**EXTRINSIC**

**INTRINSIC**

edge lengths

connectivity + vertex positions
Overview

I. Yesterday:
   1. Background & history
   2. Data structures I: signposts
   3. Data structures II: normal coords.

II. Today:
   4. Discrete Laplace operator
   5. Geodesics
   6. Open questions
Nonmanifold Laplace Operator
From Manifold to Nonmanifold

• So far, have made a simplifying assumption: all geometry is manifold
  – simplicial case: link of every interior (boundary) vertex is a single loop (path)

• Want to build algorithms that work on all meshes — not just manifold ones!

• Introduce **tufted cover** to extend intrinsic triangulations to non-manifold triangle meshes
  – provides a “bridge” between manifold algorithms & nonmanifold data
  – also extends machinery to *point clouds*…
Sources of Nonmanifold Geometry

“nonmanifold meshes” are what most people would just call *meshes*!

topological defects from scanning/reconstruction

physical interfaces (e.g., Plateau, multiple interacting liquids)

Losasso+ 2006
Central Example: Nonmanifold Laplacian

• For geometry processing, especially interested in **discrete Laplacian** on arbitrary triangle meshes.

• Formal definition of Laplace-Beltrami often assumes domain is a manifold…

• …but many characterizations of Laplacian are perfectly meaningful for non-manifold domains.
  
  – heat diffusion \( \Delta u \equiv \frac{d}{dt} u(t) \bigg|_{t=0} \)
  
  – thickening by \( \varepsilon \)
  
  – deviation from local average
  
  – variation of total area
  
  – …
Basic Problem

• **Recall:** basic approach to improve triangulation (and operators) was to perform intrinsic *edge flips*.

• **Problem:** what does it mean to *flip* a non-manifold edge…?
Tufted Cover

- **Observation:** many objects we want to construct require only manifold edges—but not manifold vertices.

- **Strange idea:** make all edges manifold by making every vertex nonmanifold!
  - make two copies of every triangle, but identify duplicated vertices
  - can then flip edges as usual

![Input](image1)

- “tufted” double cover
- intrinsic edge flip

(tufted upholstery)

(displaced for visualization only!)
Building the Tufted Cover

Step I
make two copies of each face

Step II
glue adjacent copies around shared edge

Note: assumes ordering of faces around edge (usually given by the embedding)
Tufted Laplacian

• Can now build a high-quality Laplacian for any triangle mesh in a straightforward way:
  1. Construct the tufted cover
  2. Flip to intrinsic Delaunay triangulation
  3. Build the usual cotan-Laplace matrix
     – e.g., take sum of per-triangle matrices (times 1/2)
• Since the vertex set is unchanged, resulting matrix defines $V \times V$ Laplace operator for original mesh
  • But edge weights now guaranteed to be positive
  • In practice: provides more accurate results for geometry processing applications
  • (Note: without flips just recover cot-Laplace!)

input: any triangle mesh

build tufted cover

flip to Delaunay

output: high-quality Laplace matrix
Open Question: Discrete Locality?

- What properties does this "tufted Laplacian" \( L \) exhibit?

  ✔ Symmetry. \( L^T = L \)

  ✔ Linear precision. \( Lu = 0 \) for a linear function \( u \) on a planar domain.

  ✔ Positive edge weights. Implies maximum principle (no discrete harmonic function \( Lu = 0 \) has a max/min at an interior vertex).

Locality?

Smooth setting: value of \( \Delta u(x) \) depends on arbitrarily small geometric neighborhood of \( x \) (e.g., geodesic \( \epsilon \)-ball for any \( \epsilon > 0 \)).

Discrete setting [Wardetzky et al 2007]: Value of \( (Lu)_i \) depends only on values of \( u \) at vertices no more than \( k \) edges away in the input triangulation (for universal \( k \in \mathbb{Z} \)).

…but what’s so special about the input triangulation?

Combinatorial distance can be very different from geometric distance!

Delaunay at least guarantees geodesically closest vertex is neighbor…

Open question: what’s a more “geometric” notion of discrete locality?
### Boundary Behavior

Even for **manifold** domains, tufted Laplacian provides better boundary behavior.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>interior</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>boundary</td>
<td>X</td>
<td>X</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Goal:** interpolated values should stay within range of given data.
Example—Nonmanifold Minimal Surfaces

Algorithm [Pinkall & Polthier 1993]:
- Build Laplace matrix $L_{f_k}$ for current mesh
- Solve $L_{f_k}f_{k+1} = 0$ subject to fixed boundary conditions
- Repeat until convergence

robust to low-quality meshes

avoids local extrema
Example—Differential Surface Editing

**Task:** deform shape by interpolating transformations at “control handles.”

(input) (many nonmanifold edges, resulting from mesh booleans)
Effect of Ordering

Why? Define very similar function spaces on original domain.
Tufted Point Cloud Laplacian

• Increasingly popular (e.g., machine learning): point clouds (disconnected points in $\mathbb{R}^3$)

• Several definitions of point cloud Laplacian (e.g., [Belkin et al 2009]):
  + nice convergence properties (pointwise)
  – require dense sampling
  – many nonzeros (#neighbors $\gg 6$)
  – hard-to-tune parameters

• Idea:
  – triangulate local point neighborhoods
  – take union to get nonmanifold mesh
  – build tufted Laplacian

• No parameters; sparsity + accuracy of cotan-Laplace; positive edge weights.
Point Cloud Laplacians—Comparison

[Belkin et al 2008] [Clarenz et al 2004] Tufted Laplacian

underflow

$g < 0$
Can directly apply surface-based algorithms to point clouds:

- spectral conformal parameterization [Mullen et al 2008]
- logarithmic map [Sharp et al 2019]
<table>
<thead>
<tr>
<th>Day I</th>
<th>Day II</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Background</td>
<td>4. Laplacian</td>
</tr>
<tr>
<td>2. Signposts</td>
<td>5. Geodesics</td>
</tr>
<tr>
<td>3. Normal Coords</td>
<td>6. Open Questions</td>
</tr>
</tbody>
</table>
Problem Statement

**GIVEN:** curve on a surface.

**FIND:** locally-shortest path (geodesic) in same isotopy class.

**Notice:** not globally shortest path.
The word “geodesic” often confused with “globally shortest,” but many (most?) applications in geometry processing demand curves beyond the shortest path:

- cut graphs
- geodesic Bézier curves
- segmentation boundaries
- PDE boundary conditions
- constrained Delaunay triangulation
Isotopy is a natural requirement when curves describe, e.g., cuts or region boundaries:

input regions
non-isotopic straightening (regions not well-defined)
isotopic straightening (regions are preserved)
Shortening Curves—First Attempt

Natural idea (e.g., Martínez et al 2005):
- encode curve as sequence of edge crossings
- iteratively shorten, one vertex at a time

• Challenges:
  - hard to guarantee convergence
  - explicit crossings can take $O(n^2)$ storage
  - even in plane, converges only in the limit (less progress on each iteration)
**Observation:** edges of an intrinsic triangulation are geodesic segments.

**Idea:** rather than slide a curve around on a *fixed* triangulation, can we modify the triangulation so that its edges contain the curve we want?
A Greedy Flipping Algorithm for Geodesic Paths

• Leads to remarkably simple approach:
  – **Greedily flip edges** until triangulation contains the desired geodesic path(s)
  – (similar to Lawson’s algorithm)

• Many nice features:
  ‣ configuration space is discrete (flip graph) ⇒ can reach **exact** solution in finitely many steps
  ‣ extremely fast (ms for millions of triangles)
  ‣ like curve-shortening flow, embedded loops remain embedded, shrink to geodesics (or points) in finite time
  ‣ same strategy can be used to compute geodesic loops, curve networks, …
FlipOut In Action—Geodesic Path

11912 edge flips
actual time: 12 ms
FlipOut In Action—Geodesic Loop

127 FlipOut iterations
315 edge flips
actual time: <1 ms
Unified Treatment of Geodesics + Meshing

Key idea: flip-based approach unifies curve processing + surface triangulation.
### Discrete Geodesics

- **Smooth setting:** geodesics are both *straightest* and *locally shortest*
- **Discrete setting:** these two notions diverge [Polthier & Schmies 1998]
- **We seek** *locally shortest* geodesics
  - Same as in classic algorithms (*e.g.*, Mitchell-Mount-Papadimitriou 1987)
  - *Exact* polyhedral geodesics—not, *e.g.*, numerical PDE approximation
Locally Shortest Geodesic

- Consider a curve made by a sequence of edges
- At each vertex $i$, let $\alpha_i^{\min}/\alpha_i^{\max}$ be smaller/larger angle on either side
  - Total angle is $\Theta_i = \alpha_i^{\min} + \alpha_i^{\max}$
  - Curve is locally shortest at $i$ if and only if $\alpha_i^{\min} \geq \pi$

  $\Rightarrow$ Can’t pass through positively-curved vertex ($\Theta_i < 2\pi$)
  $\Rightarrow$ Many ways to pass through negatively-curved vertex ($\Theta_i > 2\pi$)
FlipOut Algorithm—Basic Idea

• Atomic operation in Lawson’s algorithm was to greedily flip a non-Delaunay edge
• Atomic operation in our algorithm will be to greedily flip edges at non-shortest vertices
• Intuitively: pick a vertex, “pull the path as tight as possible” without crossing vertices
More precisely, consider vertex $b$ where $\alpha_b^{\text{min}} < \pi$.

**FlipOut**
- **Input:** consecutive vertices $a$, $b$, $c$ along path
- **while** $\beta_i < \pi$ for any $i \neq a,b,c$ in the wedge boundary:
  - flip first such edge $bi$ (in CCW order)
- **Output:** shorter path along remaining boundary
FlipOut Algorithm—Proof

• **Theorem.** If any edge path \((a,b,c)\) in a \(\Delta\)-complex is not locally shortest (i.e., if \(\alpha_b^{\min} < \pi\)), the FlipOut algorithm will always produce a shorter path.

• **Sketch of proof:**
  1. Any edge that needs to be flipped can be flipped
     - forms a convex pair
     - both endpoints have degree \(> 1\)
  2. Flipping an edge always decreases the number of edges that need to be flipped
  3. Upon termination, wedge boundary is always shorter than initial edge pair

• For brevity, will sketch proof assuming a simplicial complex
  - Extension to \(\Delta\)-complexes is reasonably straightforward, but annoying…
FlipOut Proof (Step 1) — Edges are Flippable

• Suppose $b_i$ is the next edge we need to flip

• The associated edge diamond is always convex:
  – By definition, $\beta_i < \pi$
  – Both $\theta_{i-1}, \theta_{i+1}$ are corners of triangles ($< \pi$)
  – The remaining corner is contained in $\alpha_b^{\min} < \pi$

• In the simplicial case, the endpoints (and all vertices) have degree $> 1$.

• Hence, $b_i$ is flippable.
  ‣ *Additional care is needed in a $\Delta$-complex*
FlipOut Proof (Step 2)—Flipping Decreases Wedge Size

- In the simplicial case, flipping edge $b_i$ always decreases the degree of $b$
  - Why? Because $b, i-1, i, i+1$ are always distinct (“kick out” vertex $i$)
- In $\Delta$-complex, can have edges where both endpoints are incident on $b$, inside wedge.
  - ...everything still works out.
  - Degree always decreases on at least one end, cannot increase on the other.
FlipOut Proof (Step 3)—Final Curve is Shorter

- Algorithm terminates when all remaining angles are $\geq \pi$
- Path along “left” boundary is convex, contained in convex hull of $abc$
- Why must this curve be shorter?
- **Crofton formula**: length of any curve is proportional to expected number of intersections w/ random line
- By convexity, every line that intersects inner path intersects outer path *at least* as many times. Hence, inner path is shorter.
**Path Straightening**

• Can use local straightening procedure to globally shorten curve:

**Path straightening algorithm**

• while curve is not locally shortest:
  – *FlipOut* smallest wedge

• Given a non-crossing curve w/ fixed endpoints, yields a non-crossing geodesic in the same isotopy class (rel. endpoints)

• **Theorem.** Takes a finite number of flips.
  ‣ Crux of proof is to use the fact that there are a finite number of intrinsic triangulations with edge lengths bounded by any fixed constant $L_0$ [Indermitte 2001, Prop 1].

• **In practice:** need very few flips (like Lawson’s algorithm)
Loop Straightening

• Same algorithm works for closed loops
• Finds closed geodesic in same isotopy class
• Q: What happens on convex domain?
• A: Contracts to a “discrete point”, i.e., a loop around a degree-1 vertex.

(*with a small modification near termination)
### Straightening Curve Networks

- In geometry processing, cuts & segmentations often specified as paths of edges.
- Usually, *nothing special about edges!* Just artifact of discretization.
- Replace with smoother, straighter curves by running FLIPOUT to convergence — or just until a user-provided length deviation.
Geodesic Bézier Curves

input
control points

midpoint subdivision
(1 round)

Bézier curve
(4 rounds)

Modeling on triangulations with geodesic curves
Morera, Carvalho, Velho (TVC 2008)
**Intrinsic Constrained Delaunay Triangulation (CDT)**

**constrained Delaunay (CDT):**
- must include given segments
- “maximally Delaunay”
- important for triangulating, e.g., simulation domains

**surface analogue:**
- start w/ paths in isotopy class
- flip to geodesics
- flip to Delaunay & refine

*top image: Jonathan Shewchuk*
PDE-Based Geometry Processing

All these tools (intrinsic CDT, refinement, Béziers, …) makes it easy to construct high-quality meshes for surface-based simulation & processing, including precise boundary conditions.
Single Source Geodesics

- Not main focus (many specialized algorithms…) but interesting to think about

- **Simple algorithm:** apply FlipOut at frontier of Dijkstra search
  
  - get a single triangulation containing all geodesics to source!
  
  - not all globally shortest (95% are; no more than 1.04% error
  
  - intuition: can’t have many “long” geodesics without crossing

- Also get an accurate logarithmic map by just reading off direction of each edge at source

Open Question: strategy to guarantee 100% of curves are globally shortest?
Recurrent Geodesic Paths on Polyhedra

Q: Starting at a vertex of a Platonic solid, can you walk straight and return to where you started without passing through any other vertex?

A: No — except for the dodecahedron.

(re-computed via FLIPOUT)

(...Any other open questions?)
Flip-Based Geodesics — Summary

- **Main takeaway:** you can find exact geodesic paths in triangle meshes by just greedily flipping edges!
  - similar to Delaunay flipping, but constructs a very different (geodesic) triangulation
  - applies to loops, curve networks, etc.
  - easy integration of geodesics & retriangulation boon for geometry processing
- Tons of work on polyhedral geodesic algorithms—especially shortest path problem…
  - given how ridiculously fast algorithms already are, “race to the bottom” on performance makes very little sense!
  - my 2¢: effort better spent on thinking about richer ways to integrate geodesics into broader geometry processing

2 million triangles
9 milliseconds
<table>
<thead>
<tr>
<th>Day I</th>
<th>Day II</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. BACKGROUND</td>
<td>4. LAPLACIAN</td>
</tr>
<tr>
<td>2. SIGNPOSTS</td>
<td>5. GEODESICS</td>
</tr>
<tr>
<td>3. NORMAL COORDS</td>
<td>6. OPEN QUESTIONS</td>
</tr>
</tbody>
</table>

**Open Questions**
What Remains to be Done?

Have come a long way with the intrinsic viewpoint…

- Alexandrov (1942) — embeddability of convex polyhedral metrics
- Regge (1961) — use intrinsic triangulations to model general relativity
- Thurston (1970s) — hyperbolic triangulations, 3-manifolds
- Rivin (1994) — first to consider intrinsic Delaunay triangulations
- Indermitte (2001) — show that intrinsic version of Lawson’s embedding is minimizing

 deformation via intrinsic triangulation

- To build this "black box," need intrinsic data structures that support fairly general geometric computation.

- A brief history of intrinsic data structures:
  - Basic data structure (connectivity + length)
  - Overlay data structure (Fisher et al 2006)
  - Signpost data structure (Sharp et al 2019)

... but we’ve still just scratched the surface!
Intrinsic Triangulations in Higher Dimensions

- Notion of intrinsic triangulation easily extends to \( n \)-dimensional case (topological triangulation w/ simplexwise Euclidean geometry)

- **Intrinsic Delaunay**: locally embed pairs of \( n \)-simplices; circumscribing \( n \)-balls must be empty

- **Fact** (Joe 1989). Can’t always reach a Delaunay configuration via bistellar flips in \( \text{dim} \geq 3 \)

- **Open problem**: *Any* algorithm for intrinsic Delaunay triangulation in \( \text{dim} \geq 3 \).
  - (Do they always exist?)
Isometric Embedding of Nonconvex Metrics

- **Problem:** given discrete metric $\ell : E \rightarrow \mathbb{R}_{>0}$, find vertex coordinates $f : V \rightarrow \mathbb{R}^3$ such that $|f_j - f_i| = \ell_{ij} \forall ij \in E$.

- **Convex:** theory & algorithms fairly well understood [Alexandrov 1942; Bobenko & Izmestiev 2006]

- Algorithms for general (nonconvex) case still not completely satisfactory:
  - Older methods from geometry processing sneak in **extrinsic** data such as dihedral angles, etc.
  - Methods that use purely **intrinsic** data (edge lengths) are unstable, over-regularized, and/or do not provide any hard guarantees

- **Grand challenge:** solve it for real!
Isometric Embedding of Polyhedral Metrics

- **Problem II:** Given intrinsic polyhedral metric, find isometric embedding in \( \mathbb{R}^3 \)

- Why is this any different from **Problem I**?
  - Don't know a priori which intrinsic triangulation admits a linear embedding!
  - E.g., cube can of course be embedded, but not using just *any* triangulation…

- **Open Question:** Is finding the *embeddable triangulation* just as hard as actually finding the embedding itself?
  - Can always cut along all \( O(n^2) \) geodesics…

(image credit: Yousuf Soliman)
List of Open Questions

Many questions seeking good answers:

1. **Intrinsic coarsening**—is there a natural way to remove cone points?
2. **Efficiency of intrinsic flip algorithm**—how to explain empirical behavior?
3. **Vector Rippa Theorem**—does Delaunay minimize some notion of vector Dirichlet energy?
4. **Geodesics**—can the flip algorithm be modified to yield 100% minimal geodesics?
5. **Intrinsic Delaunay triangulation**—any algorithm at all in dimension $\geq 3$?
6. **Discrete Laplacian**—is there a more “geometric” notion of discrete locality?
7. **Isometric embedding**—guaranteed & exact reconstruction of nonconvex metrics?
8. …

Have answers? Want more questions? Let’s talk!
Code & Demos

Give it a try!
Free & open source (MIT license)

http://geometry.cs.cmu.edu/signpost
http://geometry.cs.cmu.edu/flipout
http://geometry.cs.cmu.edu/laplacian
Summary—Intrinsic Triangulations

• Terrific example of success of math & computer science working together
  – deep theory — Riemannian geometry, normal surface theory, hyperbolic geometry,
  – important contemporary problems — e.g., robust algorithms for “awful data”

• Already provides major utility for geometry processing
  – quality guarantees for surface meshing (Delaunay refinement, constrained Delaunay)
  – structural guarantees for differential equations (maximum principle)
  – convergence guarantees for mapping (discrete uniformization)
  – bridge between manifold mesh processing ← nonmanifold meshes, point clouds

• Many open questions remain!
  – coarsening, embedding, 3D triangulation…
Thanks!

GEOMETRY PROCESSING WITH INTRINSIC TRIANGULATIONS

Keenan Crane • Carnegie Mellon University • May 8, 2021