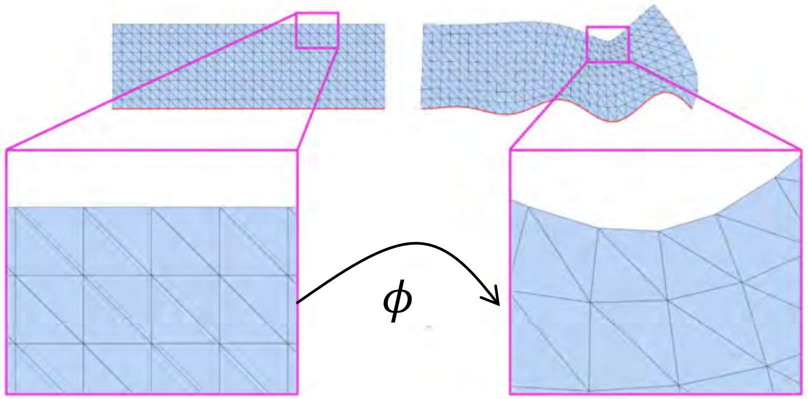


Diffeomorphism = Smooth Injective / Inversion-free Mapping

- The map $\phi = (u, v)$ can be a:

  deformation, shape representation, correspondence, parameterization, …





[Aigerman and Kovalsky 2016]

- Foundational, wherever using computers to represent shapes

  in physics, engineering, shape optimization, computer vision, mesh generation…

- Homeomorphism = Inversion-freeness (Under conditions) e.g. [Lipman 2014]

$$\det\left[\nabla u \; \nabla v\right] > 0$$

[Models from Keenan Crane]

Our Solution: (Quasi) Harmonic Maps

- Previous works, at high-level:

$$\min E(u,v) \quad \text{s.t.} \quad \det[\nabla u \; \nabla v] > 0$$

  - largely relying on constrained numerical optimization
  - solved by customized barrier / interior point methods
  - while minimizing some energy E, e.g., the Winslow functional from physics

- Starting point: quasi-harmonic map $\nabla \cdot \left[\mathrm{A}(\mathrm{x}) \, [\nabla(u,v)]\right] = (0,0)$
- Our method

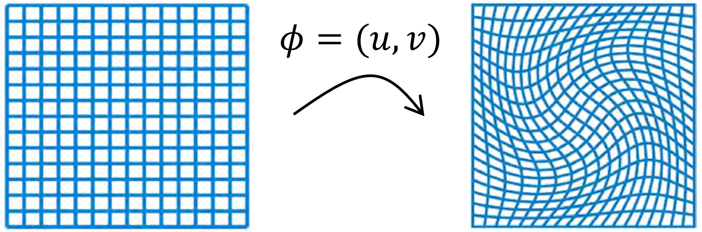
$$\min R(u,v,\mathrm{A}) \quad \text{s.t.} \quad \nabla \cdot \left[\mathrm{A}(\mathrm{x}) \, [\nabla(u,v)]\right] = (0,0)$$

- First review relevant ideas from
  - geometric graph theory
  - complex analysis / 2D PDEs

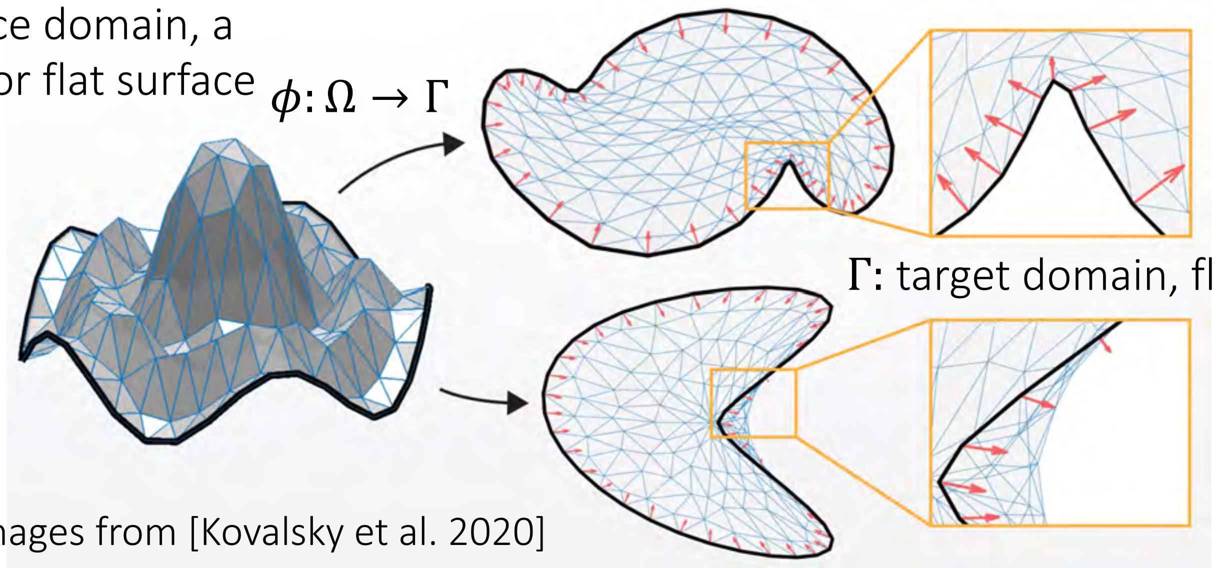
Problem: Flatten a Surface subject to Positional Constraints

Problem Setup

Suppose Ω is a two-dimensional Riemannian manifold with disk topology, and consider a planar domain $\Gamma \subset \mathbb{R}^2$ whose boundary $\partial\Gamma$ is a simple closed curve. Assume $\phi = (u, v) : \Omega \to \Gamma$ diffeomorphically maps $\partial\Omega$ onto $\partial\Gamma$. Denote the (given) boundary map as $[b_u, b_v](\cdot) : \partial\Omega \to \mathbb{R}^2$, and denote the outward normal to $\partial\Gamma$ as $\hat{\mathrm{n}}(\cdot) : \partial\Omega \to S^1$.



Ω: source domain, a curved or flat surface



Γ: target domain, flat

Images from [Kovalsky et al. 2020]

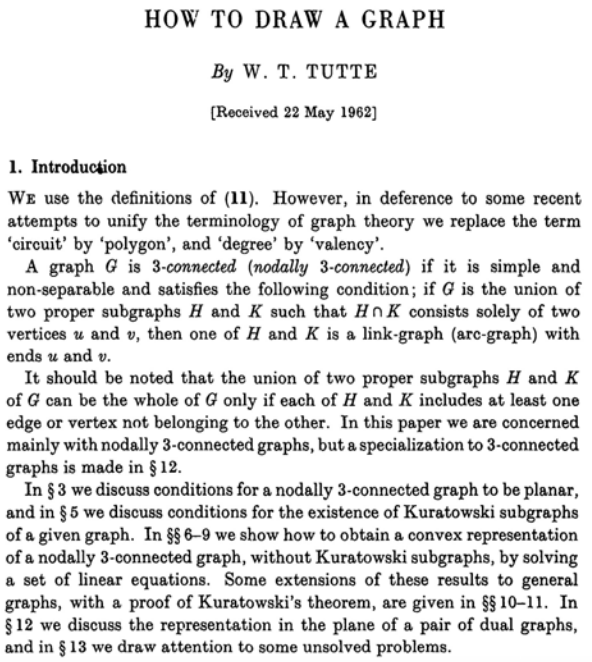
Question?

How to find a map ϕ that is diffeomorphic (and minimizes certain functional)?

Review: Tutte Embedding = Discrete Quasi-Harmonic Maps

- Fixed boundary, interior nodes placed at neighbors' **weighted** average
  - interior positions found by solving a linear system
- *Convex* boundary → edges do not intersect (injective!)
  - [Tutte 1962]: created the field of geometric graph theory

$$x_i = \sum_j w_{ij} \, x_j$$



HOW TO DRAW A GRAPH

By W. T. TUTTE

[Received 22 May 1962]

1. Introduction

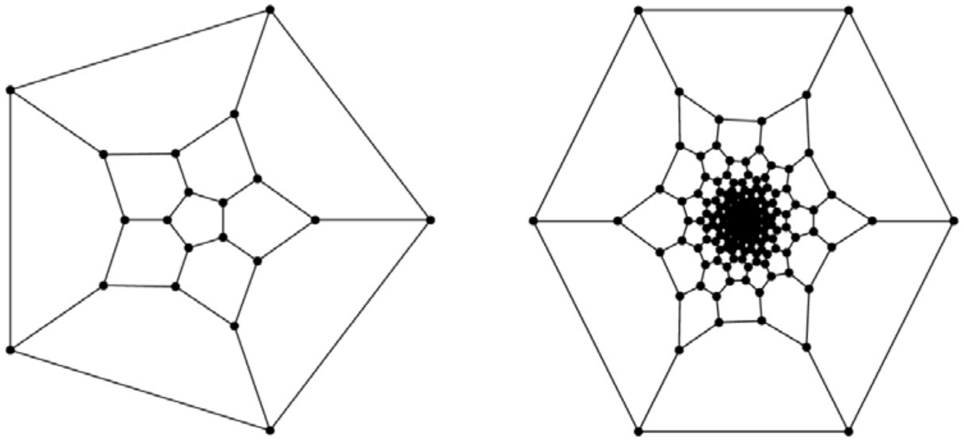
We use the definitions of (**11**). However, in deference to some recent attempts to unify the terminology of graph theory we replace the term 'circuit' by 'polygon', and 'degree' by 'valency'.

A graph G is *3-connected* (*nodally 3-connected*) if it is simple and non-separable and satisfies the following condition; if G is the union of two proper subgraphs H and K such that $H \cap K$ consists solely of two vertices u and v, then one of H and K is a link-graph (arc-graph) with ends u and v.

It should be noted that the union of two proper subgraphs H and K of G can be the whole of G only if each of H and K includes at least one edge or vertex not belonging to the other. In this paper we are concerned mainly with nodally 3-connected graphs, but a specialization to 3-connected graphs is made in § 12.

In § 3 we discuss conditions for a nodally 3-connected graph to be planar, and in § 5 we discuss conditions for the existence of Kuratowski subgraphs of a given graph. In §§ 6–9 we show how to obtain a convex representation of a nodally 3-connected graph, without Kuratowski subgraphs, by solving a set of linear equations. Some extensions of these results to general graphs, with a proof of Kuratowski's theorem, are given in §§ 10–11. In § 12 we discuss the representation in the plane of a pair of dual graphs, and in § 13 we draw attention to some unsolved problems.

William T. Tutte



[Images from Kyri Pavlou]

Review: Continuous Quasi-Harmonic Maps

- Quasi / $\mathrm{A}(\mathrm{x})$-harmonic: point x placed at neighbors' weighted average

$$\nabla \cdot [\mathrm{A}(\mathrm{x}) \nabla u(\mathrm{x})] = 0, \quad \forall \mathrm{x} \in \Omega \backslash \partial\Omega$$
$$\nabla \cdot [\mathrm{A}(\mathrm{x}) \nabla v(\mathrm{x})] = 0, \quad \forall \mathrm{x} \in \Omega \backslash \partial\Omega$$

- Dirichlet boundary condition: fix the boundary

$$u(\mathrm{x}) = b_u(\mathrm{x}), \quad \forall \mathrm{x} \in \partial\Omega$$
$$v(\mathrm{x}) = b_v(\mathrm{x}), \quad \forall \mathrm{x} \in \partial\Omega$$

- Map (u, v) diffeomorphic/injective for convex boundary
  - $\mathrm{A}(\mathrm{x}) = \mathrm{I}$: by *RKC theorem* in complex analysis
  - $\mathrm{A}(\mathrm{x}) \neq \mathrm{I}$: generalization by [Alessandrini and Nesi 2001]

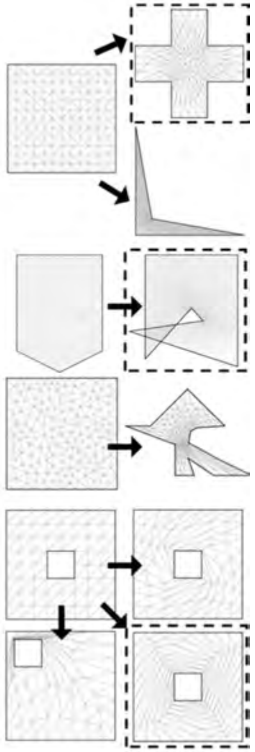
Tibor Radó

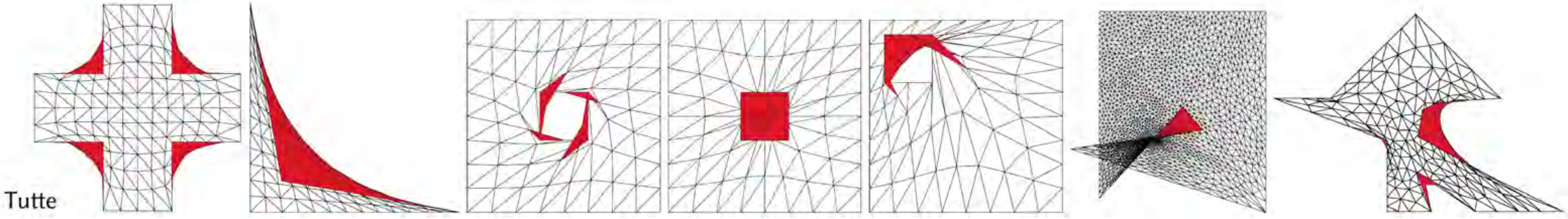
Theorem [Radó–Kneser–Choquet (RKC)]

Harmonic maps onto convex regions are diffeomorphic.

Quasi-Harmonic Maps, aka, Tutte Embedding

- Fix boundary, interior nodes placed at neighbors' **weighted** average [Tutte 1962]
- Fail for non-convex boundary: flipped triangles (in red color)---our condition fixes it!
- There is a hope with extra conditions [Gortler, Gotsman, Thurston 2006]





Idea: in $\mathrm{x}_i = \sum_j w_{ij}\, \mathrm{x}_j$, carefully choose weights w_{ij} can remove flips
Method: search within a (differentiable) family of Tutte embeddings

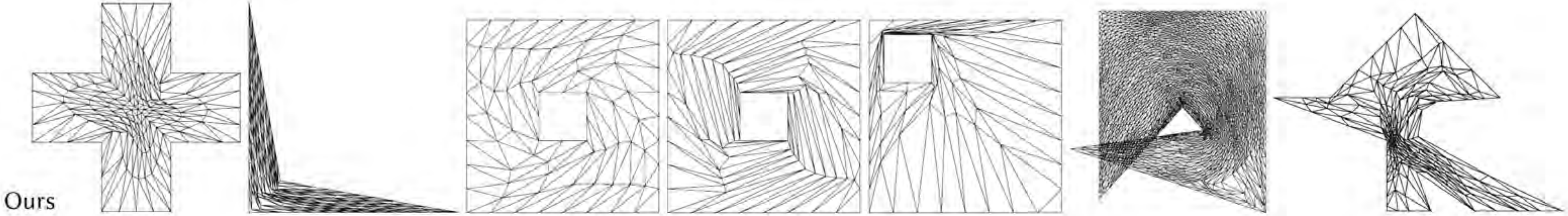


Image from
[Du et al. 2021]

Main Theory: Diffeomorphism = {Quasi-harmonic} + { Dirichlet & Neumann BCs}

Theorem (Main result: continuous version)

$\phi = (u, v)$ is a diffeomorphism if and only if there exist
(1) a positive definite tensor field $\mathrm{A}(\cdot)$ satisfying $\frac{1}{K}\mathrm{I} \preceq \mathrm{A}(\mathrm{x}) \preceq K\mathrm{I}$, (+ smooth conditions)
(2) a positive function $s : \partial\Omega \to \mathbb{R}$ that $s(\mathrm{x}) \geq S$, for some $K, S > 0$,
such that ϕ is $\mathrm{A}(\mathrm{x})$-harmonic with a special Cauchy boundary condition, i.e.:

$$\nabla \cdot [\mathrm{A}(\mathrm{x})\nabla u(\mathrm{x})] = 0 \qquad \forall \mathrm{x} \in \Omega \backslash \partial\Omega$$

$$\nabla \cdot [\mathrm{A}(\mathrm{x})\nabla v(\mathrm{x})] = 0 \qquad \forall \mathrm{x} \in \Omega \backslash \partial\Omega$$

$$u(\mathrm{x}) = b_u(\mathrm{x}) \qquad \forall \mathrm{x} \in \partial\Omega$$

$$v(\mathrm{x}) = b_v(\mathrm{x}) \qquad \forall \mathrm{x} \in \partial\Omega$$

$$\mathrm{n}(\mathrm{x})^\intercal \left[\mathrm{A}(\mathrm{x})\nabla u(\mathrm{x}) \quad \mathrm{A}(\mathrm{x})\nabla v(\mathrm{x})\right] = s(\mathrm{x})\hat{\mathrm{n}}(\mathrm{x})^\intercal \qquad \forall \mathrm{x} \in \partial\Omega$$

Main Theory: *Diffeomorphism = {Quasi-harmonic} + { Dirichlet & Neumann BCs}*

- **Main Theorem**: map (u, v) is diffeomorphic *if-and-only-if* such an $\mathrm{A}(\mathrm{x})$ exists:

Same as Tutte

- (1) Quasi-harmonic:

$$\nabla \cdot [\mathrm{A}(\mathrm{x})\nabla u(\mathrm{x})] = 0, \quad \forall \mathrm{x} \in \Omega \backslash \partial\Omega$$
$$\nabla \cdot [\mathrm{A}(\mathrm{x})\nabla v(\mathrm{x})] = 0, \quad \forall \mathrm{x} \in \Omega \backslash \partial\Omega$$

- (2) Dirichlet boundary condition: specify the boundary positions (trivial).

$$u(\mathrm{x}) = b_u(\mathrm{x}), \quad \forall \mathrm{x} \in \partial\Omega$$
$$v(\mathrm{x}) = b_v(\mathrm{x}), \quad \forall \mathrm{x} \in \partial\Omega$$

Ours

- (3) Neumann boundary condition: specify the $\mathrm{A}(\mathrm{x})$-weighted normal derivative.

$$\mathrm{n}(\mathrm{x})^{\intercal}[\mathrm{A}(\mathrm{x})\nabla u(\mathrm{x}) \quad \mathrm{A}(\mathrm{x})\nabla v(\mathrm{x})] = \hat{\mathrm{n}}(\mathrm{x})^{\intercal} \quad \forall \mathrm{x} \in \partial\Omega$$

  - $\mathbf{n}(\mathrm{x})$: normal on source domain
  - $\hat{\mathbf{n}}(\mathrm{x})$: normal on target domain

A Condition *Feasible* Computationally

Neumann boundary condition: specify the $A(x)$-weighted normal derivative

$$n(x)^{\mathsf{T}}[A(x)\nabla u(x) \quad A(x)\nabla v(x)] = \hat{n}(x)^{\mathsf{T}} \qquad \forall x \in \partial\Omega$$

- $\mathbf{n}(x)$: normal on source domain
- $\hat{\mathbf{n}}(x)$: normal on target domain

Ours: correct & *feasible* computationally

the *nature* boundary condition: the best thing you can hope for!

made possible *discrete* injectivity

$$n(x)^{\mathsf{T}}[\nabla u(x) \quad \nabla v(x)] = \hat{n}(x)^{\mathsf{T}} \qquad \forall x \in \partial\Omega$$

A possible variant:

theoretically correct

computationally *inconsistent* (with the PDE)

Our Starting Point: {Diffeomorphic Mapping} = {Inverse PDE}

$$\min_{u,v,\mathrm{A}} R(u,v,\mathrm{A})$$

$$\text{s.t.} \begin{cases} \nabla \cdot [\mathrm{A}(\mathrm{x})\nabla u(\mathrm{x})] = 0, & \forall \mathrm{x} \in \Omega \backslash \partial\Omega \\ \nabla \cdot [\mathrm{A}(\mathrm{x})\nabla v(\mathrm{x})] = 0, & \forall \mathrm{x} \in \Omega \backslash \partial\Omega \\ u(\mathrm{x}) = b_u(\mathrm{x}), & \forall \mathrm{x} \in \partial\Omega \\ v(\mathrm{x}) = b_v(\mathrm{x}), & \forall \mathrm{x} \in \partial\Omega \\ \mathrm{n}(\mathrm{x})^\intercal[\mathrm{A}(\mathrm{x})\nabla u(\mathrm{x})] = g_u(\mathrm{x}), & \forall \mathrm{x} \in \partial\Omega \\ \mathrm{n}(\mathrm{x})^\intercal[\mathrm{A}(\mathrm{x})\nabla v(\mathrm{x})] = g_v(\mathrm{x}), & \forall \mathrm{x} \in \partial\Omega \end{cases}$$

⬅

$$\min_{u,v} E(u,v)$$

$$\text{s.t.} \quad \det\left[\nabla u(\mathrm{x})\ \nabla v(\mathrm{x})\right] > 0$$

(barrier methods,
interior point methods)

Constraints: copy-paste previous conditions

Objective R: is some regularizer or energy $E(u,v)$

PDE-constrained optimization

complicated but *more efficiently* solvable with our method!

many unsuccessful attempts (augmented Lagrangian etc., too slow)

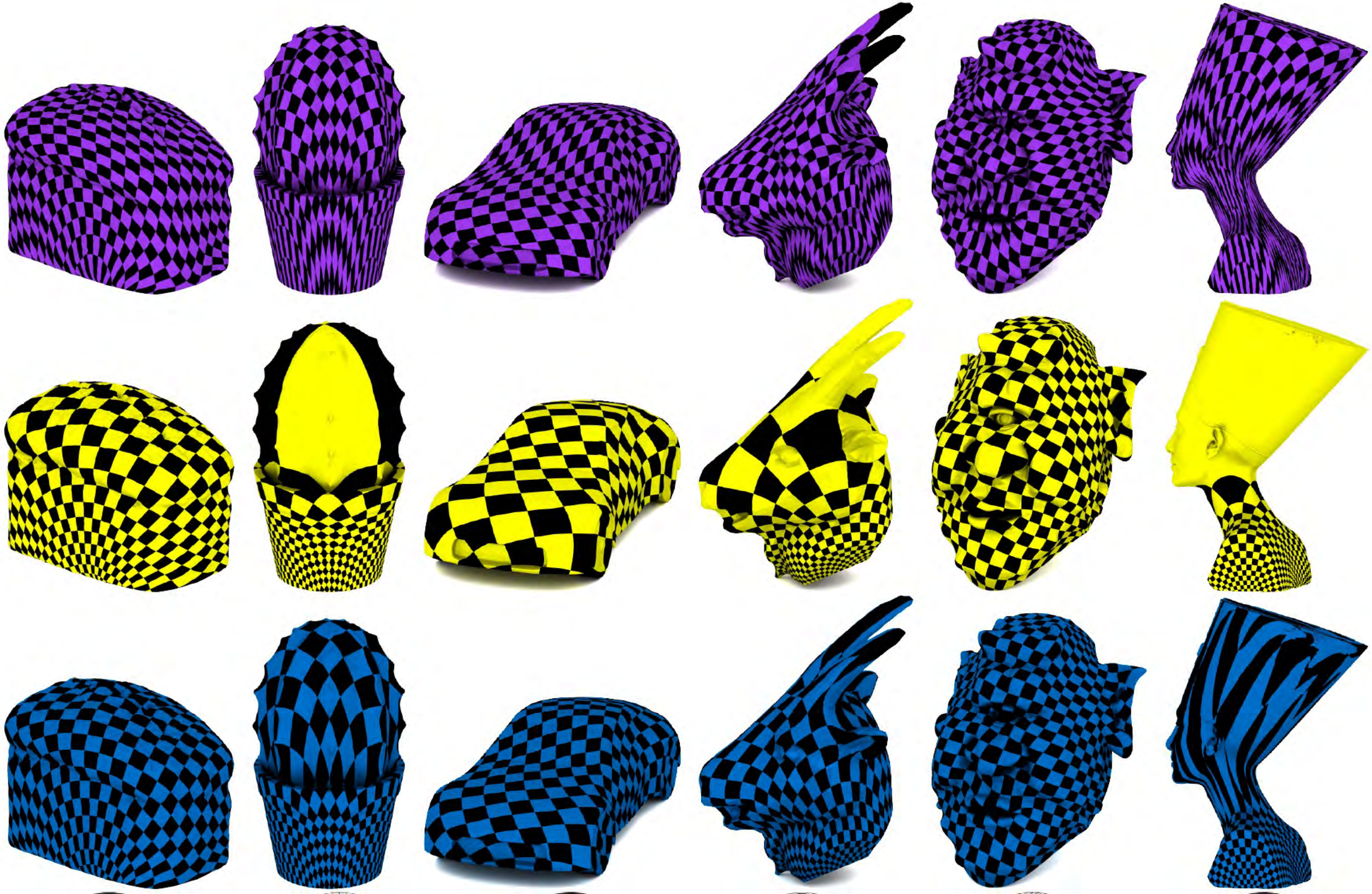
Application: Bijective Parameterizations Optimizing Different Energies

- By choosing different regularizer R.

area preserving

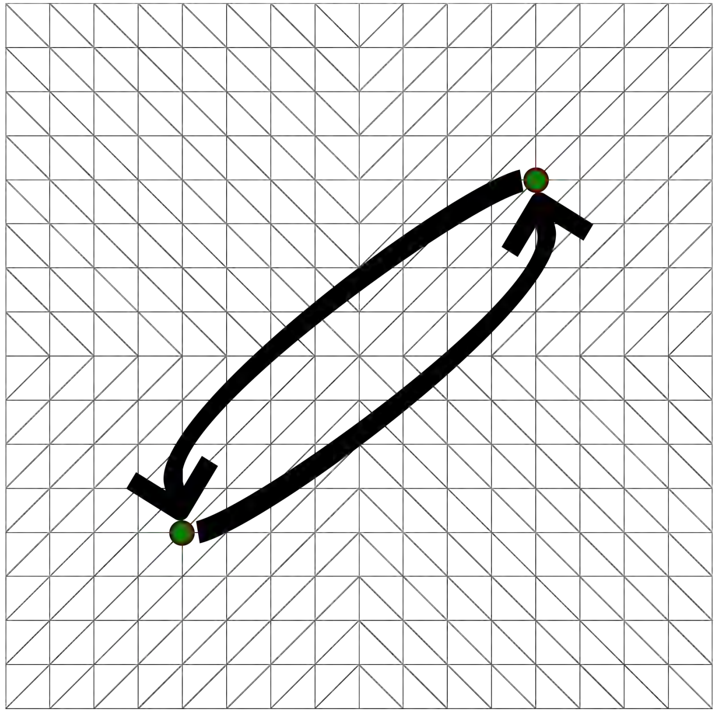
none:
$R = 0$

as-rigid-as-possible

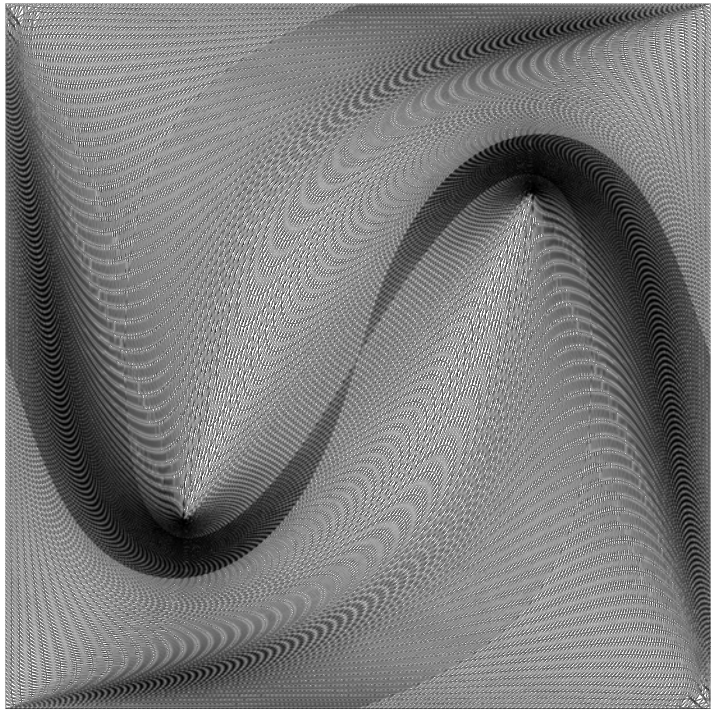


Swap Two Point Landmarks Using Our Method

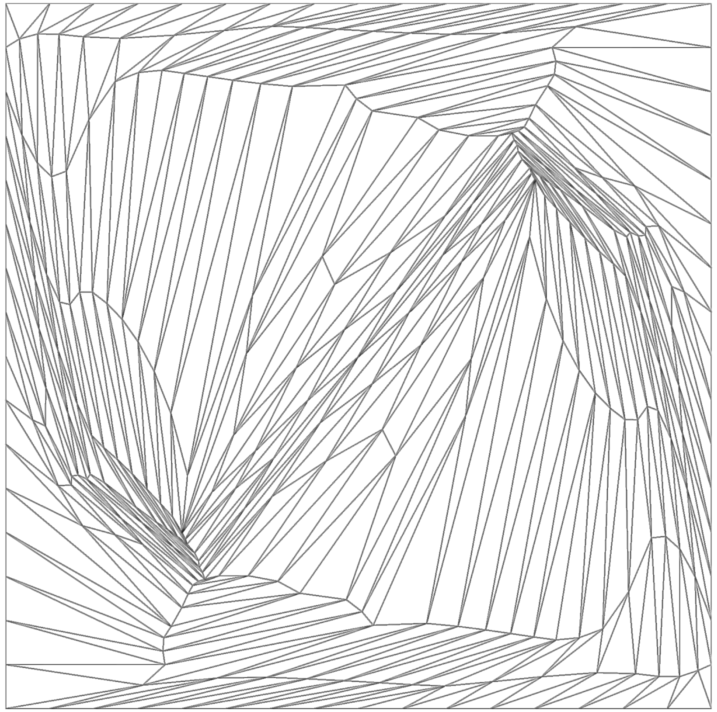
- The discrete solution can be quite different from the continuous one
- Require careful consideration from discrete geometry



swap two point landmarks



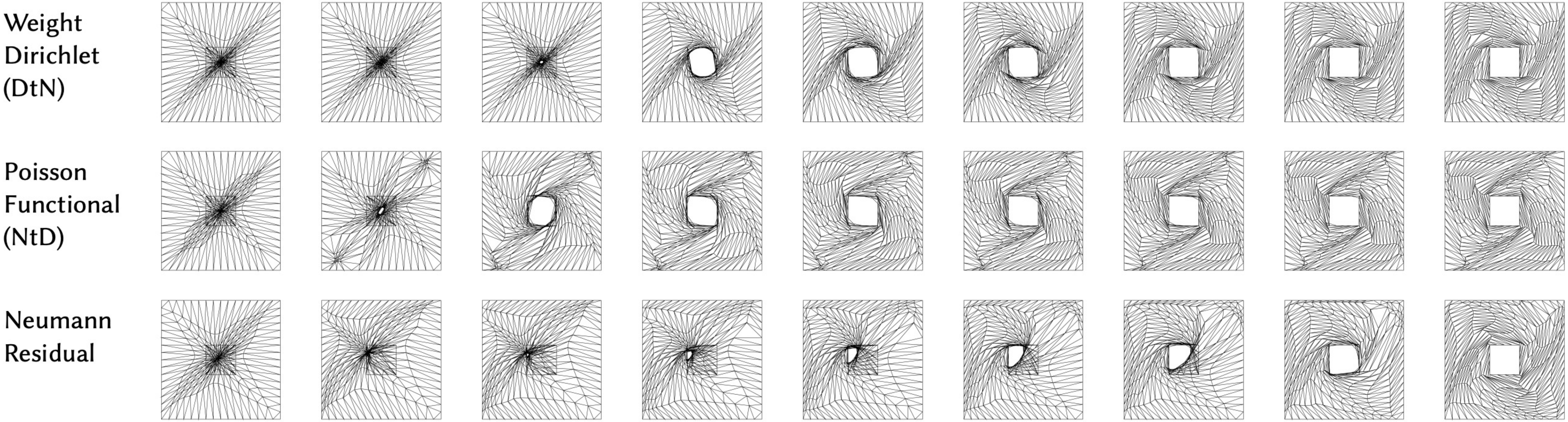
with an up-sampled mesh



our setting: no up-sampling

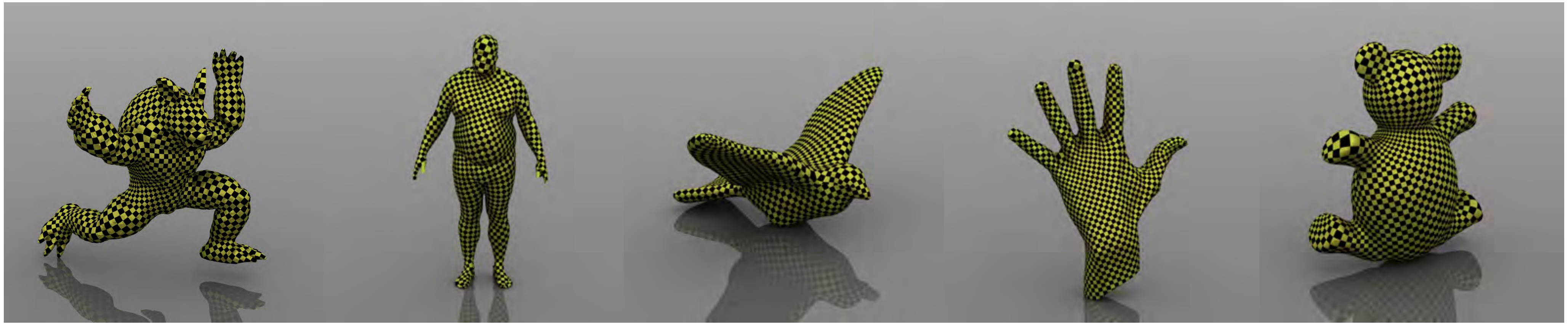
Our Framework Leads to a Family of Methods

- Different functionals in our framework provide variant means solving the problem

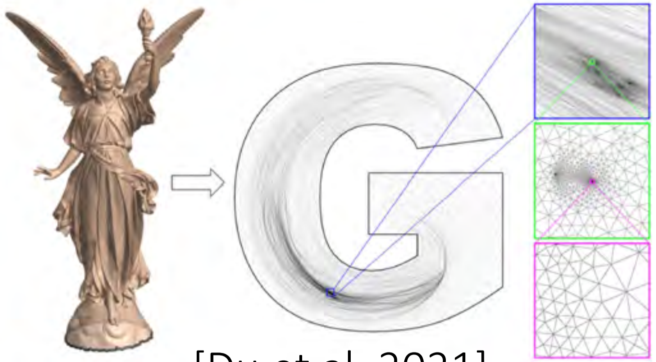


Experiments

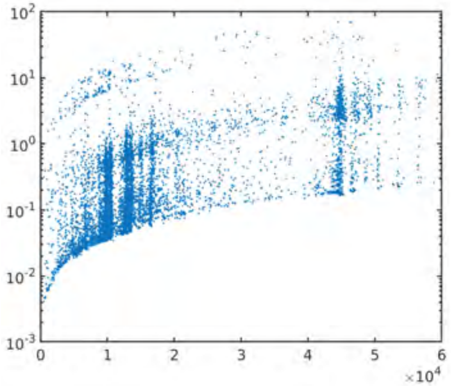
- Our method is extremely robust and fast
- Pass a challenge with 11k tests [Du et al. 2021]; up to 1000 faster



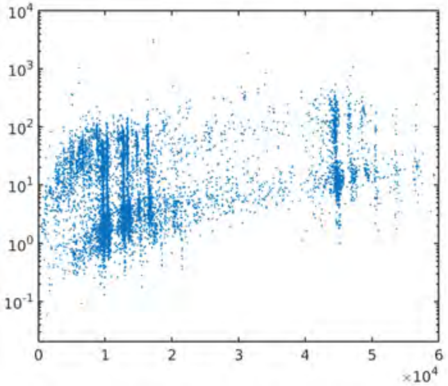
diffeomorphisms by our method, shapes already cut into disk topology



[Du et al. 2021]

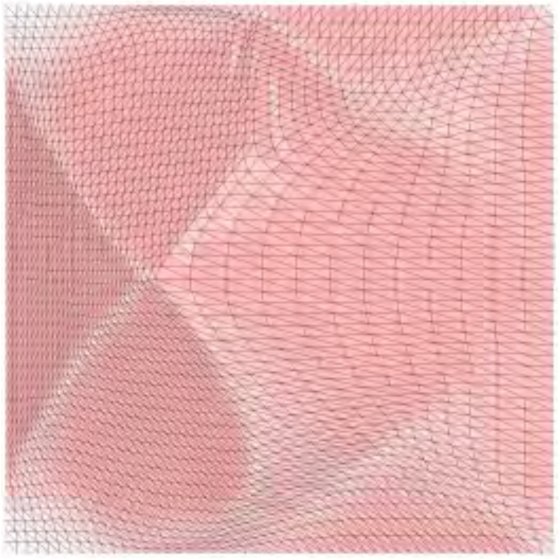


(a) Our method



(b) [Garanzha et al. 2021]

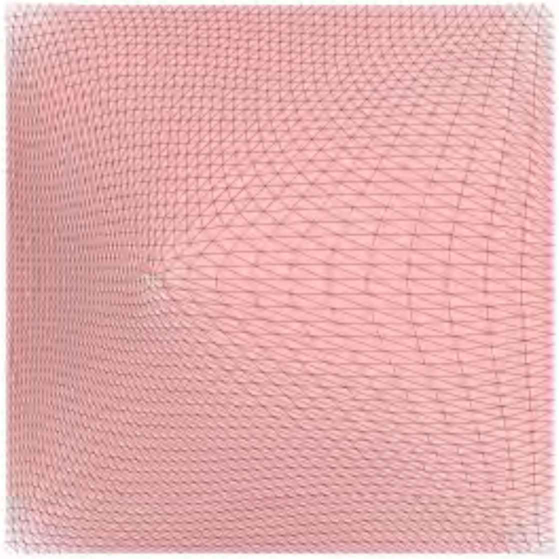
Inverse PDEs v.s. Geometric Optimization



[Lipman 2012].
E_{MIPS}=1.03.
max=3.00, mean=2.63.

aliasing patterns:

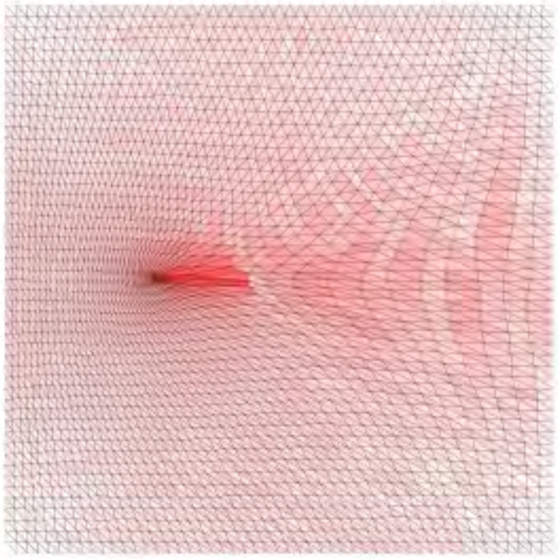
triangulation-sensitive



Ours (si-log).
E_{MIPS}=1.01.
max=3.11, mean=2.63.

smoother

50X faster



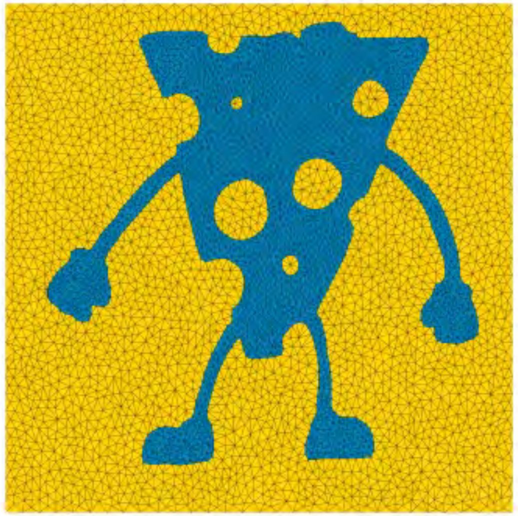
Ours (tanh+MIPS).
E_{MIPS}=0.55.
max=27.83, mean=2.00.

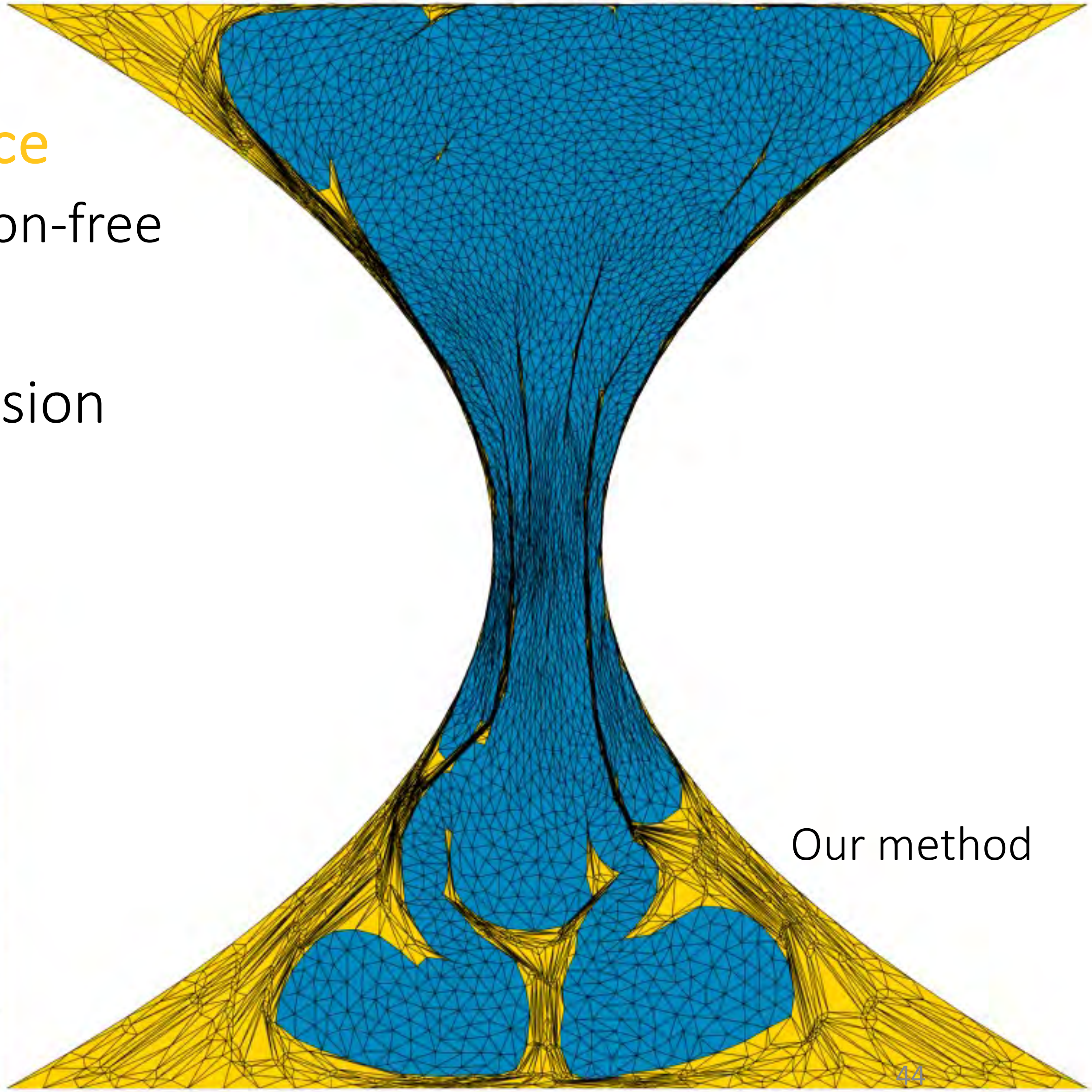
and support many objectives

Application: Collision Avoidance by way of Injectivity

- Task: Put the Cheeseman in an hour-glass
- Ours: warp the shape + surrounding space
  - ensuring shape-mesh + air-mesh inversion-free

    avoids penetration / self-intersection
- Result: a unified engine for physics & collision



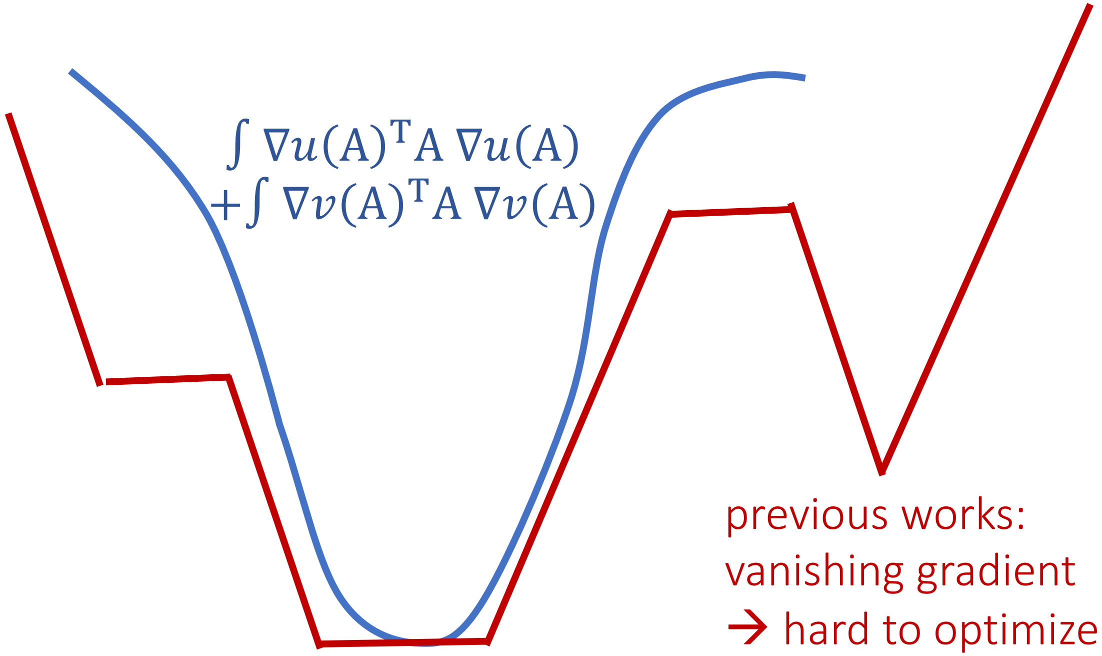


Why Inverse PDE is Better? Evidences from Inverse Problems of PDEs

Problem: $\min_{A,u,v} E(A,u,v)$

Solvers: operating in the space of:

- prior work in (u,v):
  $\min_{u,v} \{ \min_{A} E(A,u,v) \}$
  non-smooth & grad vanish
- ours in (A)
  $\min_{A} \{ \min_{u,v} E(A,u,v) \}$
  a *tight* upper bound
  smooth & C^∞ differentiable



A Geometry Approach for Topological Constraints

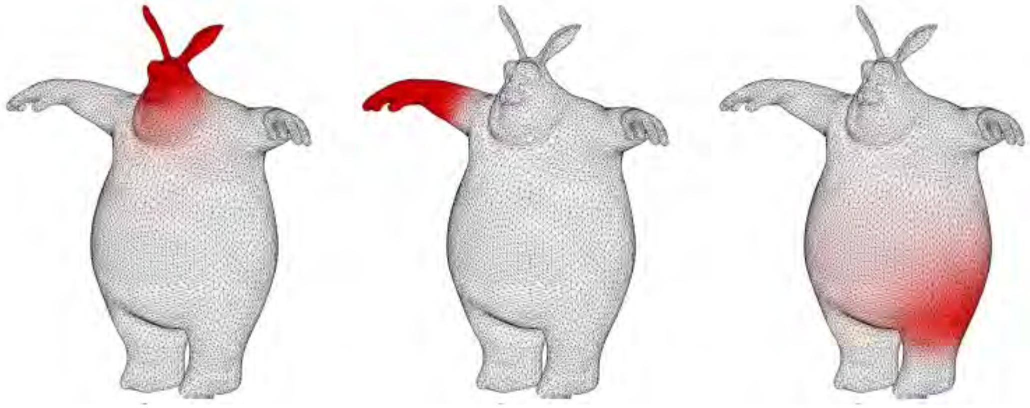
"Fast Quasi-Harmonic Weights for Geometric Data Interpolation."
Yu Wang and Justin Solomon.
ACM Transactions on Graphics (TOG) 40(4). ACM SIGGRAPH 2021.

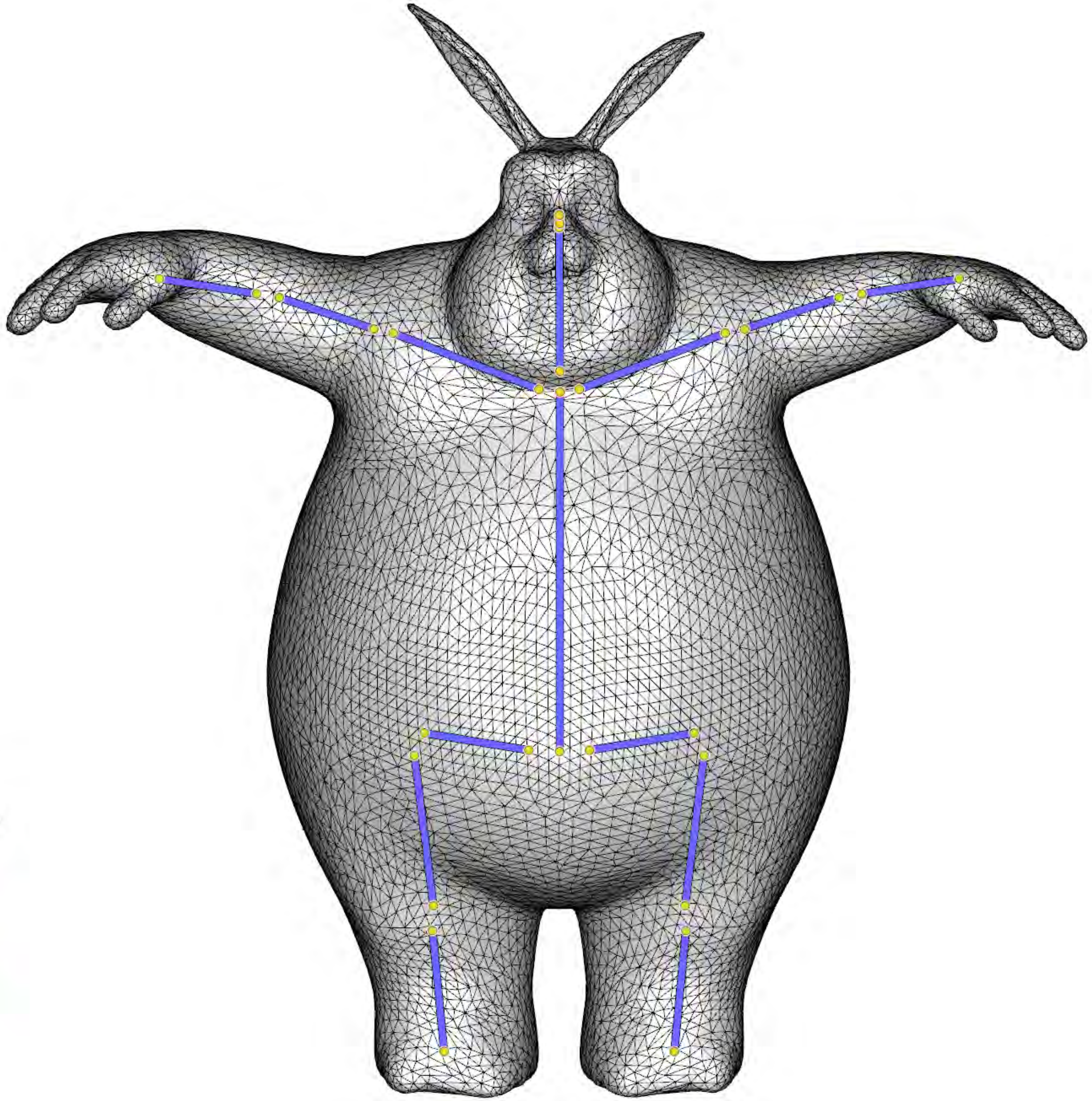
Animation with *Sparse Control*

- A tool for artist to direct animation with *sparse control*

Skinning Animation

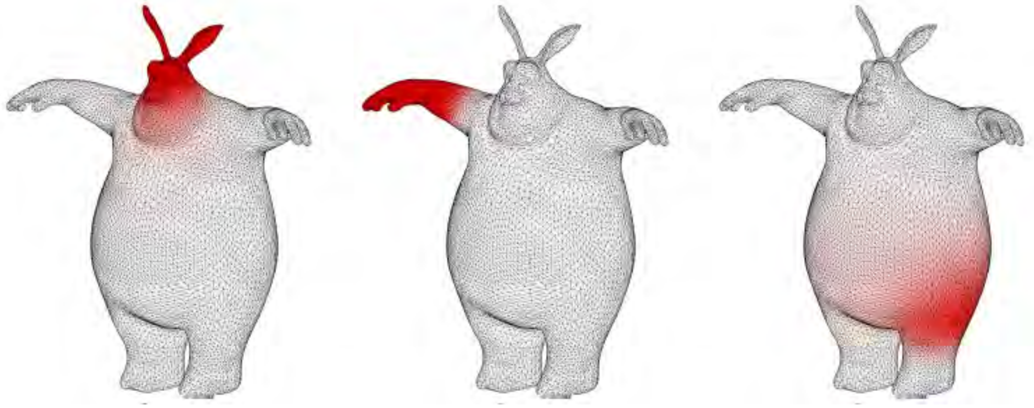
- Skinning: drive deformation by propagating transformations at skeletons to all vertices
- Fast
- Common in computer games

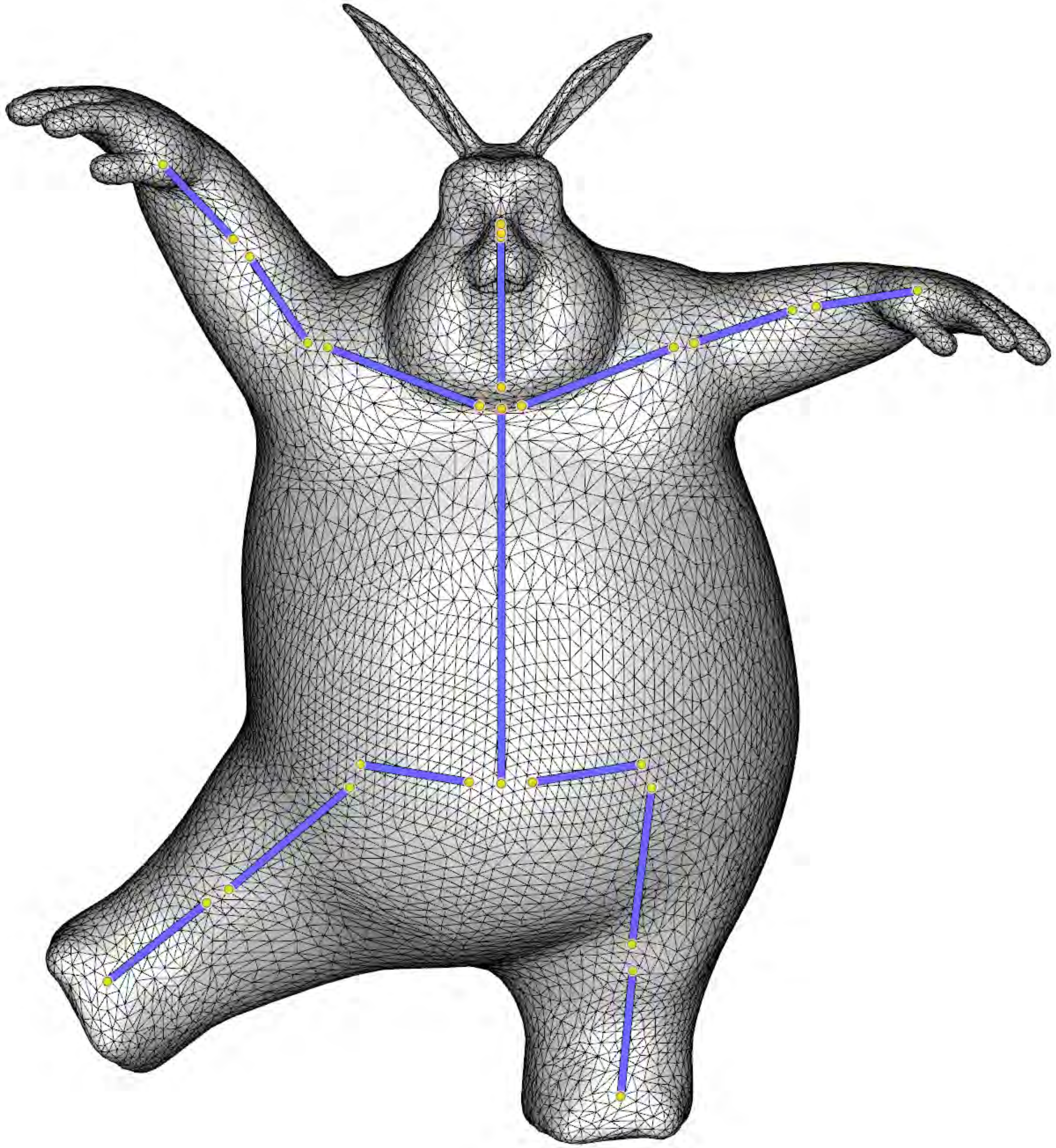




Skinning Animation

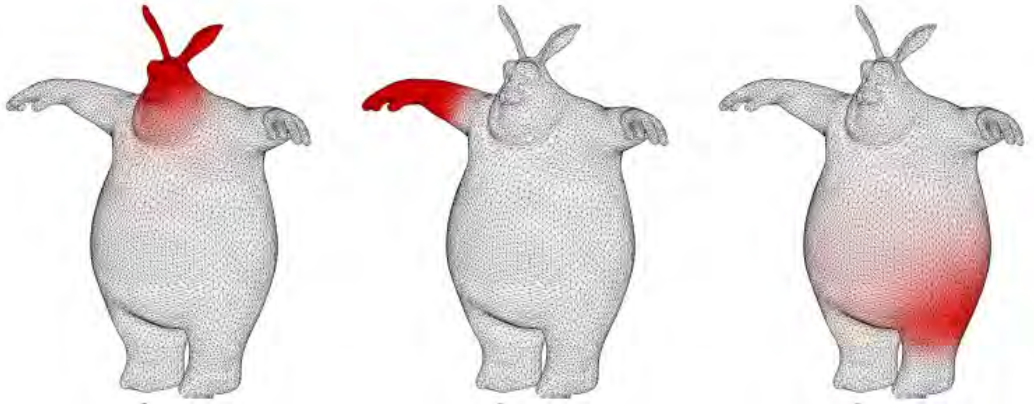
- Skinning: drive deformation by propagating transformations at skeletons to all vertices
- Fast
- Common in computer games

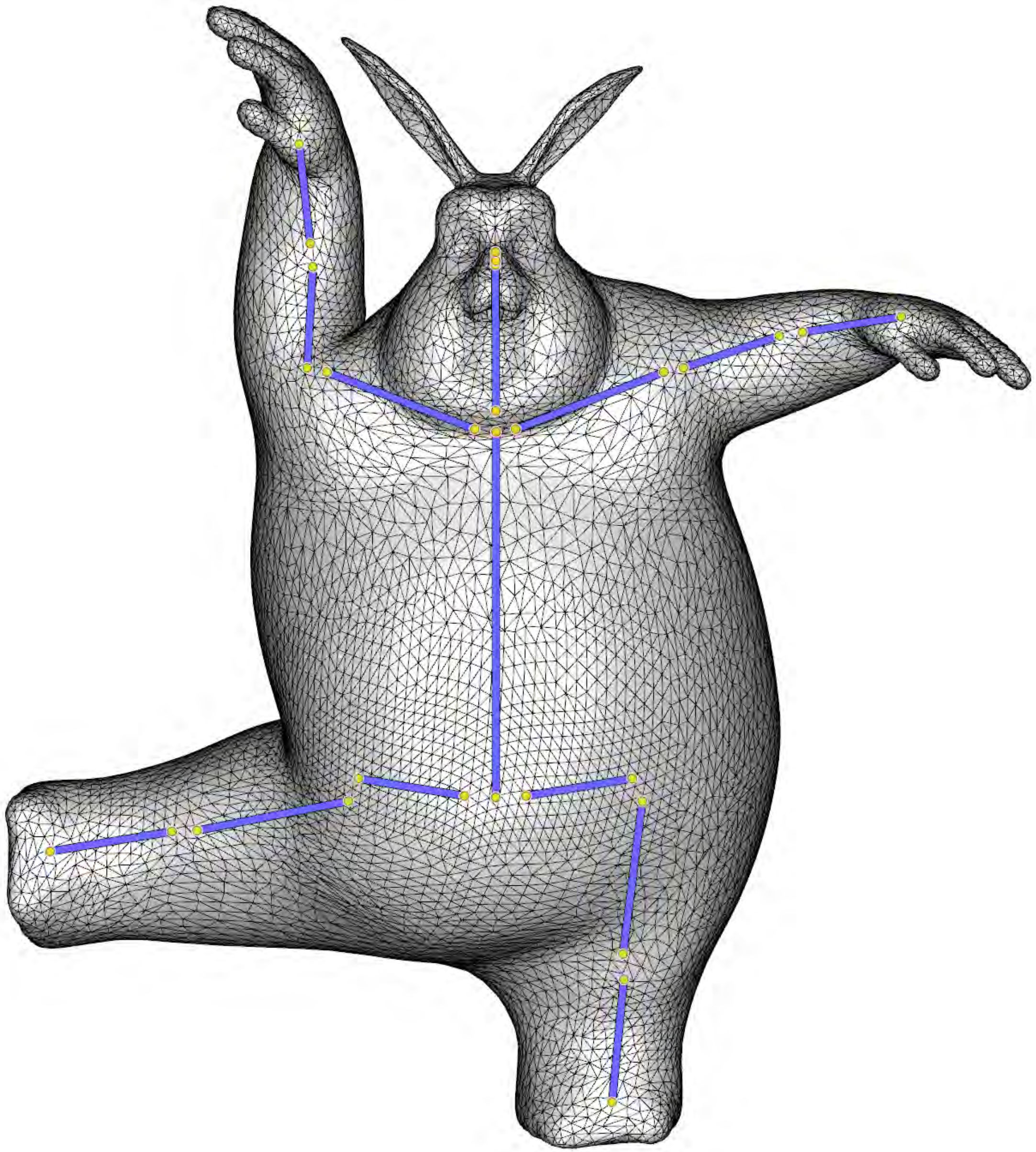




Skinning Animation

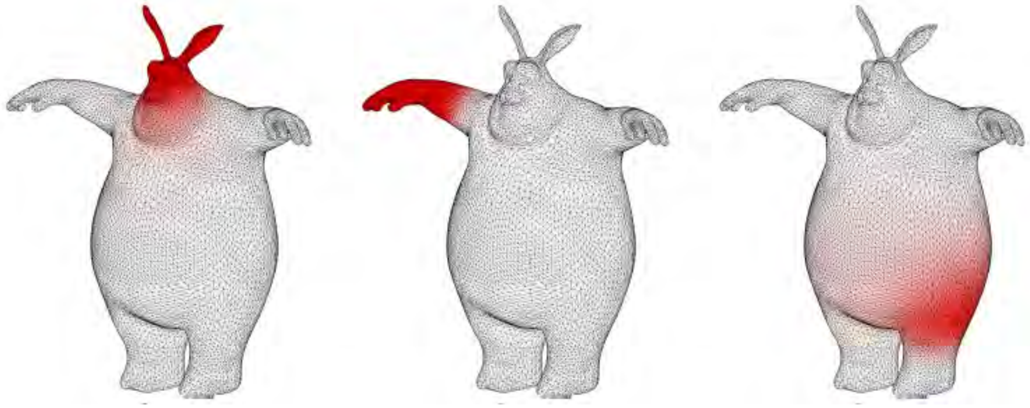
- Skinning: drive deformation by propagating transformations at skeletons to all vertices
- Fast
- Common in computer games

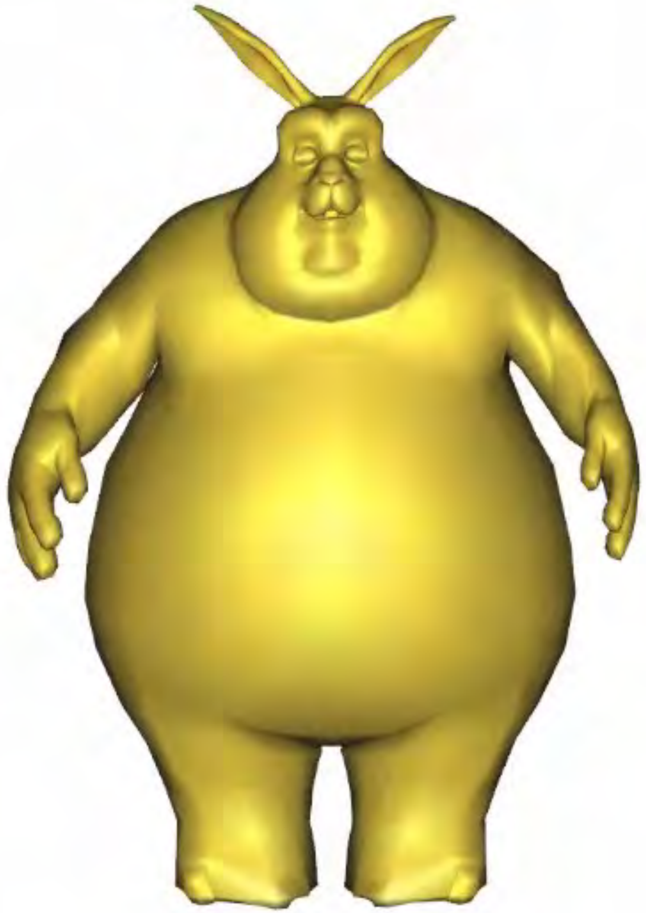




Skinning Animation

- Skinning: drive deformation by propagating transformations at skeletons to all vertices
- Fast
- Common in computer games



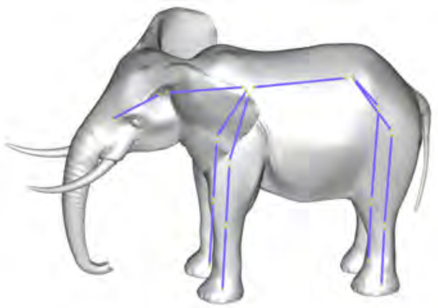


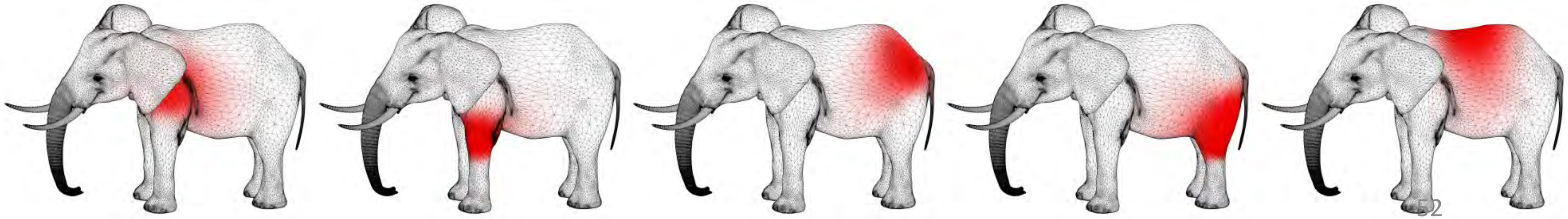
Problem: Skinning Weight Computing

- Skinning/interpolation weights: a *partition-of-unity* that decay from 1 to 0
- $w_j(\cdot)$ is a fundamental geometry quantity. E.g., $\nabla w_j(\cdot)$ defines the *foliation*.

Mathematically:

$$
\begin{aligned}
&\text{(Lagrange)} && \forall \mathbf{c}_i : w_j(\mathbf{c}_i) = \delta_{ij} && i, j = 1, \ldots, m \\
&\text{(Partition of unity)} && \forall \mathbf{x} : \sum_{j=1}^{m} w_j(\mathbf{x}) = 1 && \\
&\text{(Nonnegativity)} && \forall \mathbf{x} : w_j(\mathbf{x}) \geq 0 && j = 1, \ldots, m \\
&\text{(No local extrema)} && \forall \mathbf{x} : w_j(\mathbf{x}) \text{ not a local extremum} && j = 1, \ldots, m
\end{aligned}
$$





$w_1(\mathrm{x})$ $w_2(\mathrm{x})$ $w_3(\mathrm{x})$ $w_4(\mathrm{x})$ $w_5(\mathrm{x})$

(Monotonic) Bounded Biharmonic Weights [Jacobson et al. 2011, 2012]

A generalized B-spline, for a non-Euclidean domain Ω.

- used extensively beyond animation
- take hours on large example

$$\min_w \quad \sum_{j=1}^{m} \int_\Omega \|\Delta w_j(\mathbf{x})\|^2$$

Objective: Weight Smoothness

$s.t.$

$$\text{(Lagrange)} \quad \forall \mathbf{c}_i : w_j(\mathbf{c}_i) = \delta_{ij} \qquad i, j = 1, \ldots, m$$

$$\text{(Partition of unity)} \quad \forall \mathbf{x} : \sum_{j=1}^{m} w_j(\mathbf{x}) = 1$$

Constraint: Desired Properties

$$\text{(Nonnegativity)} \quad \forall \mathbf{x} : w_j(\mathbf{x}) \geq 0 \qquad j = 1, \ldots, m$$

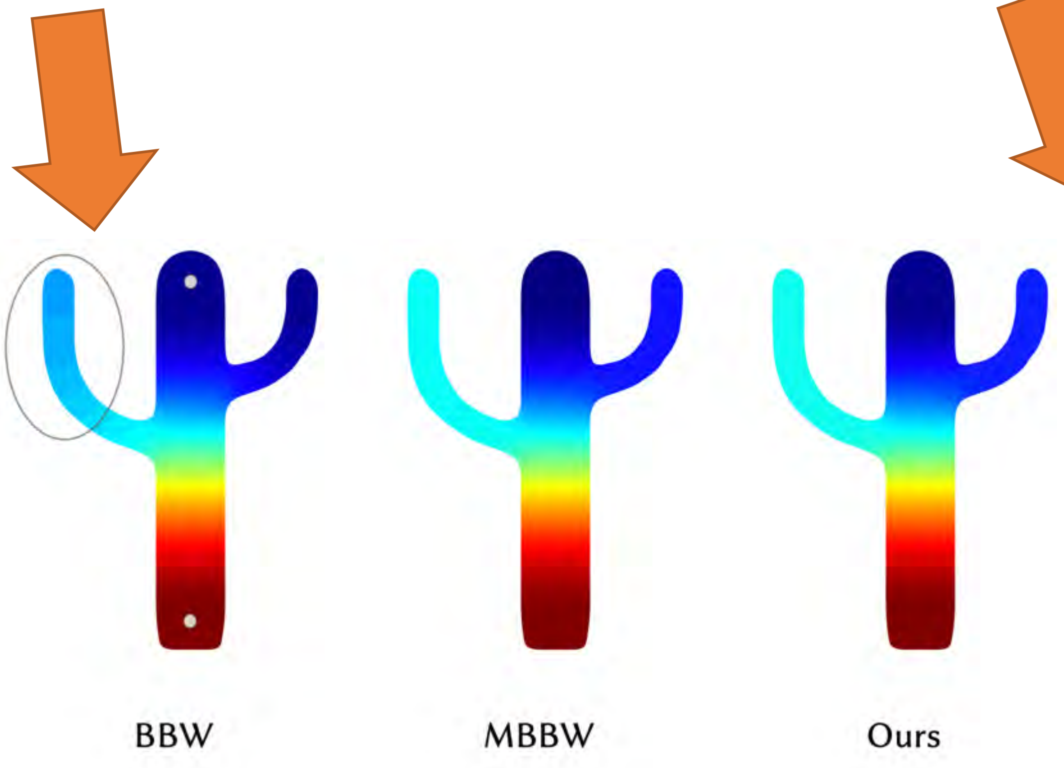
$$\text{(No local extrema)} \quad \forall \mathbf{x} : w_j(\mathbf{x}) \text{ not a local extremum} \qquad j = 1, \ldots, m$$

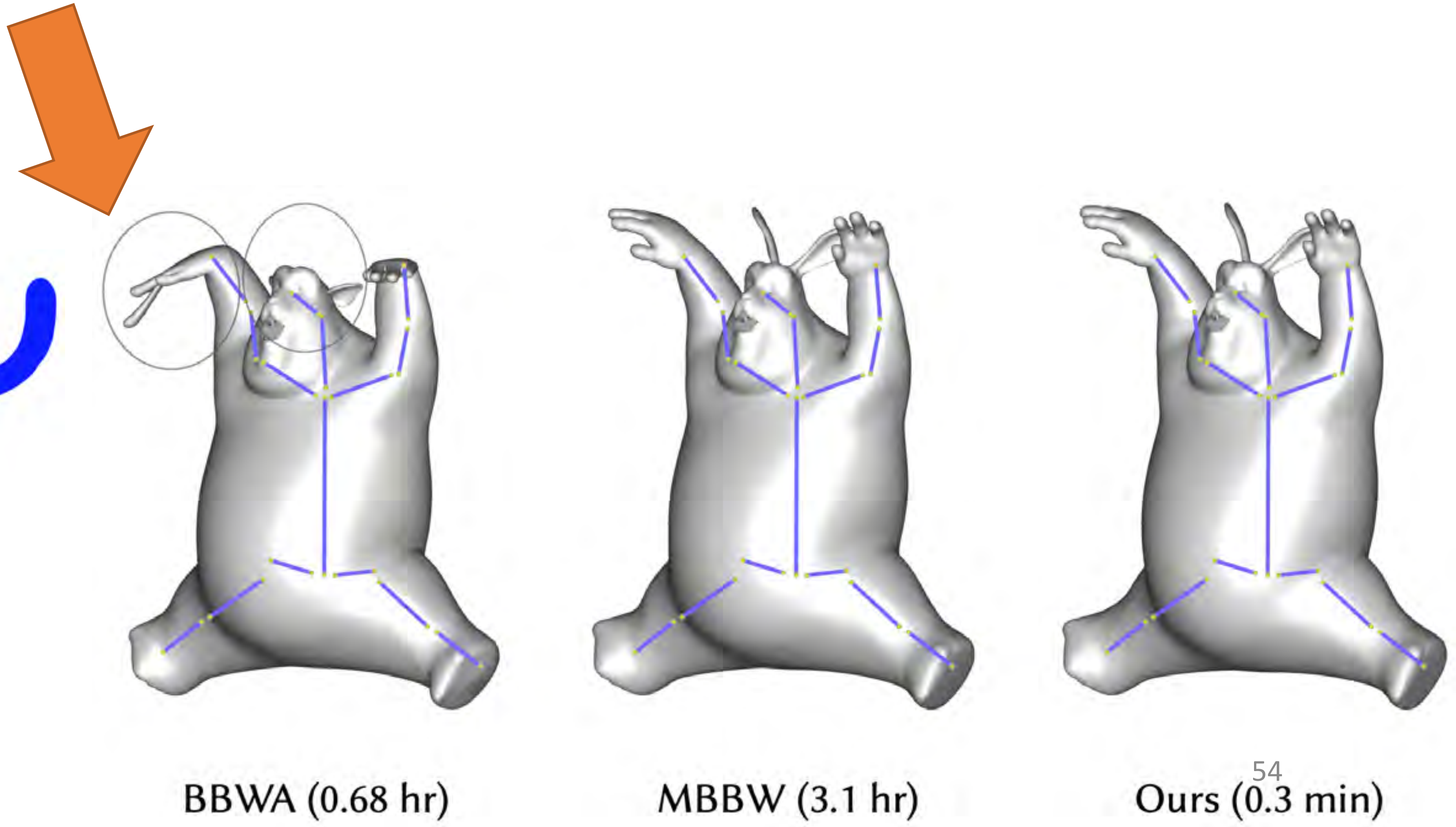
BBW or monotonic BBW (MBBW)

[Jacobson et al. 2011, 2012]

Monotonicity Constraint

- Monotonicity: no local extremum away from control handles
  - A *topological* constraint [Jacobson et al. 2012]
  - A necessary condition for *diffeomorphic* shape morphing
- Without monotonicity → noticeable artifacts





Our Model: Quasi-Harmonic Weights

- Consider solutions to the anisotropic Laplace equations

$\Delta^{\mathrm{A}} = \nabla \cdot [\mathrm{A}(\mathrm{x})\nabla]$

$$
\begin{aligned}
\Delta^{\mathbf{A}} w_j(\mathbf{x}) &= 0 && \forall j = 1, \ldots, m \\
w_j(\mathbf{c}_i) &= \delta_{ij} && \forall i, j = 1, \ldots, m \\
\nabla^{\mathbf{A}}_{\mathbf{n}} w_j(\mathbf{x}) &= 0 && \forall \mathbf{x} \in \partial\Omega/\mathbf{c}, \forall j = 1, \ldots, m.
\end{aligned}
$$

- For *any* $\mathrm{A}(\mathrm{x})$, the weights $w_j(\mathrm{x})$ satisfy:

$$
\begin{aligned}
&\text{(Lagrange)} && \forall \mathbf{c}_i : w_j(\mathbf{c}_i) = \delta_{ij} && i, j = 1, \ldots, m \\
&\text{(Partition of unity)} && \forall \mathbf{x} : \sum_{j=1}^{m} w_j(\mathbf{x}) = 1 && \\
&\text{(Nonnegativity)} && \forall \mathbf{x} : w_j(\mathbf{x}) \geq 0 && j = 1, \ldots, m \\
&\text{(No local extrema)} && \forall \mathbf{x} : w_j(\mathbf{x}) \text{ not a local extremum} && j = 1, \ldots, m
\end{aligned}
$$

For any $\mathrm{A}(\mathrm{x})$, the generated $w(\mathrm{x})$ automatically satisfy all conditions

Our Method

- Search within the family of quasi-harmonic weights

$$\Delta^{A} = \nabla \cdot [A(x)\nabla]$$

$$\min_w \quad \sum_{j=1}^{m} \int_\Omega \|\Delta w_j(\mathbf{x})\|^2$$

$$\forall \mathbf{c}_i : w_j(\mathbf{c}_i) = \delta_{ij} \qquad i, j = 1, \ldots, m$$

$$\forall \mathbf{x} : \sum_{j=1}^{m} w_j(\mathbf{x}) = 1$$

$$\forall \mathbf{x} : w_j(\mathbf{x}) \geq 0 \qquad j = 1, \ldots, m$$

$$\forall \mathbf{x} : w_j(\mathbf{x}) \text{ not a local extremum} \qquad j = 1, \ldots, m$$

$$\begin{aligned} \min_{\mathbf{A}} \quad & \textstyle\sum_{j=1}^{m} \int_\Omega \|\Delta w_j(\mathbf{x})\|^2 & \\ \text{s.t.} \quad & w_j(\mathbf{c}_i) = \delta_{ij} & \forall i, j = 1, \ldots, m \\ & \Delta^{\mathbf{A}} w_j(\mathbf{x}) = 0 & \forall \mathbf{x} \in \Omega, \forall j = 1, \ldots, m \\ & \nabla^{\mathbf{A}}_{\mathbf{n}} w_j(\mathbf{x}) = 0 & \forall \mathbf{x} \in \partial\Omega, \forall j = 1, \ldots, m \\ & \mathbf{A}(\mathbf{x}) \succeq 0 & \forall \mathbf{x} \in \Omega \end{aligned}$$

PDE-constrained optimization

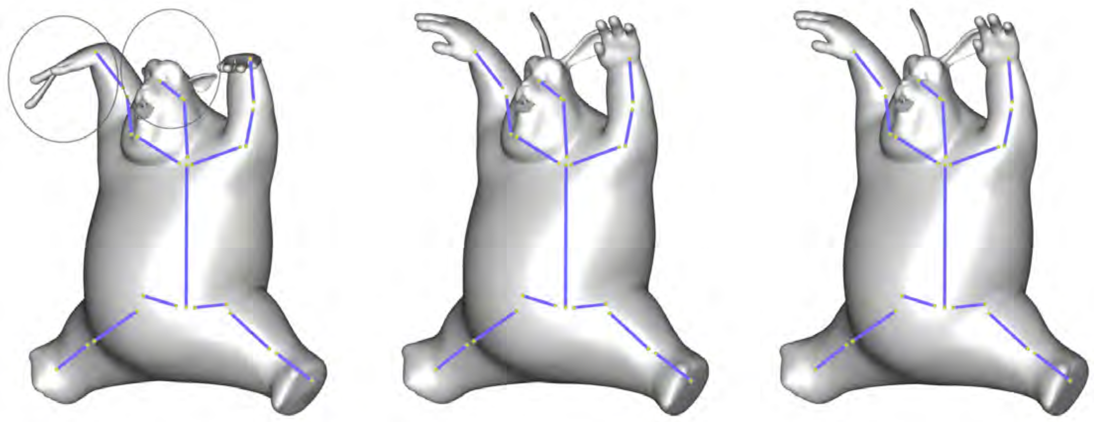
- not necessarily easier
- but we find an efficient solver

Evaluation: Timing

- Our inverse PDE solver: orders-of-magnitude faster than previous methods

Example			Smoothness Energy				Time (Sec.)			
Mesh	#Hdl.	#Ele.	BBWA	MBBW	Ours, $k=20$	$k=10$	BBWA	MBBW	Ours ($k=10$)	num. fact.
Beast	15	443442	1	1.001	0.997	1.016	707.8	840.6	12.6	0.16
Bunny	14	531392	1	0.997	1.001	1.026	2430.6	11111.3	18.0	0.41
Raptor	15	367966	1	1.002	0.992	1.051	640.1	720.3	14.9	0.11
Elephant	17	516858	1	0.996	0.987	1.008	1092.3	2379.5	14.4	0.16
Dragon	17	1187670	1	*	0.992	1.023	5022.1	*	54.8	0.60
Image	27	6952	1	0.995	0.984	1.005	7.78	17.70	0.18	0.0009
Brick	2	78529	1	1.000	0.981	1.084	17.8	53.7	1.21	0.03
Tibiman	16	84125	1	0.991	0.986	0.995	91.9	274.1	1.80	0.025

BBW [Jacobson et al. 2011]
MBBW [Jacobson et al. 2012]



BBWA (0.68 hr) MBBW (3.1 hr) Ours (0.3 min)

Geometric Computing beyond the Laplacian

- Optimization in a larger space of Laplacians.
- Design Laplacian-like operators.
- Learn Laplacian-like operators from data.

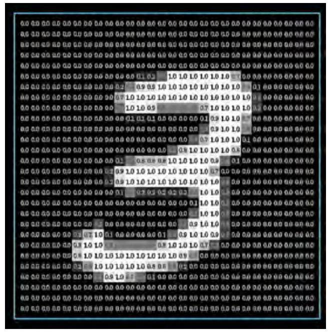
Shape Classification backed by Modern Geometric Analysis

"Steklov Spectral Geometry for Extrinsic Shape Analysis"

Yu Wang, Mirela Ben-Chen, Iosif Polterovich, Justin Solomon

ACM Transactions on Graphics 38(1)

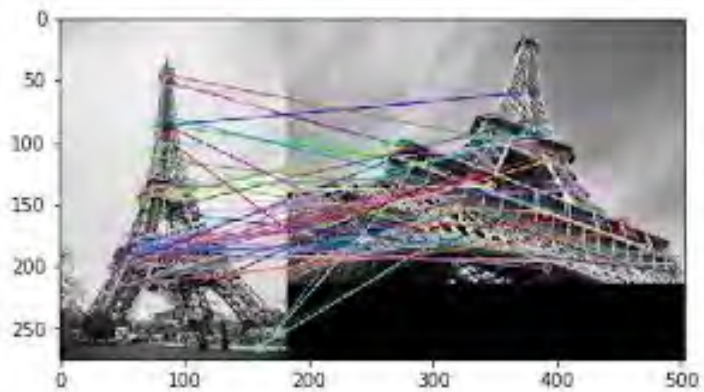
- Image processing & analysis
  - input: 2D array
- Image classification

$f($  $) = c_1$

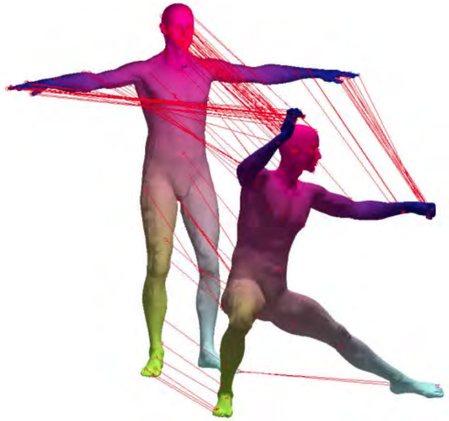
- Geometry processing & analysis
  - input: 2-dim manifold in 3D
- Shape classification

$f($  $) = c_2$

- Local features: SIFT etc.



- Local features: curvatures etc.



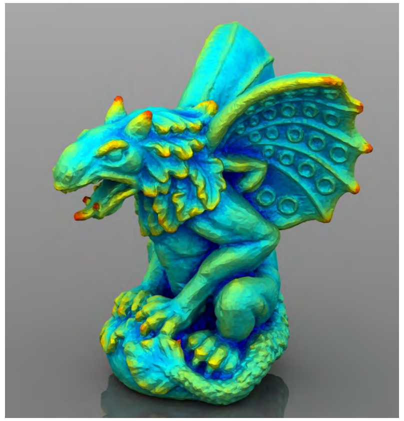
[Lombaert et al. 2013]

Geometric Computing Tasks

- Task: shape analysis and 3D vision
  - shape classification

$f($  $) = c$

  - feature extraction

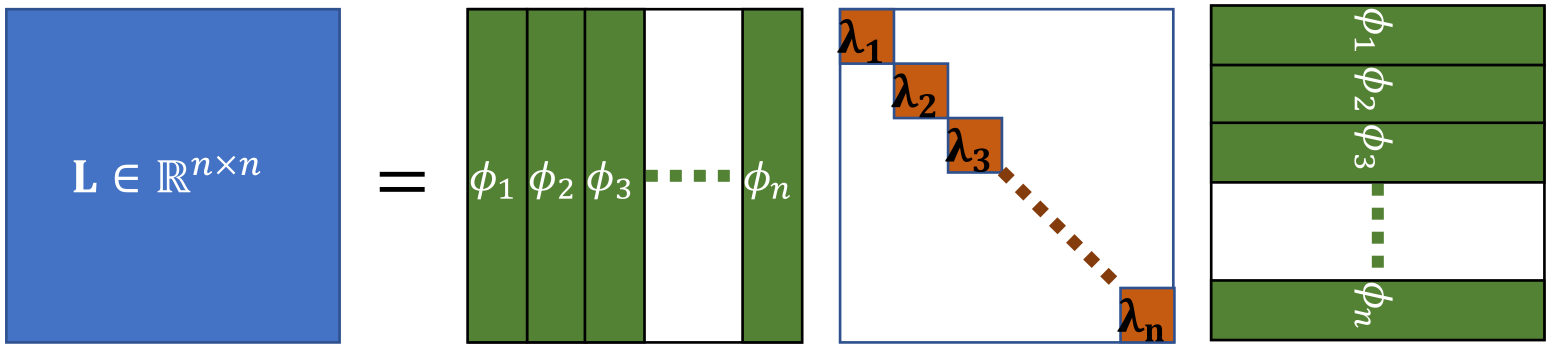
$g($  $) =$ 

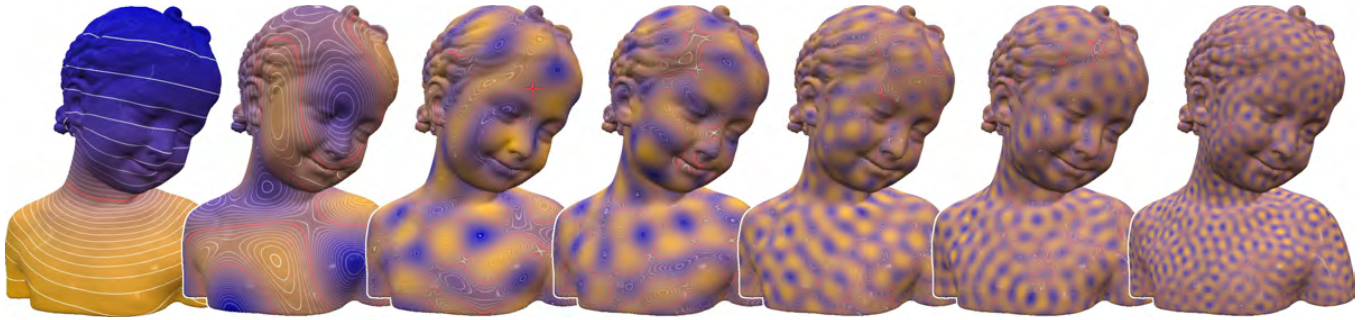
- Approach
  - Theory: insights taken from modern **spectral geometry**
  - Tools: borrowed from **computational electromagnetics**

Mathematically Justified Embedding: Laplacian Eigenvalues

- Solve the **eigenvalue problem**

$$\Delta\phi_i = \lambda\phi_i$$





Laplacian eigenmodes
[Levy 2006]

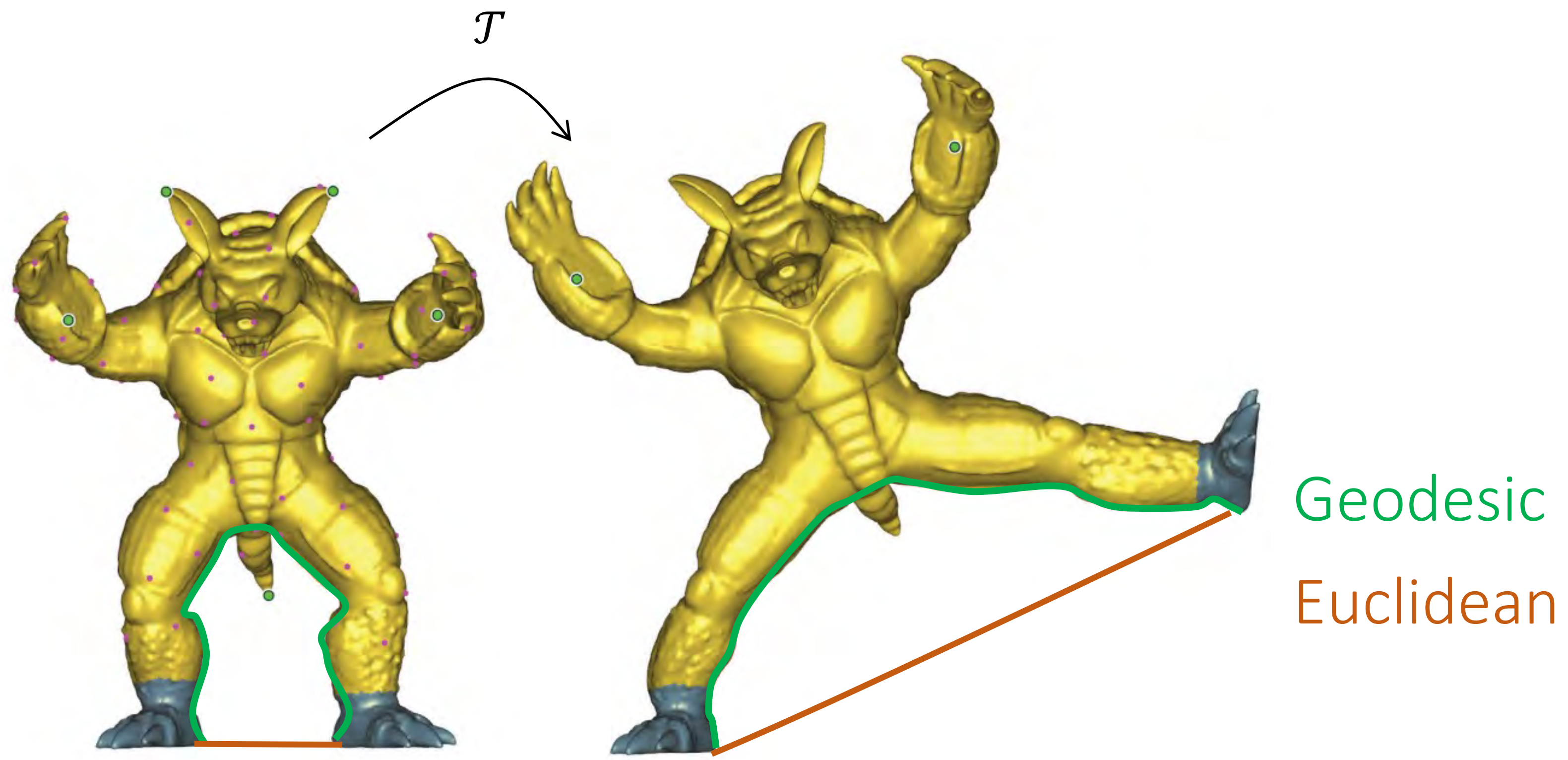
- $(\lambda_1, \lambda_2, \dots, \lambda_k) \in \mathbb{R}^k$: *Shape2Vector* scheme, *mathematically justified*

Even Better: Intrinsic Invariance

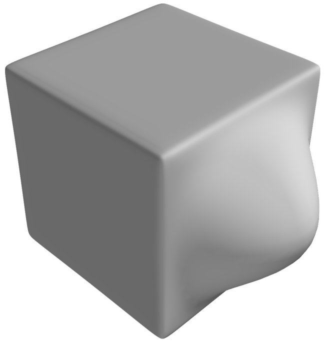
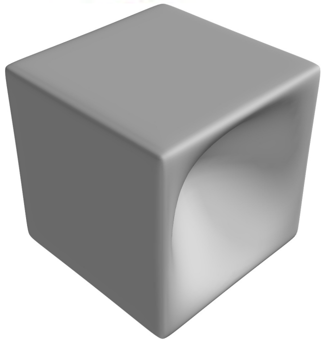
Intrinsic/Laplacian approaches are invariant to isometry ("pose invariant")

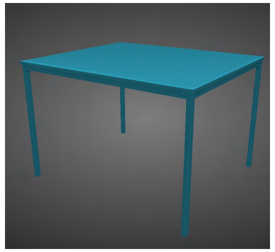
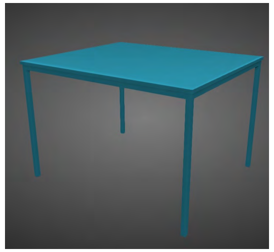
Real-world objects are usually subject to (near-) isometries

Isometry $\mathcal{T}$: length-preserving map

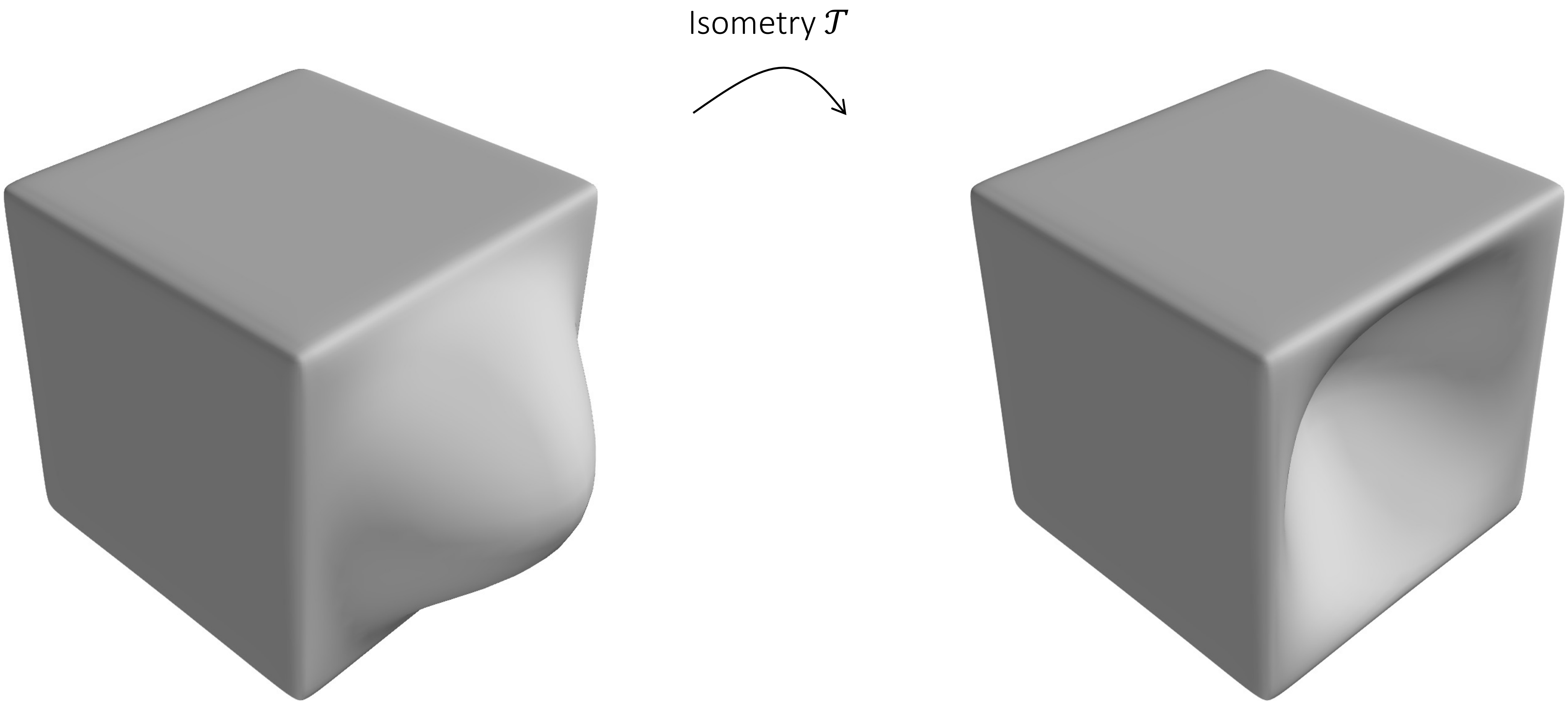


$f($  $)$ $\approx$ $f($  $)$

$f($  $)$ $=$ $f($  $)$

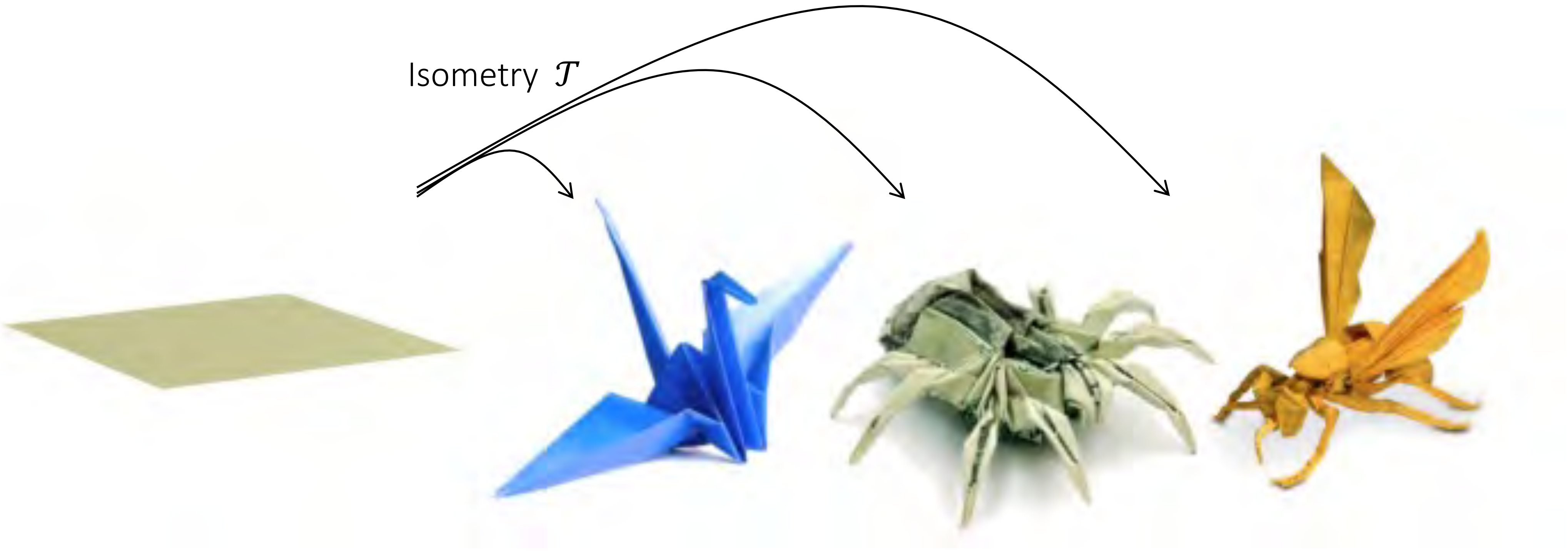
$f($  $)$ $\neq$ $f($  $)$

(Intrinsic) Laplacian can be Counterintuitive



- Same Laplacian operator → same Laplacian eigenvalues

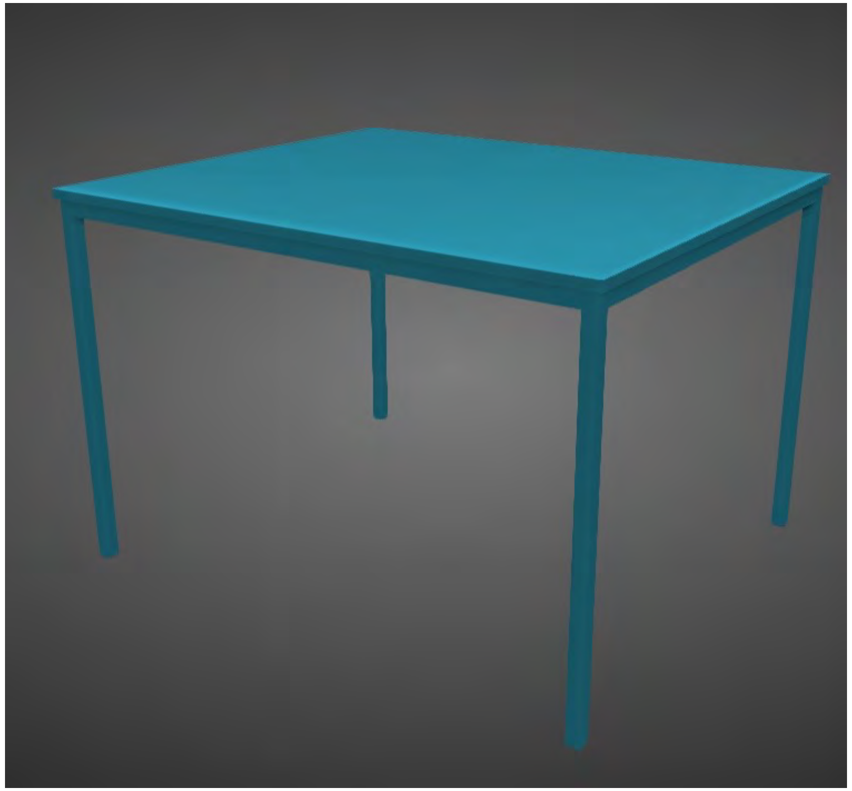
(Intrinsic) Laplacian Information is Incomplete

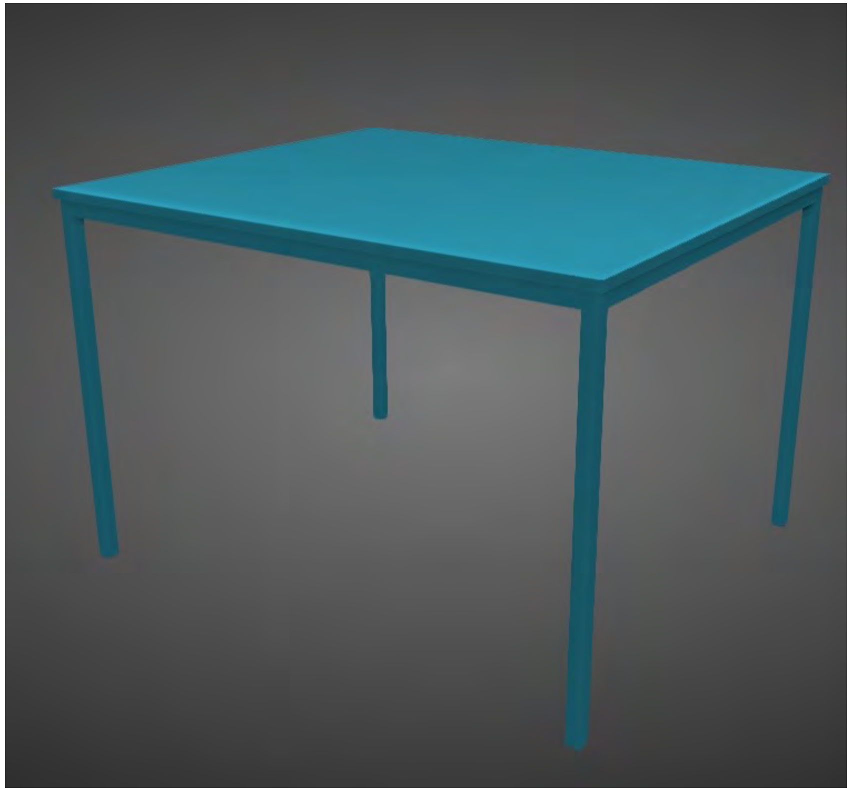


Intrinsic geometry: any origami is equivalent to a piece of flat paper!

Origami images from http://image.google.com/

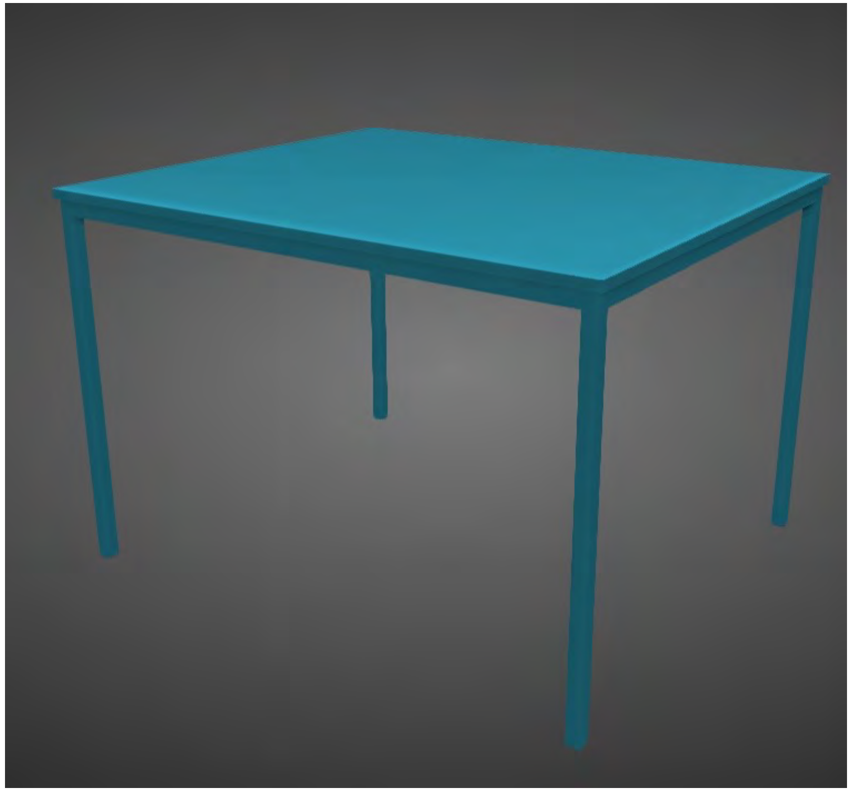
Laplacian Lacks Robustness





- Should be identical
- But completely different are their Laplacian eigenvalues---Why?

Laplacian is Sensitive to Topological Noises



- The surface connected to legs

- Not connected. The surface only *touches* the legs
- Topology errors → completely different Laplacian eigenvalues

Our Solution: Replace the Laplacian Δ with Operator $\mathcal{S}$

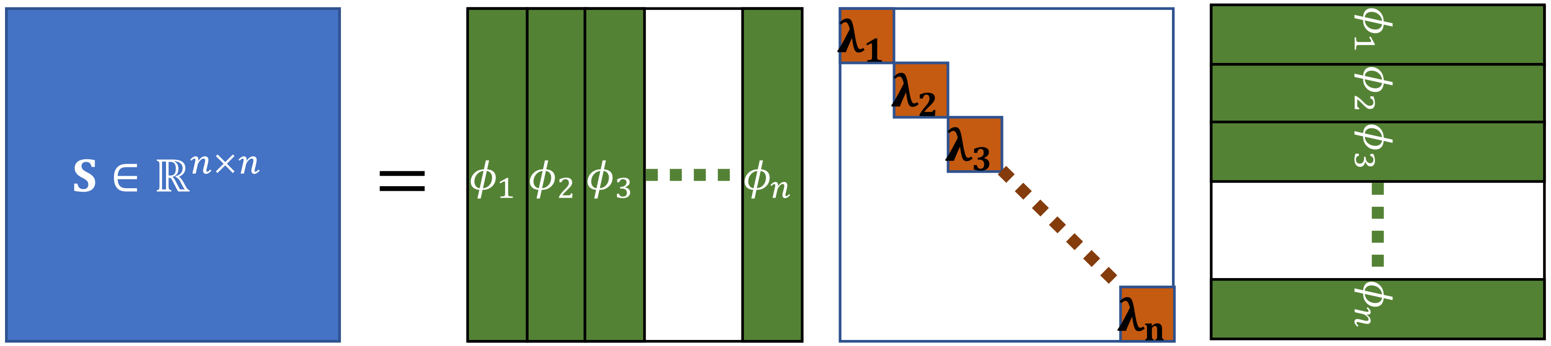
$$\Delta \to \mathcal{S}$$

$$\mathrm{L} \to \mathrm{S}$$

Our Solution: DtN Operator and Steklov Eigenvalue Problem

- Discrete Dirichlet-to-Neumann (DtN) operator: $\mathbf{S} \in \mathbb{R}^{n \times n}$

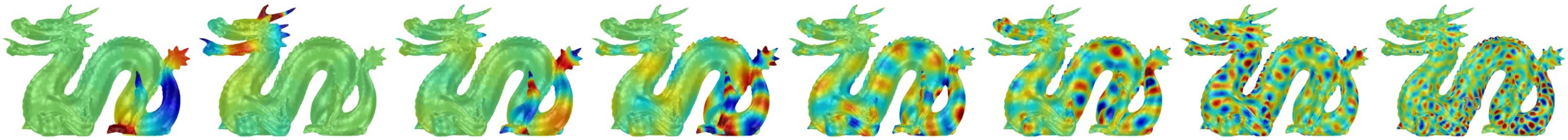
n: number of vertices



Definition

$\mathcal{S}$: the Dirichlet-to-Neumann operator. The Steklov eigenvalue problem

$$\mathcal{S}\psi = \lambda\psi$$



Operator $\mathcal{S}$: Encode Extrinsic Geometry, Backed by Spectral Geometry

Theorem (Lassas 2001)

Denote $\Omega_1, \Omega_2 \subseteq \mathbb{R}^3$ as two domains, and $\alpha : \Omega_1 \to \Omega_2$ is a bijection. Under proper assumptions, if the two domains have the same Dirichlet-to-Neumann operators (under map composition), then α must be a rigid motion.

For smooth domains in $\mathbb{R}^3$, the Steklov heat kernel admits the asymptotic expansion

[Polterovich and Sher 2015]

$$e^{-t\mathcal{S}}(x,x) = \sum_{i=0}^{\infty} e^{-t\lambda_i}\phi_i(x)^2 \sim \sum_{k=0}^{\infty} a_k(x)t^{k-2} + \sum_{l=1}^{\infty} b_l(x)t^l \log t,$$

$$a_0(x) \equiv \frac{1}{2\pi}$$

$$a_1(x) = \frac{H(x)}{4\pi}$$

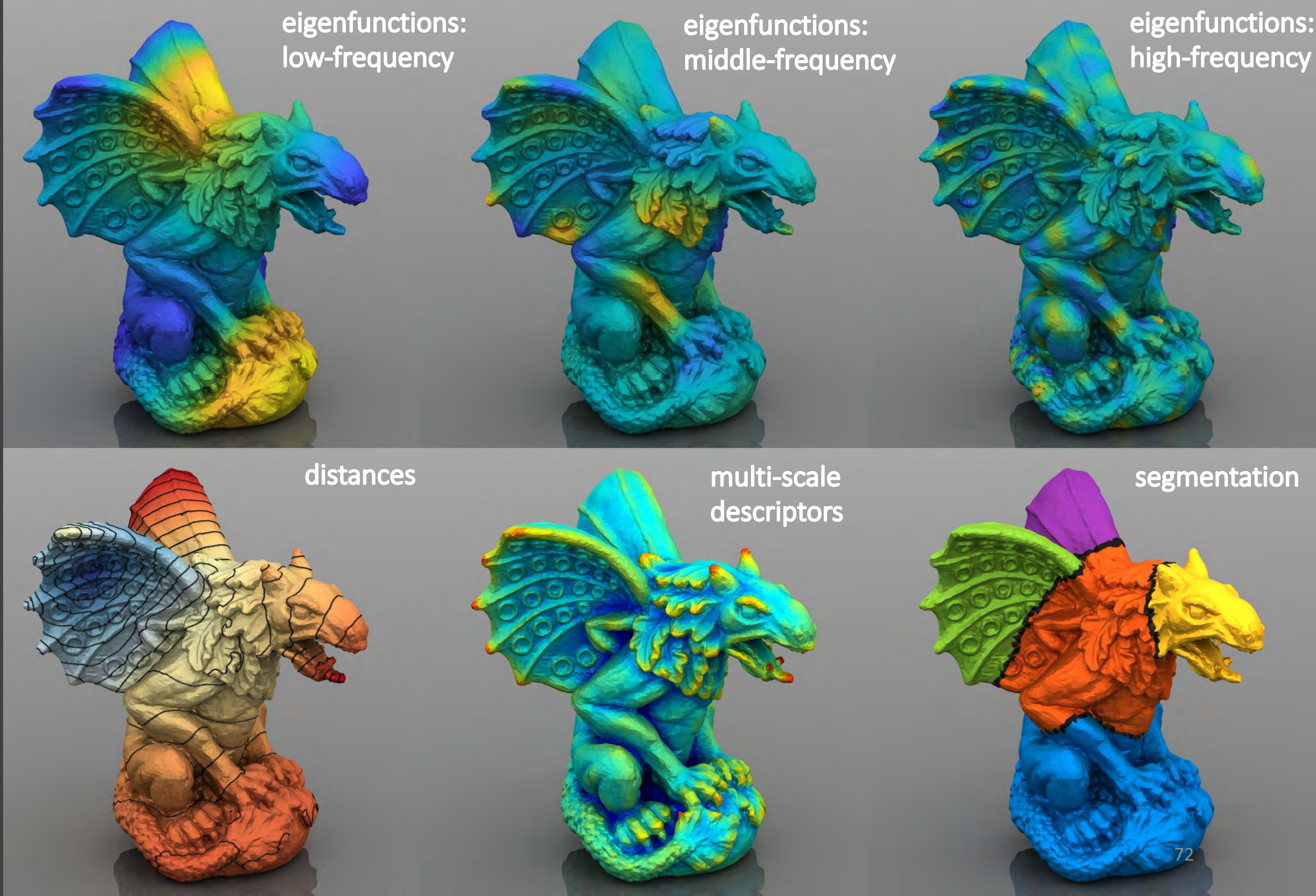
$$a_2(x) = \frac{1}{16\pi}\left(H(x)^2 + \frac{K(x)}{3}\right)$$

$H(x)$: mean curvature

$K(x)$: Gaussian curvature

 $\leftrightarrow \mathcal{S}$ $\quad f($  $) \approx f($  $)$

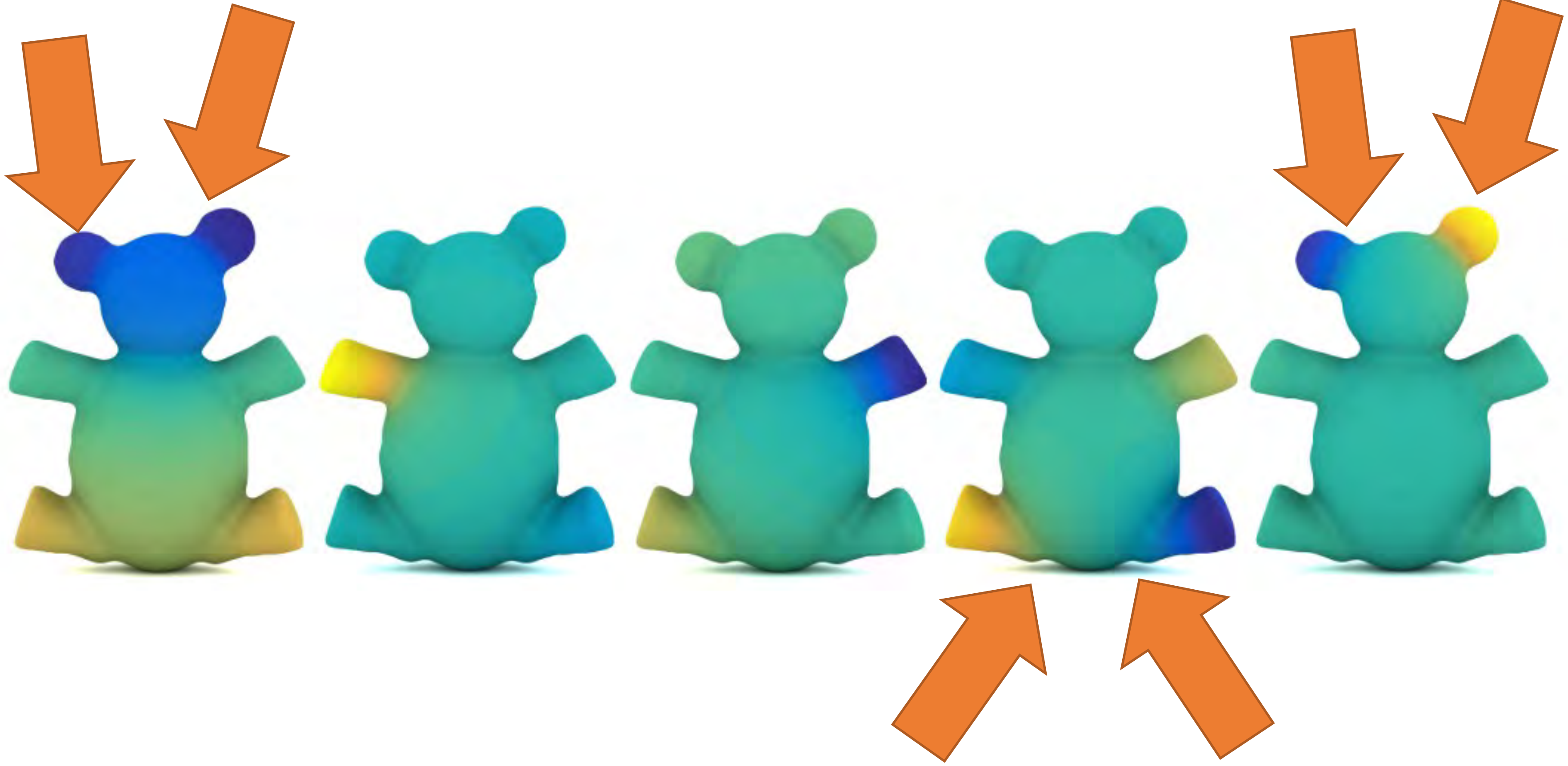
Quantities Defined using Eigenfunctions

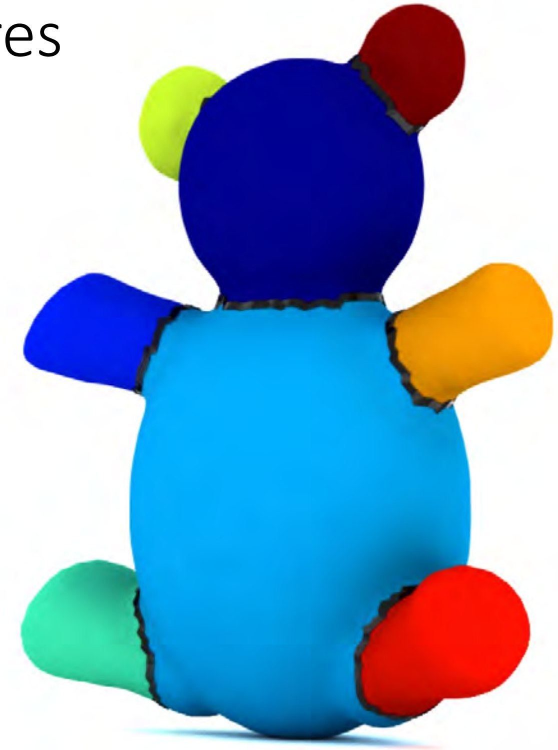


Steklov Kernels for Shape Segmentation

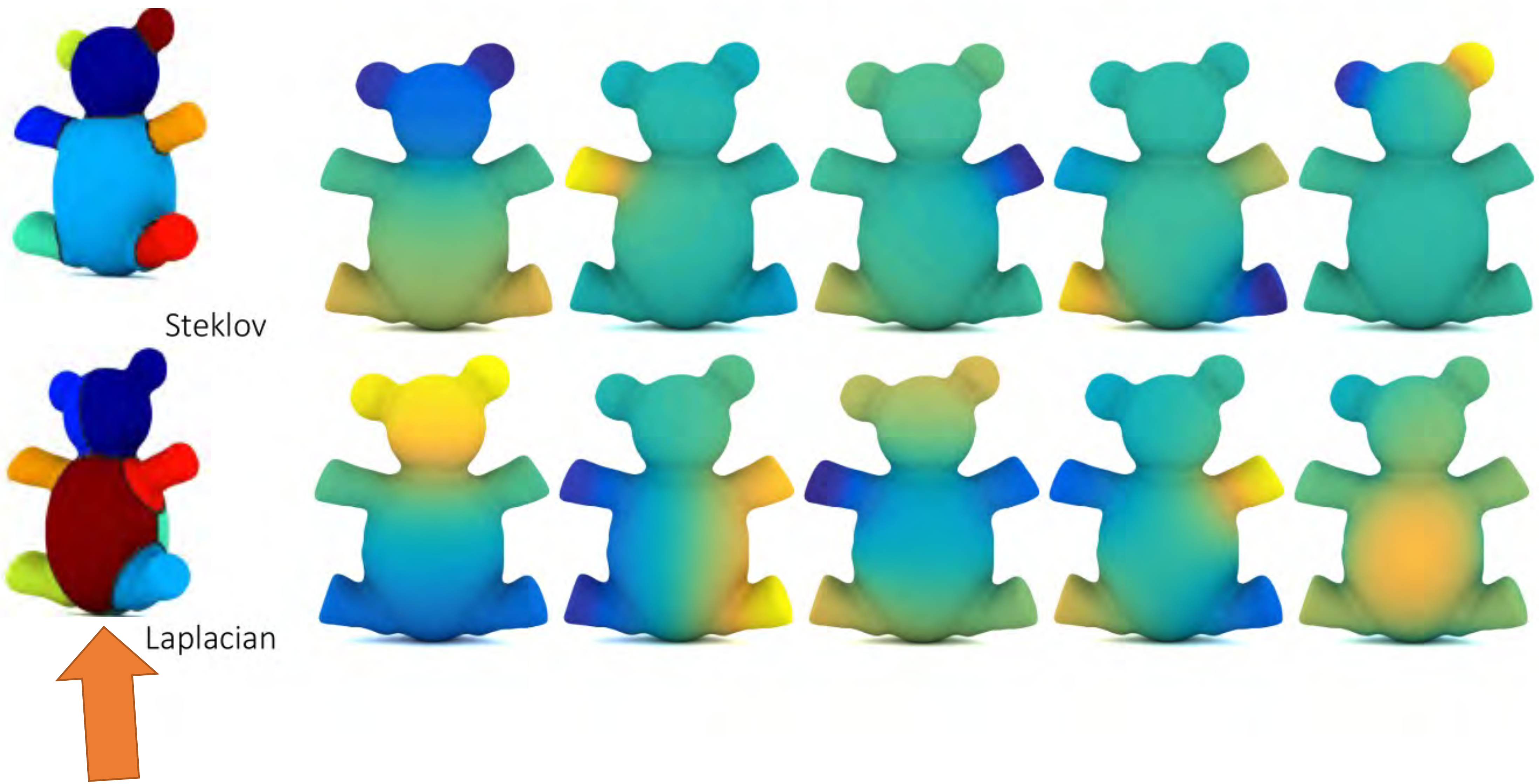
Spectral clustering with geodesics / heat kernels → shape segmentation

Level sets of Steklov eigenfunctions conform to mean curvatures

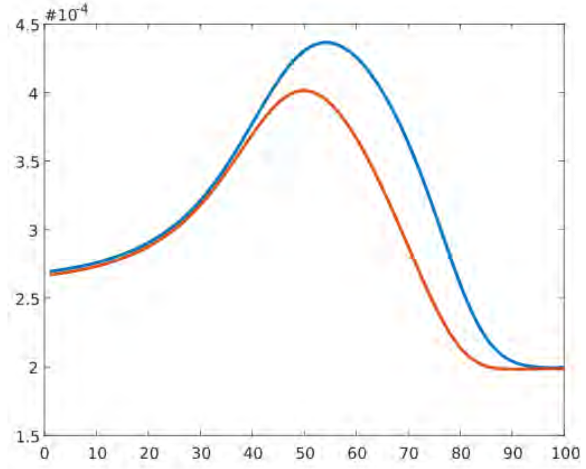




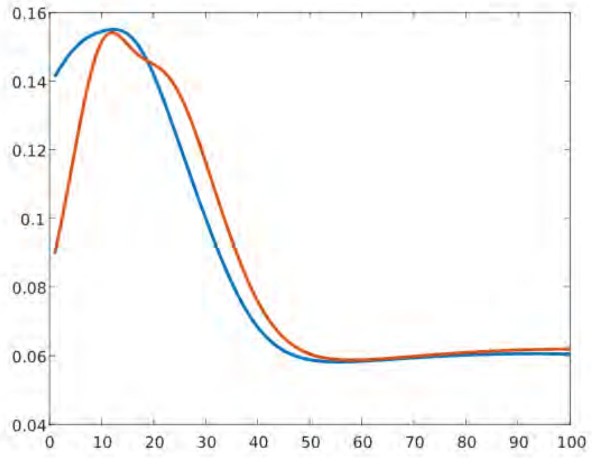
Laplacian Segmentation: Much Worse than Ours



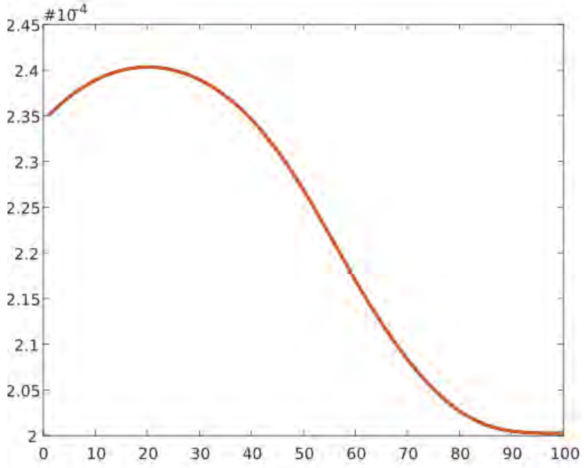
Heat Kernel Signature $h_t(x)$



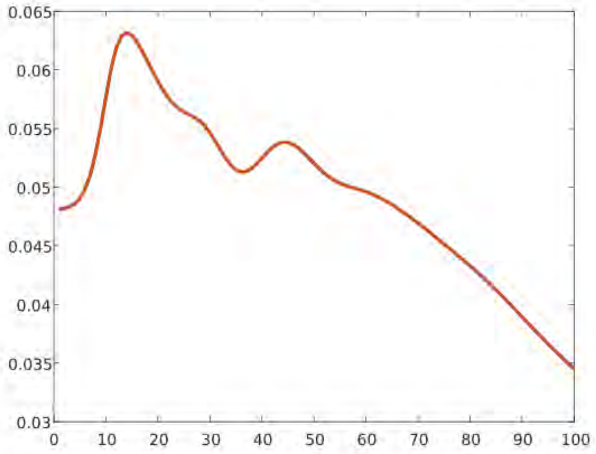
(a) Steklov Scaled-HKS



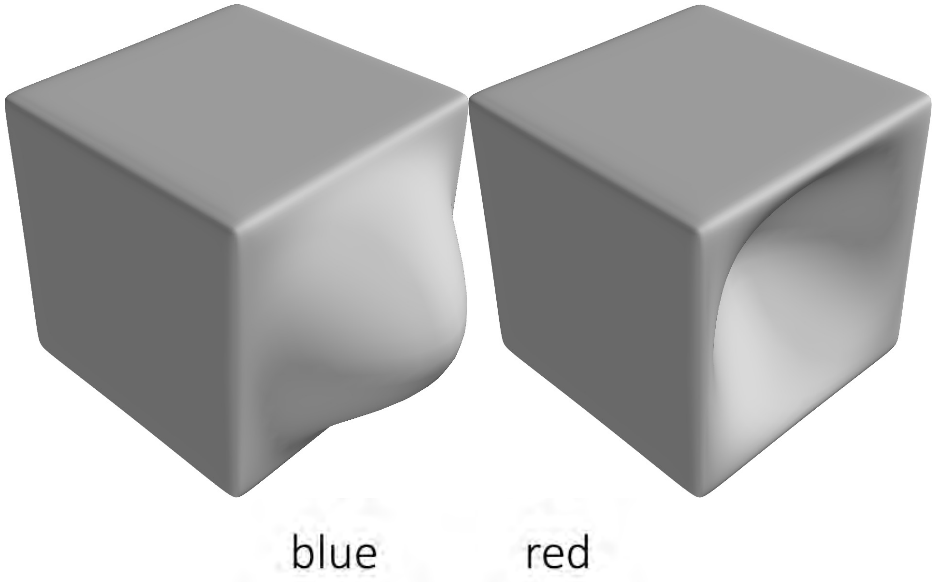
(b) Steklov WKS



(a) Laplacian Scaled-HKS

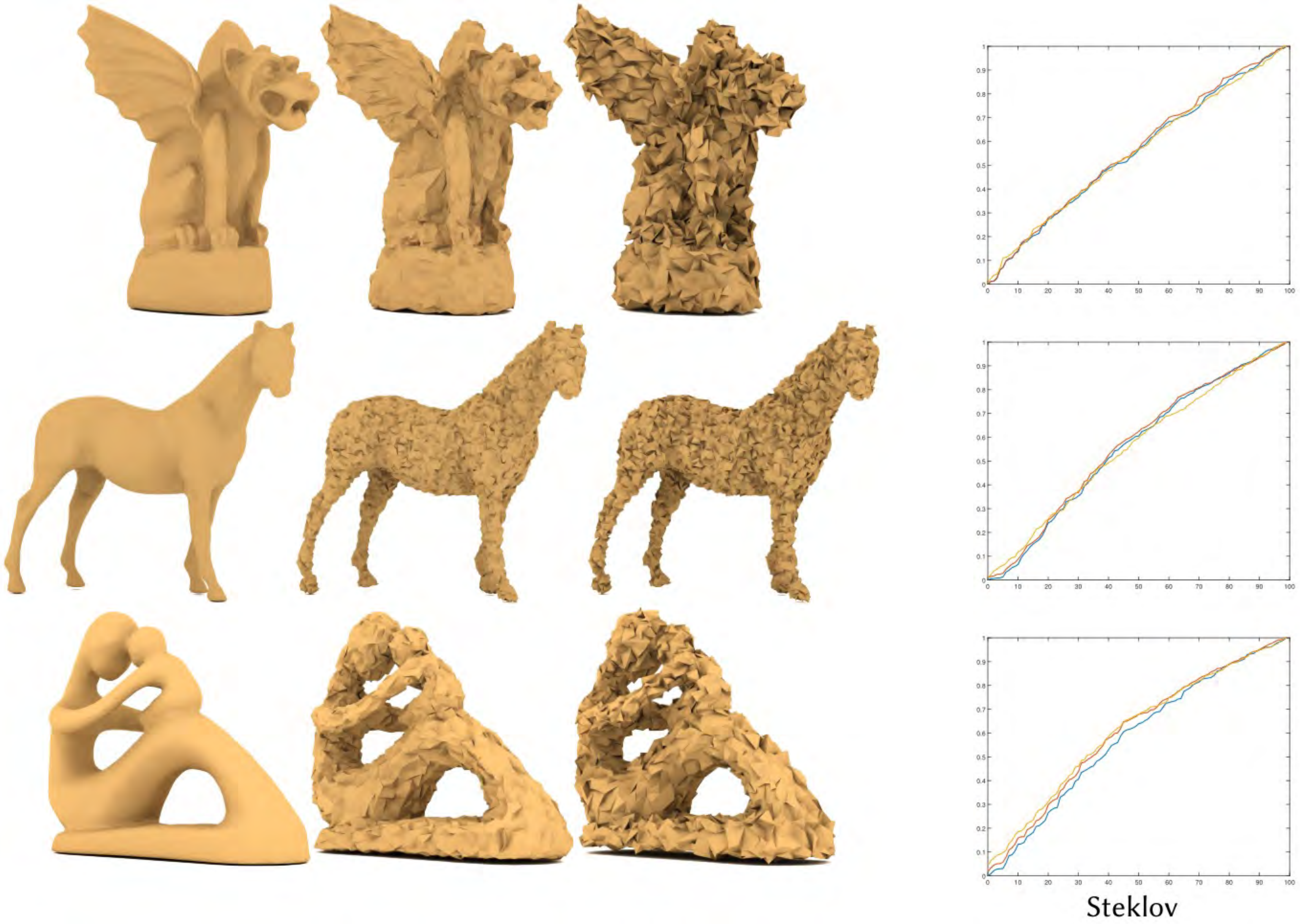


(b) Laplacian WKS



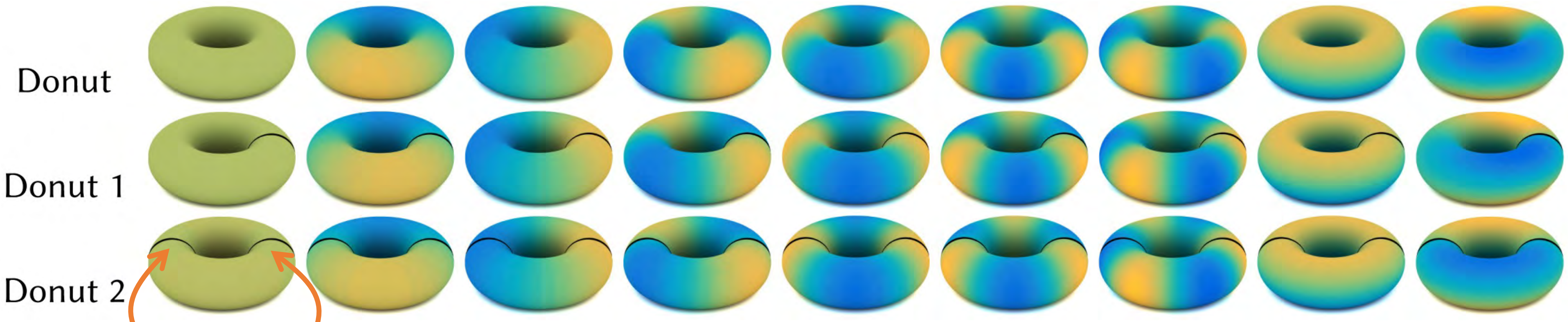
Replacing the Laplacian eigenfunctions with the Steklov ones---we can only distinguish the cubes

Robust to Noises

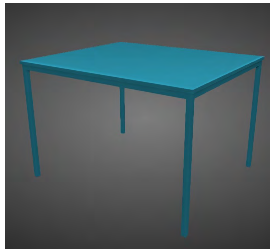
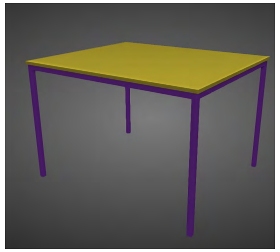


Boundary Approach: Robust to Non-watertight Surface

Steklov eigenfunctions: stable to topological changes for open surfaces

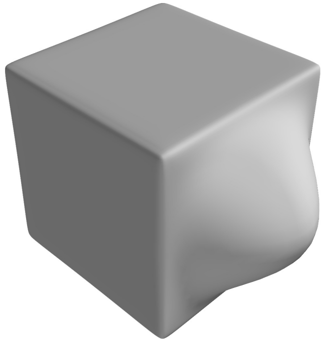
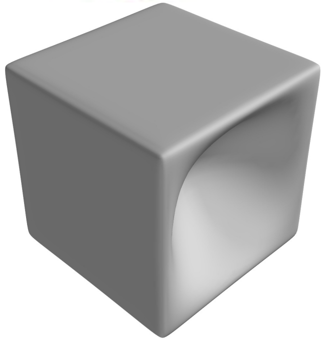


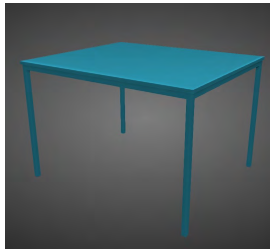
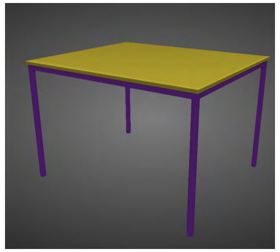
One or two cuts added

$f($  $) = f($  $)$

~~Laplacian~~ Steklov Shape Classification

$f($  $)$ $\approx$ $f($  $)$

$f($  $)$ $\neq$ $f($  $)$

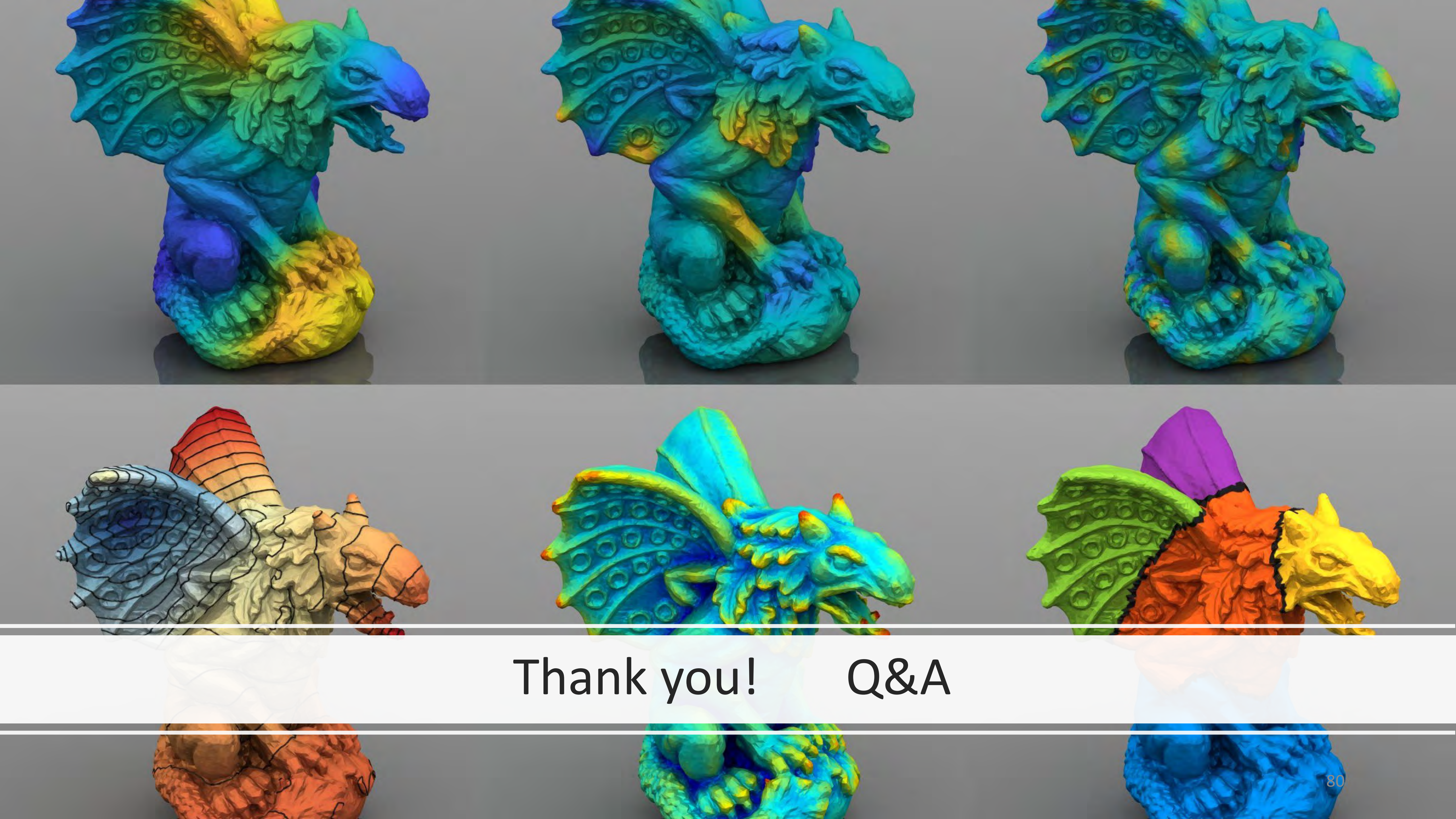
$f($  $)$ $=$ $f($  $)$

Summary: Mathematical Geometric Processing: Laplacian & Beyond

- A Tutorial on Laplacian
- Quasi-harmonic maps
  - search for a deformed Laplacian
- Extrinsic shape analysis
  - design a new operator
- Learn operator kernel from data
  - for a different operator kernel
- Future: potentially every tasks in geometric computing…

- 3min for the poll: https://tinyurl.com/15362-guestlec2

- Links to the paper: https://wangyu9.github.io/
- For further questions: wangyu9@mit.edu



Thank you! Q&A