Modélisation géométrique

Pooran MEMARI

Master IMA - Séminaires et pratique en image (PRAT) (SI955)
Janvier 2015
Outline

• Geometric representation methods:
  — Explicit and Implicit

• Curvature:
  - parametric curves
  - parametric surfaces

• Laplacian smoothing
Surface representation

- **Explicit (Parametric)**
  - Represent a surface as a continuous function from a domain in $\mathbb{R}^2$ to $S$ in $\mathbb{R}^3$.

- **Implicit**
  - Represent a surface as the zero set of a distance function defined over $\mathbb{R}^3$. 


Surface representation

• Explicit (Parametric)
  – Represent a surface as continuous function from a domain in R2 to S in R3.

- Global parameterization with a single function: not easy at all!
- Instead, collection of local parameterizations defined over simple 2D domains (charts).
- Smooth manifold if the charts are “smoothly compatible”, i.e. for any two charts \((\phi, U), (\psi, V)\),
  \[\psi \circ \phi^{-1}: \phi(U \cap V) \rightarrow \psi(U \cap V)\] is smooth.
Discrete explicit representation

Triangle meshes : triangulations

• Topology (connectivity):
  Vertices \( \mathcal{V} = \{v_1, \ldots, v_n\} \)
  Edges \( \mathcal{E} = \{e_1, \ldots, e_m\}, \quad e_i \in \mathcal{V} \times \mathcal{V} \)
  Triangles \( \mathcal{F} = \{f_1, \ldots, f_k\}, \quad f_i \in \mathcal{V} \times \mathcal{V} \times \mathcal{V} \)

• Geometry (vertex positions):
  \( \mathcal{P} = \{p_1, \ldots, p_n\}, \quad p_i \in \mathbb{R}^3 \)
Discrete representation: Meshing

- Mesh seen as a partition of a domain in $\mathbb{R}^n$ into cells.
- Two cells are disjoint or share a lower dimensional face (cell complex).
- Examples in R3:
  - 3D tetrahedrization discretizing a volume or
  - 2D triangulation representing a surface.
Topological information: Euler formula

2D triangulation: \( V - E + F - C = 1 \)

- \( F \): number of faces including the exterior face
- \( E \): number of edges
- \( V \): number of vertices
- \( C \): number of components
- \( T \): number of triangles
- \( B \): number of boundary edges
- \( C = 1 \), counting edges in two different ways: \( 2E = 3T + B \)
  - \( E = 3V - 3 - B \)
  - \( T = 2V - 2 - B \)

Closed orientable surface: \( V - E + F = 2 - 2g \)
<table>
<thead>
<tr>
<th>Name</th>
<th>Image</th>
<th>V</th>
<th>E</th>
<th>F</th>
<th>$X = V - E + F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hexahedron or Cube</td>
<td><img src="hexahedron.png" alt="Image" /></td>
<td>8</td>
<td>12</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>Tetrahedron</td>
<td><img src="tetrahedron.png" alt="Image" /></td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Great Icosahedron</td>
<td><img src="icosahedron.png" alt="Image" /></td>
<td>12</td>
<td>30</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>Sphere</td>
<td><img src="sphere.png" alt="Image" /></td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>12</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>18</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>24</td>
<td>16</td>
<td>2</td>
</tr>
<tr>
<td>Interval</td>
<td><img src="interval.png" alt="Image" /></td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Disk</td>
<td><img src="disk.png" alt="Image" /></td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>7</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>9</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Circle</td>
<td><img src="circle.png" alt="Image" /></td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Torus (Product of 2 Circles)</td>
<td><img src="torus.png" alt="Image" /></td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Double Torus</td>
<td><img src="double-torus.png" alt="Image" /></td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>-2</td>
</tr>
<tr>
<td>Triple Torus</td>
<td><img src="triple-torus.png" alt="Image" /></td>
<td>0</td>
<td>6</td>
<td>2</td>
<td>-4</td>
</tr>
</tbody>
</table>
Discrete explicit representation

• Meshing
  – Interpolating approaches
    (connects sample points by triangles)
  – Precision improved via refinement
    (tradeoff between accuracy and efficiency)
  – Bad for noisy data
    (needs optimization)
  – May lead to holes and non-manifold situations
    (needs repairing)

Will come back to meshing soon!
Surface representation

• Implicit
  - Represent a surface as the zero set of a regular (with non-vanishing derivative) real-valued function defined over R3 (distance function).

Easy and efficient for topological modifications.
Discrete implicit representation

Voxel grids
Values of the distance function on a grid

Adaptative grids
High-precision only near the surface
Implicit to explicit: extracting the surface

\[ F(x) = 0 \rightarrow \text{surface} \]

\[ F(x) < 0 \rightarrow \text{inside} \]

\[ F(x) > 0 \rightarrow \text{outside} \]

Sample

2D Marching squares
Marching cubes

Ambiguity ...
Outline

• Geometric representation methods:
  – Explicit and Implicit

• Curvature:
  - parametric curves
  - parametric surfaces

• Laplacian smoothing
Parametric curves

\[ \mathbf{x} : [a, b] \subset \mathbb{R} \to \mathbb{R}^3 \]

\[
\mathbf{x}(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} \\
\mathbf{x}_t(t) := \frac{d\mathbf{x}(t)}{dt} = \begin{pmatrix} dx(t)/dt \\ dy(t)/dt \\ dz(t)/dt \end{pmatrix}
\]
Curvature: intuition

Tangent, the first approximation
• limiting secant as the two points come together
Curvature: intuition

Circle of curvature

- Consider the circle passing through 3 points of the curve
- The limiting circle as three points come together
Gauß map

Point on curve maps to point on unit circle

Turning (winding) number, $k$

number of orbits in Gaussian image
Turning number theorem

For a closed curve, the integral of curvature is an integer multiple of $2\pi$.
Outline

• Geometric representation methods:
  – Explicit and Implicit

• Curvature:
  - parametric curves
  - parametric surfaces

• Laplacian smoothing
Parametric surfaces

• Continuous surface

\[ \mathbf{p}(u, v) = \begin{pmatrix} x(u, v) \\ y(u, v) \\ z(u, v) \end{pmatrix}, \quad (u, v) \in \mathbb{R}^2 \]

• Normal vector

\[ \mathbf{n} = \frac{\mathbf{x}_u \times \mathbf{x}_v}{\| \mathbf{x}_u \times \mathbf{x}_v \|} \]

• Assume regularity on the parameterization

\[ \mathbf{x}_u \times \mathbf{x}_v \neq 0 \quad \text{normal exists} \]
Normal curvature

The curve $\gamma$ is the intersection of the surface with the plane through $n$ and the tangent vector $t$.

Normal curvature:

$$\kappa_n(\varphi) = \kappa(\gamma(p))$$
Surface Curvatures

• Principal curvatures
  – Maximal curvature \( \kappa_1 = \kappa_{\text{max}} = \max_{\varphi} \kappa_n(\varphi) \)
  – Minimal curvature \( \kappa_2 = \kappa_{\text{min}} = \min_{\varphi} \kappa_n(\varphi) \)

• Mean curvature \( H = \frac{\kappa_1 + \kappa_2}{2} \)

• Gaussian curvature \( K = \kappa_1 \cdot \kappa_2 \)
Principal Directions: examples

Fig from wikipedia ☺️
Principal Directions

- Principal directions: tangent vectors corresponding to $\varphi_{\text{max}}$ and $\varphi_{\text{min}}$

Fig: Alliez et al.
Euler’s Theorem: Planes of principal curvature are orthogonal and independent of parameterization.

\[ \kappa(\varphi) = \kappa_1 \cos^2 \varphi + \kappa_2 \sin^2 \varphi, \quad \varphi = \text{angle with } t_1 \]
Local Surface Shape By Curvatures

**Isotropic:**
all directions are principal directions

- $K > 0$, $\kappa_1 = \kappa_2$
  - spherical (umbilical)

**Anisotropic:**
2 distinct principal directions

- $K > 0$
  - $\kappa_2 > 0$, $\kappa_1 > 0$
  - elliptic

- $K = 0$
  - $\kappa_2 = 0$
  - parabolic

- $K < 0$
  - $\kappa_2 < 0$
  - hyperbolic

Slide from Olga Sorkine, Eitan Grinspun
**Gauss-Bonnet Theorem**

- For a closed surface $\mathcal{M}$:
  \[
  \int_{\mathcal{M}} K \, dA = 2\pi \chi(\mathcal{M})
  \]

\[
\int K(\text{ball}) = \int K(\text{star}) = \int K(\text{rabbit}) = 4\pi
\]

Intuition: when sphere is deformed, new positive and negative curvature cancel out.

- Compare with planar curves: (k turning number)
  \[
  \int_{\gamma} \kappa \, ds = 2\pi \, k
  \]
Surface Curvatures

• Mean curvature: extrinsic

\[ \frac{H_p}{2} = \frac{1}{2\pi} \int_0^{2\pi} k_p(\theta) \, d\theta \]

\( H=0 \) everywhere \quad \rightarrow \quad \text{minimal surface}

• Gaussian curvature: intrinsic

\[ K_p = \lim_{A \to 0} \frac{A_G}{A} \]

\( K=0 \) everywhere \quad \rightarrow \quad \text{developable surface}
Discrete curvatures

- Mean curvature
  \[ H = ||\Delta_S x|| \]

- Gaussian curvature
  \[ G = (2\pi - \sum_j \theta_j)/A \]

- Principal curvatures
  \[ \kappa_1 = H + \sqrt{H^2 - G} \]
  \[ \kappa_2 = H - \sqrt{H^2 - G} \]
Outline

• Geometric representation methods:
  – Explicit and Implicit

• Curvature:
  - parametric curves
  - parametric surfaces

• Laplacian smoothing
The Laplace operator is defined as:

$$\Delta f = \text{div} \nabla f = \sum_i \frac{\partial^2 f}{\partial x_i^2}$$

- **Laplace operator**
- **Gradient operator**
- **2nd partial derivatives**
- **Function in Euclidean space**
- **Divergence operator**
- **Cartesian coordinates**
Laplace-Beltrami operator

- Extension of Laplace to functions on manifolds

\[ \Delta_s \mathbf{x} = \text{div}_s \nabla_s \mathbf{x} = -2H \mathbf{n} \]
Laplacian smoothing

\[ \Delta p_i = \frac{1}{2}(p_{i-1} - p_i) + \frac{1}{2}(p_{i+1} - p_i) \]

\[ p_i \leftarrow p_i + \frac{1}{2}\Delta p_i \]
Discrete Laplace-Beltrami

- Uniform discretization

\[ \Delta_{\text{uni}} f(v) := \frac{1}{|N_1(v)|} \sum_{v_i \in N_1(v)} (f(v_i) - f(v)) \]

- Weighted discretization (cotan formula)

\[ \Delta_{w} f(v) := \frac{2}{A(v)} \sum_{v_i \in N_1(v)} (\cot \alpha_i + \cot \beta_i) (f(v_i) - f(v)) \]
Discrete Laplace-Beltrami and smoothing

Fig. Desbrun et al. 99

Original
uniform weights
Cotangent weights
Introduction à la Géométrie Algorithmique

Pooran MEMARI

Master IMA - Séminaires et pratique en image (PRAT) (SI955)
Janvier 2015
Outline

• Delaunay Triangulations (& good properties)
• Voronoi Diagram (famous dual)
• Some “elegant & simple” geometric ideas ...
  – Crust algorithm
• Triangle-based meshing/reconstruction:
  – Restricted Delaunay
  – Isotropic remeshing
  – ...

* Some slides from Misha Kazhdan
Outline

• Delaunay Triangulations (& good properties)
• Voronoi Diagram ( famous dual )
• Some “elegant & simple” geometric ideas …
  – Crust algorithm
• Triangle-based meshing/reconstruction:
  – Restricted Delaunay
  – Isotropic remeshing
  – …
Convex Hulls

Definition:
Given a finite set of points $P=\{p_1,\ldots,p_n\} \subset \mathbb{R}^n$, the convex hull is the set of points consisting of the convex combinations of points in $P$:

$$\text{Convex}(P) = \left\{ \sum_{p \in P} \alpha_p p \left| \alpha_p \geq 0 \text{ and } \sum_{p \in P} \alpha_p \right\}$$
Convex Hulls

Definition:
Given a finite set of points $P=\{p_1,\ldots,p_n\}\subset \mathbb{R}^n$, the convex hull is the set of points consisting of the convex combinations of points in $P$:

$$\text{Convex}(P) = \left\{ \sum_{p \in P} \alpha_p p \middle| \alpha_p \geq 0 \quad \text{and} \quad \sum_{p \in P} \alpha_p \right\}$$
Planar Triangulations

Definition:
A \textit{triangulation} of a finite set of points \( P = \{p_1, \ldots, p_n\} \) is a decomposition of the convex hull of \( P \) into triangles with the property that:

- The set of triangle vertices equals \( P \)
- The intersections of two triangles is either empty or is a common edge or vertex.
Delaunay Triangulation

Canonical triangulation associated to any point set
Delaunay Triangulations

Definition:
A triangulation of the set $P$ is said to be Delaunay if the interior of the triangles’ circumcircles are empty.
Delaunay Triangulations

**Definition:**
A triangulation of the set $P$ is said to be *Delaunay* if the interior of the triangles’ circum-circles are empty.
Delaunay Triangulations

Definition:
A triangulation of the set $P$ is said to be Delaunay if the interior of the triangles’ circumspheres are empty.
Delaunay Triangulations

Definition:
A triangulation of the set $P$ is said to be *Delaunay* if the interior of the triangles’ circumcircles are empty.
Delaunay Edges

Definition:
An interior edge $e$ is locally Delaunay if the interiors of the circum-circles of the two triangles do not contain the triangles’ vertices.
Delaunay Edges

Definition:
An interior edge $e$ is *locally Delaunay* if the interiors of the circum-circles of the two triangles do not contain the triangles’ vertices.

Property:
An interior edge is Delaunay iff. the sum of the opposite angles is not greater than $\pi$.

$\alpha + \beta \leq \pi$
Delaunay Edges

Note:
If the sum of the opposite angles is greater than $\pi$, then flipping the edge will give a sum that is less than $\pi$.

$$\gamma + \delta = 2\pi - (\alpha + \beta)$$
Delaunay Triangulations

Property:
A triangulation is Delaunay if and only if every interior edge is locally Delaunay.
Delaunay Triangulations

Property:
A triangulation is Delaunay if and only if every interior edge is locally Delaunay.

Edge Flipping Algorithm:
Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.
Delaunay Triangulations

Property:
A triangulation is Delaunay if and only if every interior edge is locally Delaunay.

Edge Flipping Algorithm:
Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.
Delaunay Triangulations

**Property:**
A triangulation is Delaunay if and only if every interior edge is locally Delaunay.

**Edge Flipping Algorithm:**
Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.
Delaunay Triangulations

**Property:**
A triangulation is Delaunay if and only if every interior edge is locally Delaunay.

**Edge Flipping Algorithm:**
Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.
Delaunay Triangulations

Property:
A triangulation is Delaunay if and only if every interior edge is locally Delaunay.

Edge Flipping Algorithm:
Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.
Delaunay Triangulations

**Edge Flipping Algorithm:**
Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.

Is this algorithm guaranteed to terminate?
Delaunay Triangulations

Edge Flipping Algorithm:
Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.
Is this algorithm guaranteed to terminate?
Termination is proved by:
- Showing that there finitely many different triangulations.
- Defining a global “energy” that is reduced with each flip (e.g. sum of squared circum-radii.)
Delaunay Triangulations

Computing the Delaunay Triangulation:
• Incremental
• Divide and Conquer
• Sweepline (planar)
• Convex hulls of paraboloids
Why should we care?

Meshing

• **Input:** a planar straight-line graph (PSLG) or simply a planar subdivision.
• **Constraints:** shape, boundaries, internal edges to preserve, sizing, orientation (unisotropc)...
• Structured (fixed valence/vertex deg) or not
• Uniform distribution or not
• Application dependent constraints: numerical operator based with geometric interpretation.

• Delaunay triangulation has “well-formed” triangles, facilitating numerical processing over the triangulation.
2D: Triangle shape quality

- Minimum angle $\alpha$
- Circumcentric radius / smallest edge length = $\frac{1}{2} \sin(\alpha)$
- Circumcentric radius / incircle radius
- Biggest edge length / minimum height

Delaunay triangulation maximizes the smallest angle

Even more: angular vector is maximal for the lexicographic order
Delaunay triangulation’s good properties:

• Efficient & robust algorithms to compute it.
• Maximizes the smallest angle of triangles.
• Good shape triangles (at least in 2D).
• Contains the nearest neighbor graph.
• Contains the minimum spanning tree.
• Used in many different applications such as reconstruction, mesh refinement, remeshing...

See CGAL library
Outline

• Delaunay Triangulations (& good properties)
• Voronoi Diagram ( famous dual )
• Some “elegant & simple” geometric ideas ...
  – Crust algorithm
• Triangle-based meshing/reconstruction:
  – Restricted Delaunay
  – Isotropic remeshing
  – ...


Voronoi Diagram

Definition:
The Voronoi Diagram of the set $P$ is the partition of space into cells $V(p)$ such that for all $q \in V(p)$, $q$ is closer to $p$ than to any other point $p' \in P$. 
Voronoi Diagram

The Voronoi Diagram of $P$ is the dual of the Delaunay Triangulation.
Voronoi Diagram

The Voronoi Diagram of $P$ is the dual of the Delaunay Triangulation:

– **2D:**
  - Every vertex of the triangulation is dual to a polygon in the diagram.
  - Every edge of the triangulation is dual to an edge of the diagram.
  - Every triangle of the triangulation is dual to a vertex of the diagram.
The Voronoi Diagram of $P$ is the dual of the Delaunay Triangulation:

- **3D:**
  - Every vertex of the triangulation is dual to a polyhedron in the diagram.
  - Every edge of the triangulation is dual to an face of the diagram.
  - Every triangle of the triangulation is dual to an edge of the diagram.
  - Every tetrahedron of the triangulation is dual to a vertex of the diagram.
Delaunay and simple geometric ideas...

- Curve reconstruction from sample points:
  Delaunay triangulation of the point set $E$ covers the convex hull of $E$, but ...
Delaunay and simple geometric ideas...

• Curve reconstruction from sample points
And in 3D?

- In 3D some Voronoi vertices are not near medial axis even for dense sampling.

- Poles: a subset of Voronoi vertices which approximates medial axis (Amenta and Bern 98)
Outline

• Delaunay Triangulations (& good properties)
• Voronoi Diagram (famous dual)
• Some “elegant & simple” geometric ideas ...
  – Crust algorithm
• Triangle-based meshing/reconstruction:
  – Restricted Delaunay
  – Isotropic remeshing
  – ...

Restricted Delaunay Triangulation

Goal:
Given a surface $S$ and a set of points $P$ in $S$, we would like to compute a good triangulation of $P$ that is true* to the surface.

*Note that not every point set $P$ has to admit a true triangulation.
Restricted Delaunay Triangulation

Approach (Take 1):
We could compute a Voronoi Diagram on $S$ using the notion of distances on the surface, and then take the dual to get a Delaunay Triangulation.
Restricted Delaunay Triangulation

Approach (Take 1):
We could compute a Voronoi Diagram on $S$ using the notion of distances on the surface, and then take the dual to get a Delaunay Triangulation.
Restricted Delauney Triangulation

Approach (Take 1):
We could compute a Voronoi Diagram on $S$ using the notion of distances on the surface, and then take the dual to get a Delaunay Triangulation.
Restricted Delaunay Triangulation

Approach (Take 1):
We could compute a Voronoi Diagram on S using the notion of distances on the surface, and then take the dual to get a Delaunay Triangulation.

Challenges:

1. Measuring distances on a surface can be expensive.
2. The dual complex may not be a manifold (or even have any triangles).
Restricted Delaunay Triangulation

Approach (Take 1):
We could compute a Voronoi Diagram on $S$ using the notion of distances on the surface, and then take the dual to get a Delaunay Triangulation.

Challenges:

1. Measuring distances on a surface can be expensive.

2. The dual complex may not be a manifold (or even have any triangles).
Restricted Delaunay Triangulation

Approach (Take 1):
We could compute a Voronoi Diagram on $S$ using the notion of distances on the surface, and then take the dual to get a Delaunay Triangulation.

Challenges:
1. Measuring distances on a surface can be expensive.
2. The dual complex may not be a manifold (or even have any triangles).
Restricted Delaunay Triangulation

Approach (Take 2):
Instead of trying to compute a Voronoi Diagram using distances on the surface, compute a regular Voronoi Diagram and look at its restriction to the surface.
Restricted Delaunay Triangulation

Approach (Take 2):
Instead of trying to compute a Voronoi Diagram using distances on the surface, compute a regular Voronoi Diagram and look at its restriction to the surface.
Restricted Delaunay Triangulation

Approach (Take 2):
Instead of trying to compute a Voronoi Diagram using distances on the surface, compute a regular Voronoi Diagram and look at its restriction to the surface.

- Add a Delaunay edge between vertices \( p, p' \in P \) if their Voronoi regions meet on the surface.
- Add a Delaunay triangle between vertices \( p, p', p'' \in P \) if their Voronoi regions meet on the surface.
Restricted Delaunay Triangulation

Approach (Take 2):
Instead of trying to compute a Voronoi Diagram

Note:
• The Voronoi regions of the vertices of a Delaunay edge meet on the surface iff. the dual Voronoi face intersects the surface.
• The Voronoi regions of the vertices of a Delaunay triangle meet on the surface iff. The dual Voronoi edge intersects the surface.

\( p, p' \in P \) if their Voronoi regions meet on the surface.
– Add a Delaunay triangle between vertices \( p, p', p'' \in P \) if their Voronoi regions meet on the surface.
Restricted Delaunay Triangulation

Approach (Take 2):
Instead of trying to compute a Voronoi Diagram using distances on the surface, compute a regular Voronoi Diagram and look at its restriction to the surface.

Note:
• The Voronoi regions of the vertices of a Delaunay edge meet on the surface iff. the dual Voronoi face intersects the surface.
• The Voronoi regions of the vertices of a Delaunay triangle meet on the surface iff. The dual Voronoi edge intersects the surface.

Note that there is (still) no guarantee that the restricted Delaunay Triangulation is manifold.

– Add a Delaunay triangle between vertices \( p, p', p'' \in P \) if their Voronoi regions meet on the surface.
Restricted Delaunay [Boissonnat & Oudot ‘05]

Goal:
Use the restricted Delaunay Triangulation, to triangulate the points $P \subset S$.

Approach:
Ensure that the complex is manifold by inserting a additional points when it is not.
Restricted Delaunay [Boissonnat & Oudot ‘05]

General Idea:
The restricted Delaunay Triangulation will fail to be manifold when the samples are not well-spaced.
Restricted Delaunay [Boissonnat & Oudot ‘05]

Definition:
The *medial axis* or *skeleton* of a shape is the set of points that are simultaneously closest to two points on \( S \).

*Note that only the interior skeleton is drawn here.*
Restricted Delaunay [Boissonnat & Oudot ‘05]

Definition:
The *reach* of a point on $S$ is its distance to the nearest point on the medial axis. This provides a measure of:

- Curvature
- Proximity of surface sheets

*Note that only the interior skeleton is drawn here.*
Restricted Delaunay [Boissonnat & Oudot ‘05]

Note:
If we intersect a surface with a ball and the set of points on the intersection have reach smaller than the radius of the ball, then the intersection is connected.
Restricted Delaunay [Boissonnat & Oudot ‘05]

General Idea:
The restricted Delaunay Triangulation will fail to be manifold when the samples are not well-spaced.

More Specifically:
We want points on the Delaunay Triangulation to be closer to each other than their reach.
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:

Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation
While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:
  Add the intersection of the triangle’s dual with the surface
  (Locally) update the Delaunay Triangulation
Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:

Compute the Delaunay Triangulation.

**Compute the restricted D. Triangulation**

While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:

- Add the intersection of the triangle’s dual with the surface
- (Locally) update the Delaunay Triangulation
- Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:

Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation

While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:
   Add the intersection of the triangle’s dual with the surface
   (Locally) update the Delaunay Triangulation
   Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:
Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation
While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:
   Add the intersection of the triangle’s dual with the surface
(Locally) update the Delaunay Triangulation
Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:
Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation
While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:
   Add the intersection of the triangle’s dual with the surface
   (Locally) update the Delaunay Triangulation
Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:
Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation
While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:
   Add the intersection of the triangle’s dual with the surface
(Locally) update the Delaunay Triangulation
Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:
Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation
While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:
  Add the intersection of the triangle’s dual with the surface
  (Locally) update the Delaunay Triangulation
Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:

Compute the Delaunay Triangulation.

Compute the restricted D. Triangulation

While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:

Add the intersection of the triangle’s dual with the surface

(Locally) update the Delaunay Triangulation

Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:
Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation
While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:
   Add the intersection of the triangle’s dual with the surface
   (Locally) update the Delaunay Triangulation
   Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:
Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation
While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:
  Add the intersection of the triangle’s dual with the surface
  (Locally) update the Delaunay Triangulation
Update the Restricted D. Triangulation
**Restricted Delaunay** [Boissonnat & Oudot ‘05]

**Algorithm:**

Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation

While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:

Add the intersection of the triangle’s dual with the surface

*(Locally) update the Delaunay Triangulation*

Update the Restricted D. Triangulation
Restricted Delaunay [Boissonnat & Oudot ‘05]

Algorithm:
Compute the Delaunay Triangulation.
Compute the restricted D. Triangulation
While there are triangles whose circumsphere’s radius is larger than a fraction of the reach:
   Add the intersection of the triangle’s dual with the surface
   (Locally) update the Delaunay Triangulation
   Update the Restricted D. Triangulation

*Note that the algorithm would still keep going*
Restricted Delaunay [Boissonnat & Oudot ‘05]

Implementation Requirements:

• A single computation of a (restricted) Delaunay Triangulation plus local updates.
• The ability to evaluate the reach of a surface point.
• The ability to intersect the dual Voronoi edge with the surface.
Restricted Delaunay [Boissonnat & Oudot ‘05]

Properties:
• With the appropriate scaling, the method returns a manifold, non-self-intersecting, triangulation with the same topology as $S$.
• May over-refine in flat regions.
• Requires a strictly positive reach (which is not satisfied by triangle meshes).
Restricted Delaunay [Boissonnat & Oudot ‘05]

Surface mesh generation algorithm:

Do{
• Take a bad shaped facet $f$
• Insert furthest dual($f$)$\cap S$ in Del
• Update Del restricted to $S$
}
untill all facets are well shaped
Surface mesh algorithm guarantees

- Well shaped triangles (lower bound on angles)
- Resulting mesh is manifold.
- Homeomorphic to input surface under some sampling conditions (dense sampling where curvature is high or near features).
- $\varepsilon$-sampling: distance from any surface point to nearest sample is at most small constant $\varepsilon$ times distance to medial axis. Zero at sharp corners
- Good approximation in terms of Haussdorff distance and normals.

Boissonnat and Oudot 2005.
Restricted Delaunay

• Polyhedral Domains (Input conforming):
  – Angle restricted: Chew89, Ruppert92, Miller-Talmor-Teng-Walkington95, Shewchuk98.

• Smooth Surfaces (Topology conforming):
  – Chew93 (w/out guarantee), Cheng-Dey-Edelsbrunner-Sullivan01 (skin surfaces), Boissonnat-Oudot03 and Cheng-Dey-Ramos-Ray04, Oudot-Rineau-Yvinec06 (Volumes).

• Non-smooth:
  – Boissonnat-Oudot06 (Lipschitz surfaces).
  – Cheng-Dey-Ramos07 (piecewise smooth complexes).
Outline

• Delaunay Triangulations (& good properties)
• Voronoi Diagram ( famous dual )
• Some “elegant & simple” geometric ideas ...
  – Crust algorithm
• Triangle-based meshing/reconstruction:
  – Restricted Delaunay
  – Isotropic remeshing
  – ...

Isotropic Remeshing [Alliez et al. ‘03]

Observation:
Given a parameterization of $S$ over a 2D domain, we can pull back a triangulation of the 2D domain to a triangulation of the mesh.

*May have intersecting triangles
Isotropic Remeshing [Alliez et al. ‘03]

Observation:
Given a parameterization of S over a 2D domain, we can pull back a triangulation of the 2D domain to a triangulation of the mesh.*

*May have intersecting triangles
Isotropic Remeshing [Alliez et al. ‘03]

Questions:
1. Which parameterization do we choose?
2. How do we triangulate the 2D domain?
Isotropic Remeshing [Alliez et al. ‘03]

1. **Which parameterization do we choose?**
Use a conformal parameterization. The distortion is strictly due to scaling, so we can undo that by appropriately tesselating the 2D domain.

[Global Conformal Surface Parameterization, Gu and Yau]
Isotropic Remeshing [Alliez et al. ‘03]

2. How do we triangulate the 2D domain?
If we have a point sampling, we can compute the (constrained) Delaunay triangulation...
So how do we choose the point set?
Isotropic Remeshing [Alliez et al. ‘03]

Goal:
We would like to undue the area distortion caused by the conformal map.

[Global Conformal Surface Parameterization, Gu and Yau]
Isotropic Remeshing [Alliez et al. ‘03]

Goal:
We would like to undue the area distortion caused by the conformal map.

Approach:
Use the distortion to sample the 2D domain adaptively.
Isotropic Remeshing [Alliez et al. ‘03]

Goal:
We would like to undo the area distortion caused by the conformal map.

Approach:
Use the distortion to sample the 2D domain adaptively.

Challenge:
Just because the points are randomly distributed, that doesn’t make them uniform. [Isotropic Surface Remeshing, Alliez et al.]
Isotropic Remeshing [Alliez et al. ‘03]

Update/Solve for well-distributed positions.
Given a density function $\rho$, solve for a point set $P$ and a partition of the 2D domain:

$$\Omega = \bigcup_{p \in P} R_p$$

that minimizes:

$$E(P, R) = \sum_{p \in P} \int_{x \in R_p} \rho(x) \|x - p\|^2 \, dx$$
Isotropic Remeshing [Alliez et al. ‘03]

\[ E(P, R) = \sum_{p \in P} \int_{x \in R_p} \rho(x) \|x - p\|^2 \, dx \]

Lloyd Relaxation:
Though finding the optimal solution is hard, improving on a solution is easy.

[Isotropic Surface Remeshing, Alliez et al.]
Isotropic Remeshing [Alliez et al. ‘03]

\[ E(P, R) = \sum_{p \in P} \int_{x \in R_p} \rho(x) \| x - p \|^2 \, dx \]

Lloyd Relaxation:
Though finding the optimal solution is hard, improving on a solution is easy.

Observations:
- Given the positions \( P \), the \( R_p \) minimizing the energy are the Voronoi regions of \( p \in P \).
Isotropic Remeshing [Alliez et al. ‘03]

\[ E(P, R) = \sum_{p \in P} \int_{x \in R_p} \rho(x) \| x - p \|^2 \, dx \]

Lloyd Relaxation:
Though finding the optimal solution is hard, improving on a solution is easy.

Observations:

– Given the positions \( P \), the \( R_p \) minimizing the energy are the Voronoi regions of \( p \in P \).

– Given the regions \( R_p \), the \( p \) minimizing the energy are the \( \rho \)-weighted centers of \( R_p \).
Isotropic Remeshing [Alliez et al. ‘03]

\[ E(P, R) = \sum_{p \in P} \int_{x \in R_p} \rho(x) \| x - p \|^2 \, dx \]

Implementation:
Iteratively alternate between computing the Voronoi regions of the points in \( P \), and computing the centers of the regions.
Isotropc Remeshing [Alliez et al. ‘03]

\[ E(P, R) = \sum_{p \in P} \int_{x \in R_p} \rho(x) \left\| x - p \right\|^2 dx \]

Applying this using the distortion weights from the conformal map, we get an isotropic tessellation.
Isotropic Remeshing [Alliez et al. ‘03]

\[ E(P, R) = \sum_{p \in P} \int_{x \in R_p} \rho(x) \|x - p\|^2 \, dx \]

Adapting the weights to take into account, curvature, you can get curvature-adapted tessellations.
Isotropic Remeshing [Alliez et al. ‘03]

\[ E(P, R) = \sum_{p \in P} \int_{x \in R_p} \rho(x) \| x - p \|^2 \, dx \]

Constraining the Delaunay Triangulation, you can preserve edges in the triangulation.
And many other interesting problems in this domain...

Computational geometry and geometry processing:
fascinating research fields
  – needs theoretical guarantees
  – Needs efficient and practical results
Questions?

memari@telecom-paristech.fr