Geodesic Distance

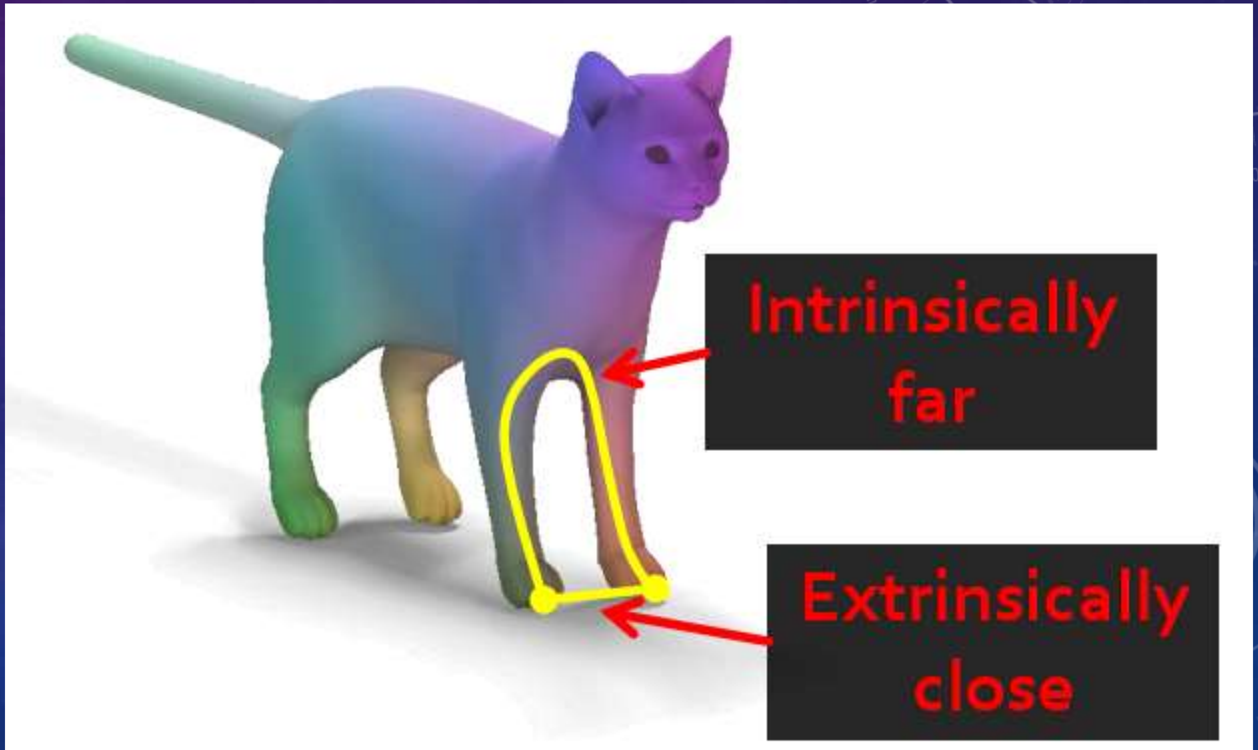
USTC, 2024 Spring

Qing Fang, fq1208@mail.ustc.edu.cn

https://qingfang1208.github.io/

Geodesic distance

- Length of the shortest path - constrained not to leave the manifold



Definition

Definition 6.1 (Geodesic distance). *The* geodesic distance *between two points* $\mathbf{p}, \mathbf{q} \in \mathcal{M}$ *on a submanifold* $\mathcal{M}$ *is given by*

$$d_{\mathcal{M}}(\mathbf{p}, \mathbf{q}) := \begin{cases} \inf_{\gamma:[0,1]\to\mathcal{M}} & L[\gamma] \\ \quad \textit{subject to} & \gamma(0) = \mathbf{p} \\ & \gamma(1) = \mathbf{q} \\ & \gamma \in C^1([0,1]). \end{cases} \tag{6.1}$$

Here, the curve γ *connects* $\mathbf{p}$ *to* $\mathbf{q}$*, and we are minimizing arc length as defined in* (3.2). *A curve* γ *realizing this infimum is known as a* global (minimizing) geodesic *curve.*

Length of a Curve

- Reparameterization in arc length:

$$\gamma: [a, b] \to \mathcal{M}, \qquad L[\gamma] = \int_a^b \left\| \frac{\partial \gamma}{\partial t} \right\| dt$$

- When the length of curve is a local minima?

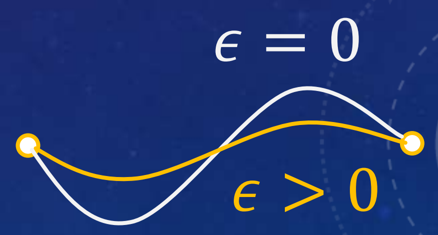
Variational approach

- Extend it to a one-parameter (ϵ) family of parametrized curves

$$\tilde{\gamma}(\cdot,\epsilon):[a,b]\to\mathcal{M}, \qquad \gamma(t)=\tilde{\gamma}(t,0)\ \forall\ t\in[a,b]$$

- The infinitesimal perturbation made by moving ϵ about $\epsilon=0$

$$\frac{\partial\tilde{\gamma}}{\partial\epsilon}:[a,b]\to TP(\mathcal{M})$$



Variational approach

- Extend it to a one-parameter (ϵ) family of parametrized curves

$$\tilde{\gamma}(\cdot,\epsilon):[a,b] \to \mathcal{M}, \qquad \gamma(t) = \tilde{\gamma}(t,0)\ \forall\ t \in [a,b]$$

- The infinitesimal perturbation made by moving ϵ $\dfrac{\partial\tilde{\gamma}}{\partial\epsilon}:[a,b] \to TP(\mathcal{M})$

$$E[\tilde{\gamma}(\cdot,\epsilon)] = \int_a^b \left\|\frac{\partial\tilde{\gamma}}{\partial t}\right\| dt \to E_{\epsilon=0} = L[\gamma]$$

Variational approach

$$\frac{\partial E}{\partial \epsilon} = \frac{\partial}{\partial \epsilon}\int_a^b \sqrt{\left\langle \frac{\partial \tilde{\gamma}}{\partial t}, \frac{\partial \tilde{\gamma}}{\partial t} \right\rangle}\, dt = \int_a^b \frac{\partial}{\partial \epsilon} \sqrt{\left\langle \frac{\partial \tilde{\gamma}}{\partial t}, \frac{\partial \tilde{\gamma}}{\partial t} \right\rangle}\, dt = \int_a^b \frac{\left\langle \frac{\partial \tilde{\gamma}}{\partial t}, \frac{\partial^2 \tilde{\gamma}}{\partial \epsilon \partial t} \right\rangle}{\sqrt{\left\langle \frac{\partial \tilde{\gamma}}{\partial t}, \frac{\partial \tilde{\gamma}}{\partial t} \right\rangle}}\, dt$$

$$\left.\frac{\partial E}{\partial \epsilon}\right|_{\epsilon=0} = \int_a^b \left\langle \frac{\partial \tilde{\gamma}}{\partial t}, \frac{\partial^2 \tilde{\gamma}}{\partial \epsilon \partial t} \right\rangle dt = \left.\left\langle \frac{\partial \tilde{\gamma}}{\partial t}, \frac{\partial \tilde{\gamma}}{\partial \epsilon} \right\rangle\right|_{t=a}^{b} - \int_a^b \left\langle \frac{\partial^2 \tilde{\gamma}}{\partial t^2}, \frac{\partial \tilde{\gamma}}{\partial \epsilon} \right\rangle dt = -\int_a^b \left\langle \frac{\partial^2 \gamma}{\partial t^2}, \frac{\partial \tilde{\gamma}}{\partial \epsilon} \right\rangle dt$$

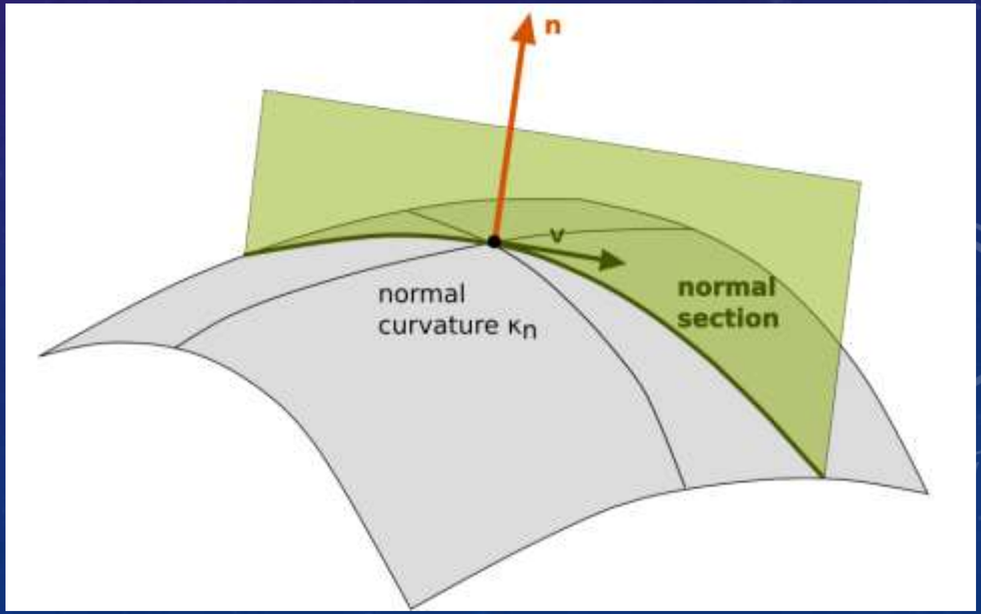
Variational approach

As $\left.\frac{\partial E}{\partial \epsilon}\right|_{\epsilon=0} = -\int_a^b \left\langle \kappa_\gamma, \frac{\partial \tilde{\gamma}}{\partial \epsilon} \right\rangle dt$ and $\frac{\partial \tilde{\gamma}}{\partial \epsilon} \in TP(\mathcal{M})$, local minima for any perturbation

$$\Longrightarrow \frac{\partial^2 \gamma}{\partial t^2} \perp TP(\mathcal{M})$$

Geodesic curve : $\kappa_g = 0$ $(\kappa^2 = \kappa_n^2 + \kappa_g^2)$

Straightest curve

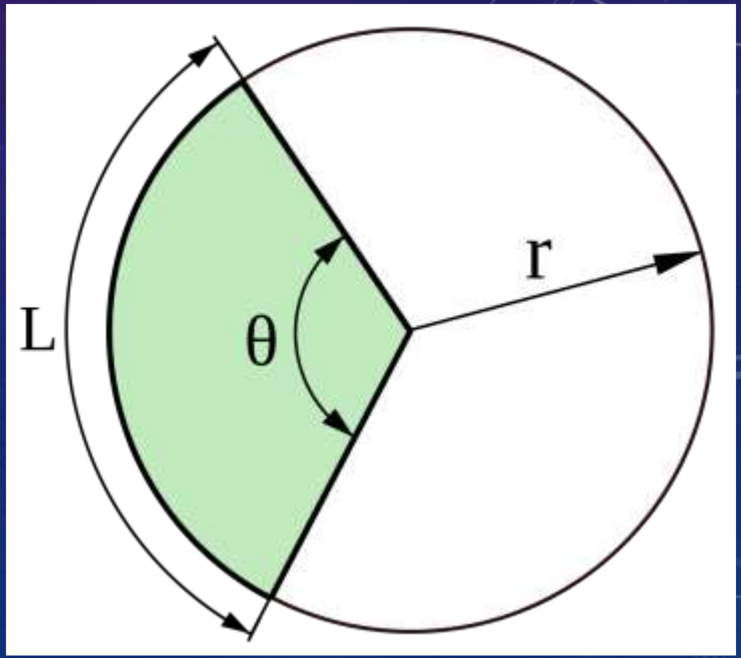


Geodesic distance

- Length of globally shortest geodesic curve

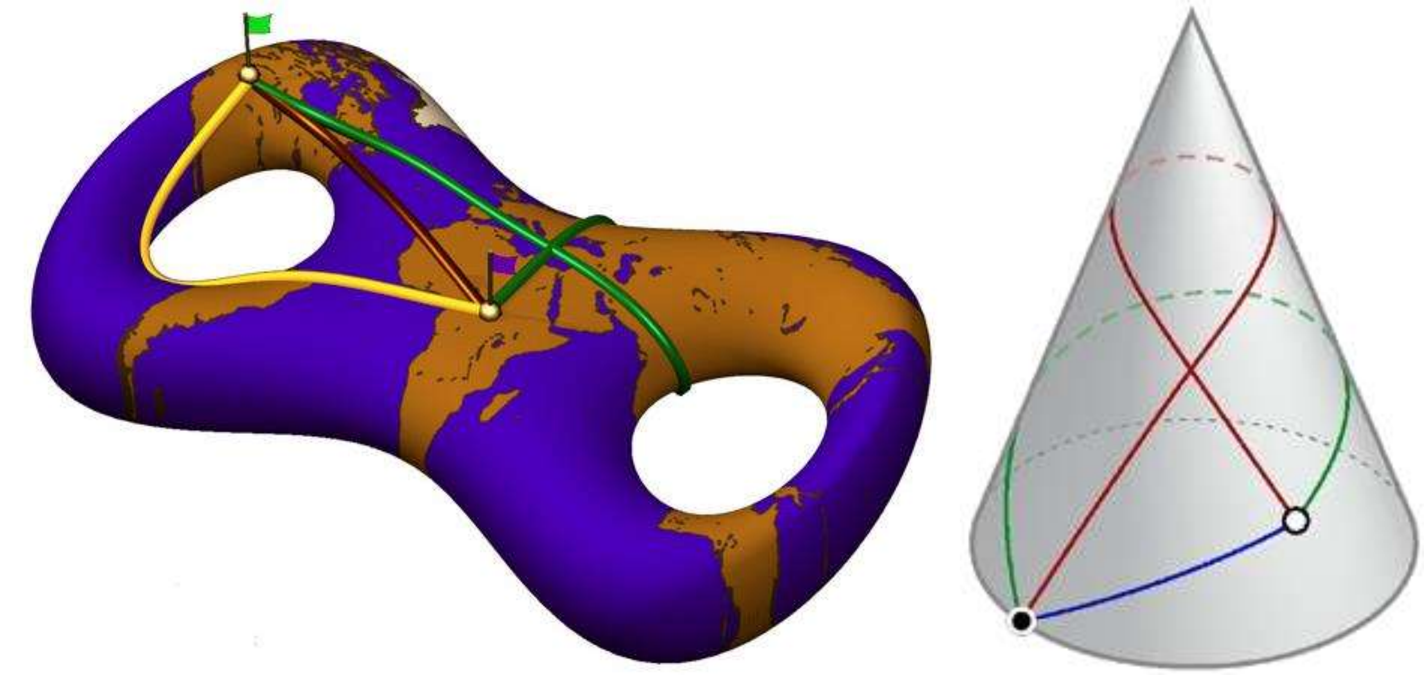
The straight line connecting two points on a plane.

The minor great arc on the sphere.



Geodesic distance

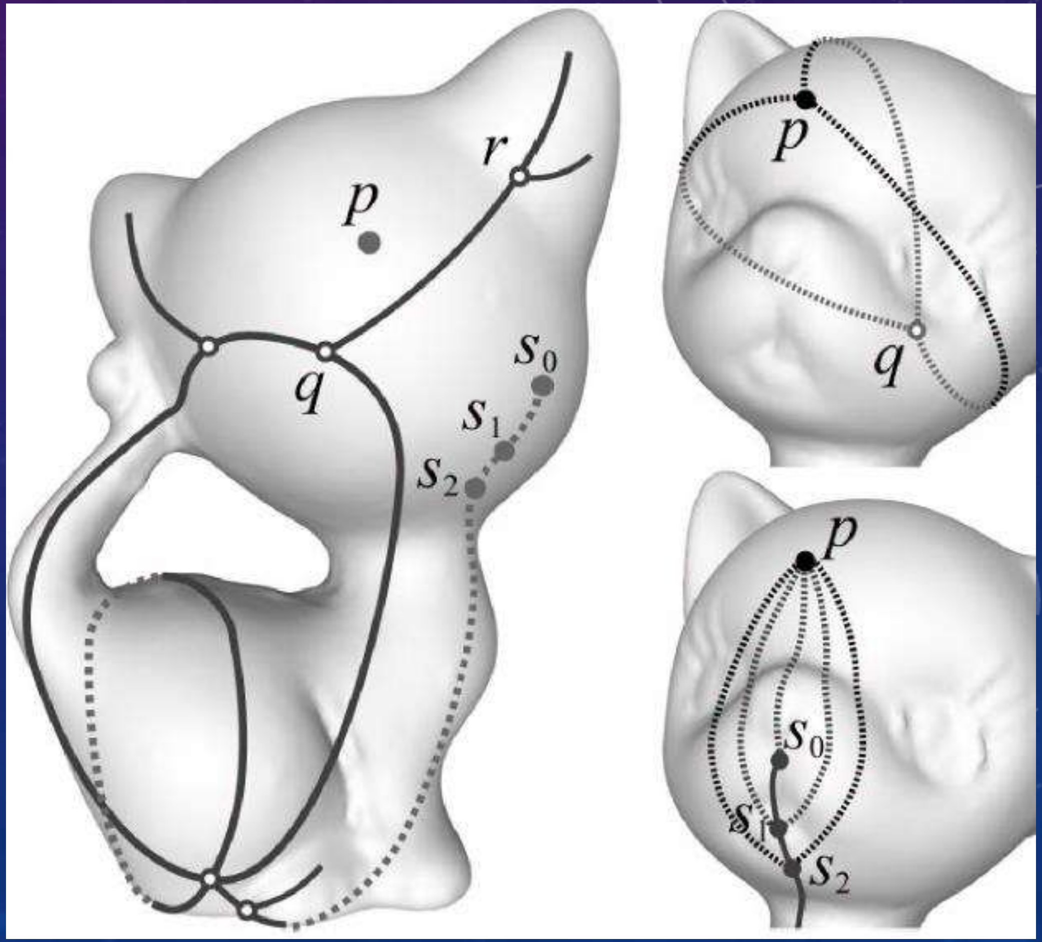
- Length of globally shortest geodesic curve
- Local minima may not be global



Cut locus

For a given point p, the injectivity radius is the radius $r > 0$ of the largest geodesic ball $B_p(r)$ such that there is a unique geodesics from any point $q \in B_p(r)$ back to p.

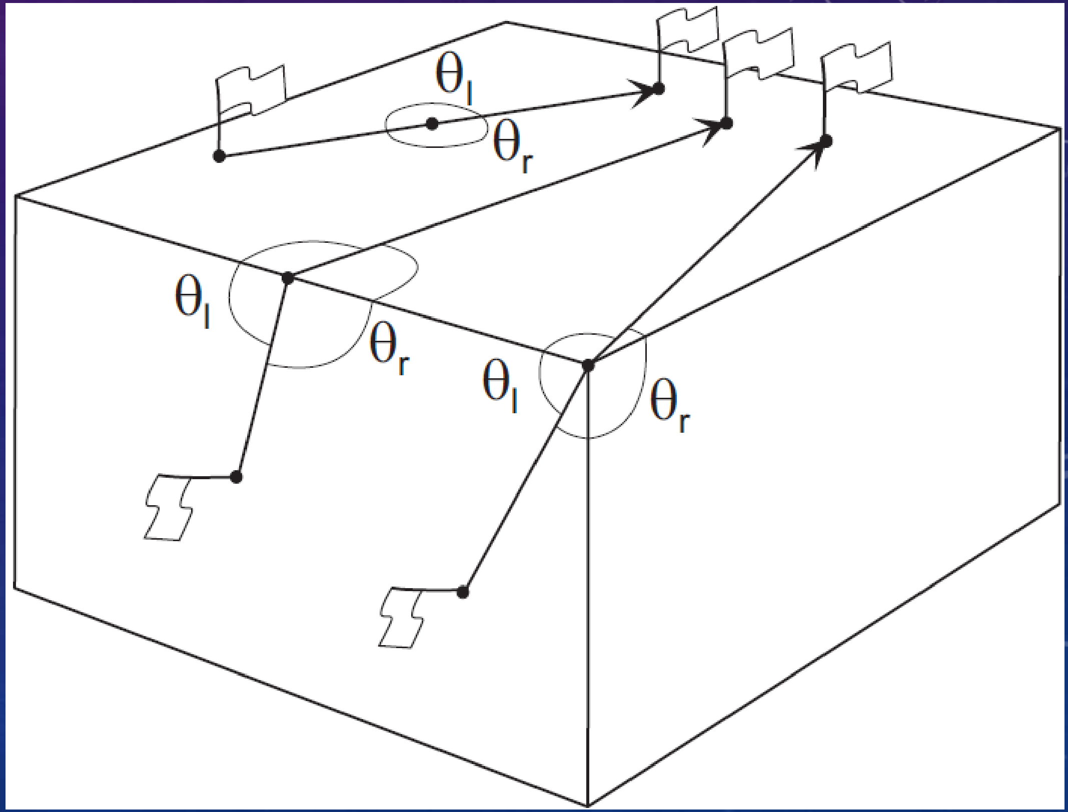
Outside this radius, there are points q with two or more geodesics to p. The collection of all such points is called the cut locus.



Straightest geodesics

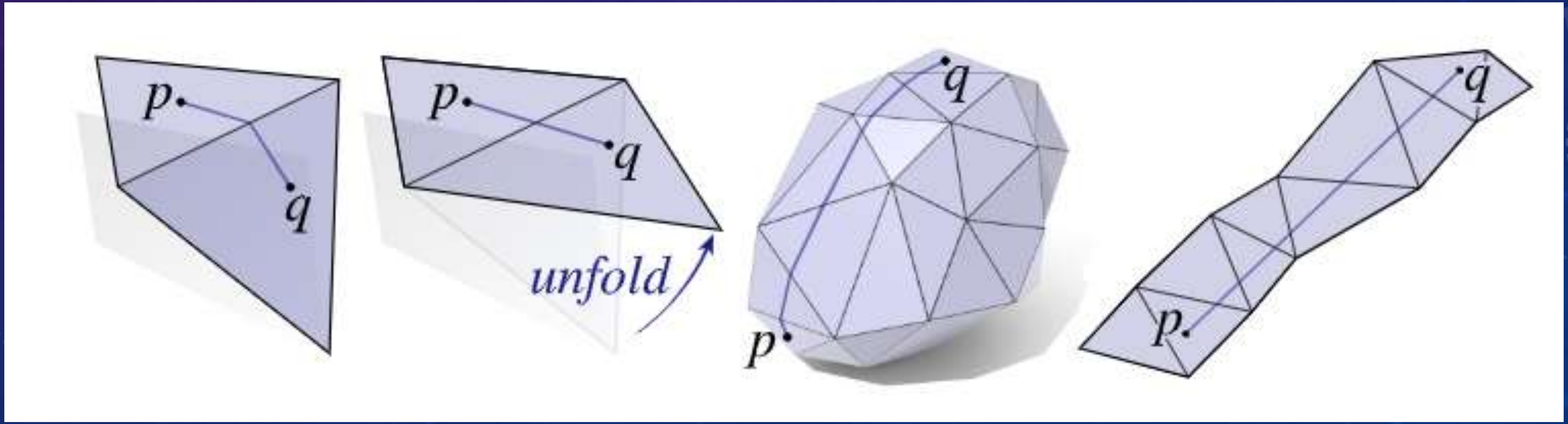
- Polthier and Schmies. Shortest Geodesics on Polyhedral Surfaces. SIGGRAPH course notes 2006

  Equal left and right angles



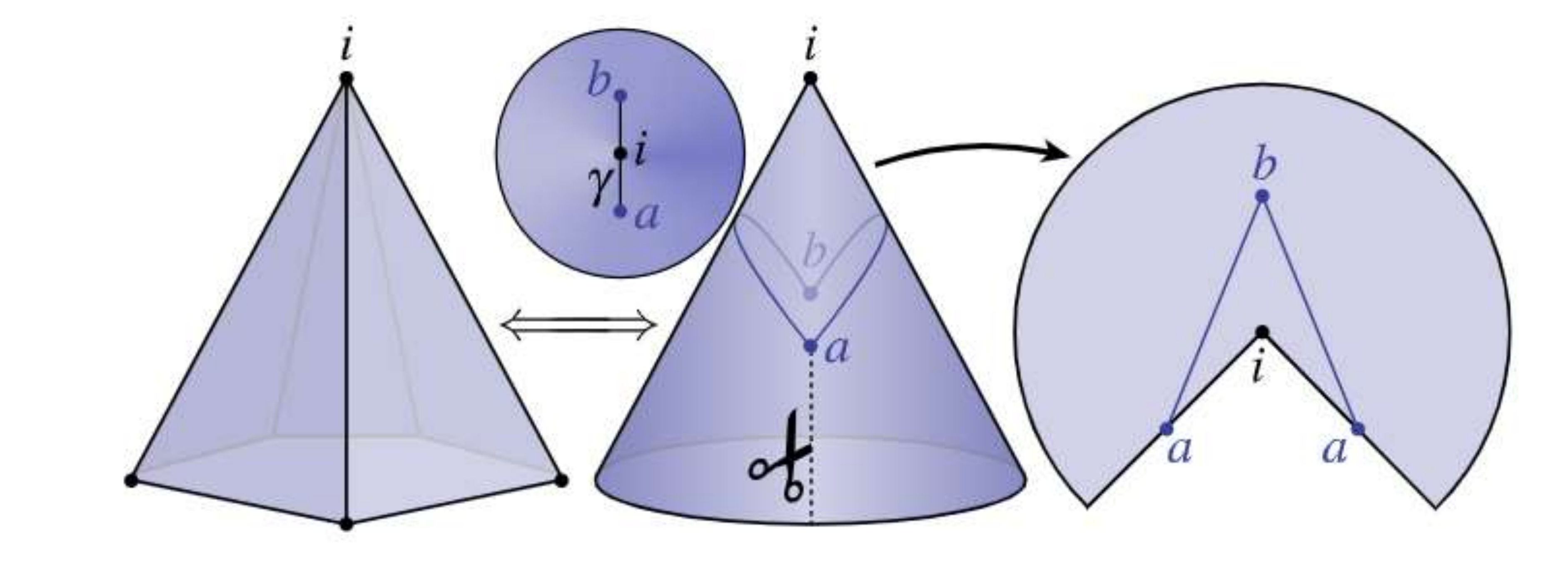
Unfolding triangles

- A geodesic on a polyhedral surface is therefore equivalent to a straight line along some planar triangle strip—so long as it does not pass through any vertices.



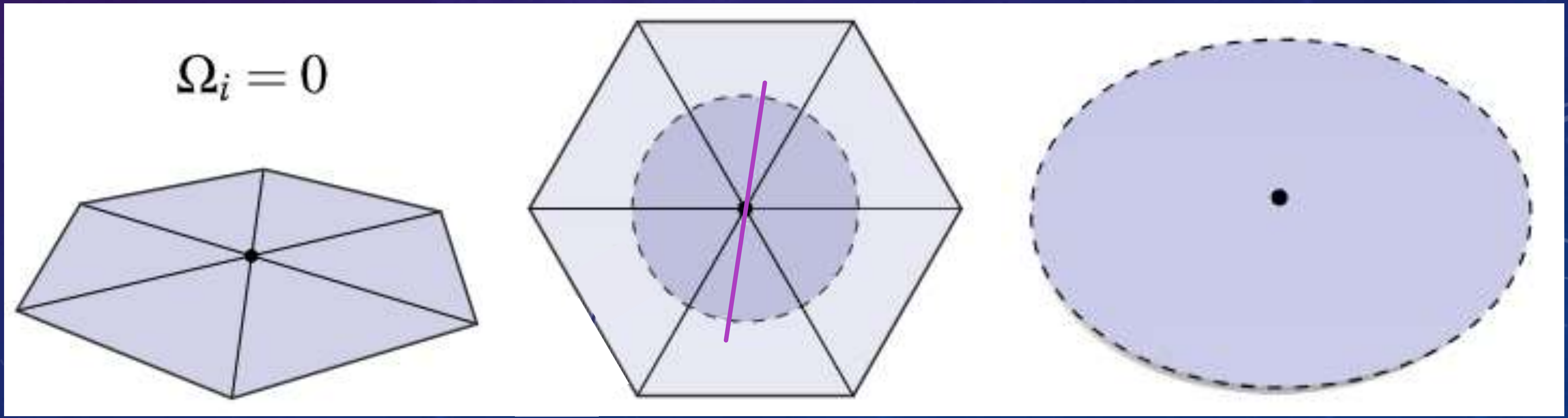
If path enters a vertex

- $K > 0$: straightest geodesic is never shortest



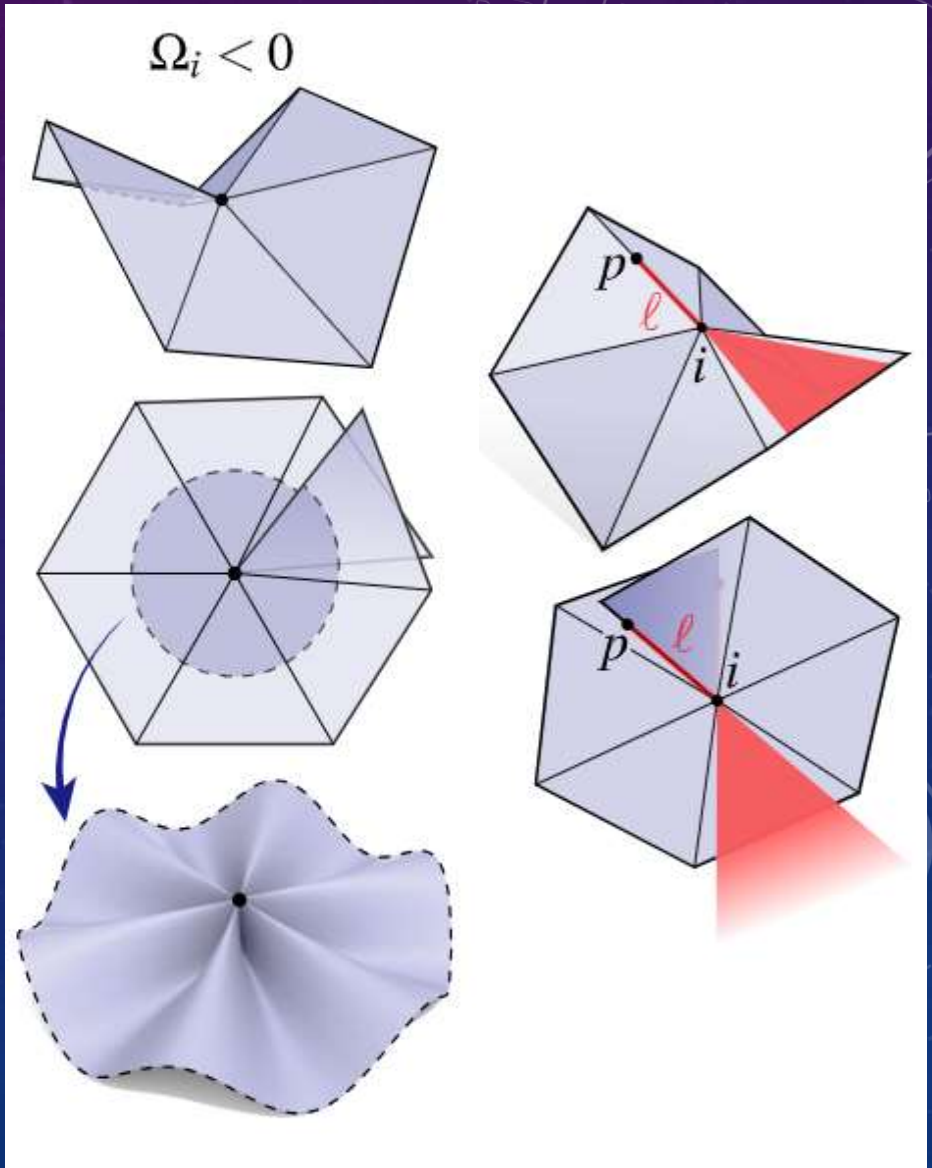
If path enters a vertex

- $K > 0$: straightest geodesic is never shortest
- $K = 0$: one shortest and straightest

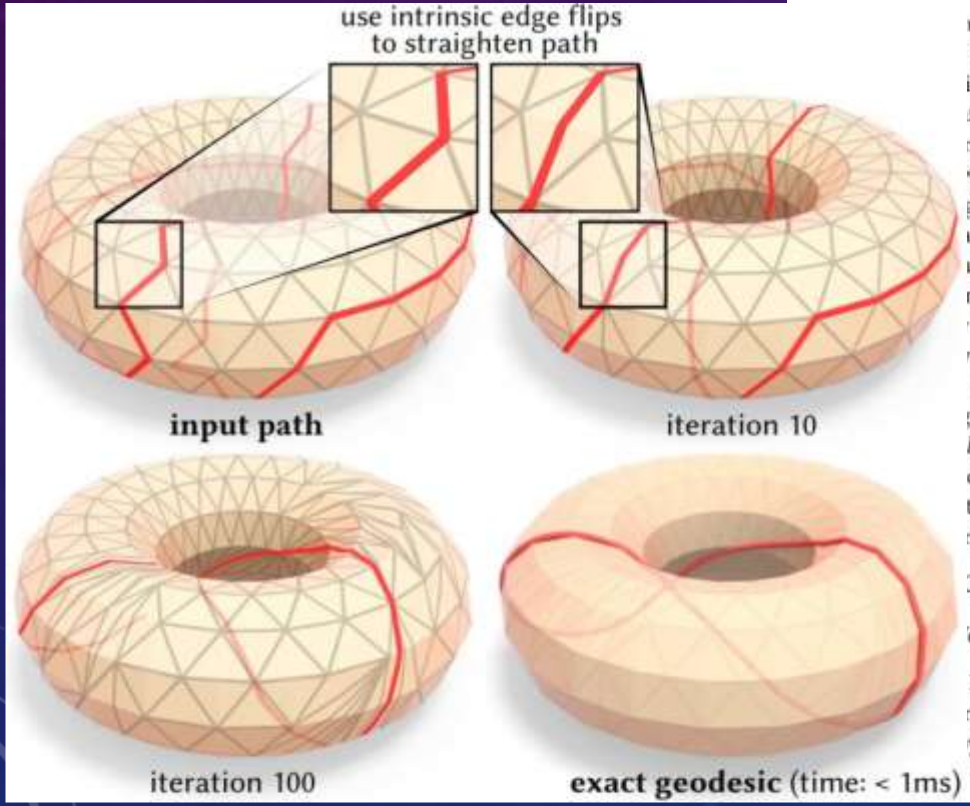


If path enters a vertex

- $K > 0$: straightest geodesic is never shortest
- $K = 0$: one shortest and straightest
- $K < 0$: multiple shortest but one straightest



New algorithm for geodesic paths



You Can Find Geodesic Paths in Triangle Meshes by Just Flipping Edges

NICHOLAS SHARP and KEENAN CRANE, Carnegie Mellon University

uces a new approach to computing geodesics on polyhe- basic idea is to iteratively perform *edge flips*, in the same ic Delaunay flip algorithm. This process also produces a forming to the output geodesics, which is immediately ı geometry processing and numerical simulation. More OUT algorithm transforms a given sequence of edges into geodesic while avoiding self-crossings (formally: it finds a ıme *isotopy class*). The algorithm is guaranteed to termi- ımber of operations; practical runtimes are on the order nds, even for meshes with millions of triangles. The same applied to curves beyond simple paths, including closed orks, and multiply-covered curves. We explore how the tasks such as straightening cuts and segmentation bound- eodesic Bézier curves, extending the notion of *constrained lations (CDT)* to curved surfaces, and providing accurate ons for partial differential equations (PDEs). Evaluation tasets such as *Thingi10k* indicates that the method is both ıt, even for low-quality triangulations.

Computing methodologies → **Shape modeling**.

ords and Phrases: geodesic, edge flip, triangulation

Format:
ıd Keenan Crane. 2020. You Can Find Geodesic Paths in y Just Flipping Edges. *ACM Trans. Graph.* 39, 6, Article 249 15 pages. https://doi.org/10.1145/3414685.3417839

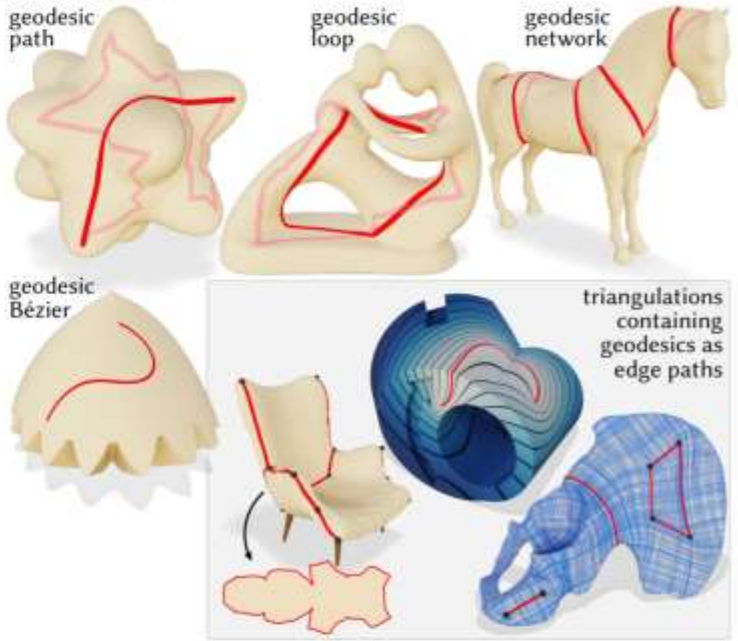
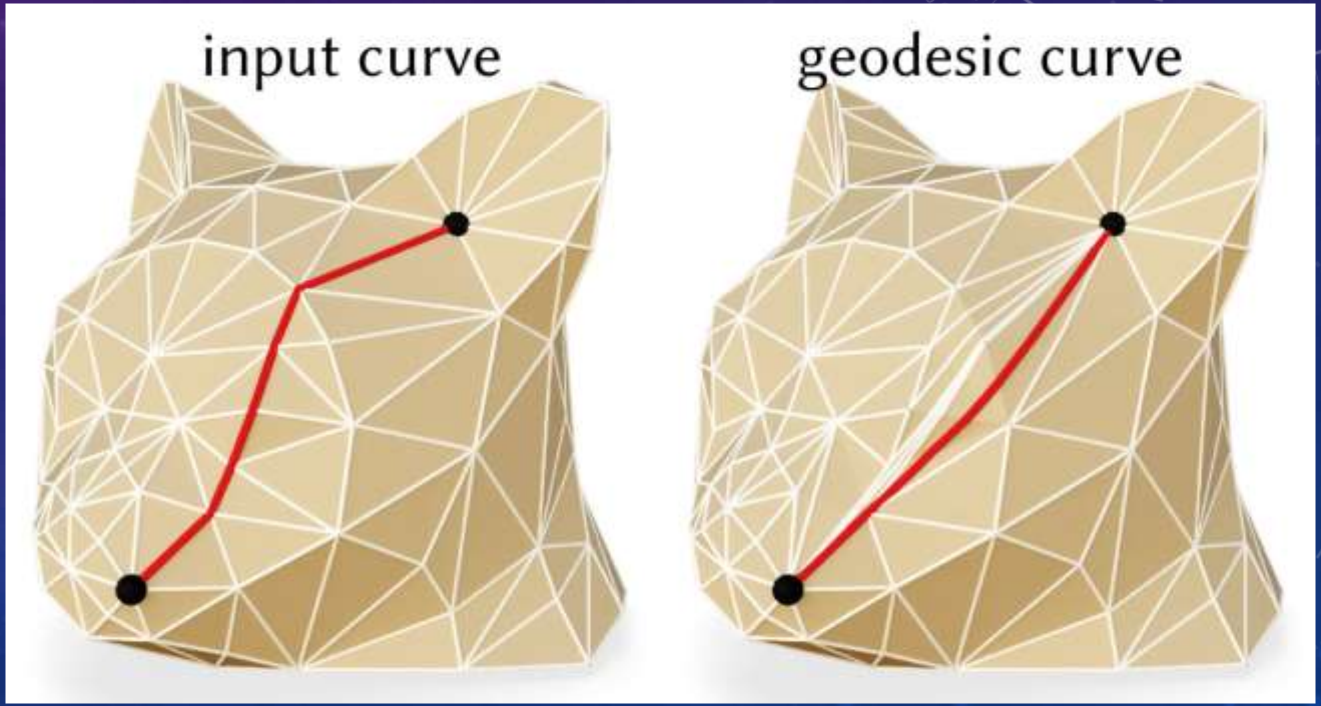


Fig. 1. We introduce an edge-flip based algorithm for computing geodesic paths, loops, and networks on triangle meshes. The algorithm also yields a triangulation containing these curves as edges, which can be used directly for subsequent geometry processing (*e.g.*, for cutting, or for solving PDEs).

Problem

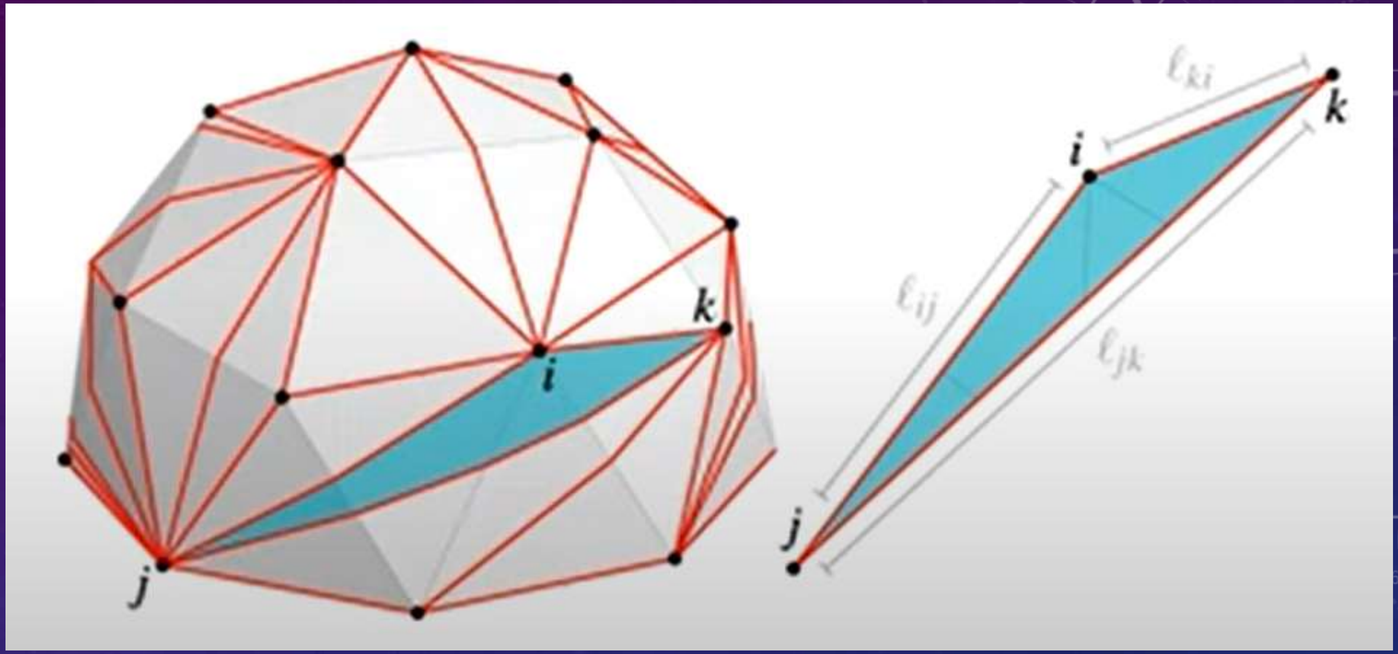
- Input: a path (or loop/network) on the surface of a mesh
- Output: an exact geodesic path (or loop/network)

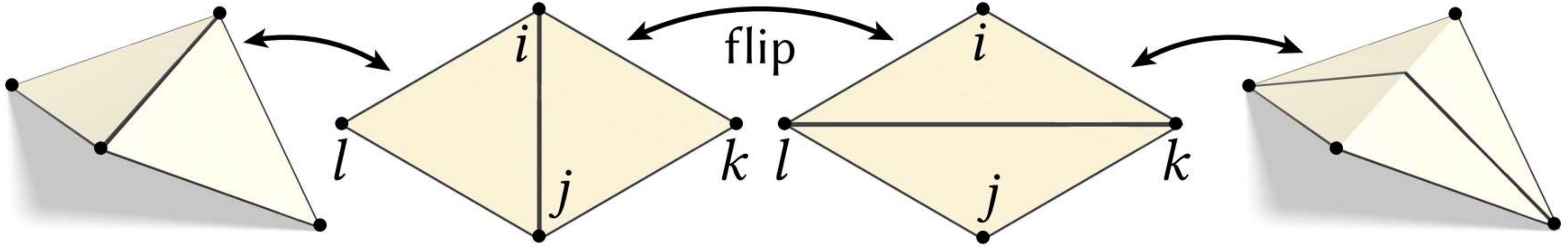


Intrinsic flip

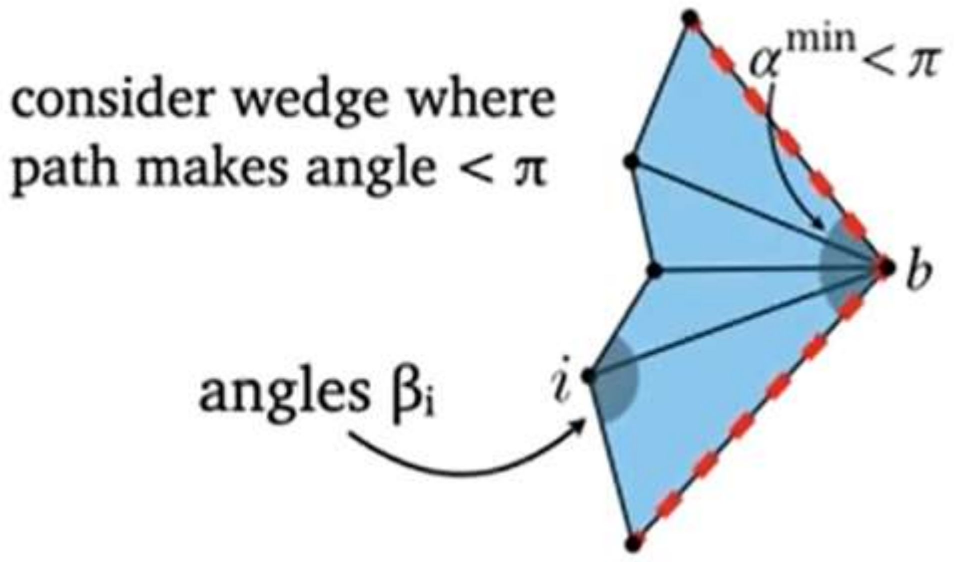
- Flip edges on a mesh

(unfolding)





FlipOut subroutine



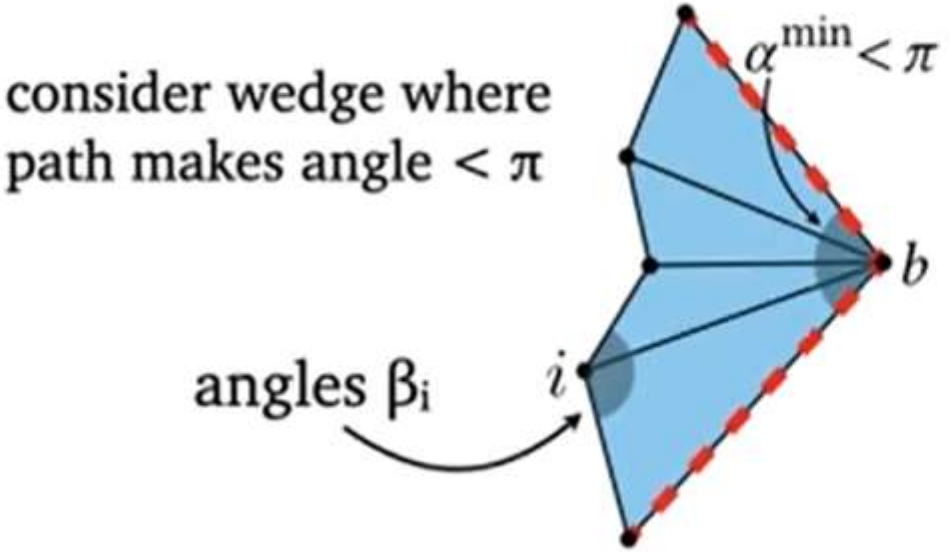
func **FlipOut()**

Input: path through vertices a,b,c

- while any $\beta_i < \pi$
  - flip first such edge *bi*

Output: shorter path along boundary

FlipOut subroutine

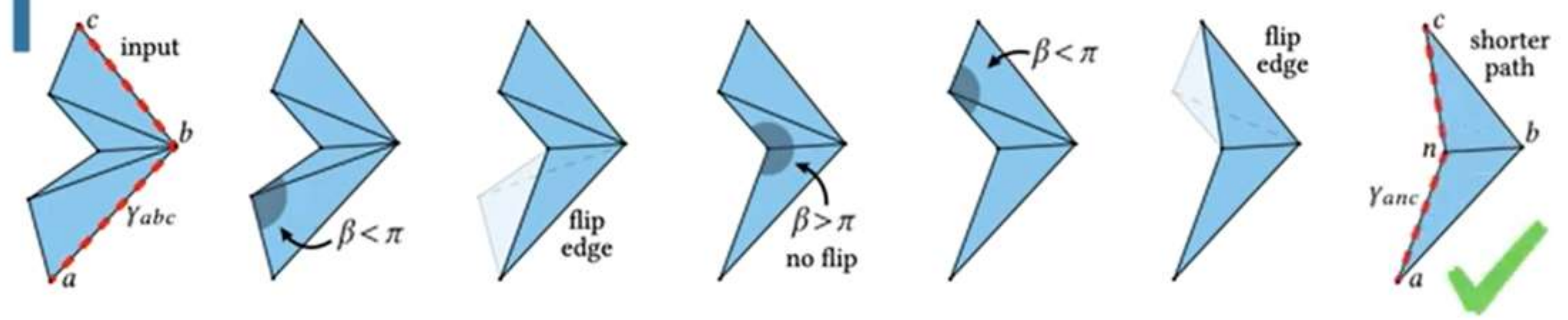


func **FlipOut()**

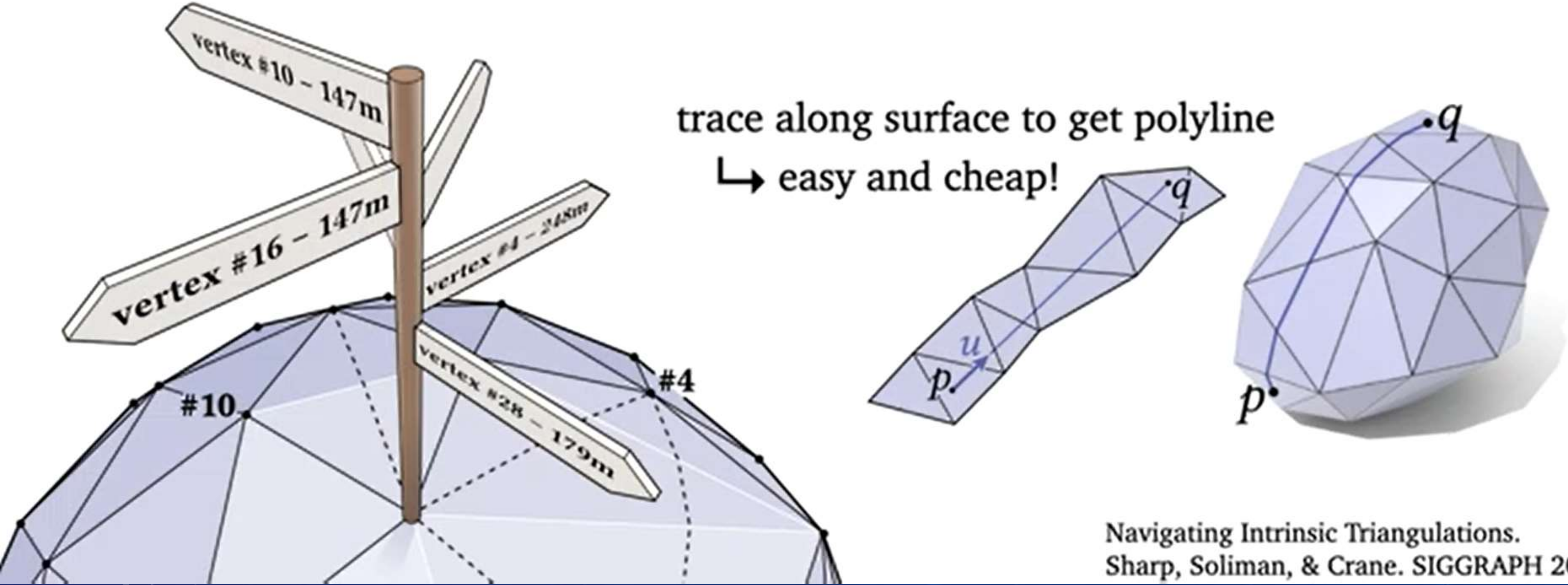
Input: path through vertices a,b,c

- while any $\beta_i < \pi$
  - flip first such edge *bi*

Output: shorter path along boundary



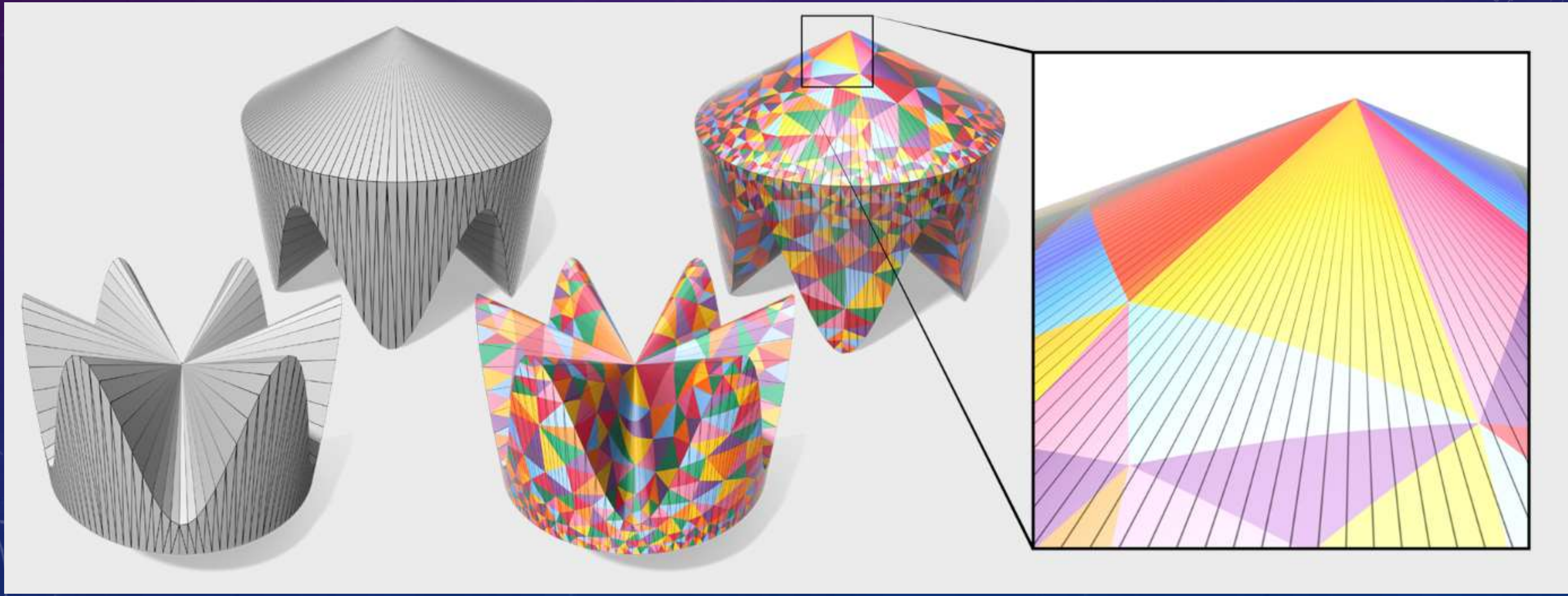
Implementation details



Navigating Intrinsic Triangulations.
Sharp, Soliman, & Crane. SIGGRAPH 2018

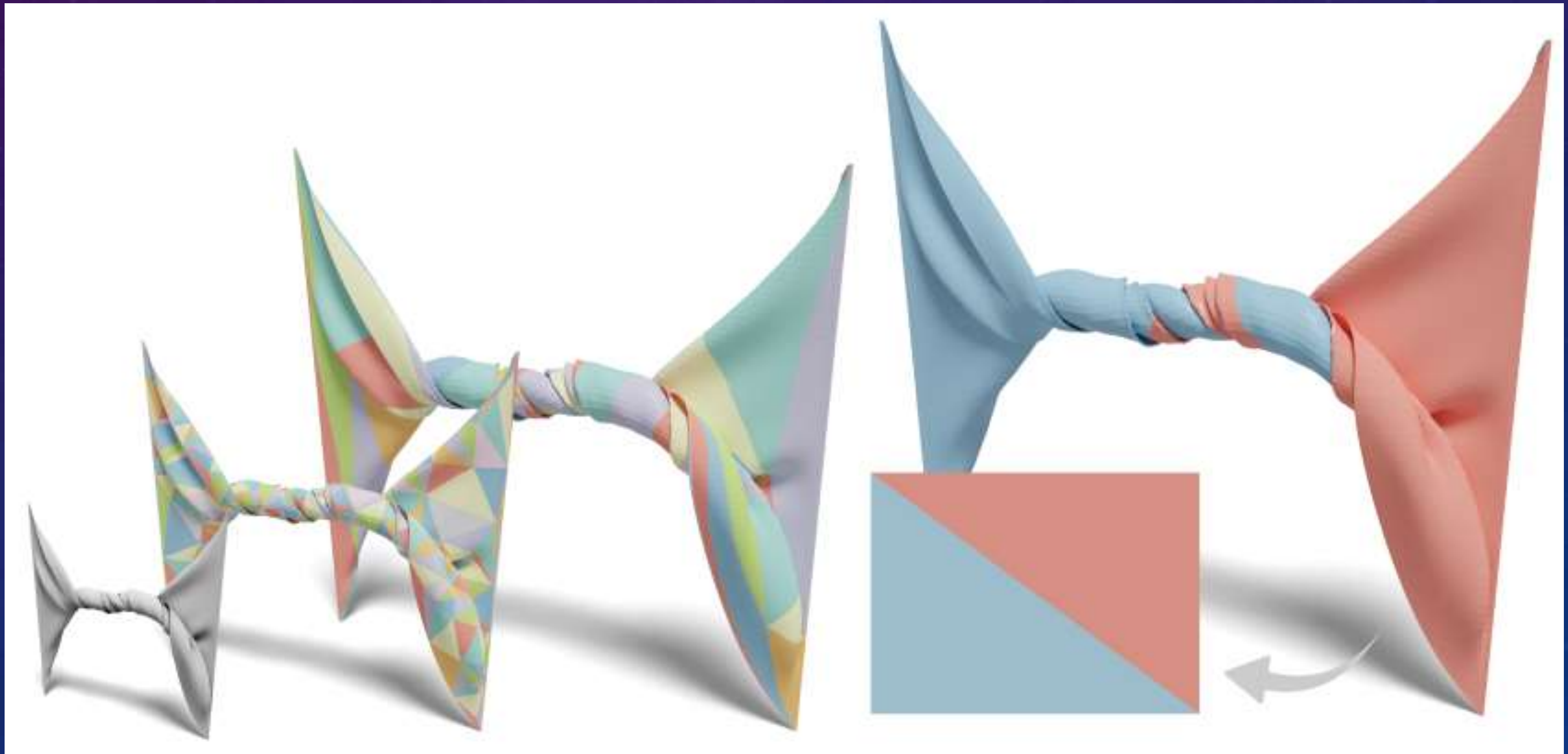
Extensions

Navigating Intrinsic Triangulations



Extensions

Surface Simplification using Intrinsic Error Metrics



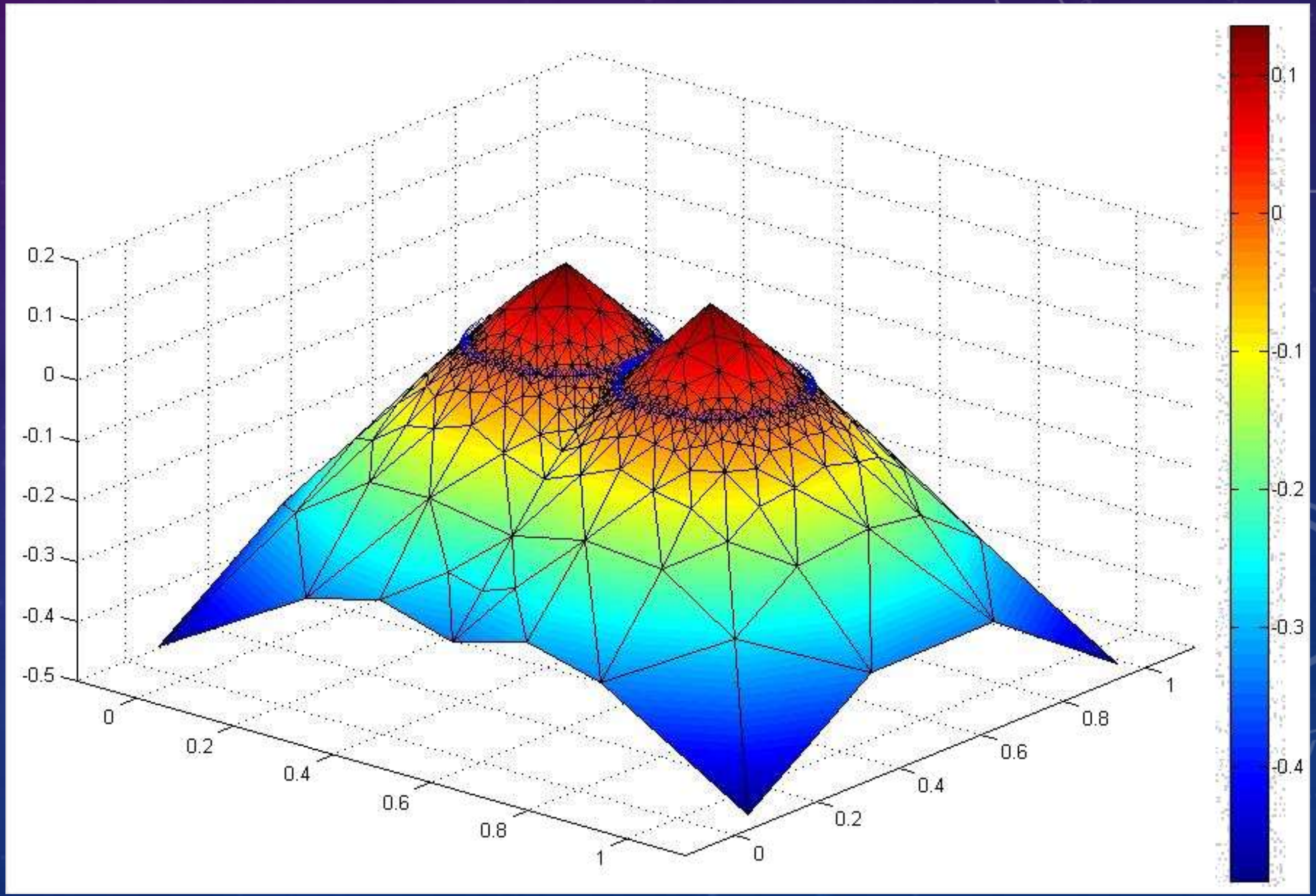
Eikonal equation

Distance like functions:

$$\|\nabla\phi(p)\|_2 = 1, \forall p \in \mathcal{M}$$

For example,

Signed distance function.



Geodesic queries

- Point to point
- Single source
- Multiple Source
- All-Pairs Geodesic Distances

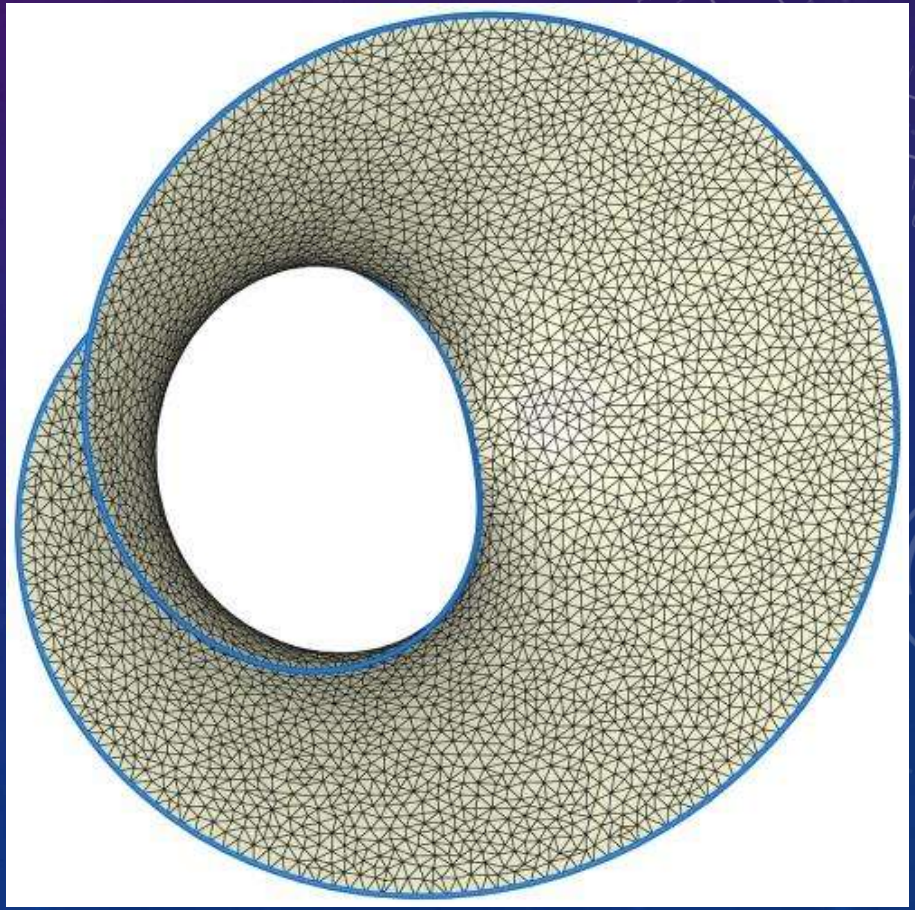
$$f : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$$

Methods

- Fast marching – modified Dijkstra’s algorithm
- Heat method – solving PDES
- Optimization-based method

Discrete geodesic distance

- Discrete meshes → graphs



Discrete geodesic distance

- Discrete meshes → graphs
- Approximate geodesics as paths along edges

Dijkstra's algorithm

- Initialize

$$
\begin{aligned}
v_0 &= \text{Source vertex} \\
d(v) &= \text{Current distance to vertex } v \\
S &= \text{Vertices with known optimal distance}
\end{aligned}
$$

Initialization:

$$
\begin{aligned}
d(v_0) &= 0 \\
d(v) &= \infty \; \forall v \in V \setminus \{v_0\} \\
S &= \{\}
\end{aligned}
$$

Dijkstra’s algorithm

- Initialize
- Update

During each iteration, S remains optimal

Complexity:

$$O(|E| + |V| \log |V|)$$

$$v_0 = \text{Source vertex}$$

$$d(v) = \text{Current distance to vertex } v$$

$$S = \text{Vertices with known optimal distance}$$

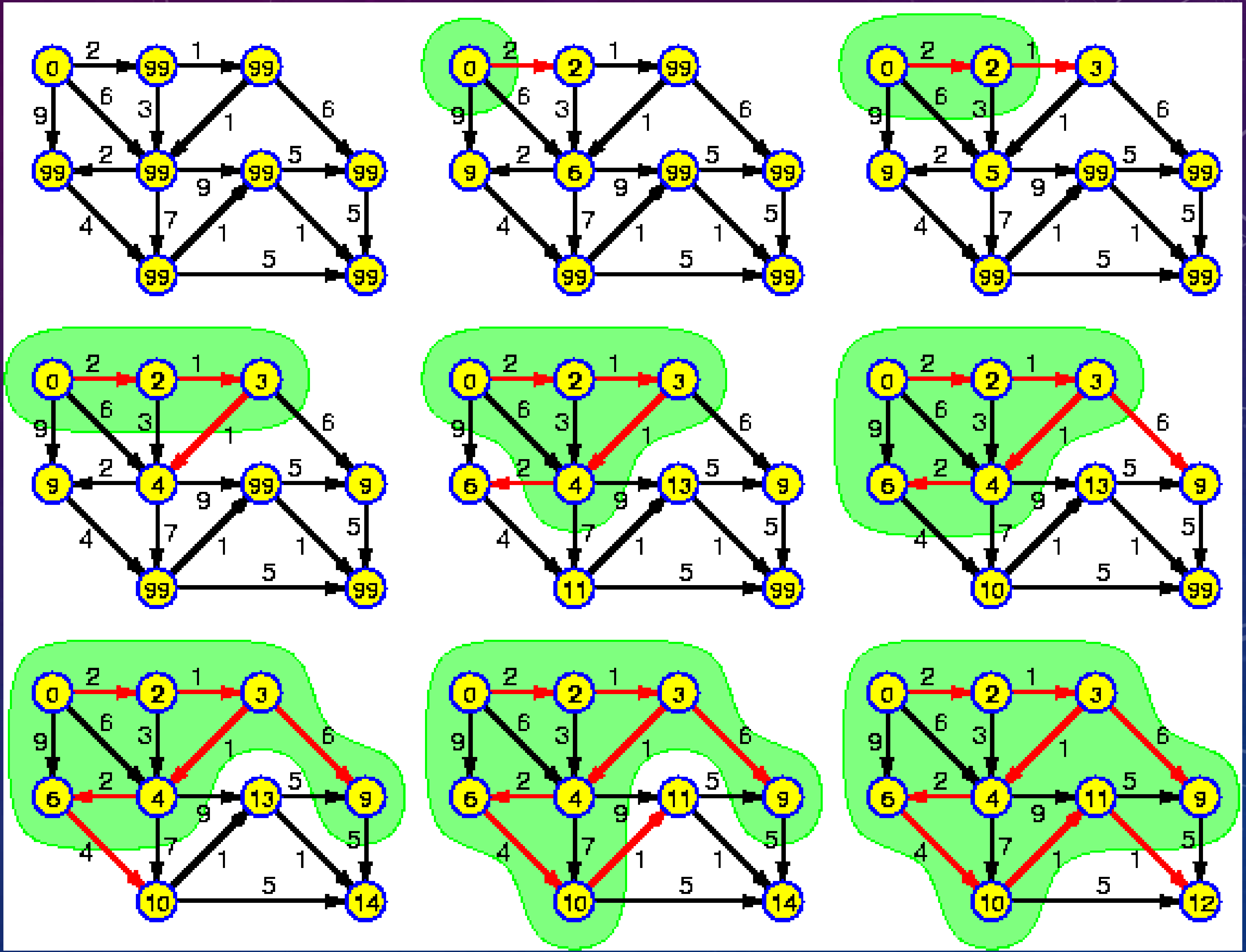
Iteration *k*:

$$v = \arg \min_{v \in V \setminus S} d(v)$$

$$S \leftarrow S \cup \{v\}$$

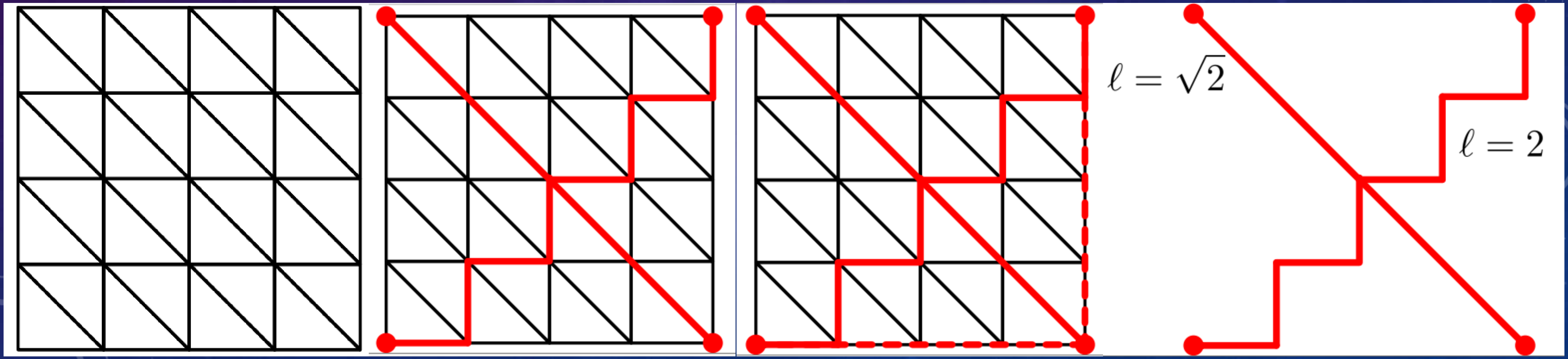
$$d(u) \leftarrow \min\{d(u), d(v) + w(e)\} \ \forall e = (u, v) \in E$$

Example



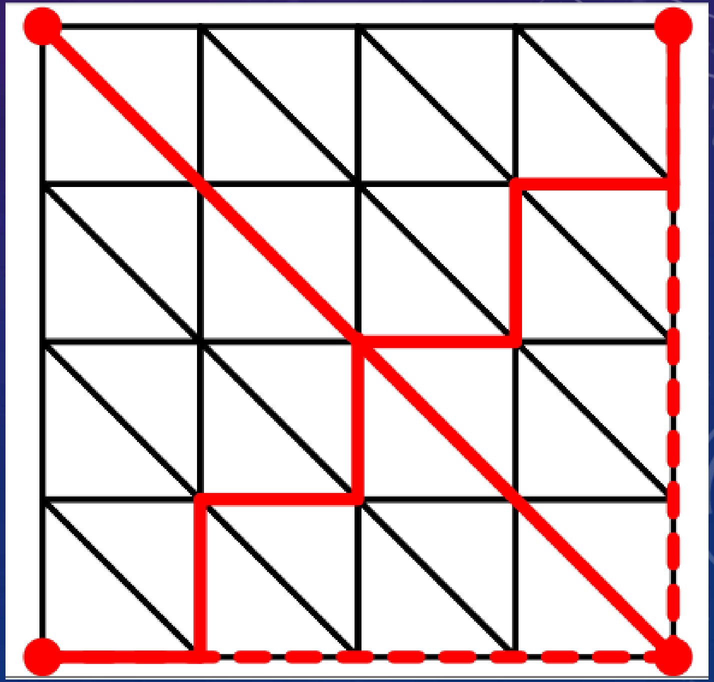
Discrete geodesic distance

- Discrete meshes → graphs
- Approximate geodesics as paths along edges



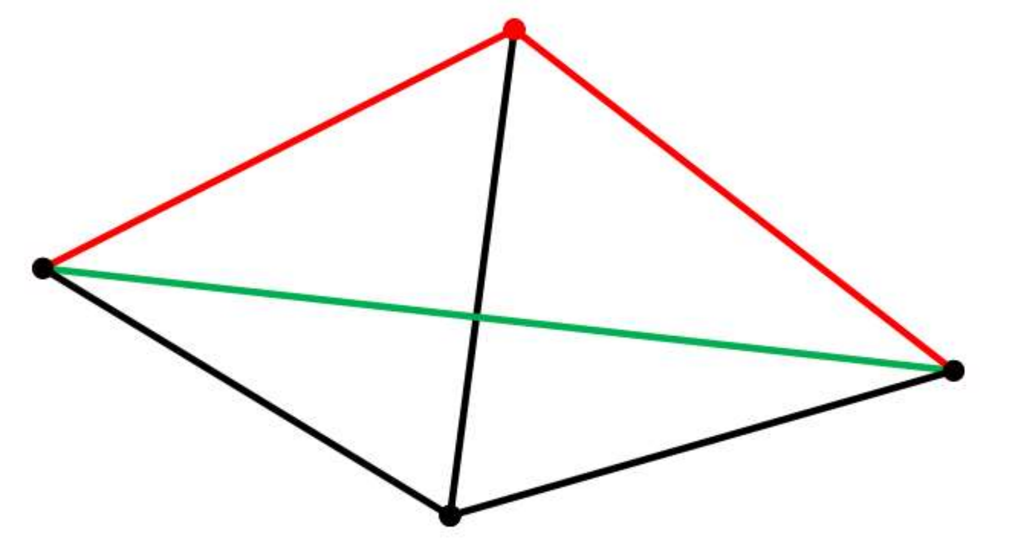
Discrete geodesic distance

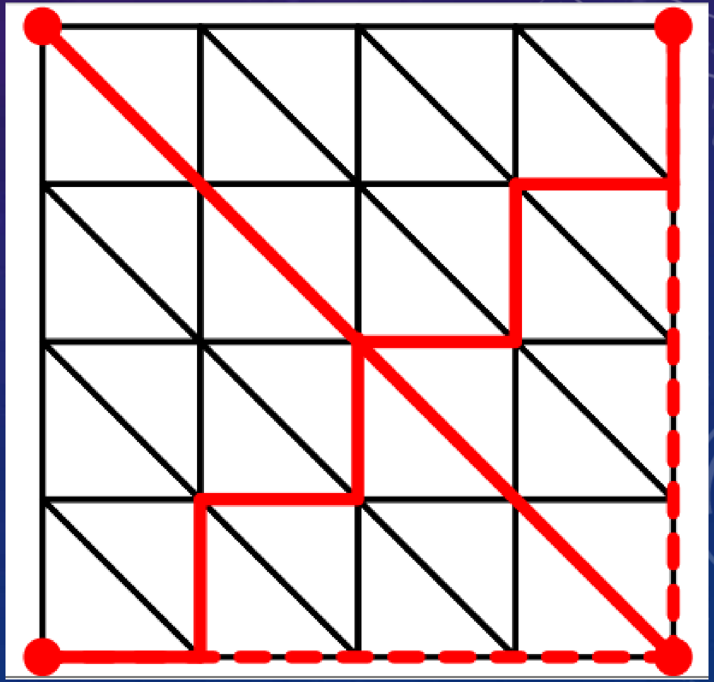
- Discrete meshes → graphs
- Approximate geodesics as paths along edges
  - Asymmetric
  - Anisotropic
  - May not improve under refinement



Discrete geodesic distance

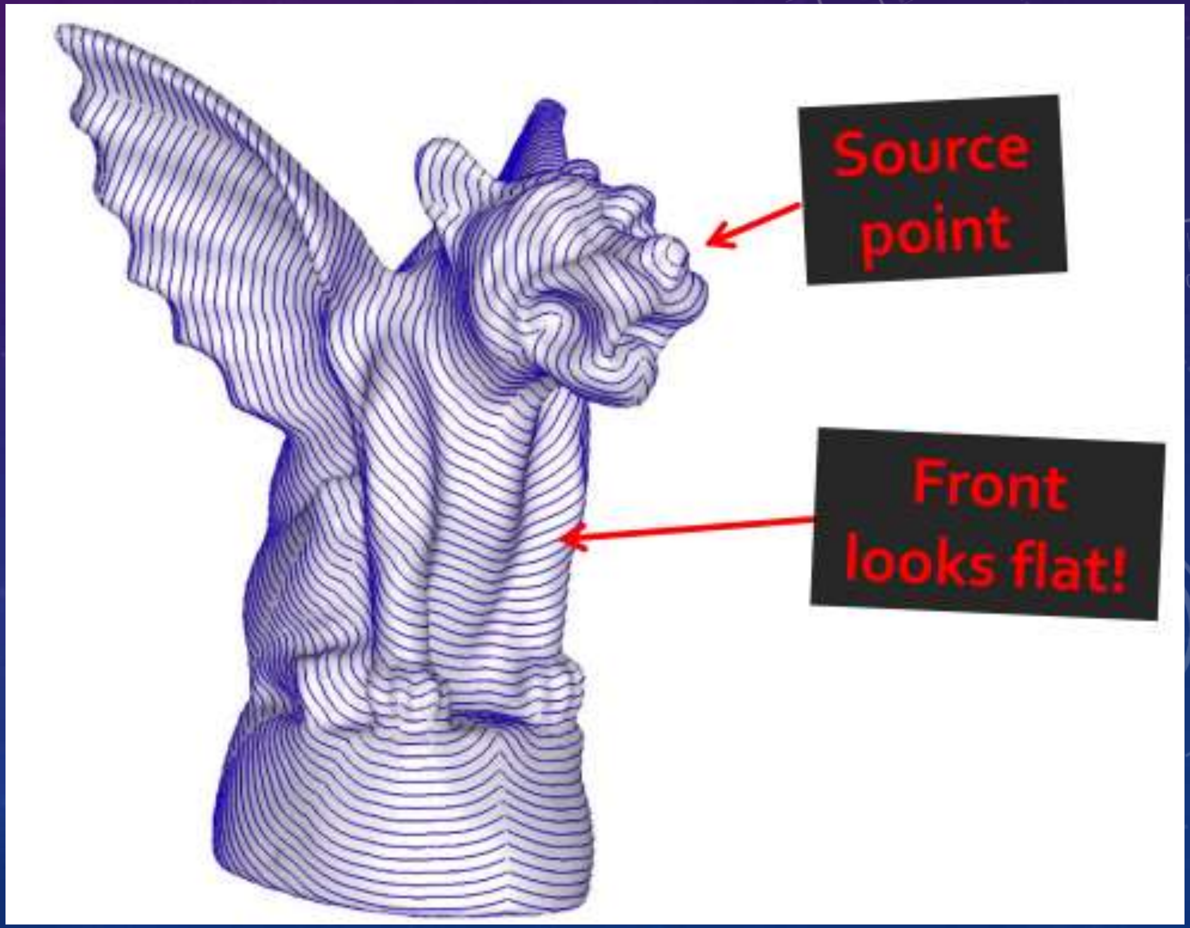
- Graph shortest-path does not converge to geodesic distance.
- Geodesic distances need special discretization.





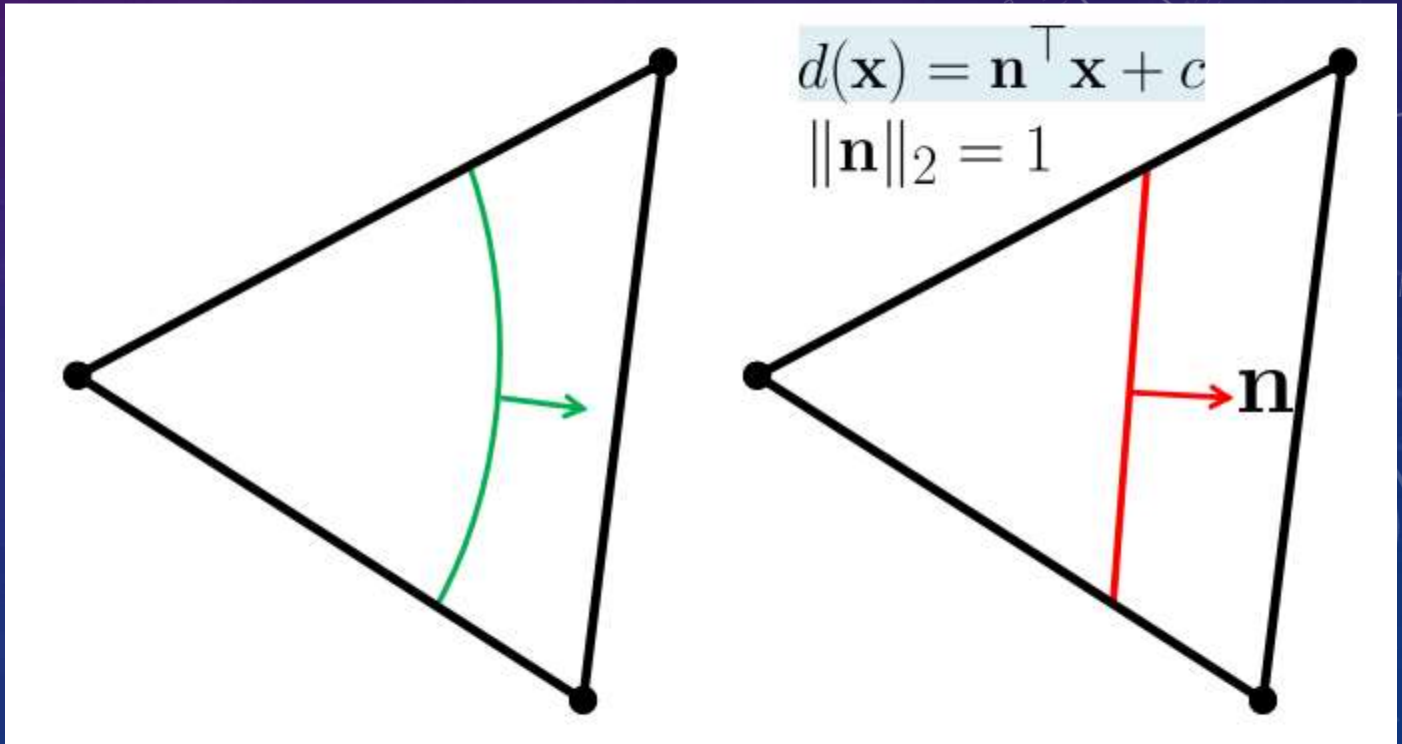
Motivation

- On a triangle mesh, level sets of the (true) distance function are composed of lots of small circles
- Less and less curved the farther we move from the source point.



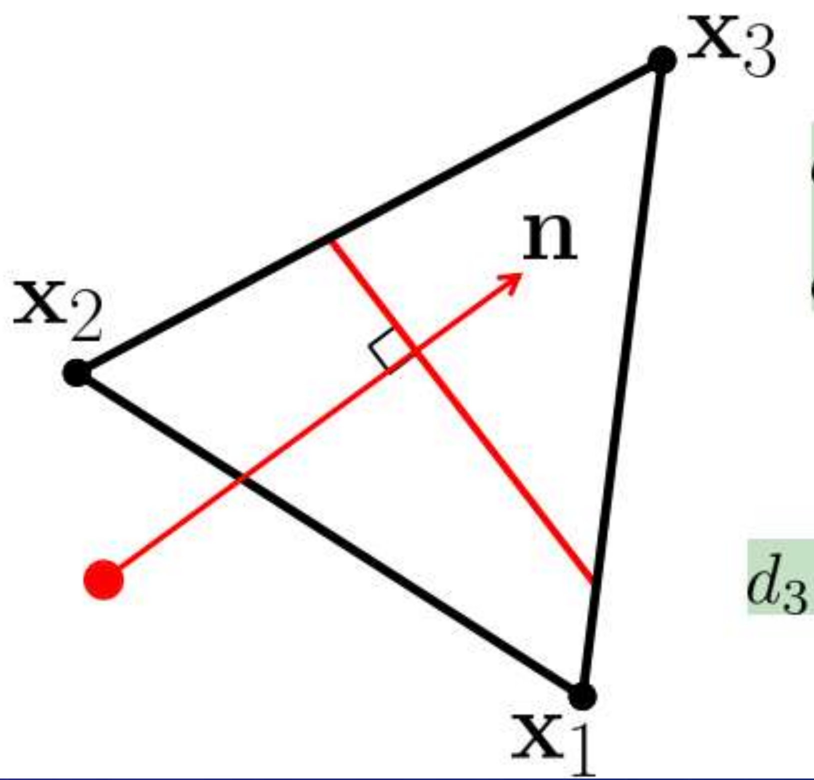
Approximation

Approximate geodesics by assuming that within a single triangle the distance function is well approximated by one whose level sets are straight lines



Planar calculation

- $\|n\| = 1$
- Solving quadratic in c



Given:

$$d_1 = \mathbf{n}^\top \mathbf{x}_1 + c$$

$$d_2 = \mathbf{n}^\top \mathbf{x}_2 + c$$

$$\mathbf{d} = X^\top \mathbf{n} + c\mathbf{1}$$

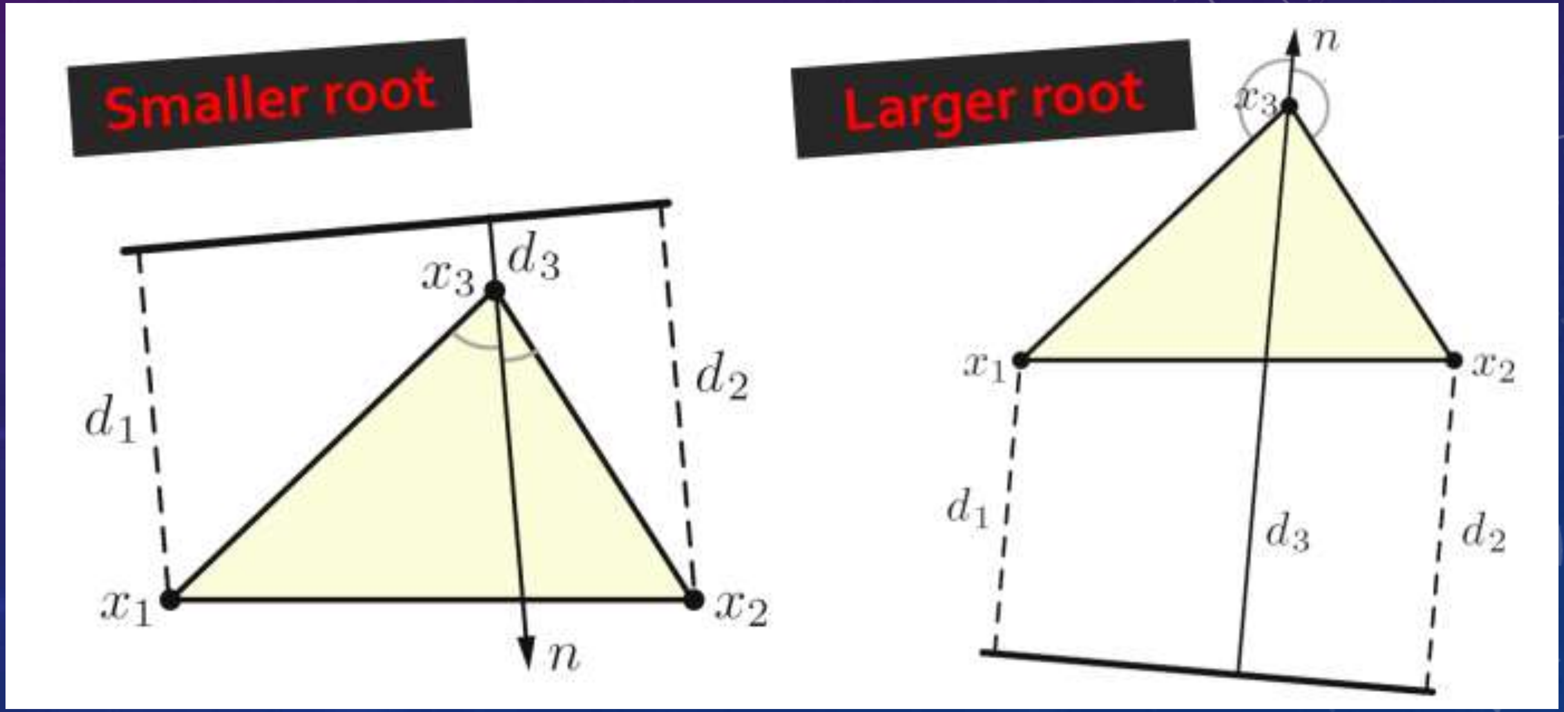
Find:

$$d_3 = \mathbf{n}^\top \cancelto{0}{\mathbf{x}_3} + c = c$$

$$\begin{aligned} 1 &= \mathbf{n}^\top \mathbf{n} \\ &= (\mathbf{d} - c\mathbf{1})^\top X^{-1} X^{-\top} (\mathbf{d} - c\mathbf{1}) \\ &= [\mathbf{1}^\top (X^\top X)^{-1} \mathbf{1}] c^2 + [-2\mathbf{1}^\top (X^\top X)^{-1} \mathbf{d}] c + [\mathbf{d}^\top (X^\top X)^{-1} \mathbf{d}]. \end{aligned}$$

Two roots

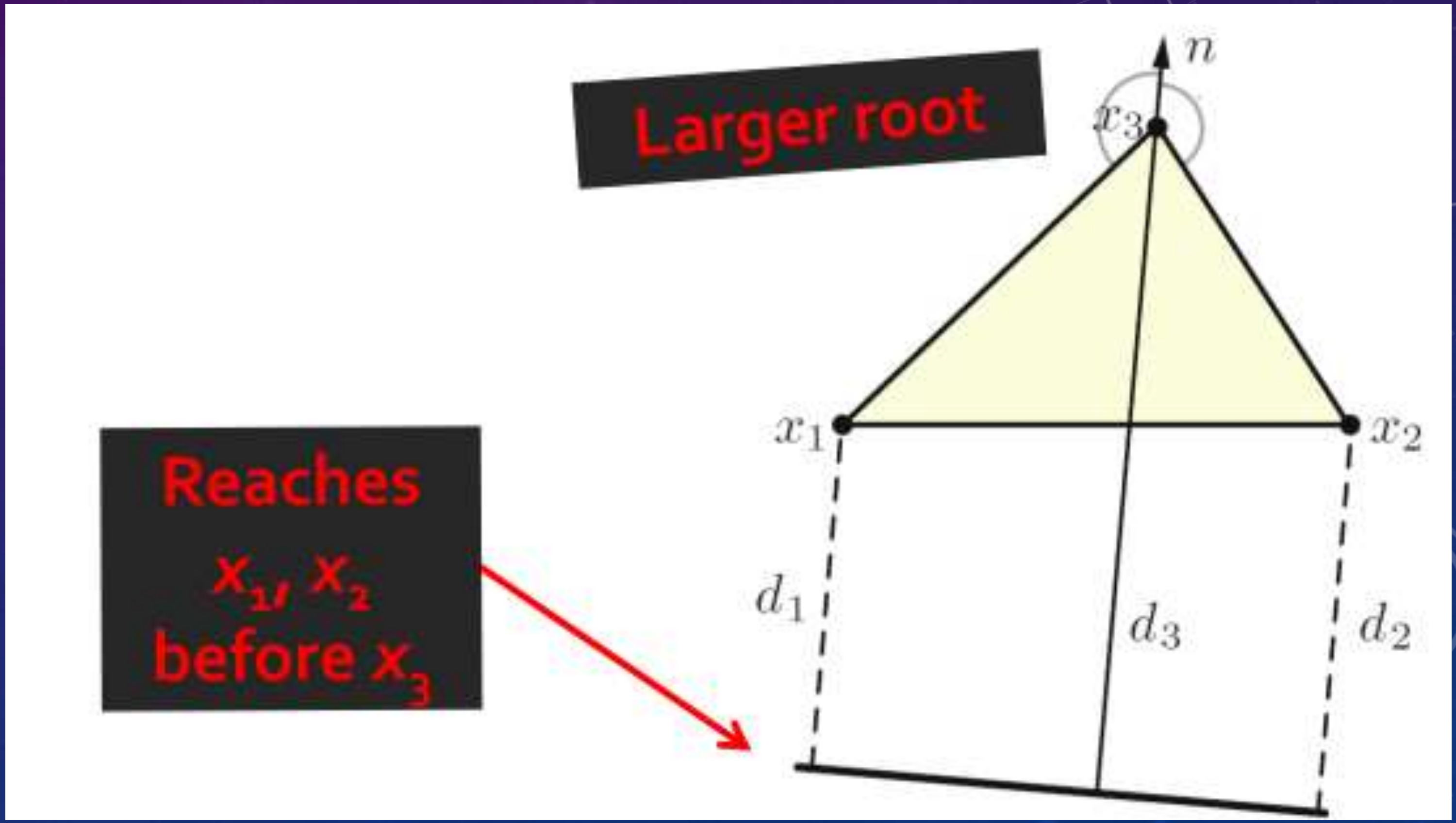
- Two orientations for the normal



Two roots

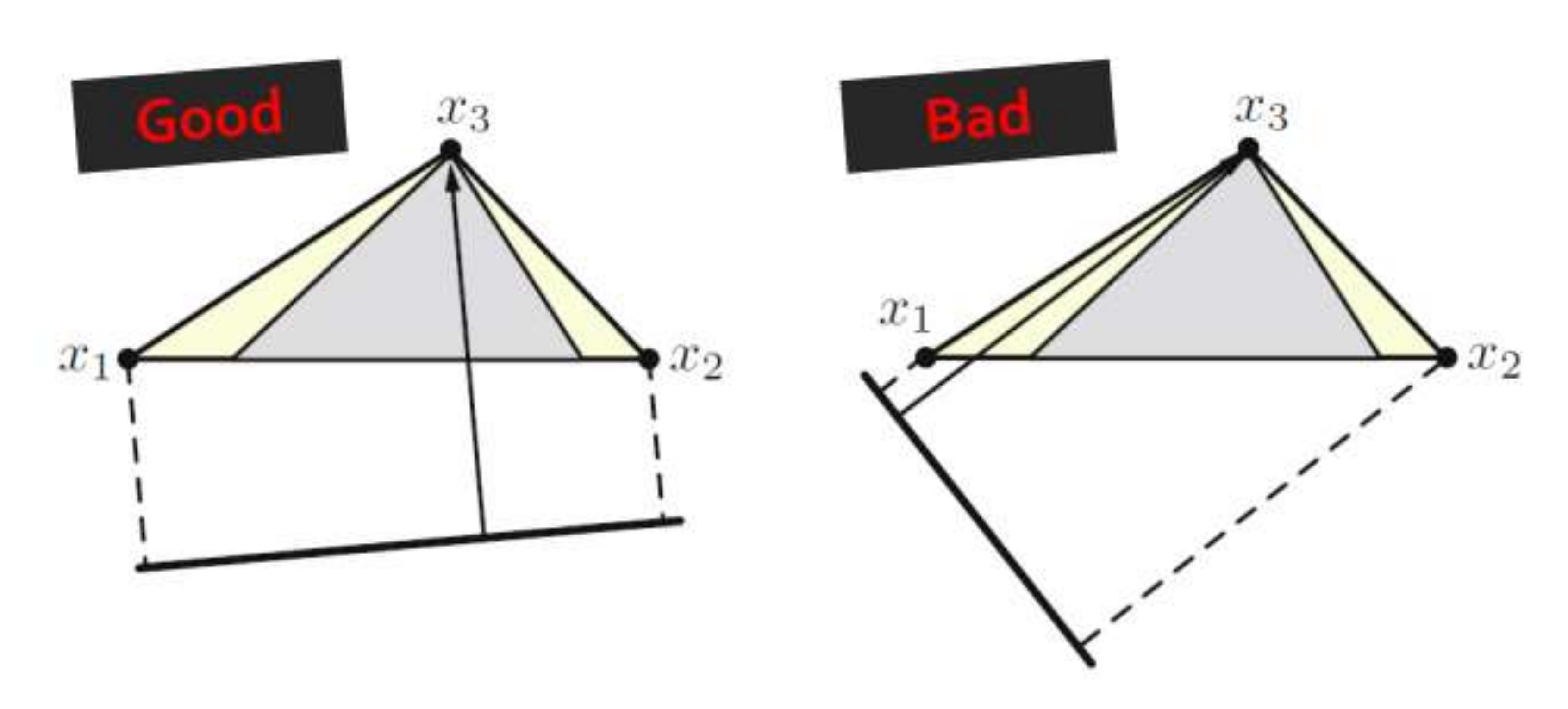
- Two orientations for the normal
- Front from outside the triangle

$d_3 \geq \max(d_1, d_2)$



Additional issue

- Obtuse triangles



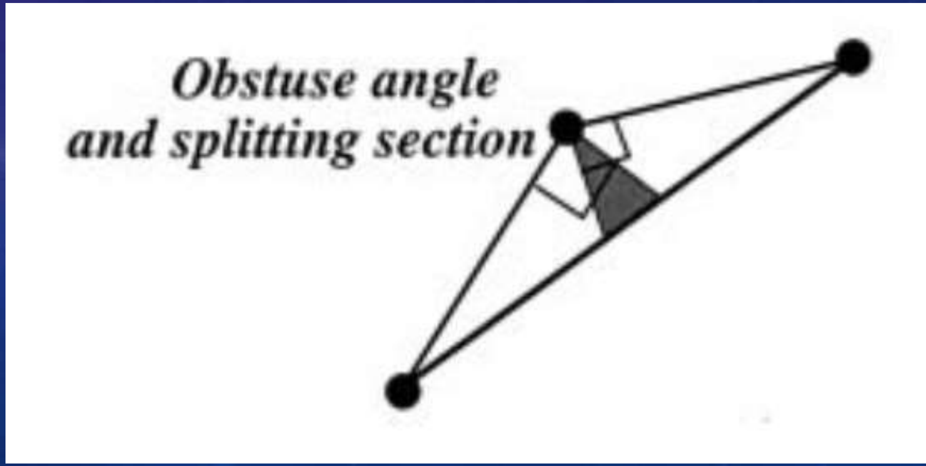
Fixing the issue

- Alternative edge-based update:

$$d_3 \leftarrow \min\{d_3, d_1 + \|x_3 - x_1\|, d_2 + \|x_2 - x_1\|\}$$

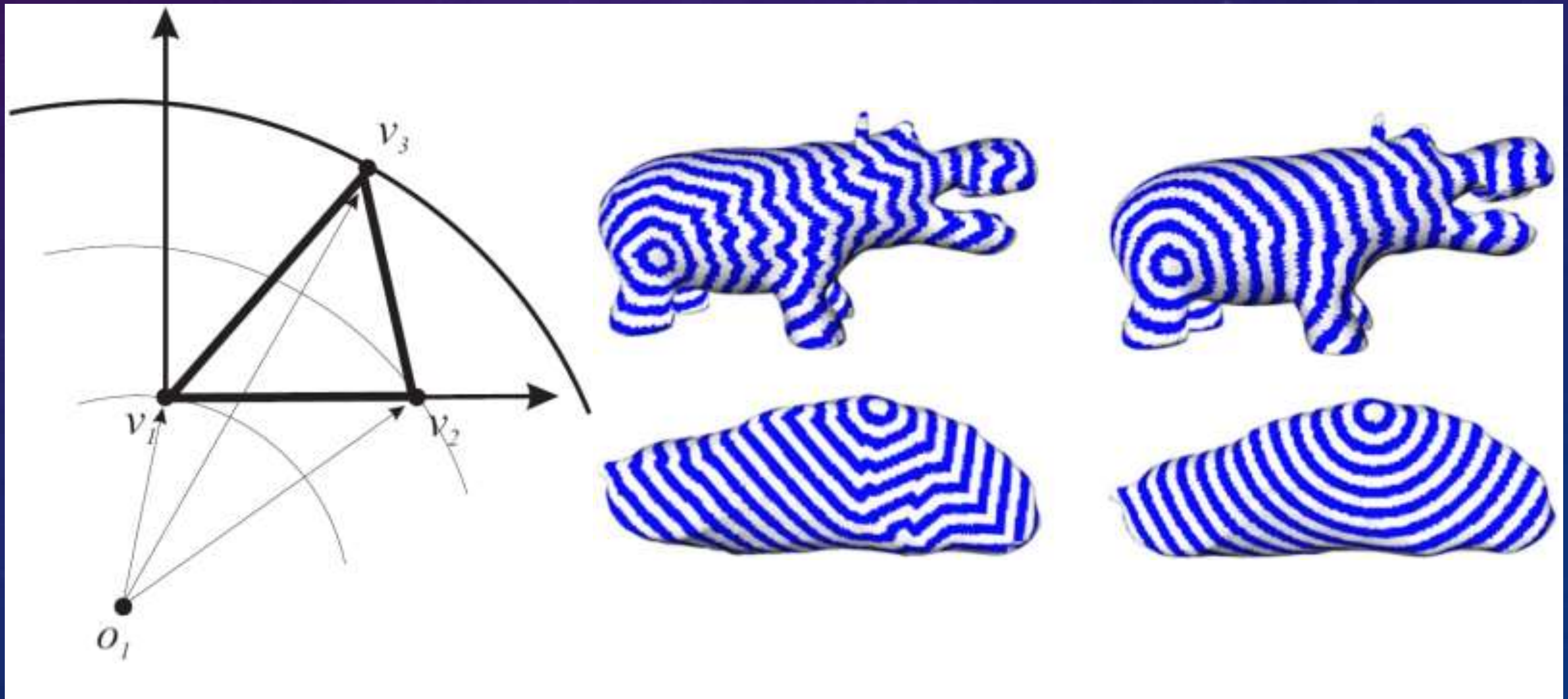
- Add connections as needed

  [Kimmel and Sethian 1998]



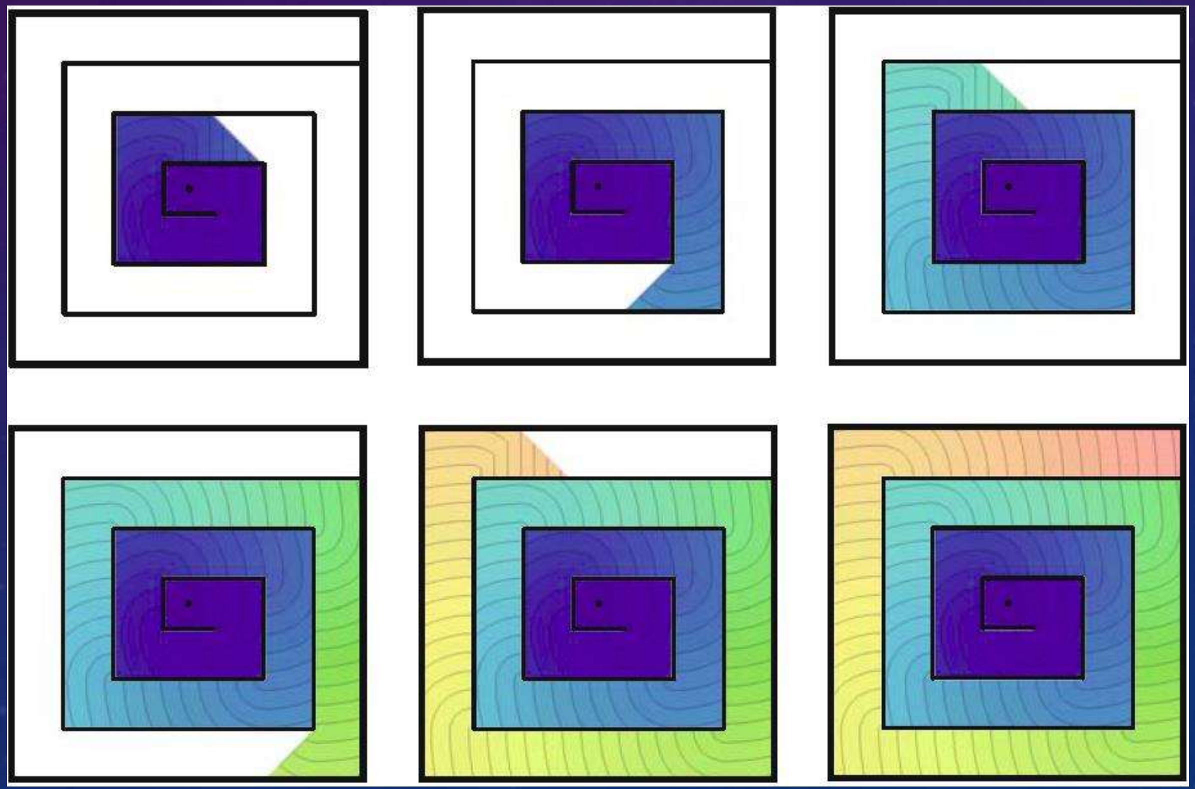
Modifying fast marching

- Circular wavefront [Novotni and Klein 2002]



Modifying fast marching

- Grids and parameterized surfaces [Novotni and Klein 2002]



Raster scan and/or parallelize

Methods

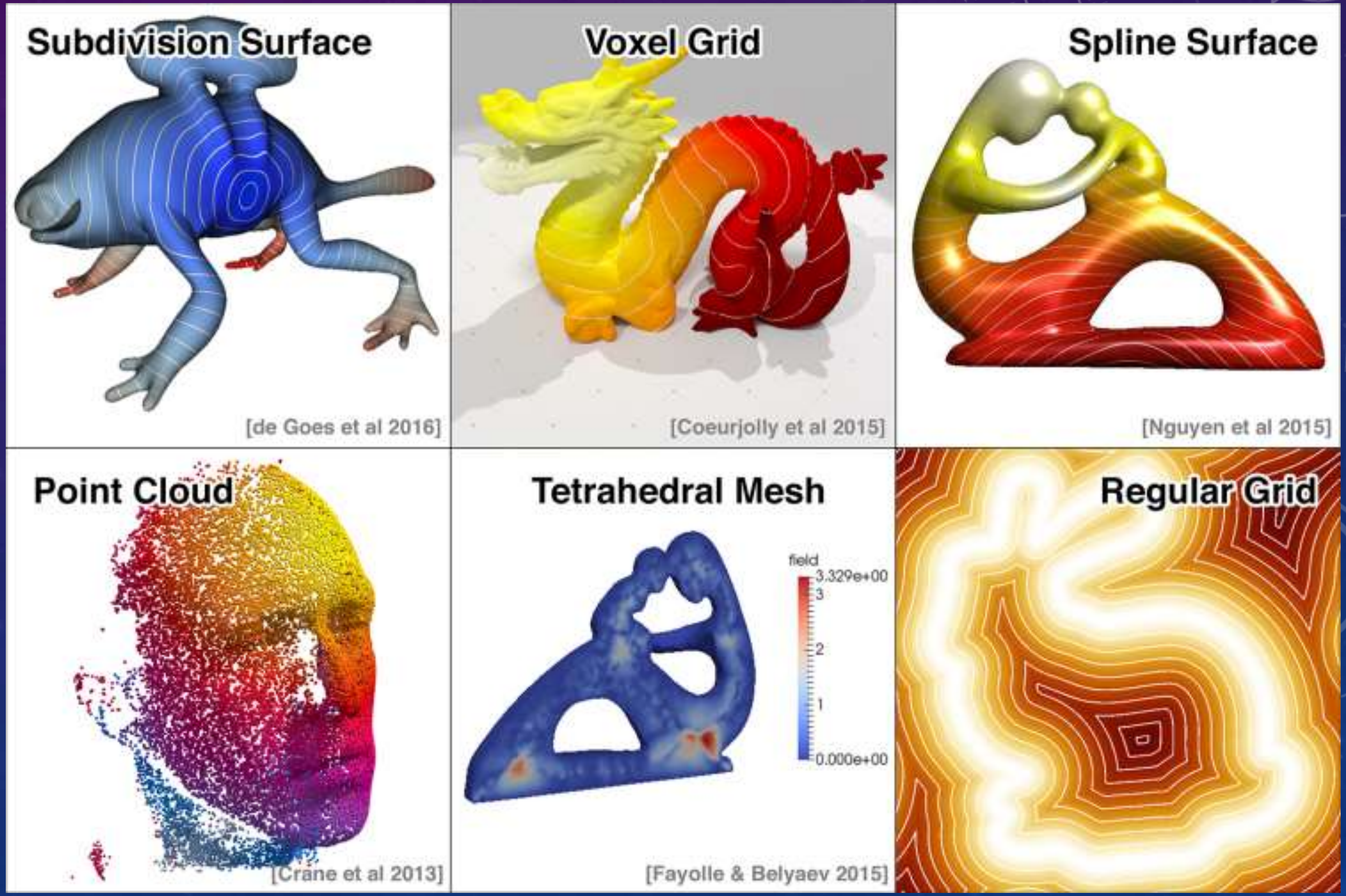
- Fast marching – modified Dijkstra’s algorithm
- Heat method – solving PDES
- Optimization-based method

Heat method

The heat method is a general principle that can be applied to any geometric data structure, as long as one knows how to take the gradient of a scalar function.

Applied data structure

- Subdivision surfaces
- Voxel grids
- Spline surfaces
- Point clouds
- Tetrahedral meshes
- Regular grids



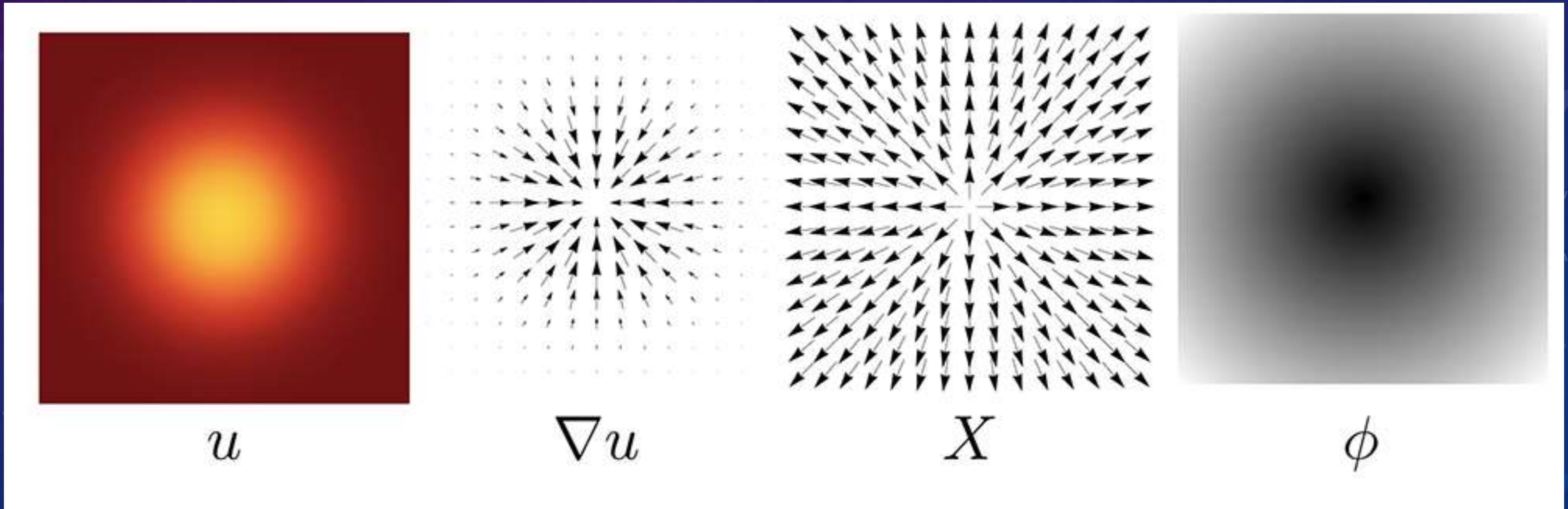
Motivation

- Eikonal equation $||\nabla\phi(p)||_2 = 1, \forall p \in \mathcal{M}$
- Recover u from gradient field $\min_u \int ||\nabla\phi - \vec{g}||^2 \to \Delta\phi = \nabla \cdot \vec{g}$
- How to estimate gradient $\vec{g}$? – from heat transfer in a short time

$$\frac{\partial u}{\partial t} = \Delta u \Longrightarrow \nabla u(t) \parallel \nabla\phi, as\ t \to 0$$

Outline

Heat diffusion u in a brief period of time. The temperature gradient ∇u is normalized and negated to be X. Recovering distance ϕ from gradient.



Temporal discretization

- Heat equation

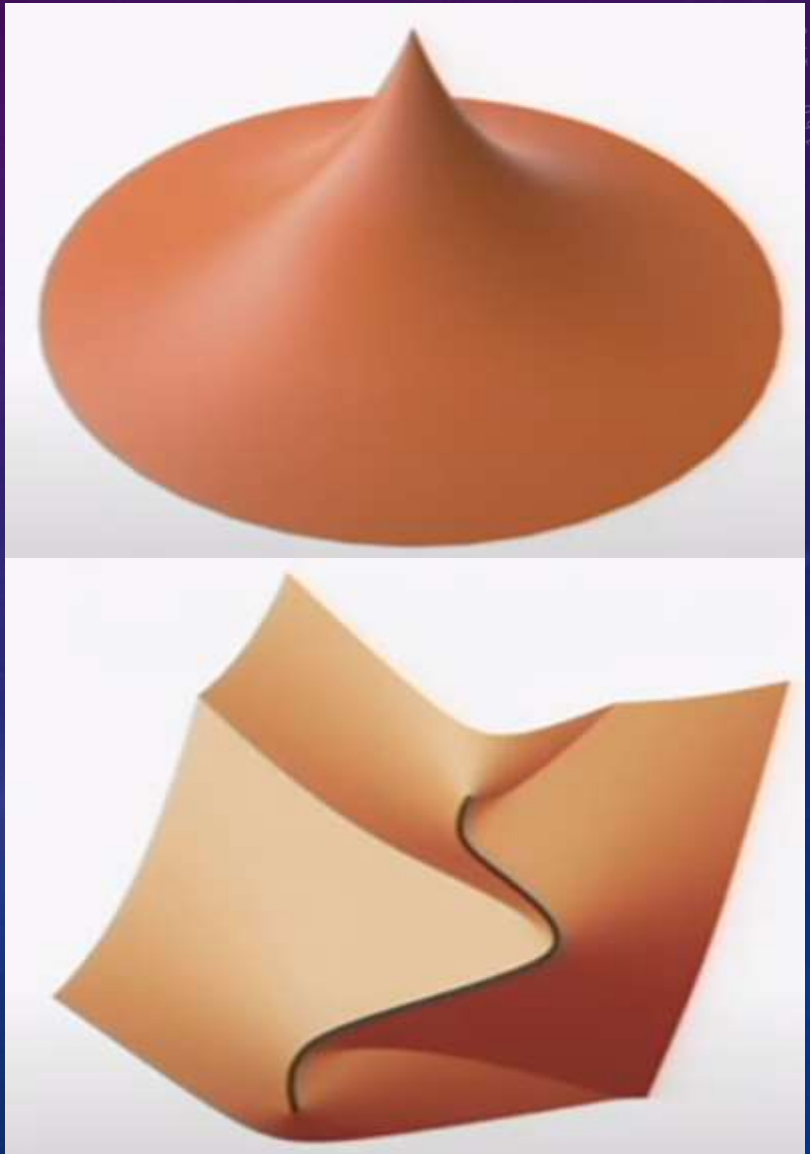
$$\frac{\partial u}{\partial t} = \Delta u$$

- Backward Euler

$$\frac{u_t - u_0}{t} = \Delta u_t$$

- Linear elliptic equation

$$(\mathrm{id} - t\Delta)u_t = u_0$$

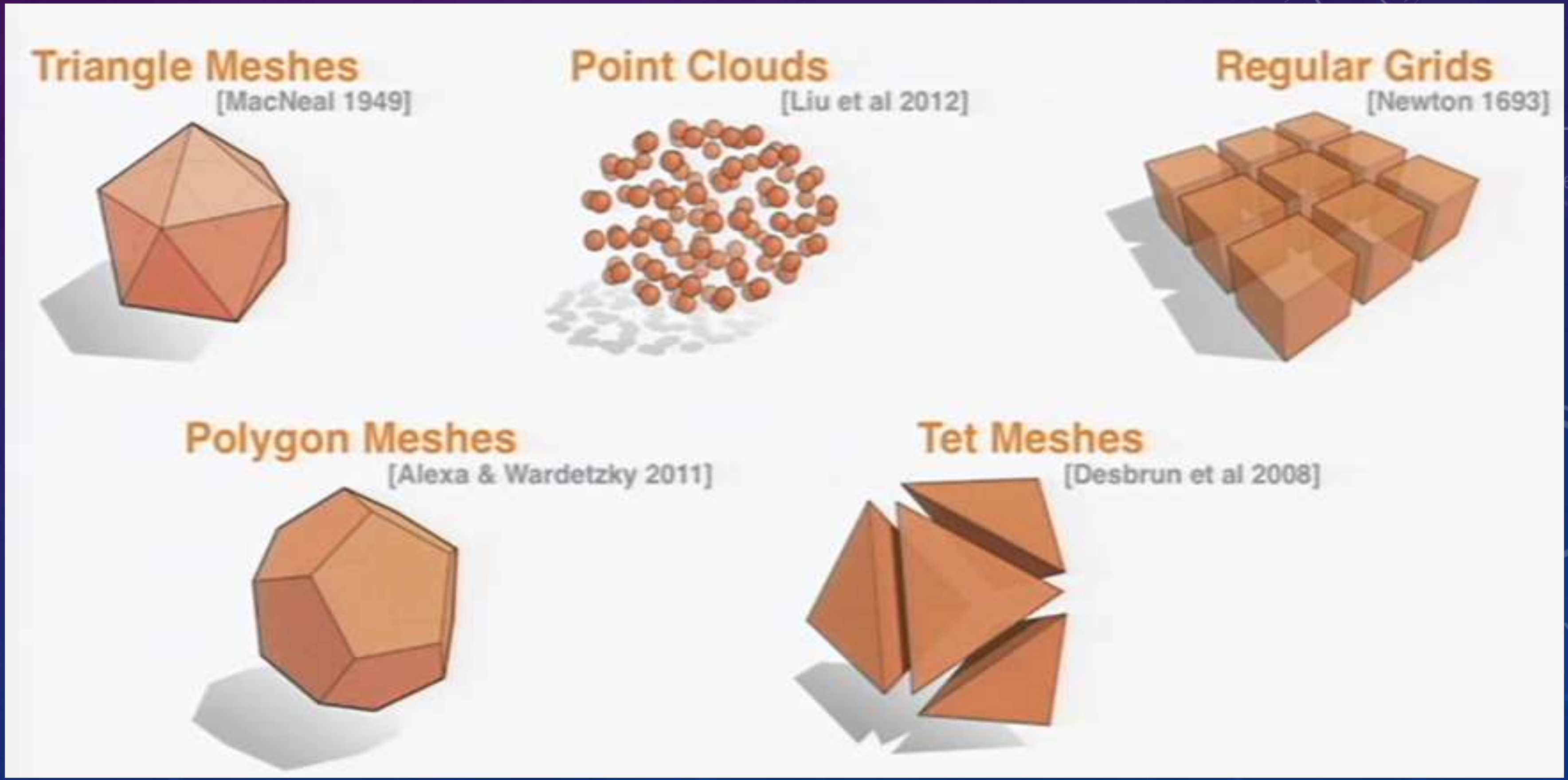


$$u\Big|_{t=0} = \delta(p)$$

$$u\Big|_{t=0} = \delta(\gamma)$$

Spatial discretization

$\Delta \rightarrow L$



Comparison

Table 1. Comparison with fast marching and exact polyhedral distance

		Heat method				Fast marching			
Model	**Triangles**	**Precompute (s)**	**Solve**	**Max error (%)**	**Mean error (%)**	**Time (s)**	**Max error (%)**	**Mean error (%)**	**Exact time (s)**
Bunny	28k	**0.21**	**0.01s** (28x)	3.22	**1.12**	0.28	**1.06**	1.15	0.95
Isis	93k	**0.73**	**0.05s** (21x)	1.19	**0.55**	1.06	**0.60**	0.76	5.61
Horse	96k	**0.74**	**0.05s** (20x)	1.18	**0.42**	1.00	**0.74**	0.66	6.42
Kitten	106k	**1.13**	**0.06s** (22x)	0.78	**0.43**	1.29	**0.47**	0.55	11.18
Bimba	149k	**1.79**	**0.09s** (29x)	1.92	0.73	2.62	**0.63**	**0.69**	13.55
Aphrodite	205k	**2.66**	**0.12s** (47x)	1.20	**0.46**	5.58	**0.58**	0.59	25.74
Lion	353k	**5.25**	**0.24s** (24x)	1.92	0.84	10.92	**0.68**	**0.67**	22.33
Ramses	1.6M	**63.4**	**1.45s** (68x)	0.49	**0.24**	98.11	**0.29**	0.35	268.87

Best speed/accuracy in bold; speedup in orange.

Intrinsic flip

- Fast marching – modified Dijkstra’s algorithm
- Heat method – solving PDES
- Optimization-based method

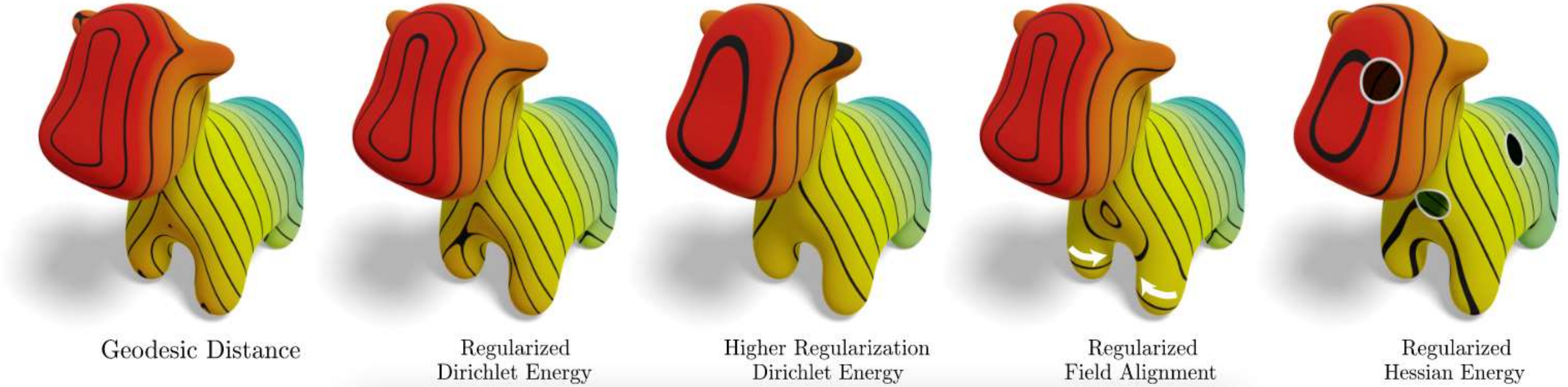
A Convex Optimization Framework for Regularized Geodesic Distances

Michal Edelstein
Technion - Israel Institute of Technology
Haifa, Israel
smichale@cs.technion.ac.il

Nestor Guillen
Texas State University
San Marcos, TX, USA
nestor@txstate.edu

Justin Solomon
Massachusetts Institute of Technology (MIT)
Cambridge, MA, USA
jsolomon@mit.edu

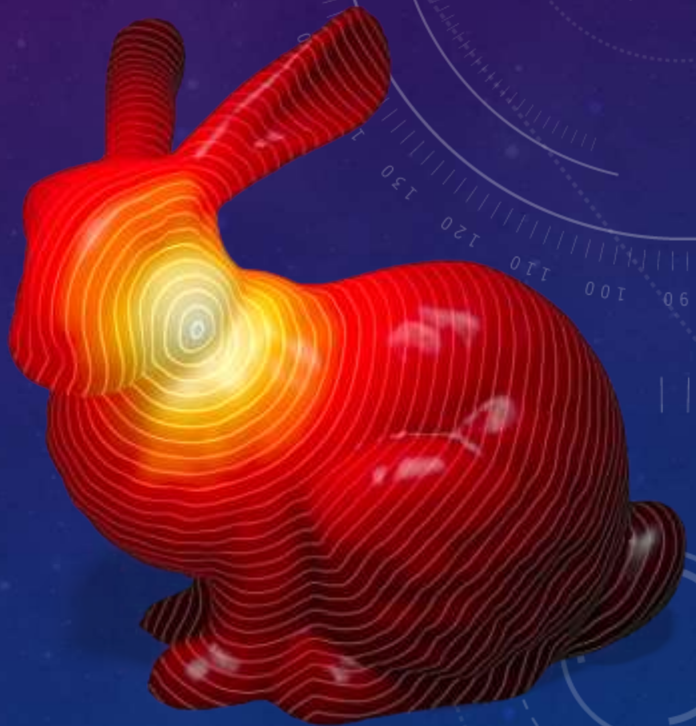
Mirela Ben-Chen
Technion - Israel Institute of Technology
Haifa, Israel
mirela@cs.technion.ac.il



Convex optimization

- Eikonal equation $\|\nabla u(p)\|_2 = 1, \forall p \in \mathcal{M}$
- Maximizing integration of u s.t. $\|\nabla u(p)\|_2 \leq 1$

$$
\begin{aligned}
\text{Minimize}_u \quad & -\int_\Omega u(x)\ \mathrm{dVol}(x) \\
\text{subject to} \quad & |\nabla u(x)| \leq 1 \text{ for all } x \in \Omega \setminus \{x_0\} \\
& u(x_0) = 0.
\end{aligned}
$$



Regularized geodesic distances

- Dirichlet energy :

$$\varepsilon(u) = \int \|\nabla u\|^2 dA$$

$$\boxed{\begin{array}{ll} \text{Minimize}_u & \alpha\mathcal{E}(u) - \int_M u(x)\ \mathrm{dVol}(x) \\ \text{subject to} & |\nabla u(x)| \leq 1 \text{ for all } x \in M \setminus E \\ & u(x) \leq 0 \text{ for all } x \in E. \end{array}}$$

$$\mathcal{E}(u) = \int_M F(\nabla u(x), x)\ \mathrm{dVol}(x),$$

Geodesic Distance (a) | Regularized Dirichlet Energy (b) | Higher Regularization Dirichlet Energy (c)

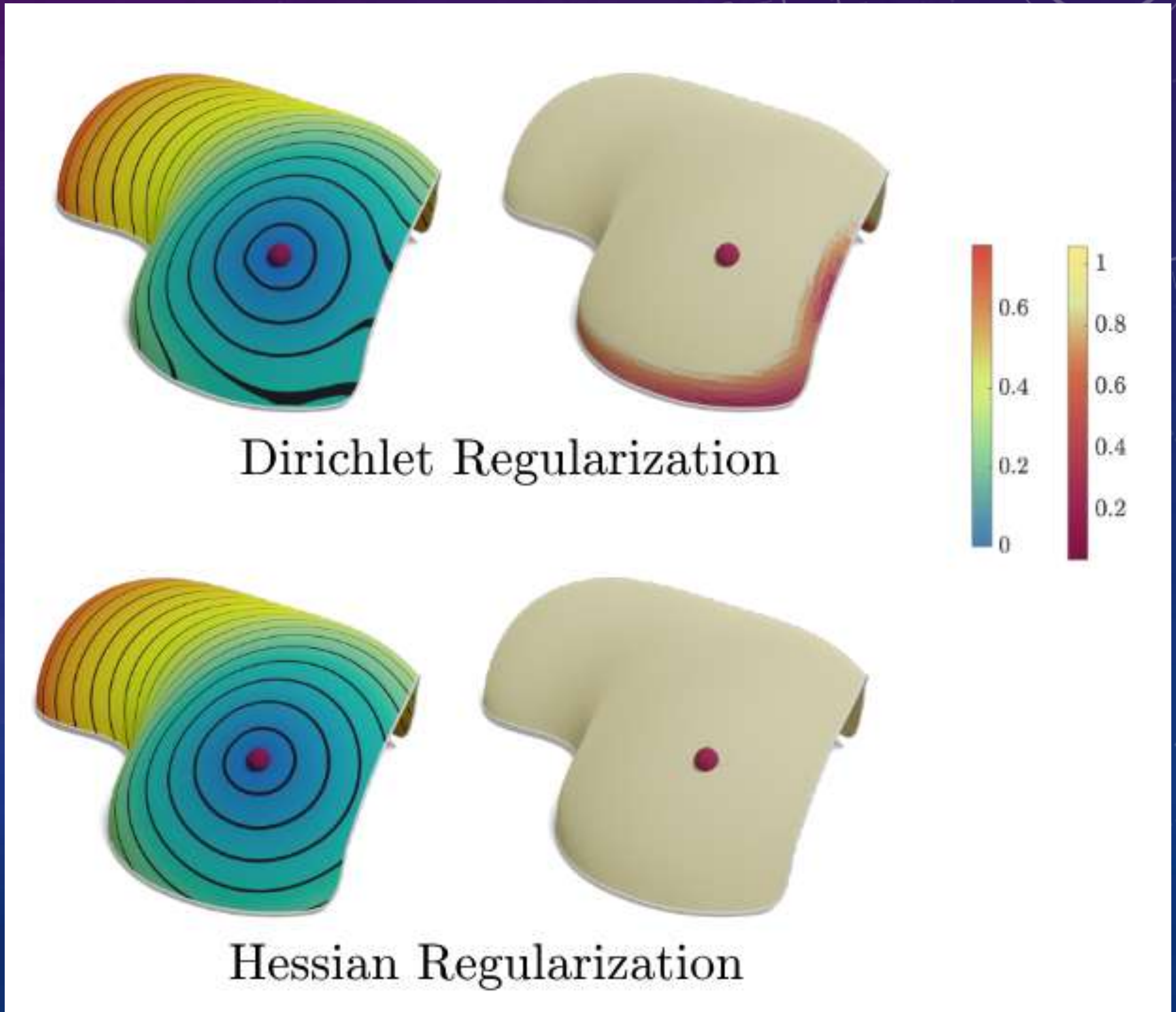
Regularized geodesic distances

- Dirichlet energy :

$$\varepsilon(u) = \int ||\nabla u||^2 dA$$

- Hessian energy:

$$\varepsilon(u) = \int ||\nabla^2 u||^2 dA$$



Theory guarantee

$$
\begin{array}{ll}
\text{Minimize}_u & \alpha\mathcal{E}(u) - \int_M u(x)\,\mathrm{dVol}(x) \\
\text{subject to} & |\nabla u(x)| \le 1 \text{ for all } x \in M \setminus E \\
& u(x) \le 0 \text{ for all } x \in E.
\end{array}
$$

$$\mathcal{E}(u) = \int_M F(\nabla u(x), x)\,\mathrm{dVol}(x),$$

THEOREM 3.1. *There is a unique minimizer for problem* (3).

THEOREM 3.2. *Let u_α denote the minimizer to the optimization problem* (3). *Then, as $\alpha \to 0$*

$$\max_{x \in M} |d(x, E) - u_\alpha(x)| \to 0,$$

where $d(x, E)$ is the geodesic distance from x to the set E.

Spatial discretization

$$
\begin{aligned}
\text{Minimize}_u \quad & -A_{\mathcal{V}}^T u + \alpha F_M(\mathrm{G}u) & \\
\text{subject to} \quad & |(\mathrm{G}u)_f| \le 1 & \text{for all } f \in \mathcal{F} \\
& u_i \le 0 & \text{for all } i \in E,
\end{aligned}
$$

$$
\begin{aligned}
\text{Minimize}_u \quad & -A_{\mathcal{V}}^T u + \tfrac{\alpha}{2} u^T W u & \\
\text{subject to} \quad & |(\mathrm{G}u)_f| \le 1 & \text{for all } f \in \mathcal{F} \\
& u_i \le 0 & \text{for all } i \in E.
\end{aligned}
$$

$$
\begin{aligned}
\text{Minimize}_u \quad & -A_{\mathcal{V}}^T u + \tfrac{1}{2}\alpha u^T W u + \sum_{f \in \mathcal{F}} \chi(|z_f| \le 1) & \\
\text{subject to} \quad & (\mathrm{G}u)_f = z_f & \text{for all } f \in \mathcal{F} \\
& u_i \le 0 & \text{for all } i \in E,
\end{aligned}
$$

ALGORITHM 1: ADMM.

input : M, α, W, E
output : $u \in \mathbb{R}^n$ - distance to E
initialize $\rho \in \mathbb{R}$; `// penalty parameter`
$z \leftarrow \mathbf{0}^{3m}$; `// auxiliary variable, Gu = z`
$y \leftarrow \mathbf{0}^{3m}$; `// dual variable`
$\rho \leftarrow \rho\sqrt{A}$
while *algorithm did not converge* **do** `// See Supp. 8`
solve $(\alpha W + \rho W_D)u = A_{\mathcal{V}} - \mathrm{D}y + \rho \mathrm{D}z$ s.t. $u_E = 0$
$z_f \leftarrow \mathrm{Proj}(\frac{1}{\rho} y_f + (\mathrm{G}u)_f, \mathbb{B}^3)$ for all $f \in \mathcal{F}$
$y \leftarrow y + \rho(\mathrm{G}u - z)$
end

All pairs distance

$$\mathcal{E}_{M\times M}(U) := \frac{1}{2}\int_{M\times M} |\nabla_1 U(x,y)|^2 + |\nabla_2 U(x,y)|^2 \,\mathrm{dVol}(x,y)$$

$$\begin{array}{ll} \text{Minimize}_U & \alpha\mathcal{E}_{M\times M}(U) - \int_{M\times M} U(x,y)\,\mathrm{dVol}(x,y) \\ \text{subject to} & |\nabla_1 U(x,y)| \le 1 \text{ in } \{(x,y) \mid x \ne y\} \\ & |\nabla_2 U(x,y)| \le 1 \text{ in } \{(x,y) \mid x \ne y\} \\ & U(x,y) \le 0 \text{ on } \{(x,y) \mid x = y\} \end{array}$$

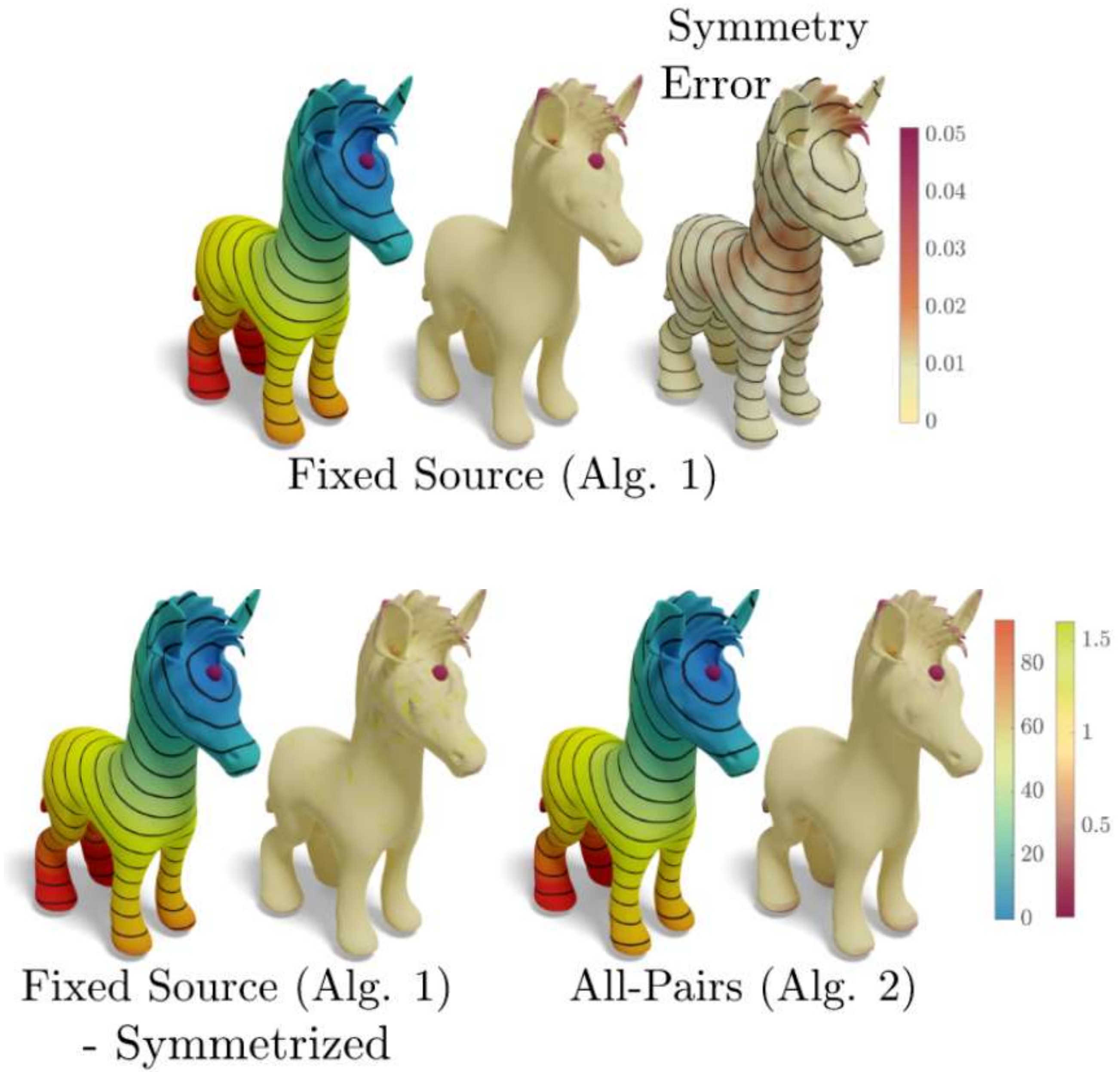
THEOREM 6.2. *The function $U_\alpha(x,y)$ is symmetric in x and y.*

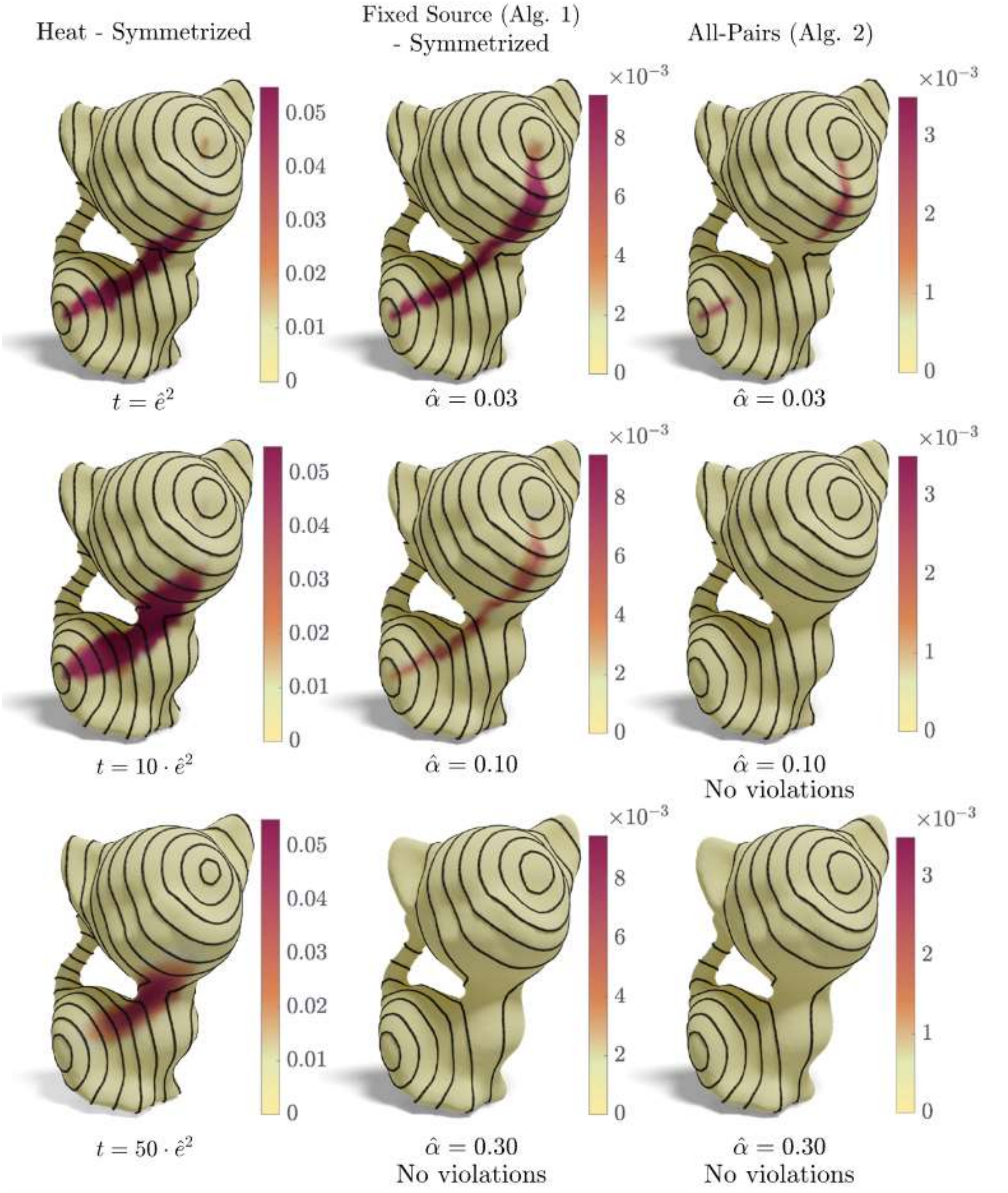
THEOREM 6.3. *As $\alpha \to 0$, we have*

$$\|d(x,y) - U_\alpha(x,y)\|_{L^\infty(M\times M)} \to 0.$$

ALGORITHM 2: Symmetric All-Pairs ADMM.

input : M, α
output: $U \in \mathbb{R}^{n\times n}$; `// dual consensus variable`
initialize $\rho_1, \rho_2 \in \mathbb{R}$; `// penalty parameters`
  $Z, Q \leftarrow \mathbf{0}^{3m\times n}$; `// auxiliary variables GX = Z, GR = Q`
  $Y, S \leftarrow \mathbf{0}^{3m\times n}$; `// dual variables`
  $H, K \leftarrow \mathbf{0}^{n\times n}$; `// dual consensus variables`
  $\rho_1 \leftarrow \rho_1\sqrt{A}, \quad \rho_2 \leftarrow \rho_2\sqrt{A^{-1}}$
  $W_P \leftarrow (\alpha+\rho_1)W_D + \rho_2 M_{\mathcal{V}}, \quad M_P \leftarrow \frac{1}{2}A_{\mathcal{V}}A_{\mathcal{V}}^T M_{\mathcal{V}}^{-1}$
while *algorithm did not converge* **do** `// See Supp. 9`
  solve for X
    $W_P X = M_P - \mathrm{D}Y + \rho_1\mathrm{D}Z - M_{\mathcal{V}}H + \rho_2 M_{\mathcal{V}}U$
  solve for R
    $W_P R = M_P - \mathrm{D}S + \rho_1\mathrm{D}Q - M_{\mathcal{V}}K + \rho_2 M_{\mathcal{V}}U^T$
  $(Z_{(\cdot,i)})_f \leftarrow$
    $\mathrm{Proj}\left(\frac{1}{\rho_1}(Y_{(\cdot,i)})_f + (\mathrm{G}X_{(\cdot,i)})_f, \mathbb{B}^3\right)$ for all $i \in \mathcal{V}, f \in \mathcal{F}$
  $(Q_{(\cdot,i)})_f \leftarrow$
    $\mathrm{Proj}\left(\frac{1}{\rho_1}(S_{(\cdot,i)})_f + (\mathrm{G}R_{(\cdot,i)})_f, \mathbb{B}^3\right)$ for all $i \in \mathcal{V}, f \in \mathcal{F}$
  $U = \max\left(\frac{H+K^T}{2\rho_2} + \frac{X+R^T}{2}, 0\right)$; $\quad U_{i,i} = 0$ for all $i \in \mathcal{V}$
  $Y \leftarrow Y + \rho_1(\mathrm{G}X - Z); \quad S \leftarrow S + \rho_1(\mathrm{G}R - Q)$
  $H \leftarrow H + \rho_2(X - U); \quad K \leftarrow K + \rho_2(R - U^T)$
end


Symmetry Error
0.05
0.04
0.03
0.02
0.01
0
Fixed Source (Alg. 1)
80
60
40
20
0
1.5
1
0.5
Fixed Source (Alg. 1) - Symmetrized
All-Pairs (Alg. 2)


Heat - Symmetrized
Fixed Source (Alg. 1) - Symmetrized
All-Pairs (Alg. 2)
$\times 10^{-3}$
$t = \hat{e}^2$
$\hat{\alpha} = 0.03$
$\hat{\alpha} = 0.03$
$t = 10 \cdot \hat{e}^2$
$\hat{\alpha} = 0.10$
$\hat{\alpha} = 0.10$
No violations
$t = 50 \cdot \hat{e}^2$
$\hat{\alpha} = 0.30$
No violations
$\hat{\alpha} = 0.30$
No violations