Computational Geometry Tools and Applications in Computer Vision

Part A: Basics of Computational Geometry

Pierre Alliez
Inria Sophia Antipolis – Méditerranée
pierre.alliez@inria.fr

ECCV'16
EUROPEAN CONFERENCE ON COMPUTER VISION
October 8 – 16, 2016 | Amsterdam | the Netherlands

Outline

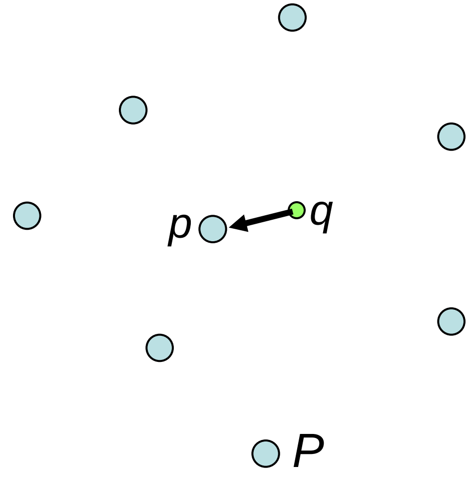
- Sample problems
  - 2D, 3D
  - Focus
- Voronoi diagram & Delaunay triangulation
- Shape reconstruction
- Mesh generation

Sample Problems

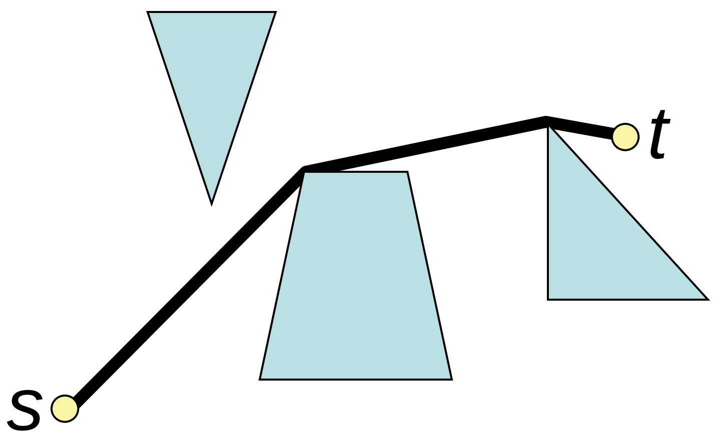
2D

Nearest Neighbor

- Problem definition:
  - Input: a set of points (*sites*) P in the plane and a query point q.
  - Output: The point $p \in P$ closest to q among all points in P.
- Rules of the game:
  - One point set, multiple queries



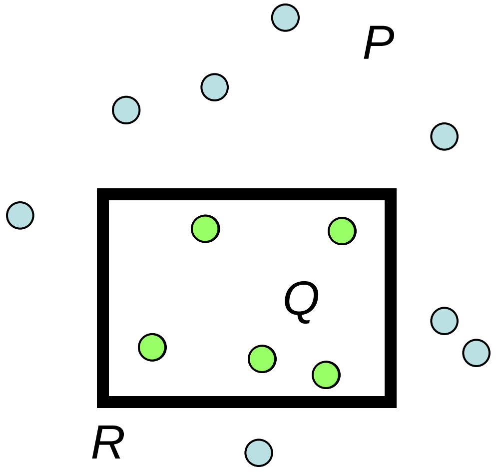
Shortest Path



- Problem definition:
    - Input: Obstacles locations and *query* endpoints *s* and *t*.
  - Output: shortest path between *s* and *t* that avoids all obstacles.
- Rules of the game: One obstacle set, multiple queries.

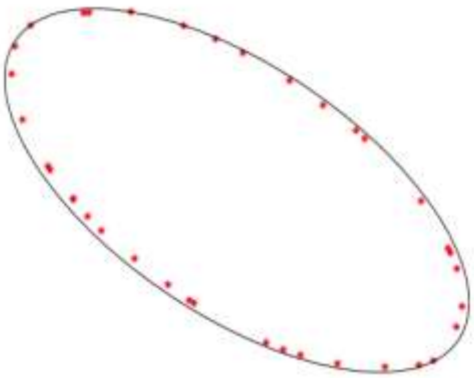
Range Searching and Counting

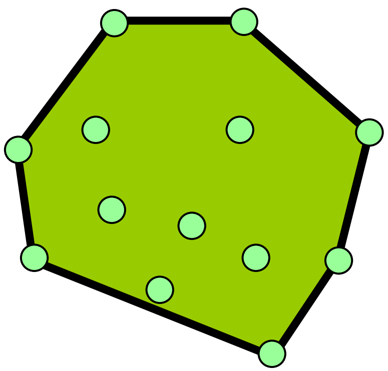
- Problem definition:
  - Input: Set of points *P* in the plane and query rectangle *R*
  - Output: (report) subset $Q \subseteq P$ contained in *R*.
    (count) size of *Q*.
- Rules of the game:
  - One point set, multiple queries.

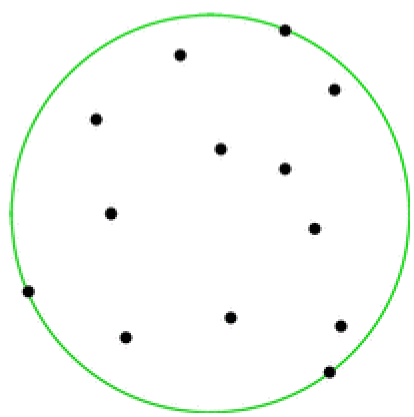


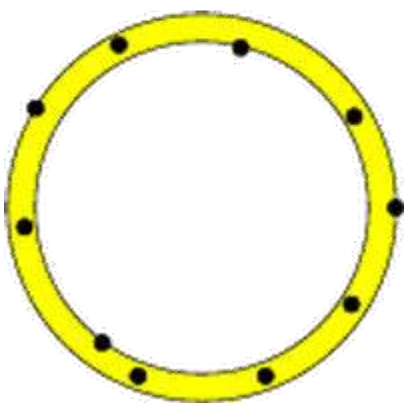
Bounding Volumes

- Problem definition:
  - Input: Set of points *P* in the plane
  - Output: (report) Smallest enclosing polygon, disk, ellipse, annulus, rectangles, parallelograms, k>=2 axis-aligned rectangles



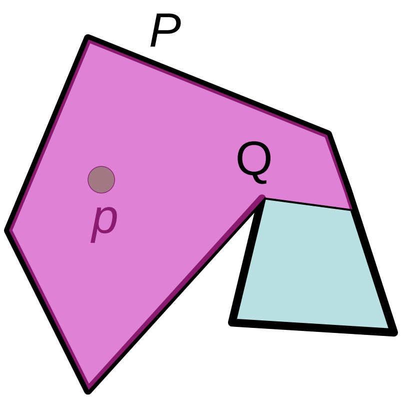






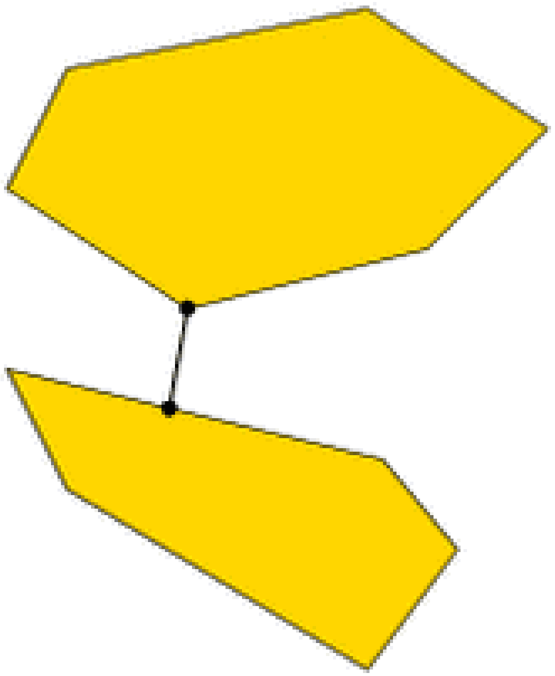
Visibility

- Problem definition:
  - Input: Polygon *P* in the plane, query point *p*.
  - Output: Polygon Q $\subseteq$ P, visible to p.
- Rules of the game:
  - One polygon, multiple queries



Optimal Distances

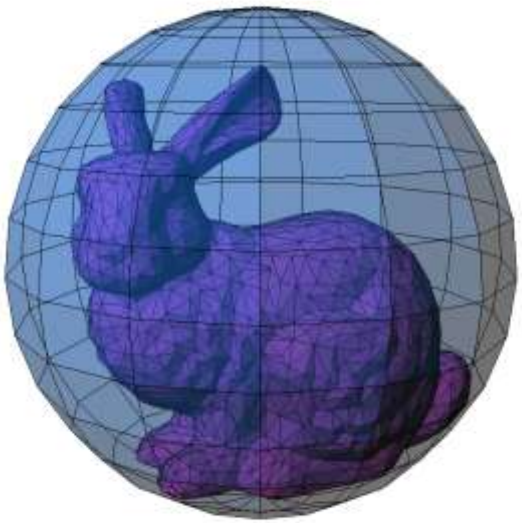
Distance between convex hulls of two point sets in Euclidean space (in dD!)

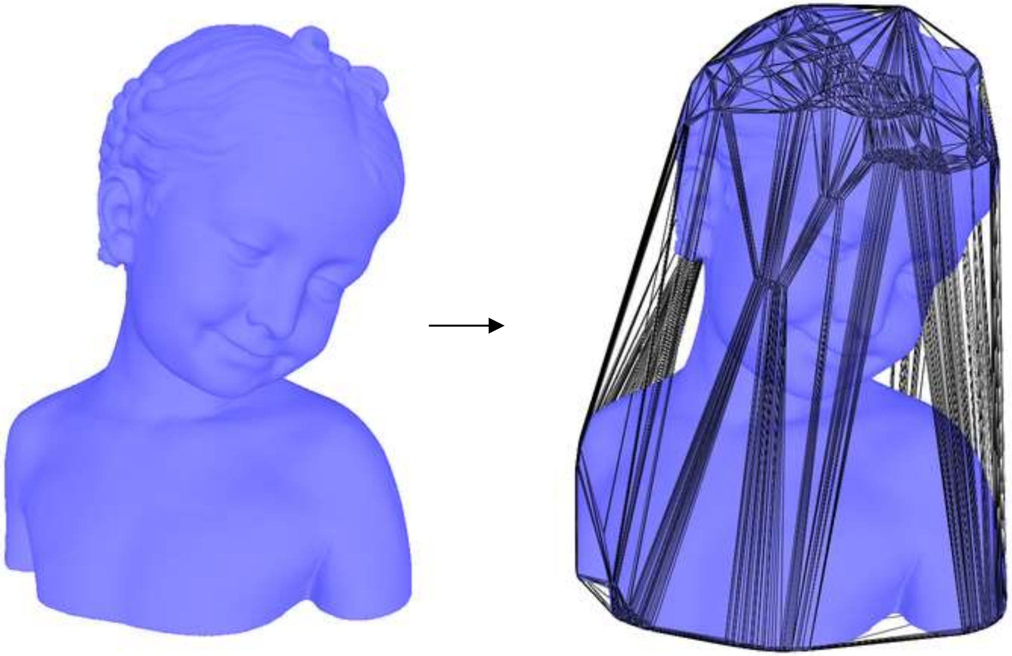


3D

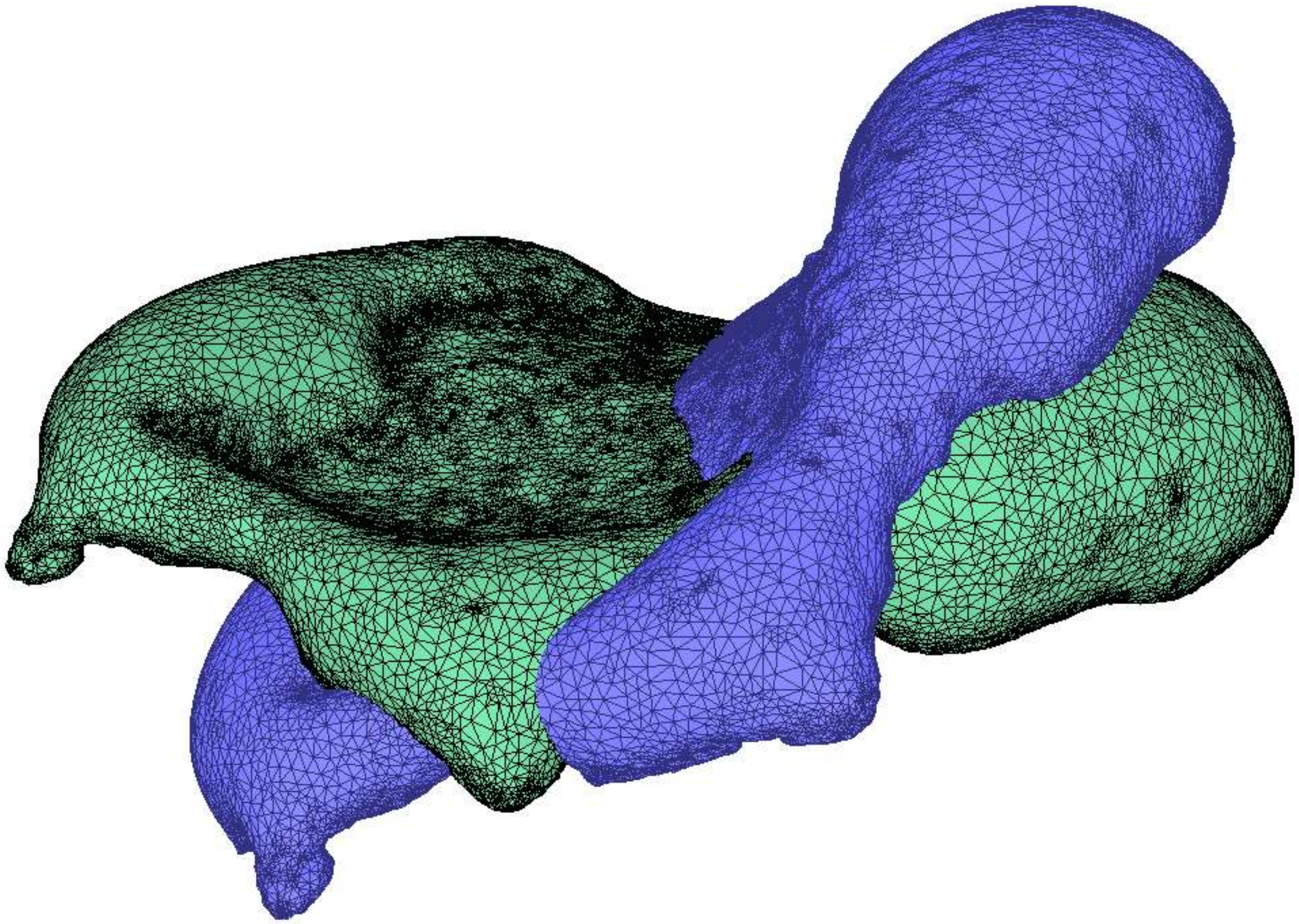
Bounding Volumes

- Convex hull
- Bounding sphere
- Bounding sphere of spheres

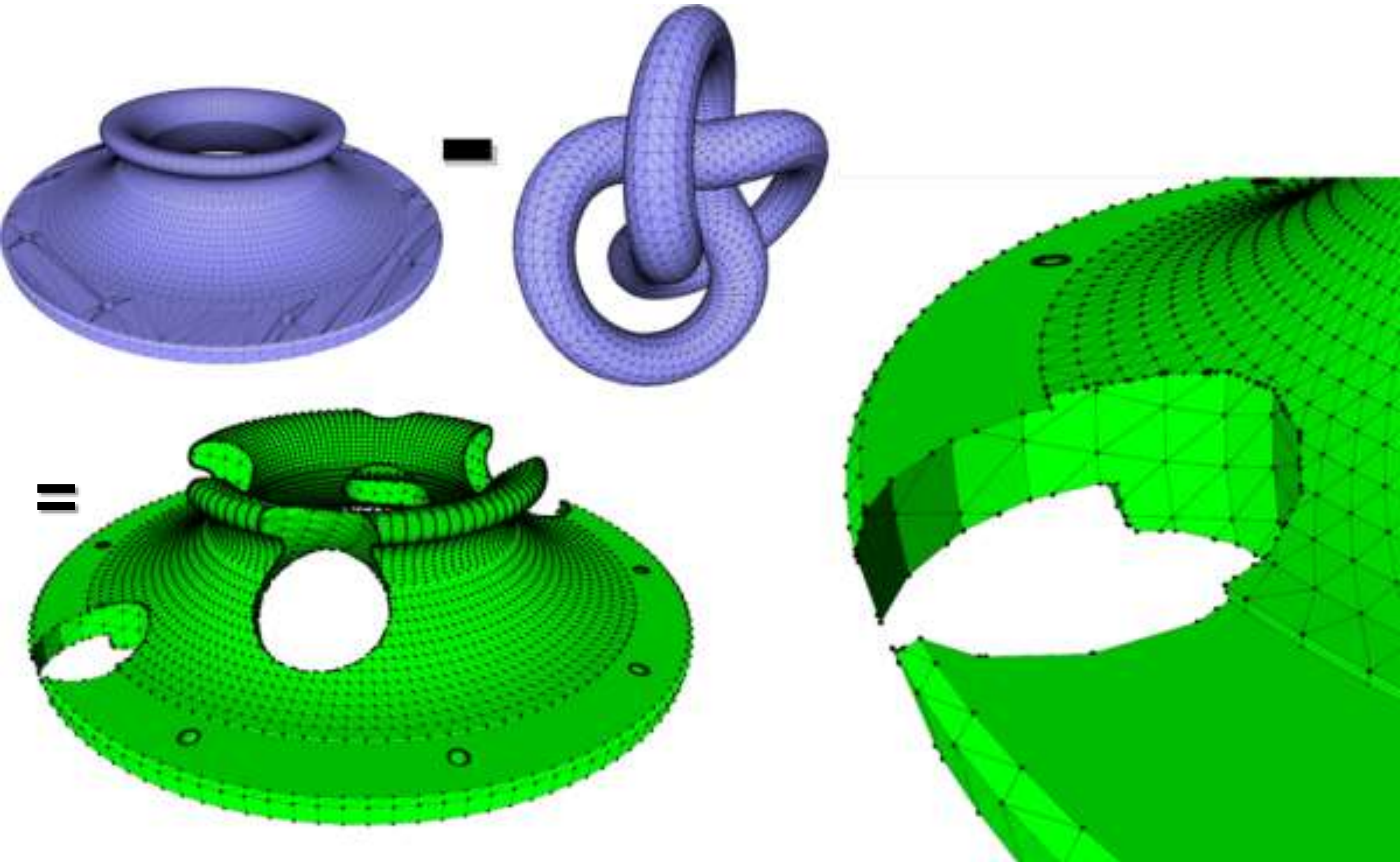




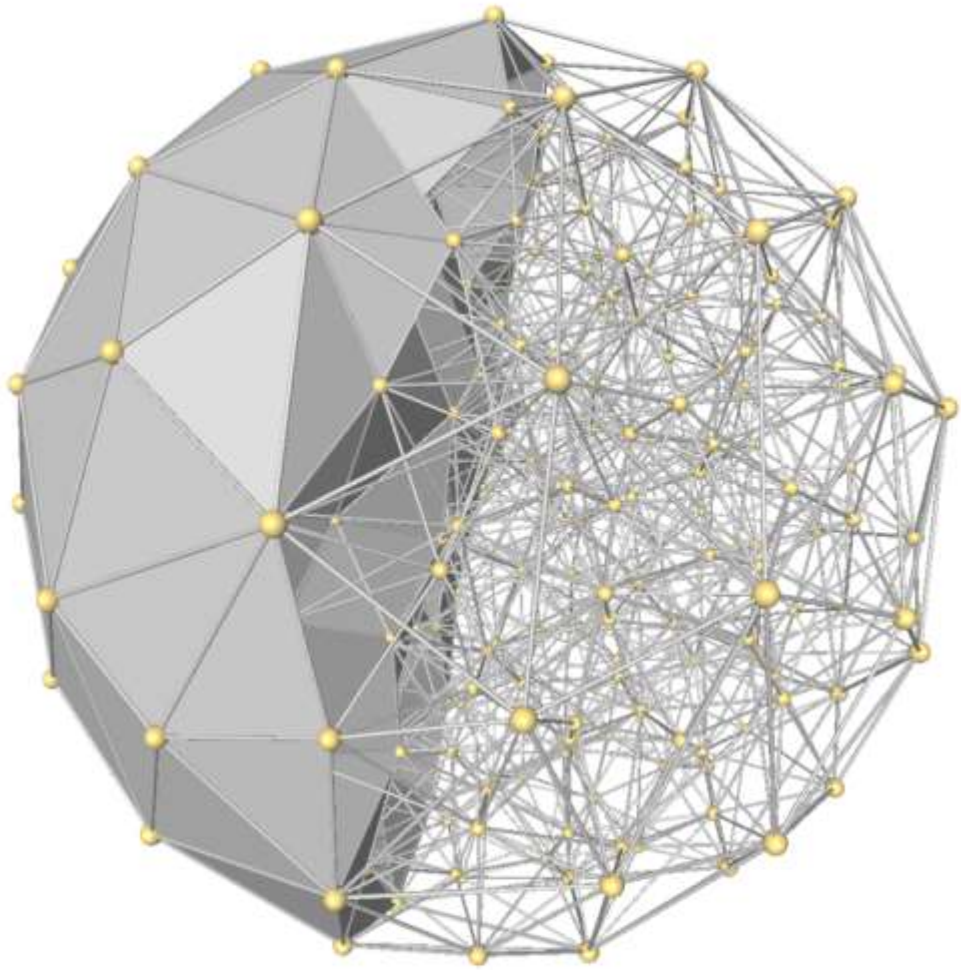
Intersections



Boolean Operations



Triangulations



Advances

Advances on Algorithms

- Correctness
- Complexity
  - Worst case
  - Average (real-world) cases
- Memory
- Reliability
  - Arithmetic of real-world computers
  - Degenerate cases

"Geometric Computing"

- Robustness
  - Real-world data

Focus for Today…

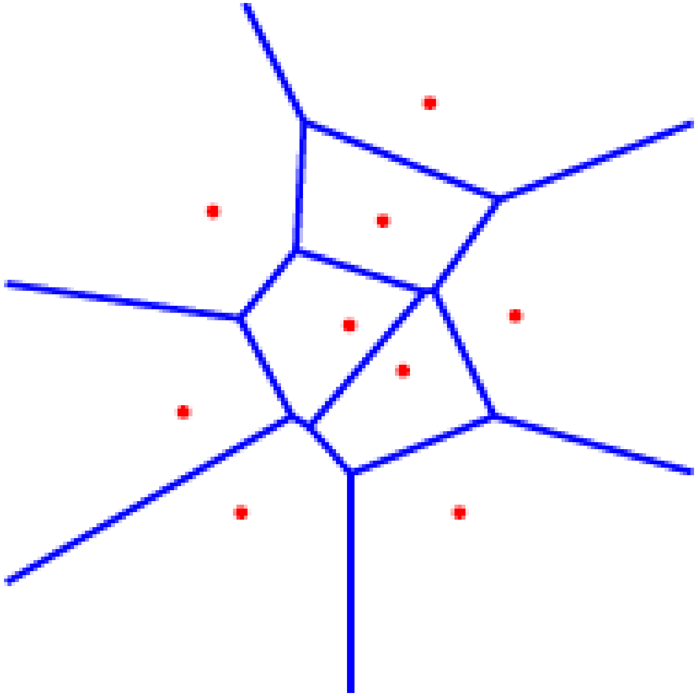
- Voronoi diagrams
- Delaunay triangulations
- Mesh generation
  - Delaunay-based
  - 2D, surface, 3D
- Shape reconstruction
  - Delaunay filtering

Voronoi diagrams
Delaunay Triangulations

Voronoi Diagram

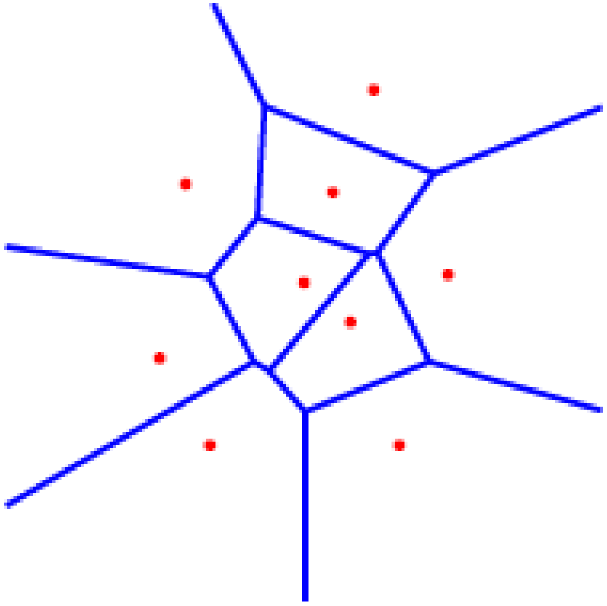
Let $\mathcal{E} = \{\mathbf{p_1}, \ldots, \mathbf{p_n}\}$ be a set of points (so-called sites) in $\mathbb{R}^d$. We associate to each site $\mathbf{p_i}$ its Voronoi region $V(\mathbf{p_i})$ such that:

$$V(\mathbf{p_i}) = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x} - \mathbf{p_i}\| \leq \|\mathbf{x} - \mathbf{p_j}\|, \forall j \leq n\}.$$



Voronoi Diagram

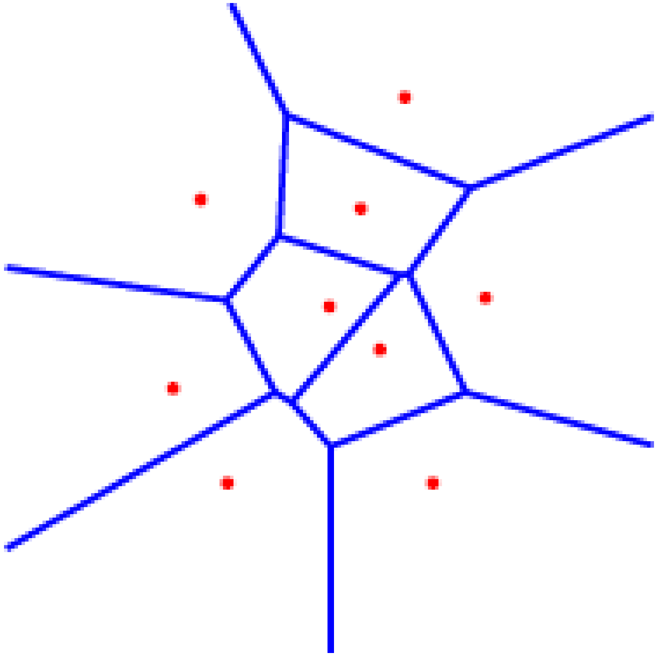
- The collection of the non-empty Voronoi regions and their faces, together with their incidence relations, form a cell complex called the **Voronoi diagram** of E.
- The locus of points which are equidistant to two sites and is called a **bisector**, all bisectors being affine subspaces of IR^d (lines in 2D).



demo

Voronoi Diagram

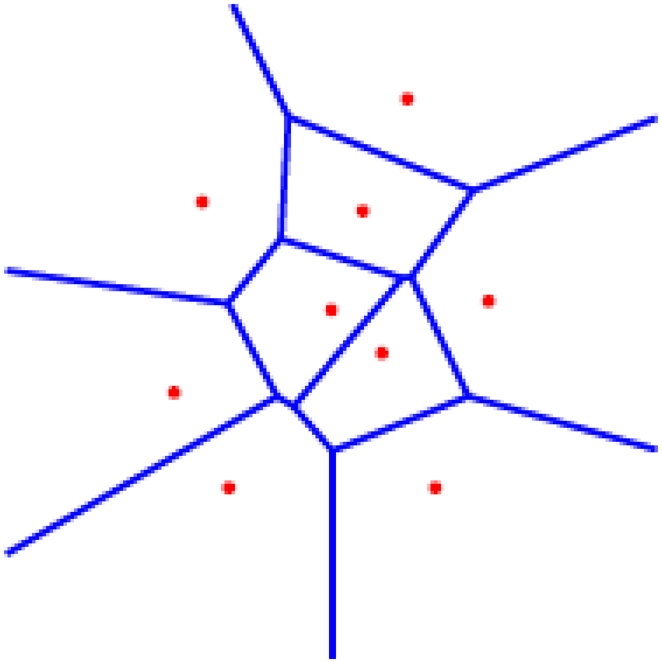
A Voronoi cell of a site *pi* defined as the intersection of closed half-spaces bounded by bisectors. Implies: All Voronoi cells are **convex**.



demo

Voronoi Diagram

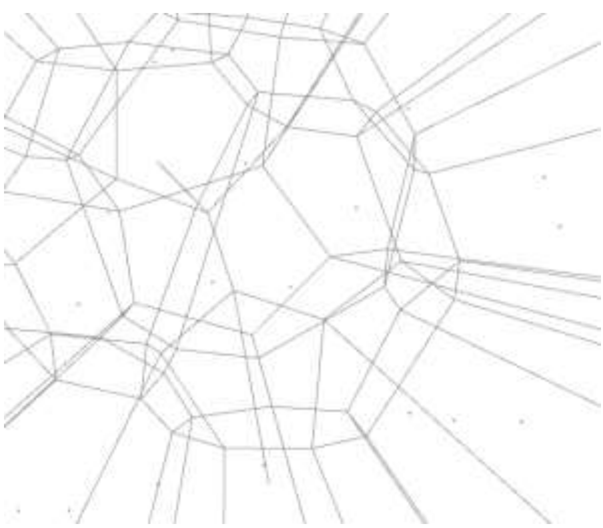
Voronoi cells may be **unbounded** with unbounded bisectors. Happens when a site is on the boundary of the convex hull of E.

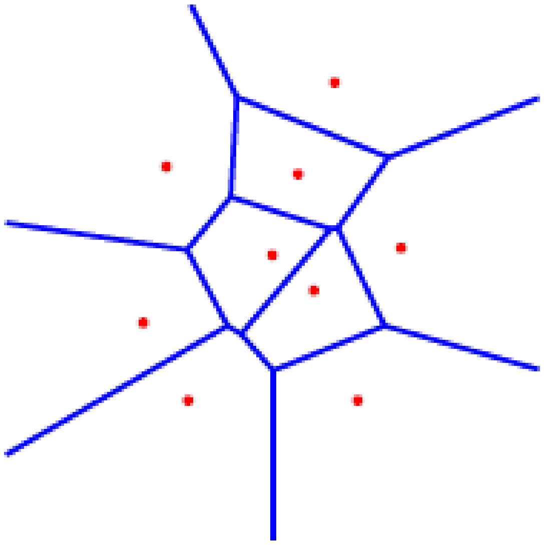


demo

Voronoi Diagram

- **Voronoi cells** have **faces** of different dimensions.
- In 2D, a face of dimension k is the intersection of 3 - k Voronoi cells. A **Voronoi vertex** is generically equidistant from three points, and a **Voronoi edge** is equidistant from two points.



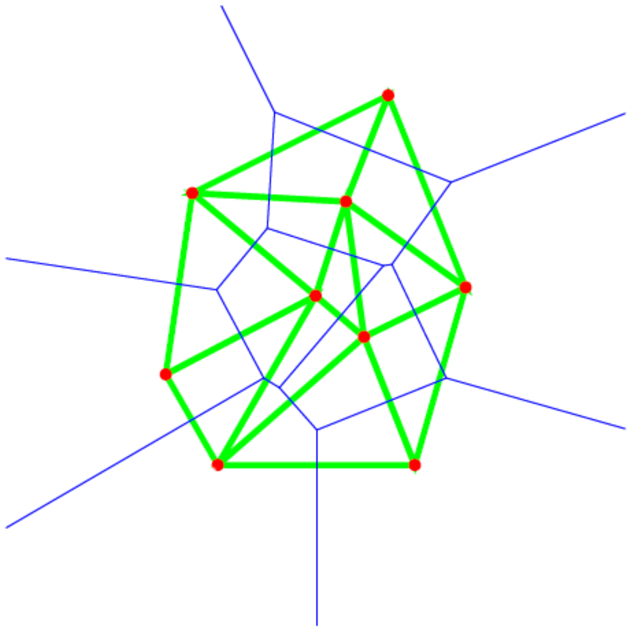


Delaunay Triangulation

- Dual structure of the Voronoi diagram.

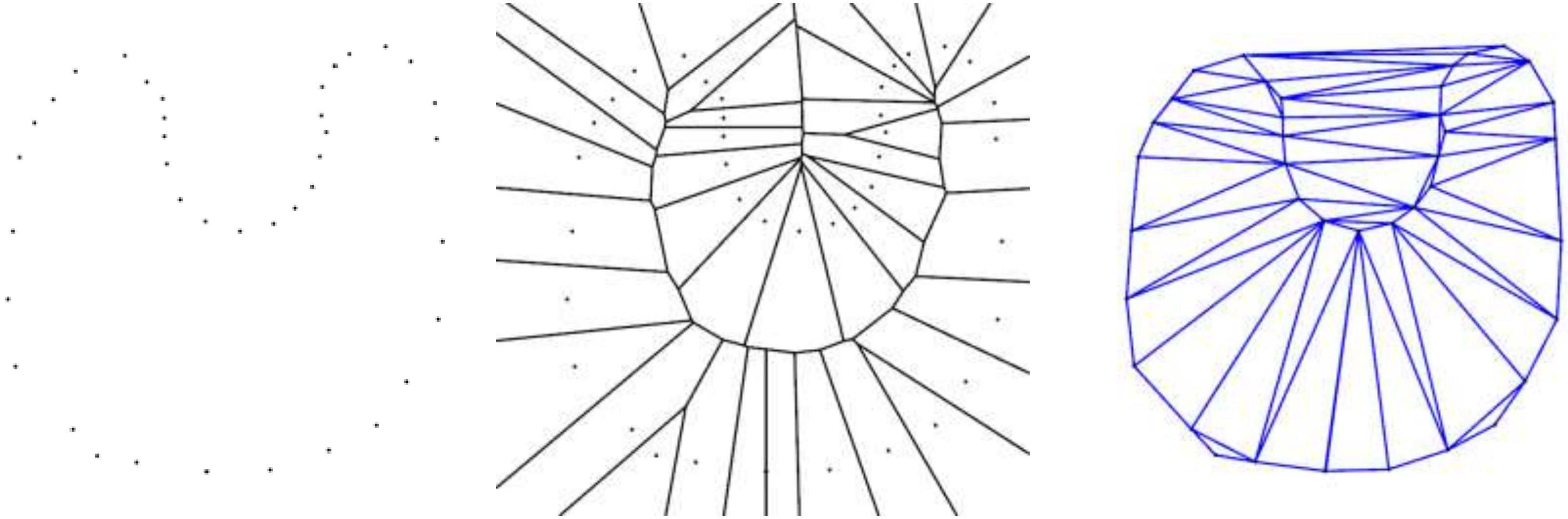
- The Delaunay triangulation of a set of sites E is a simplicial complex such that k+1 points in E form a Delaunay simplex if their Voronoi cells have nonempty intersection

demo



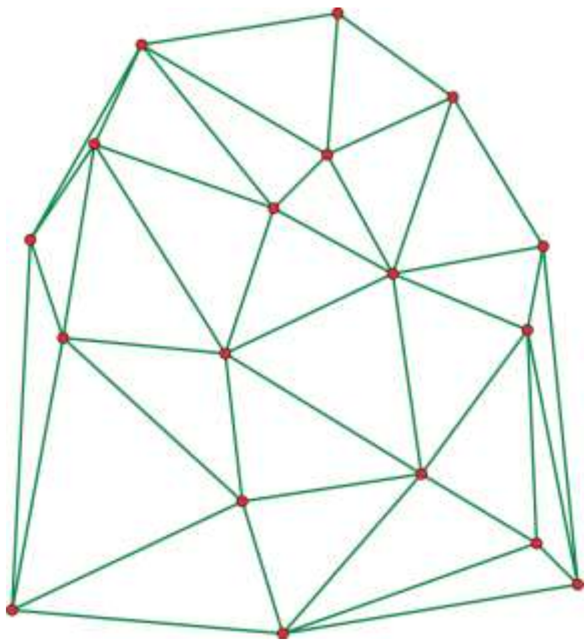
Delaunay Triangulation

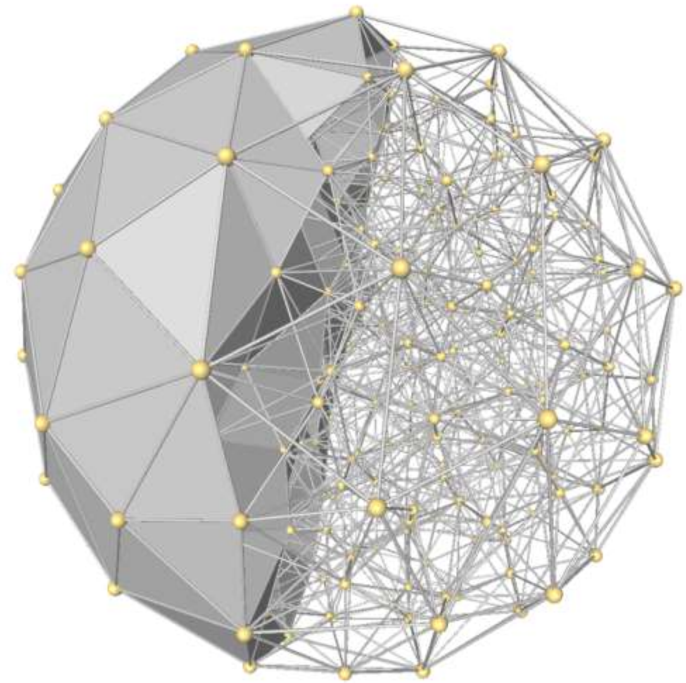
- The Delaunay triangulation of a point set E covers the convex hull of E.



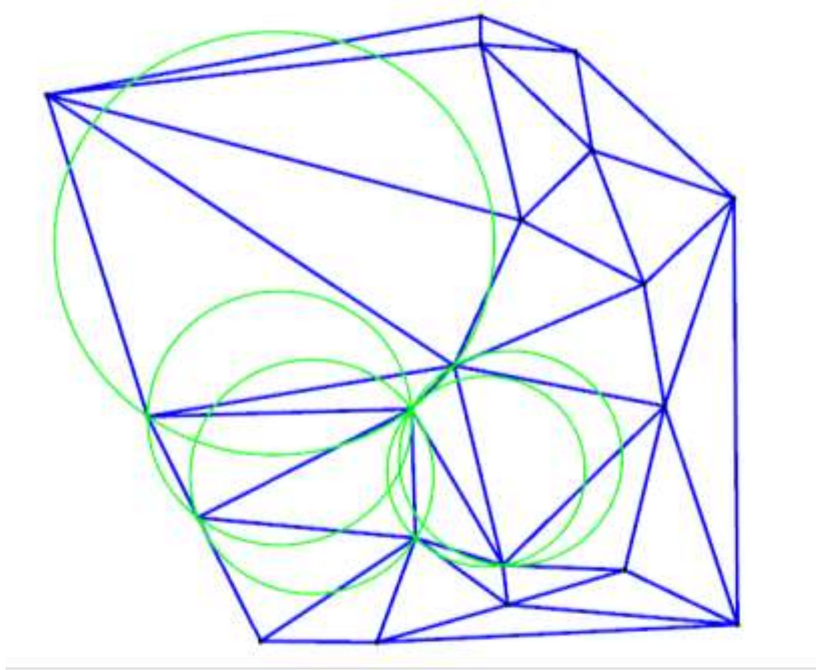
Delaunay Triangulation

- canonical triangulation associated to any point set





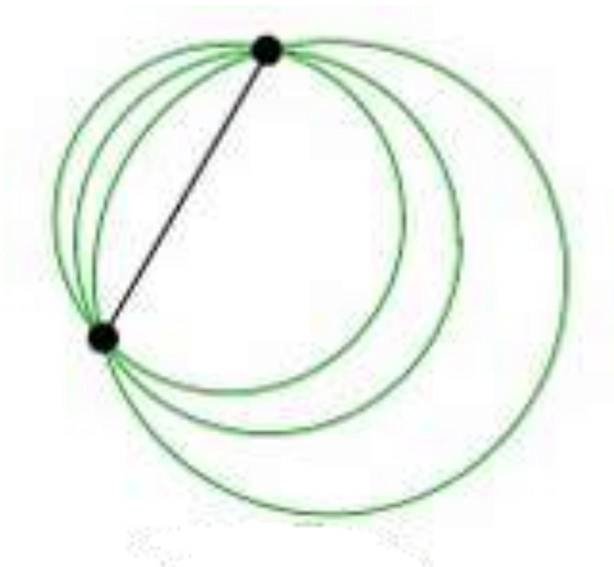
Local Property



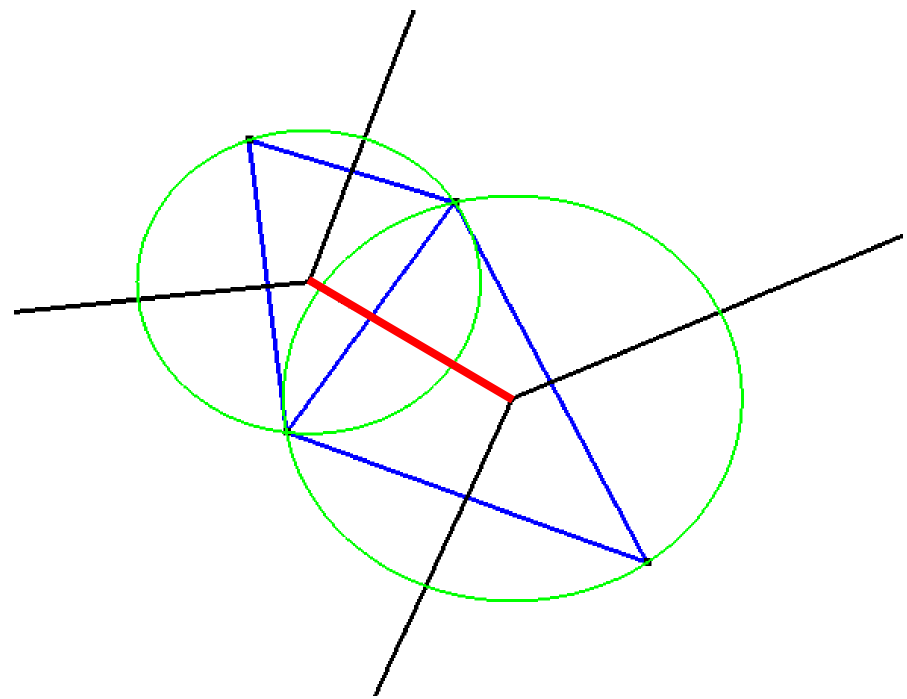
demo

Empty circle: A triangulation T of a point set E such that any d-simplex of T has a circumsphere that does not enclose any point of E is a Delaunay triangulation of E. Conversely, any k-simplex with vertices in E that can be circumscribed by a hypersphere that does not enclose any point of E is a face of the Delaunay triangulation of E.

Empty Circles



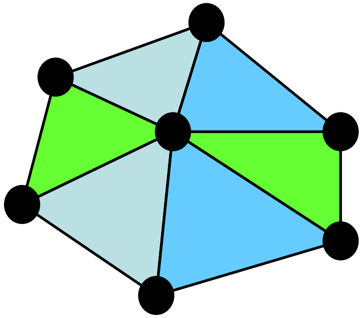
Circumscribing circles
-> pencil of circles



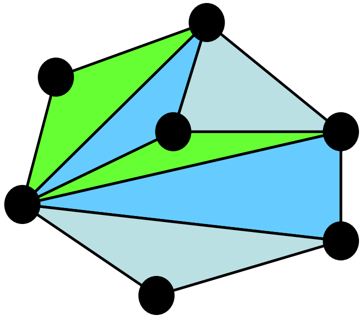
Voronoi edge
Locus of circle centers

Global Properties

- **In 2D**: « **quality** » **triangulation**
  - Smallest triangle angle: The Delaunay triangulation of a point set E is the triangulation of E which **maximizes the smallest angle**.
  - Even stronger: The triangulation of E whose **angular vector** is **maximal** for the lexicographic order is the Delaunay triangulation of E.



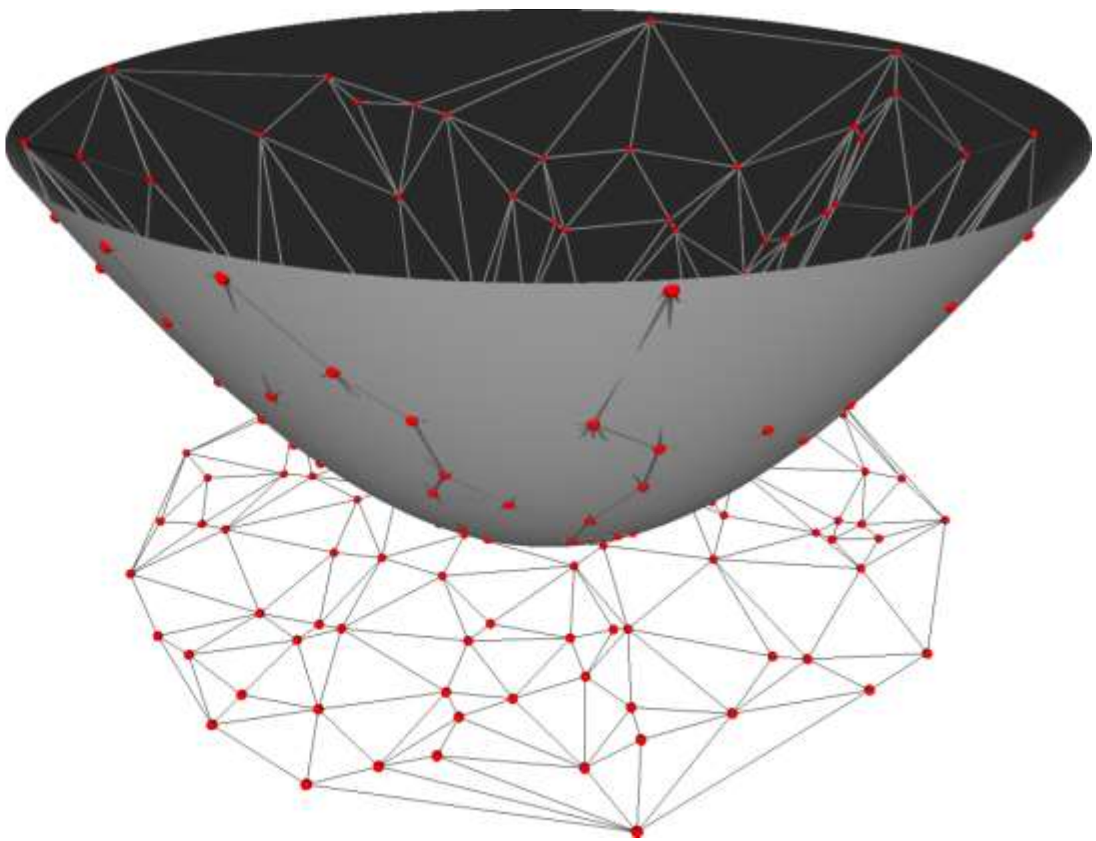
good



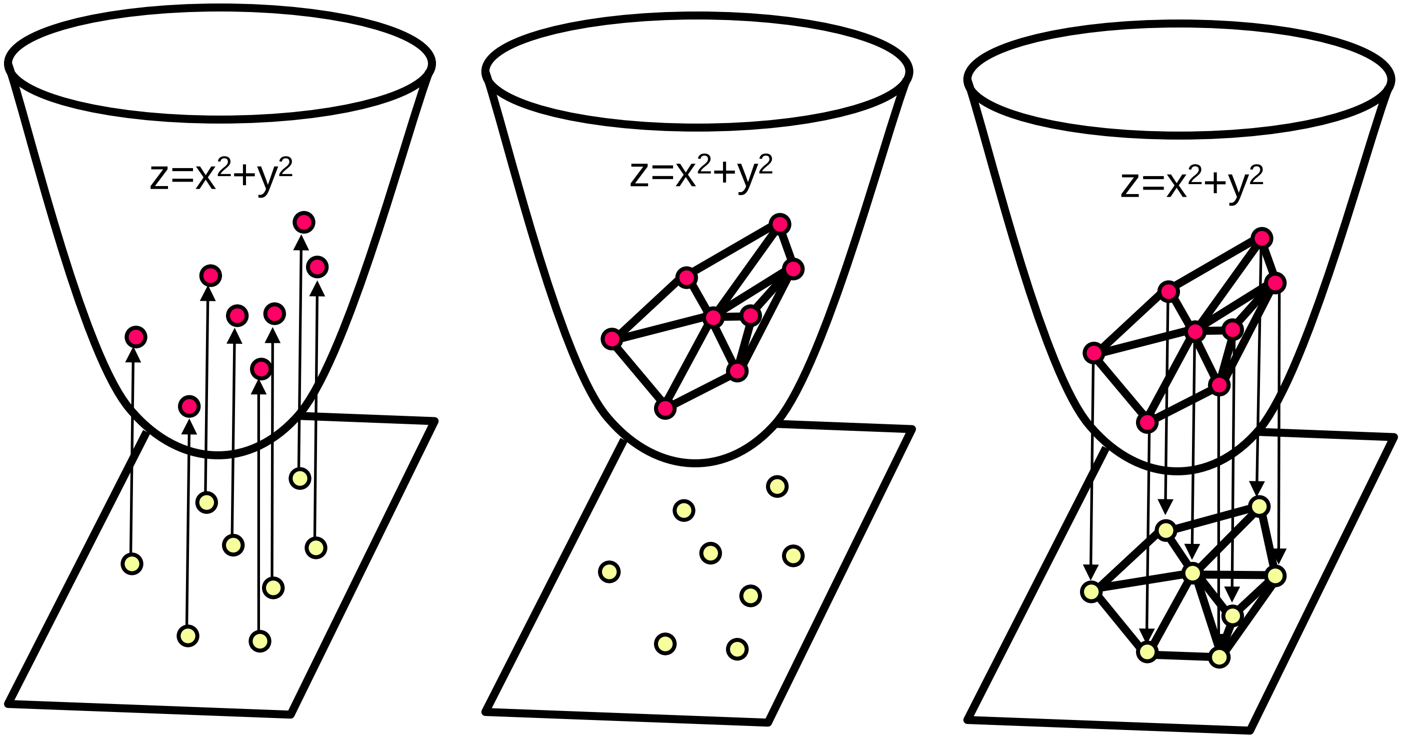
bad

Delaunay Triangulation

Duality on the paraboloid: Delaunay triangulation obtained by projecting the lower part of the convex hull.



Delaunay Triangulation

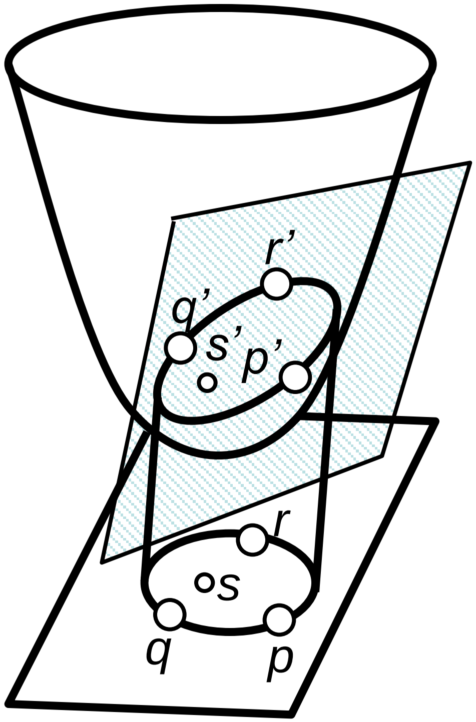


Project the 2D point set onto the 3D paraboloid → Compute the 3D lower convex hull → Project the 3D facets back to the plane.

demo

Duality

- The intersection of a plane with the paraboloid is an ellipse whose projection to the plane is a circle.
- s lies within the circumcircle of p, q, r iff s' lies on the lower side of the plane passing through p', q', r'.
- $p, q, r \in S$ form a Delaunay triangle iff p', q', r' form a face of the convex hull of S'.

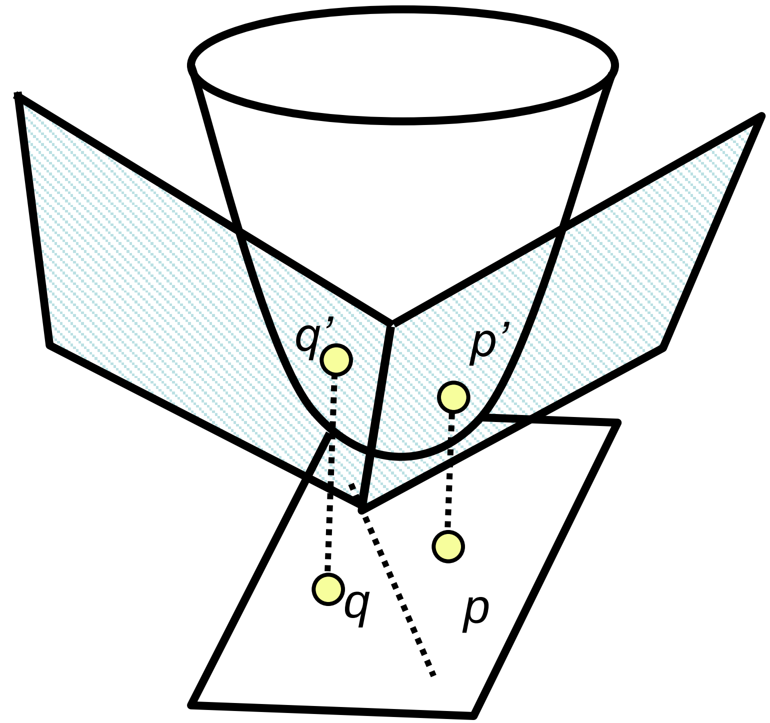


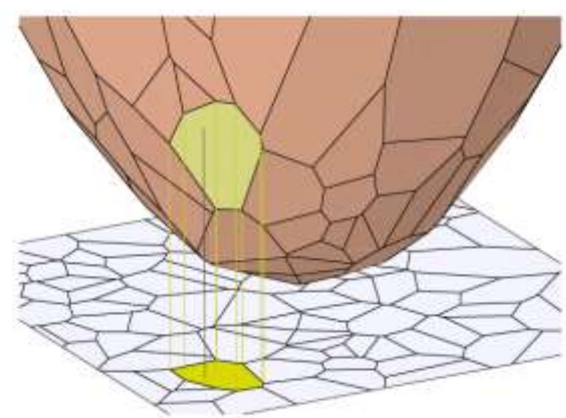
Voronoi Diagram

- Given a set S of points in the plane, associate with each point $p=(a,b)\in S$ the plane tangent to the paraboloid at p:

$$z = 2ax+2by-(a2+b2).$$

- VD(S) is the projection to the (x,y) plane of the 1-skeleton of the convex polyhedron formed from the intersection of the halfspaces above these planes.

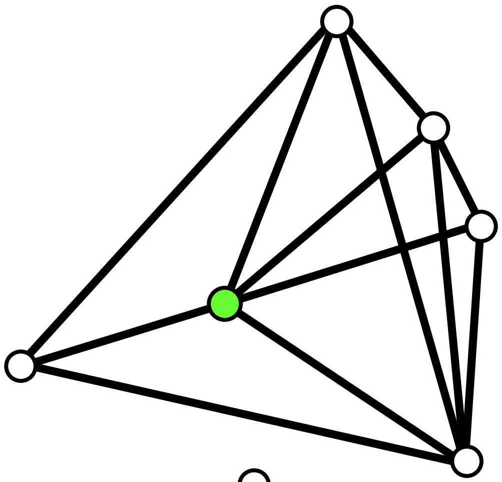


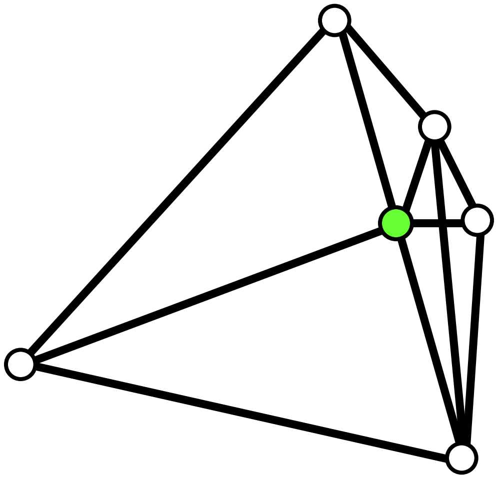


O($n$log$n$) Delaunay Triangulation Algorithm

Incremental algorithm:

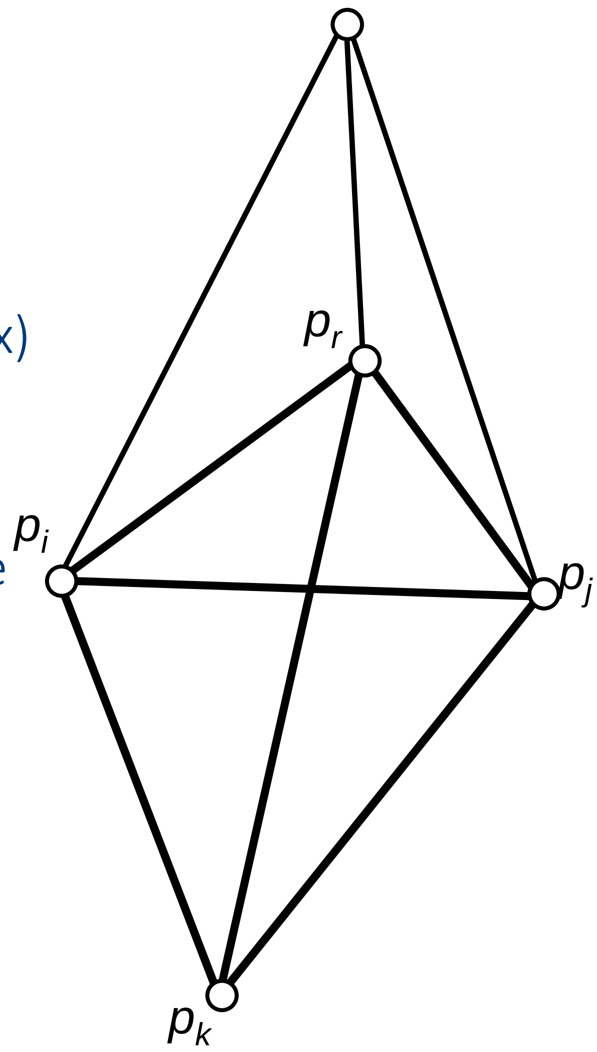
- Form bounding triangle which encloses all the sites.
- Add the sites one after another in random order and update triangulation.
- **If the site is inside an existing triangle:**
  - Connect site to triangle vertices.
  - Check if a 'flip' can be performed on one of the triangle edges. If so – check recursively the neighboring edges.
- **If the site is on an existing edge:**
  - Replace edge with four new edges.
  - Check if a 'flip' can be performed on one of the opposite edges. If so – check recursively the neighboring edges.



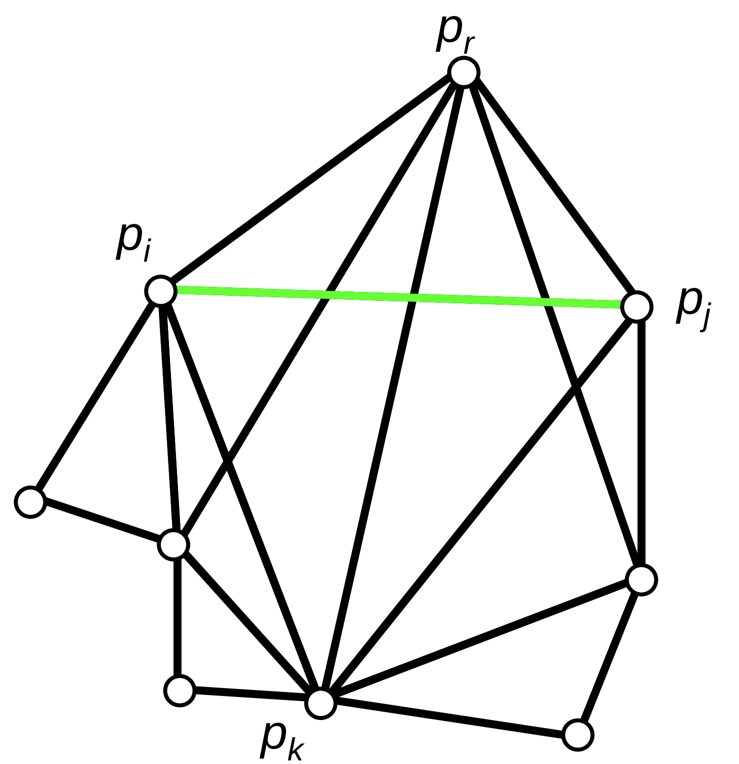


Flipping Edges

- A new vertex p_r is added, causing the creation of edges.
- The legality of the edge $p_i p_j$ (with opposite vertex) p_k is checked.
- If $p_i p_j$ is illegal, perform a flip, and recursively check edges $p_i p_k$ and $p_j p_k$, the new edges opposite p_r.
- Notice that the recursive call for $p_i p_k$ cannot eliminate the edge $p_r p_k$.
- **Note:** All edge flips replace edges opposite the new vertex by edges incident to it!

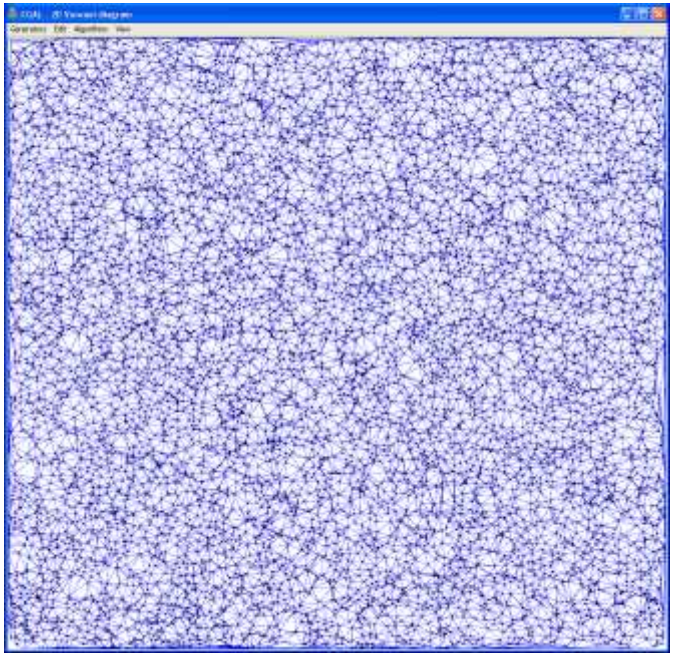


Flipping Edges - Example



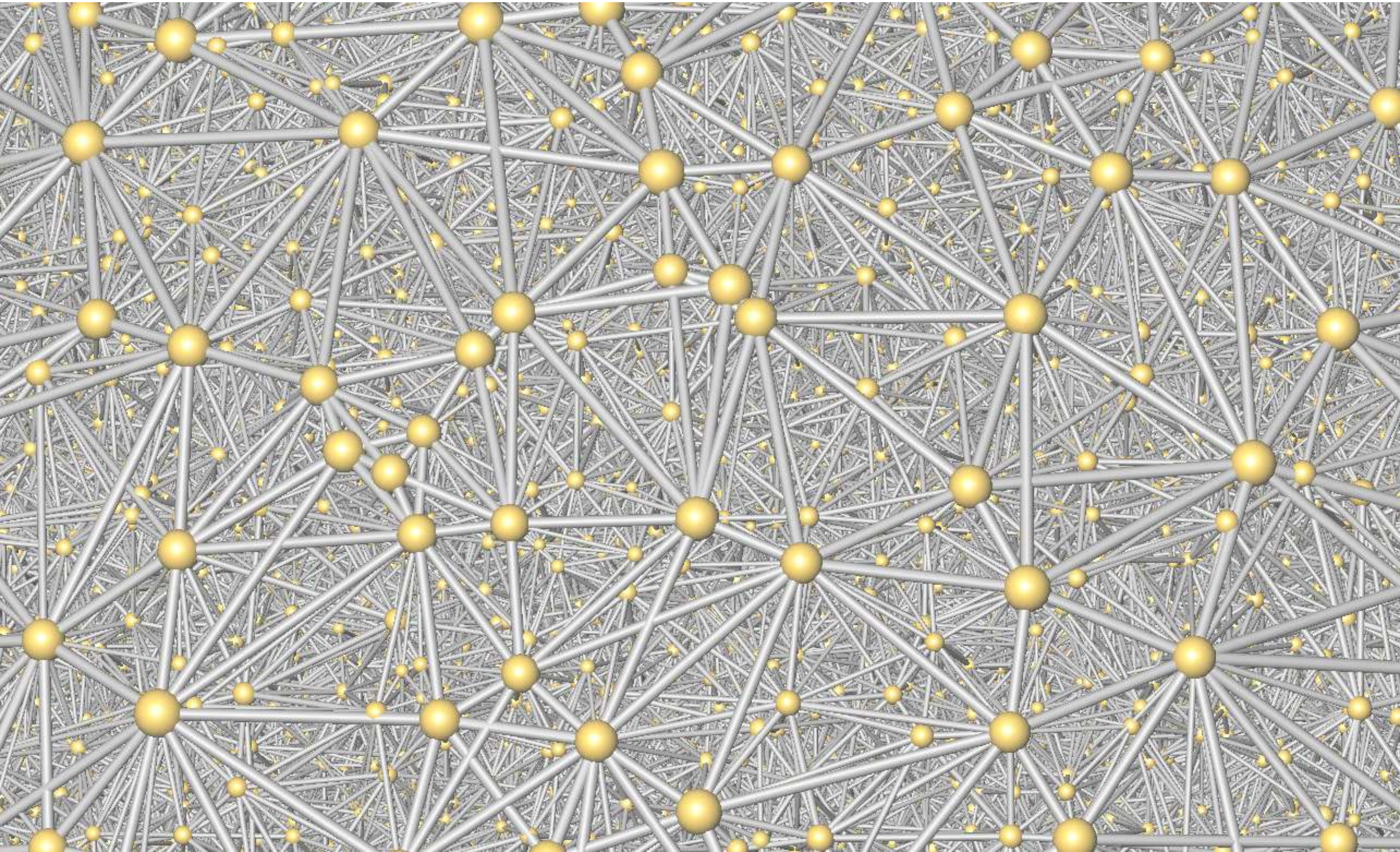
Algorithm Complexity

- **Point location for every point**: O(log n) time.
- **Flips**: Θ(n) expected time in total (for all steps).
- **Total expected time**: O(n log n).
- **Space**: Θ(n).



demo

3D Delaunay Triangulation

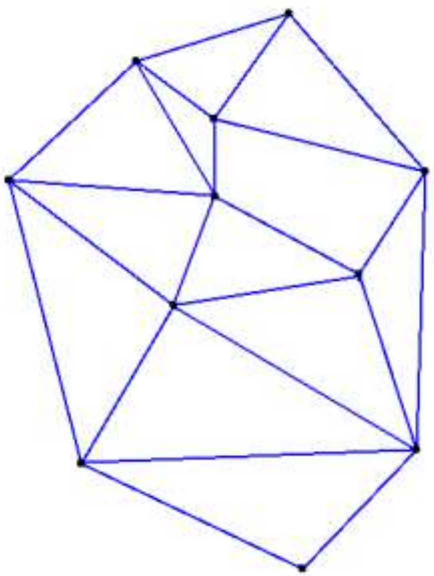


Variants

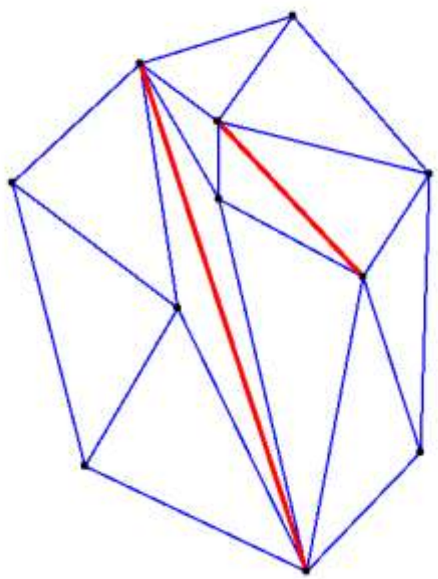
- Constraints
- Periodic
- Weighted
- Generators: segments, circles

2D Constrained Delaunay Triangulation

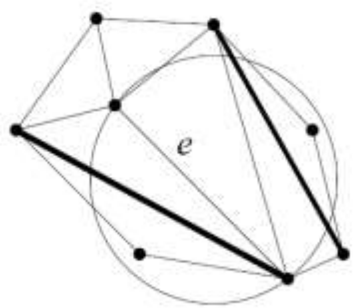
Let (P, S) be a PSLG. The constrained triangulation T(P, S) is constrained Delaunay iff the circumcircle of any triangle t of T encloses no vertex visible from a point in the relative interior of t.

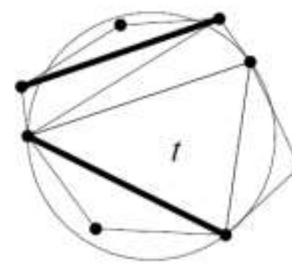


unconstrained



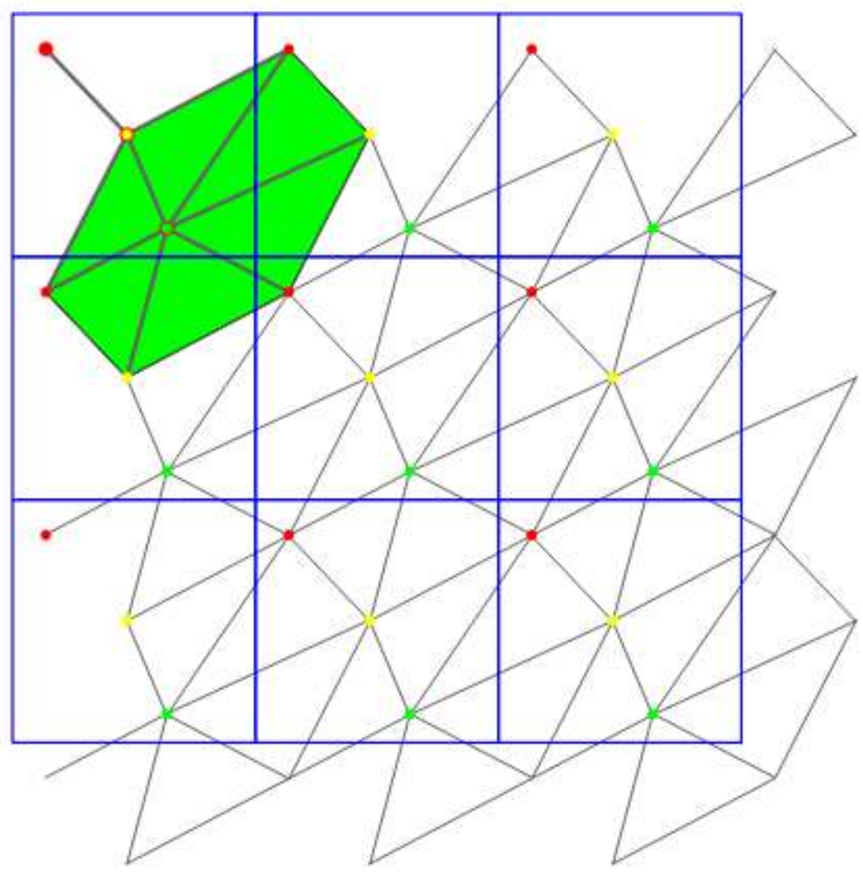
constrained





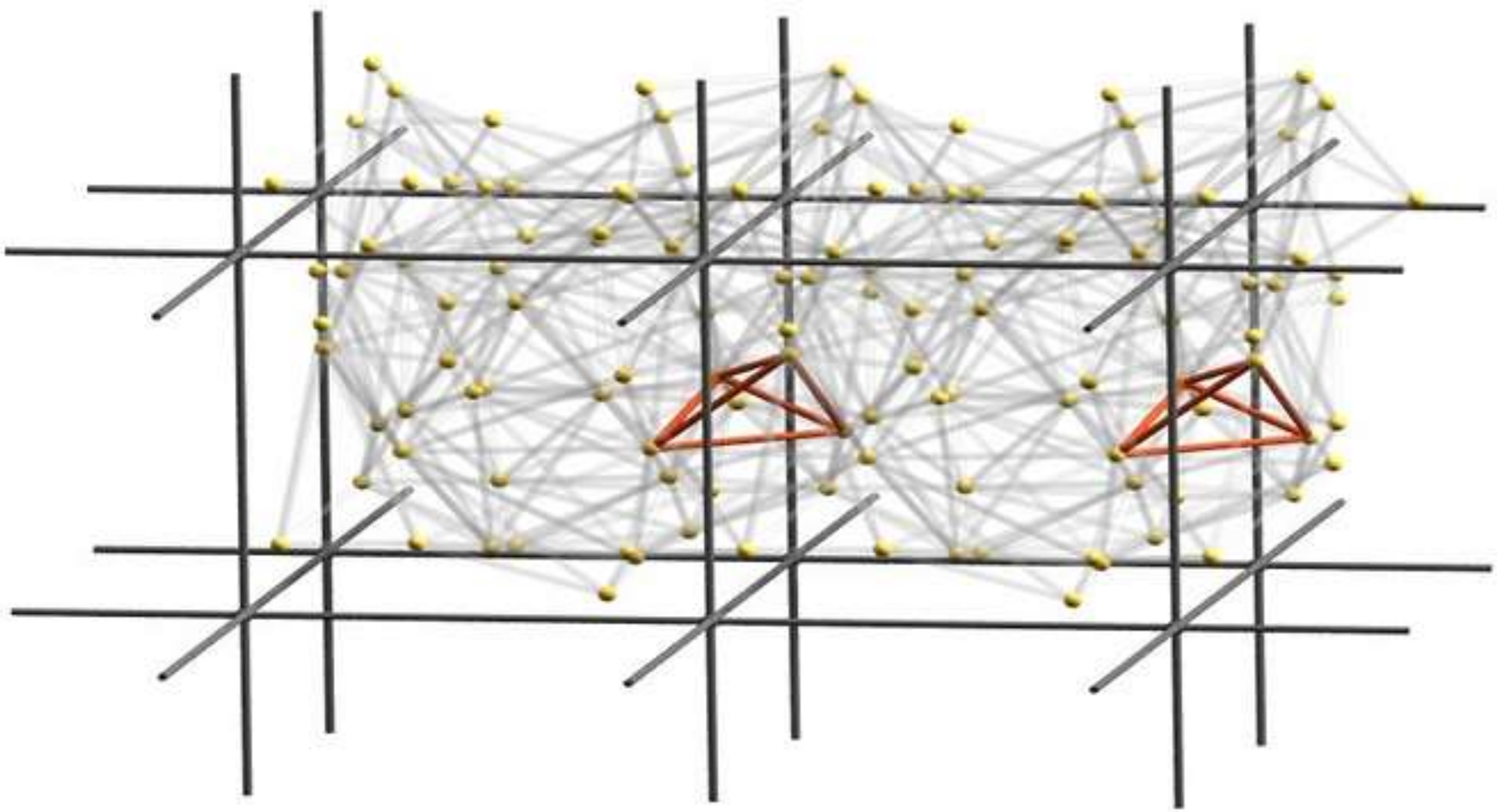
Periodic Delaunay Triangulation

Points in 2D flat torus

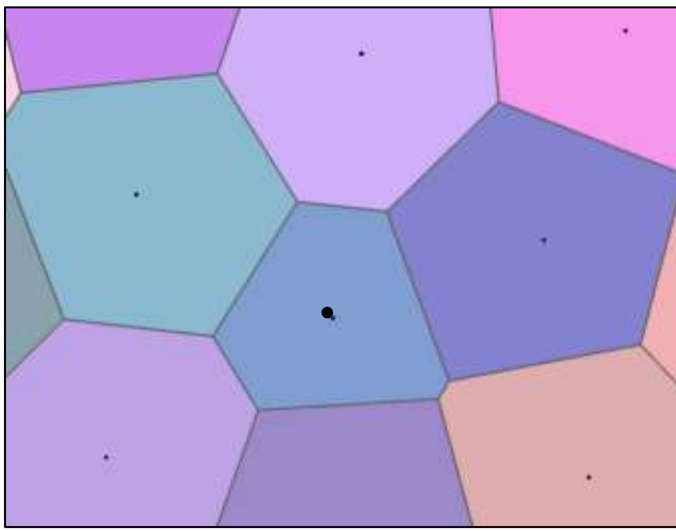


Periodic Delaunay Triangulation

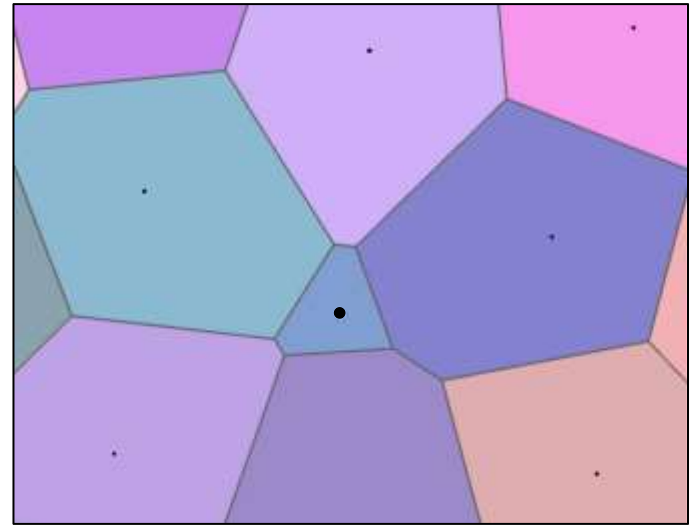
- Points in 3D flat torus



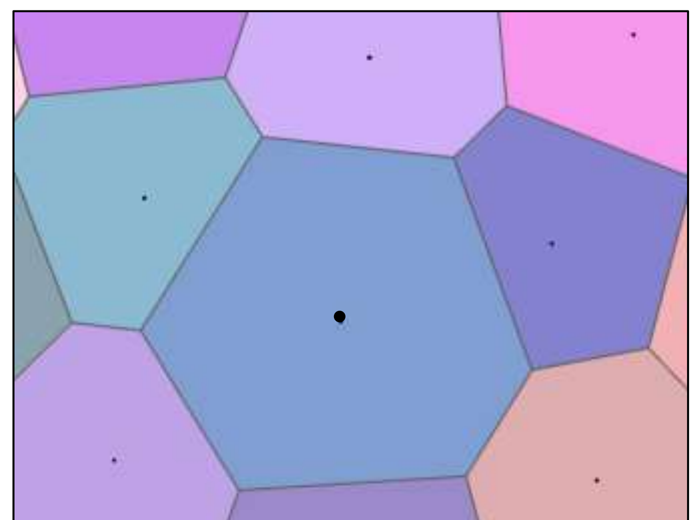
Power Diagram



Unweighted

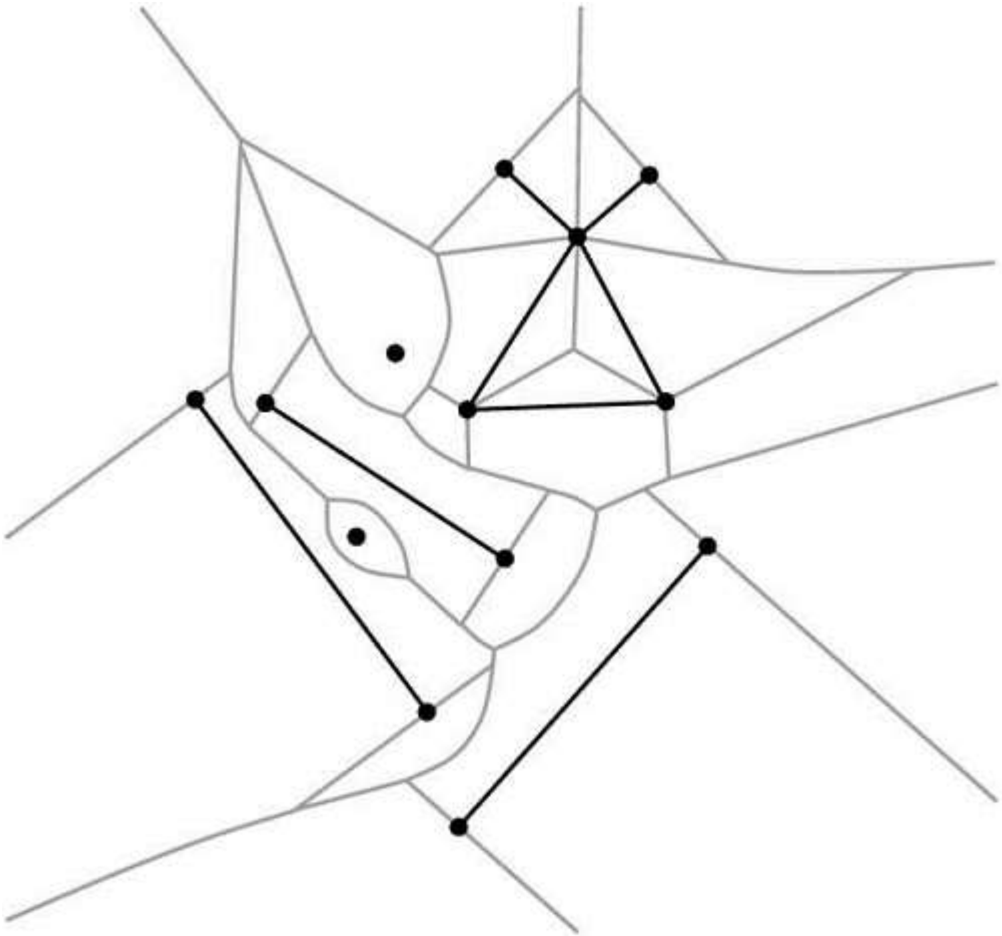


Small weight

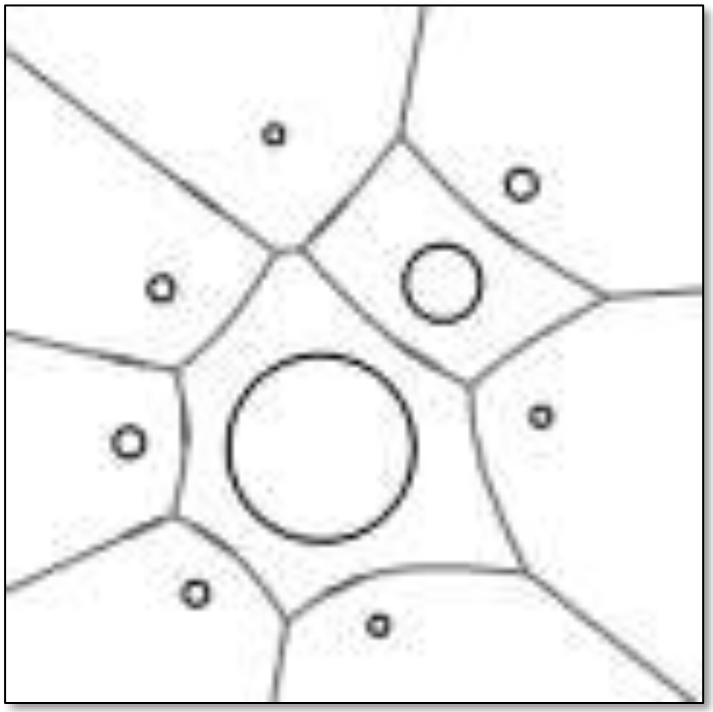


Large weight

Generators = Line Segments



Apollonius Diagram / Graph



Shape Reconstruction

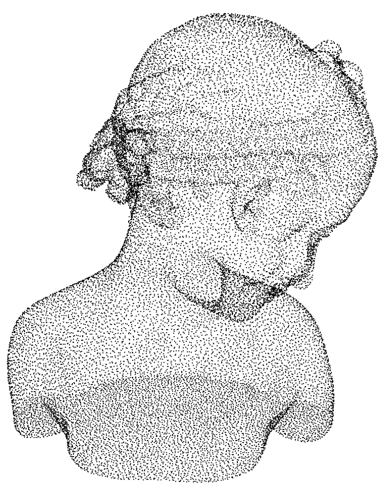
Reconstruction Problem

Input: point set *P* sampled over a surface *S*:

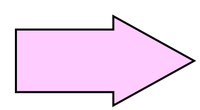
- Non-uniform sampling
- With holes
- With uncertainty (noise)

Output: surface

- Approximation of *S* in terms of topology and geometry
- Desired:
  - Watertight
  - Intersection free



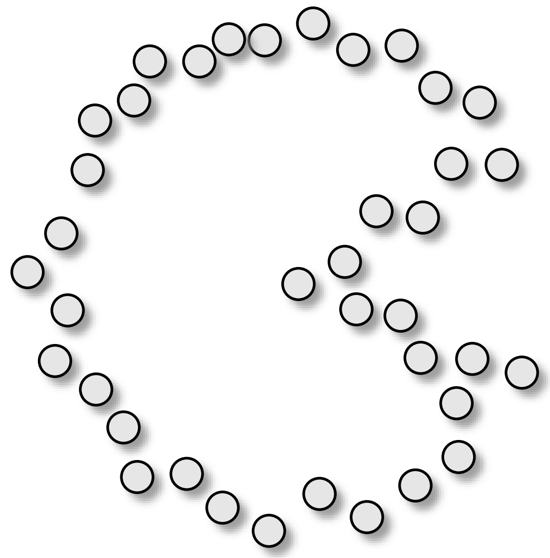
point set



reconstruction

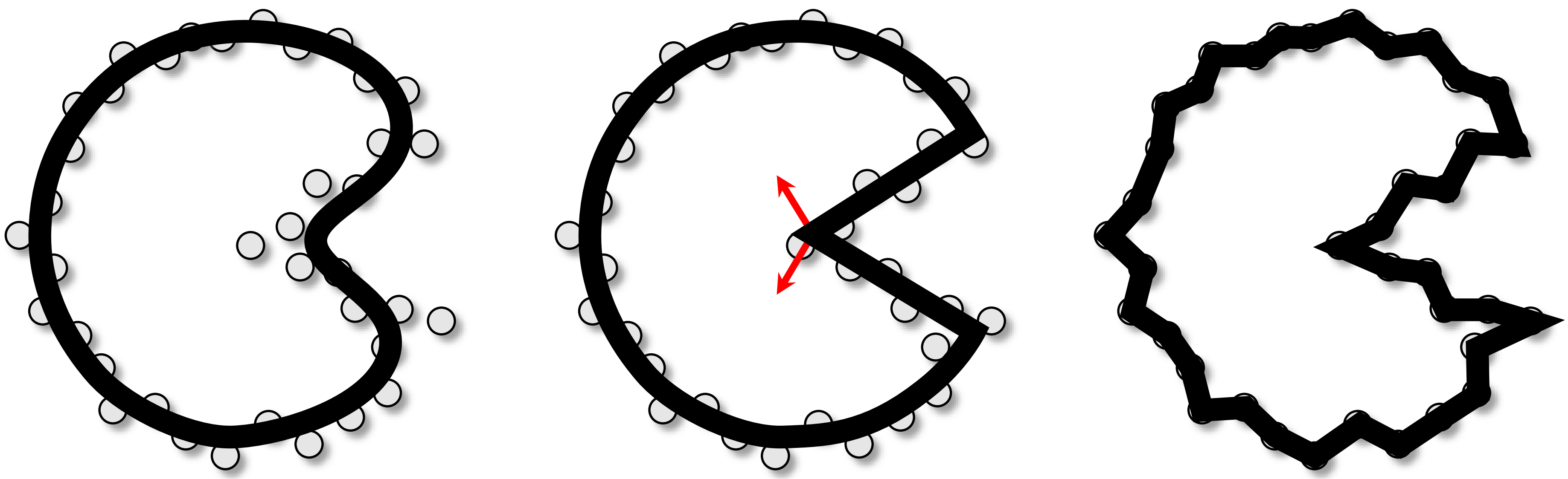
surface

Ill-posed Problem



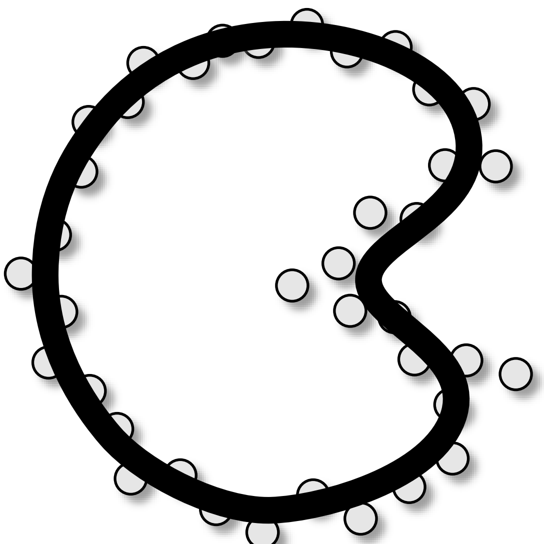
Many candidate shapes for the
reconstruction problem!

Ill-posed Problem

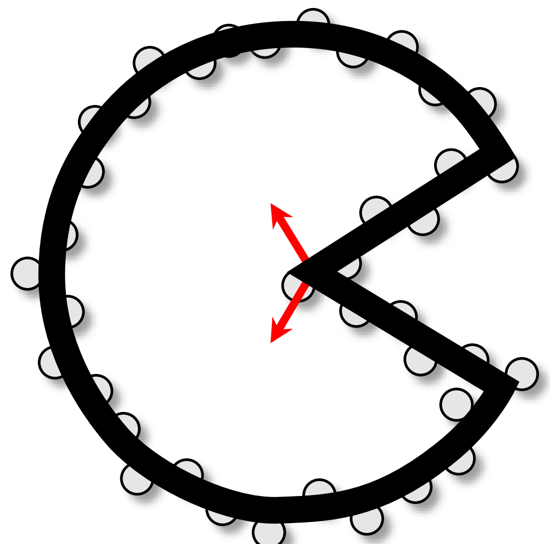


Many candidate shapes for the reconstruction problem! How to pick?

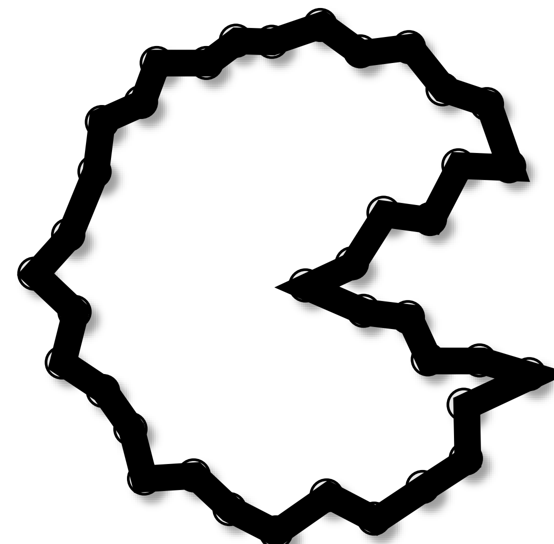
Priors



Smooth



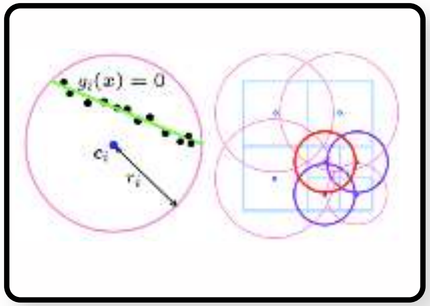
Piecewise Smooth



“Simple”

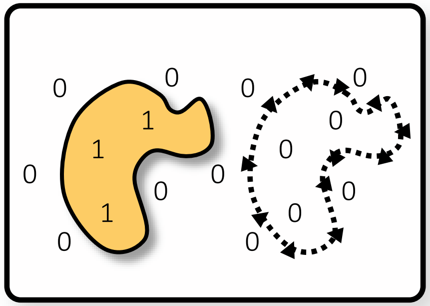
Surface Smoothness Priors

Local Smoothness



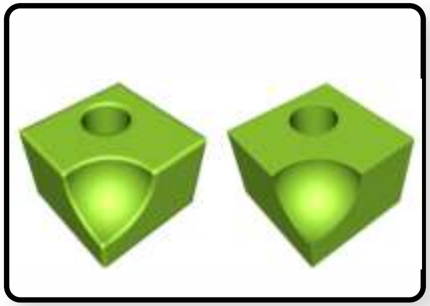
Local fitting
No control away from data
Solution by interpolation

Global Smoothness



Global: linear, eigen, graph cut, …
Robustness to missing data

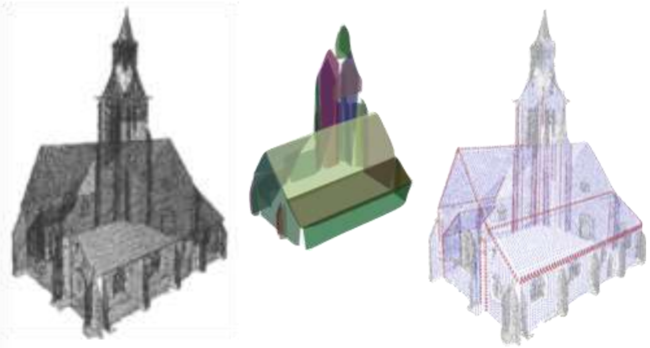
Piecewise Smoothness



Sharp near features
Smooth away from features

Domain-Specific Priors

Surface Reconstruction
by Point Set Structuring

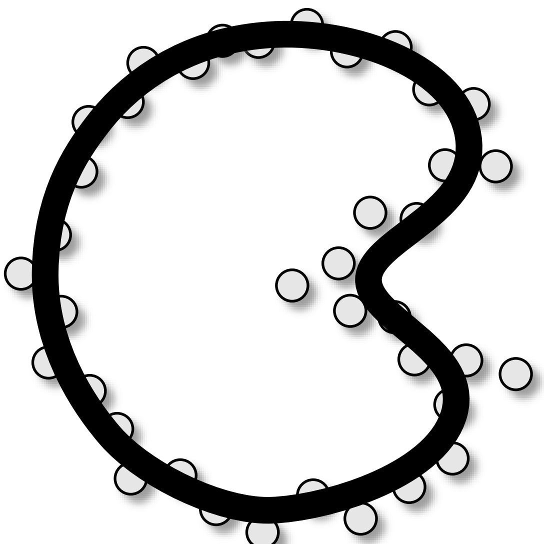


[Lafarge - A. EUROGRAPHICS 2013]

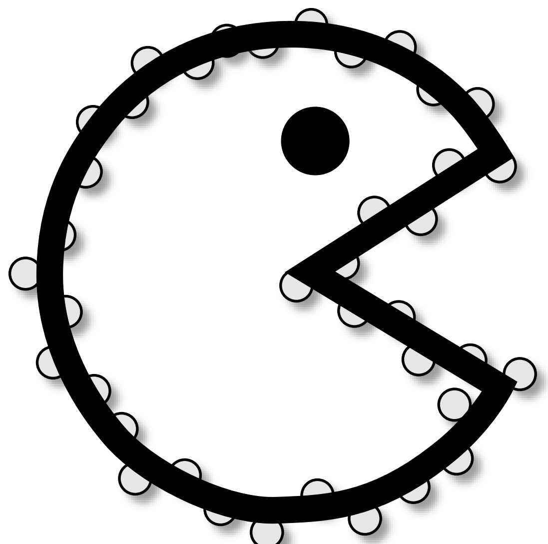
LOD Reconstruction
for Urban Scenes

[Verdie, Lafarge - A. ACM Transactions on Graphics 2015]

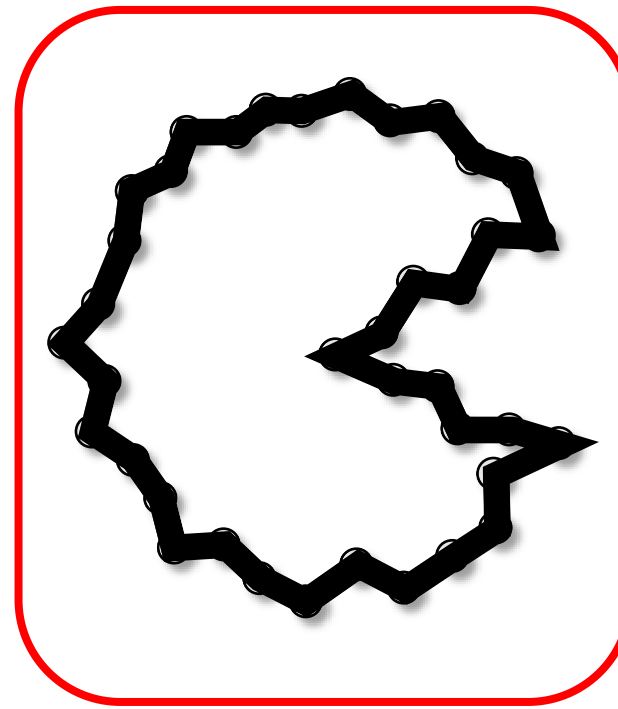
Warm-up



Smooth



Piecewise Smooth

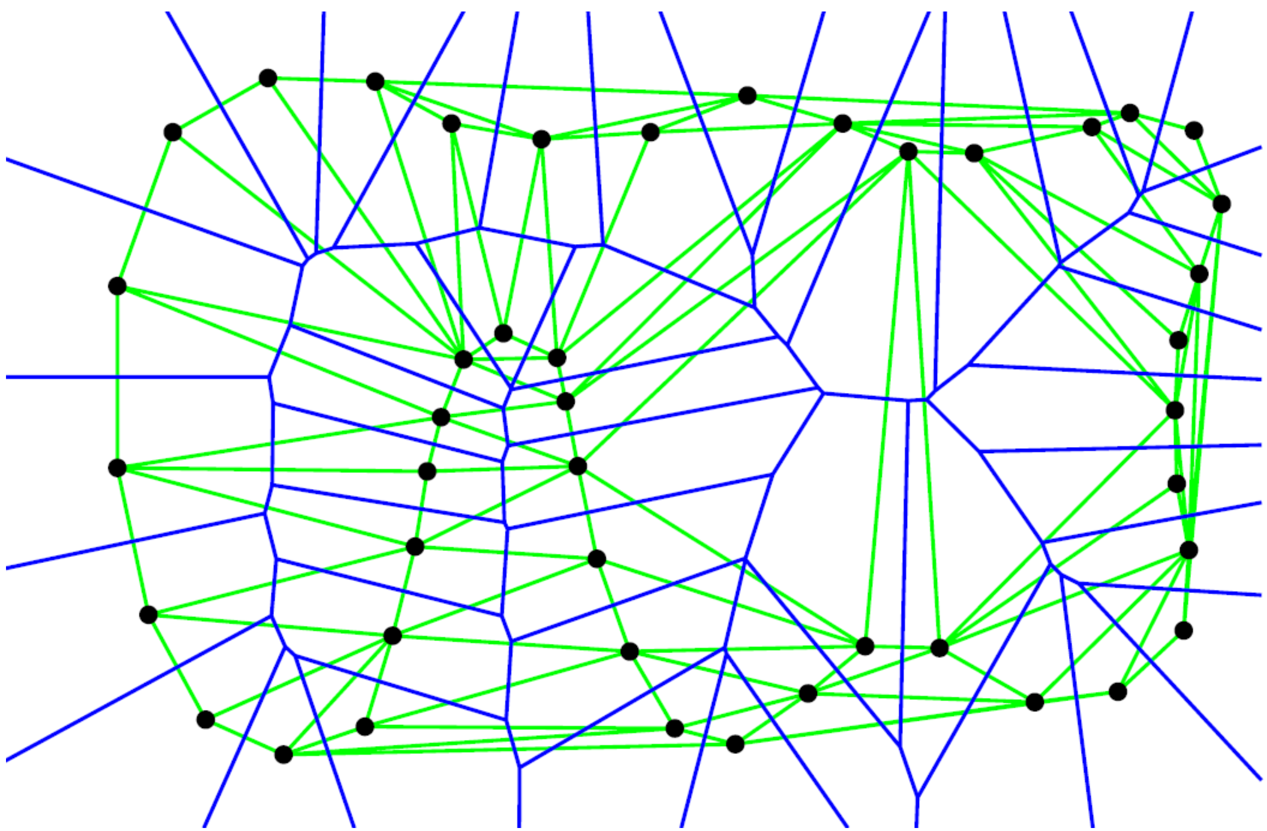


“Simple”

Voronoi / Delaunay

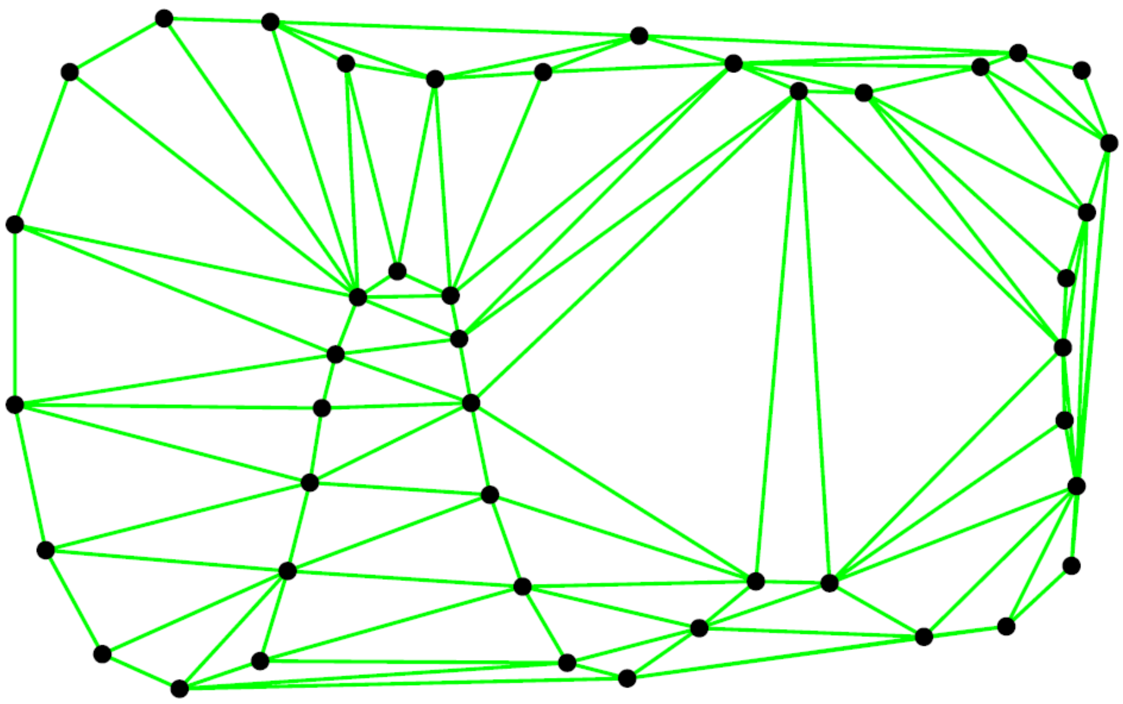
Let $\mathcal{E} = \{\mathbf{p_1}, \ldots, \mathbf{p_n}\}$ be a set of points (so-called sites) in $\mathbb{R}^d$. We associate to each site $\mathbf{p_i}$ its Voronoi region $V(\mathbf{p_i})$ such that:

$$V(\mathbf{p_i}) = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x} - \mathbf{p_i}\| \leq \|\mathbf{x} - \mathbf{p_j}\|, \forall j \leq n\}.$$



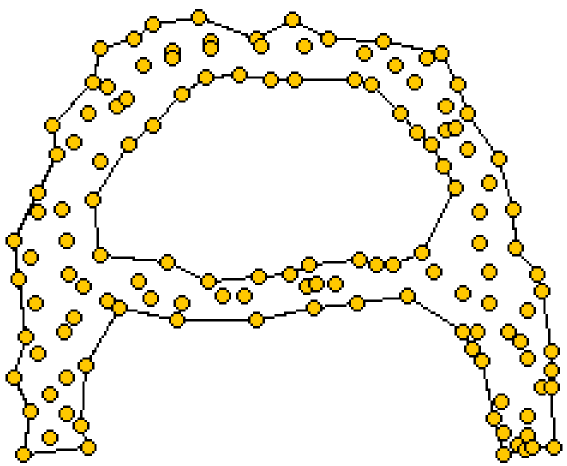
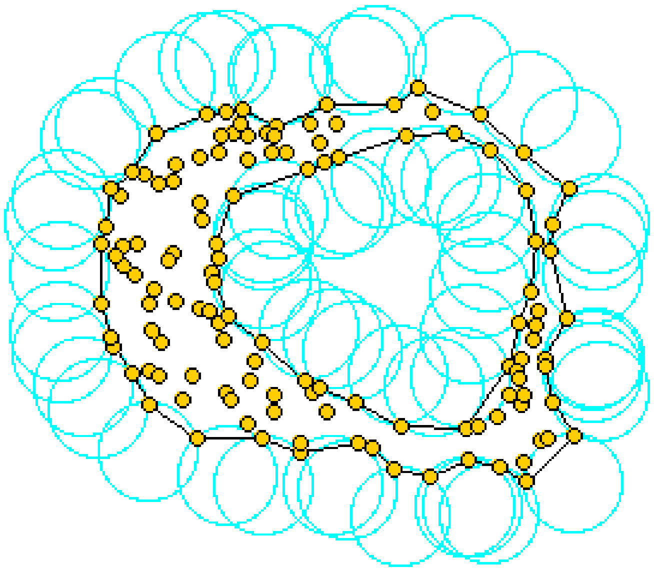
Delaunay-based

- **Key idea**: assuming dense enough sampling, reconstructed edges are Delaunay edges.



Alpha-shapes

Alpha-Shapes

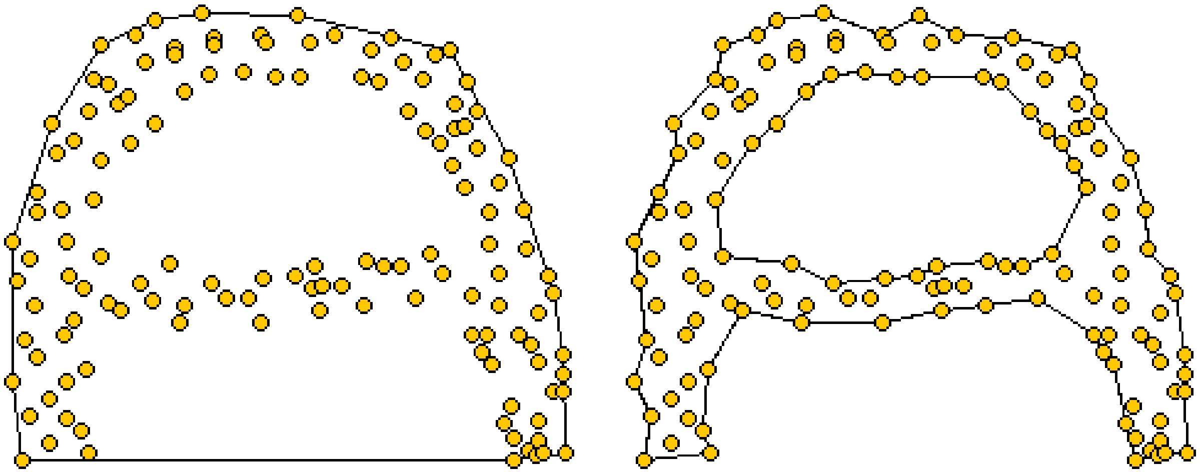
 

Segments: point pairs that can be touched by an empty disc of radius alpha.

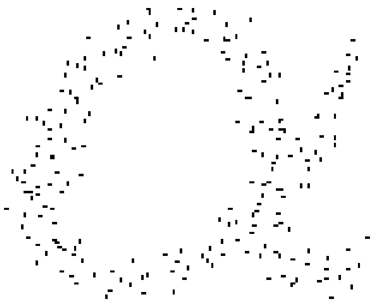
[Edelsbrunner et al.]

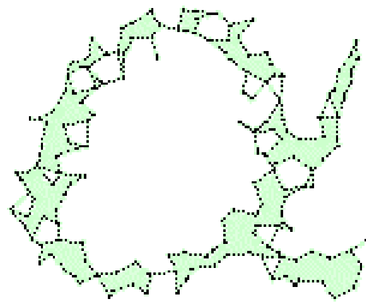
Alpha-Shapes

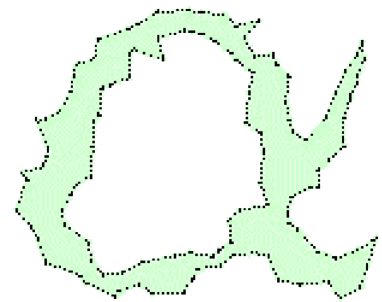
- In 2D: family of piecewise linear simple curves constructed from a point set P.
- Subcomplex of the Delaunay triangulation of P.
- Generalization of the concept of the convex hull.



Alpha-Shapes

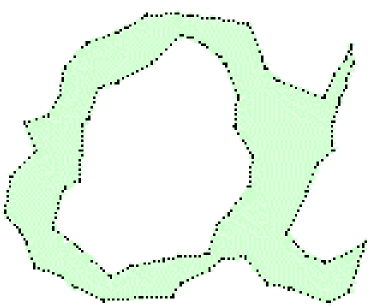


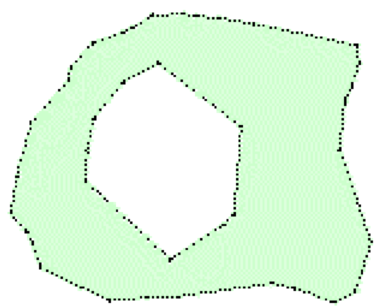


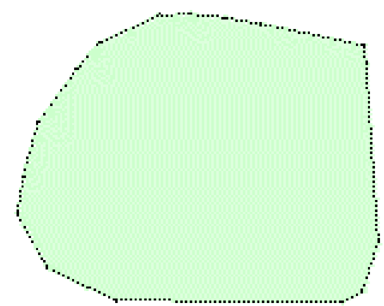


$\alpha = 0$

Alpha controls the desired level of detail







$\alpha = \infty$

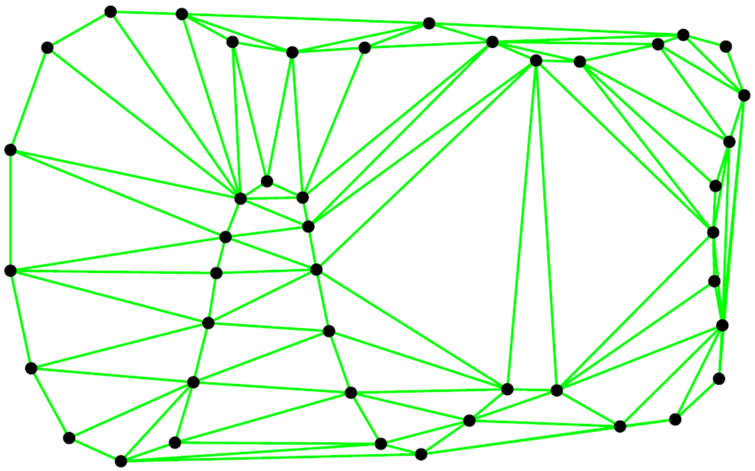
Convex hull!

Crust

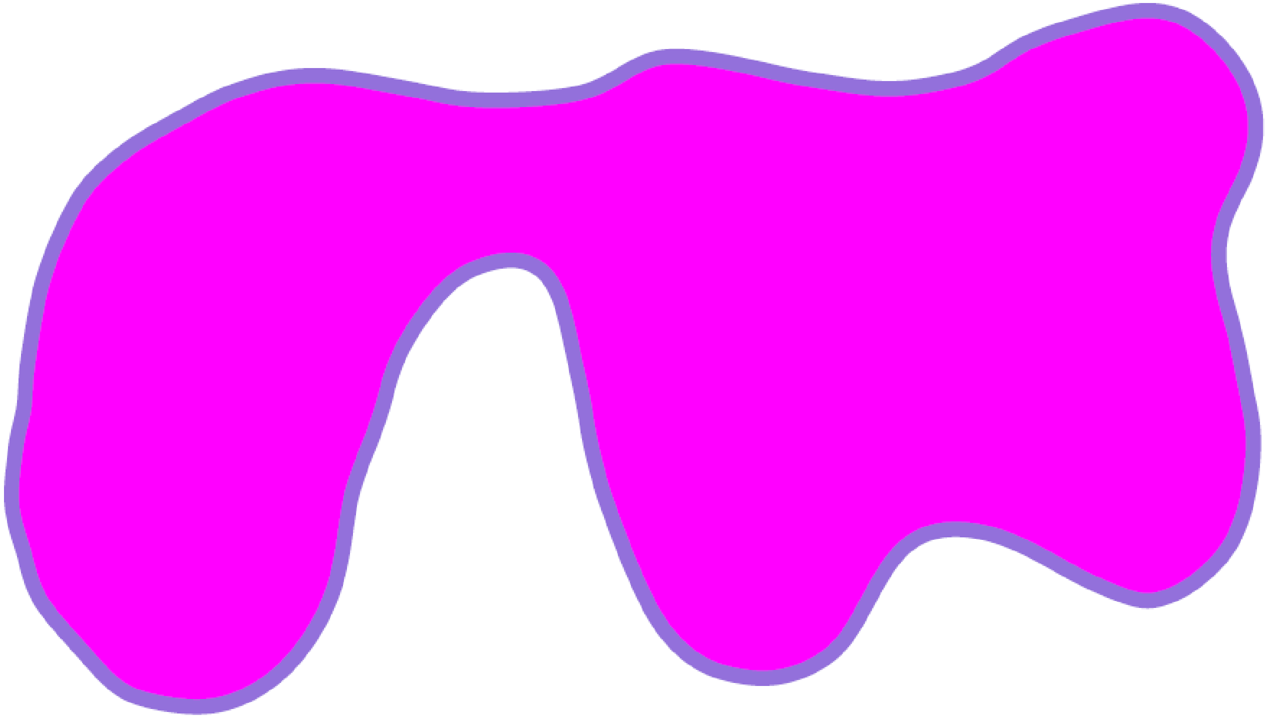
Delaunay-based

- **Key idea**: assuming dense enough sampling, reconstructed edges are Delaunay edges.

- **First define**
  - Medial axis
  - Local feature size
  - Epsilon-sampling

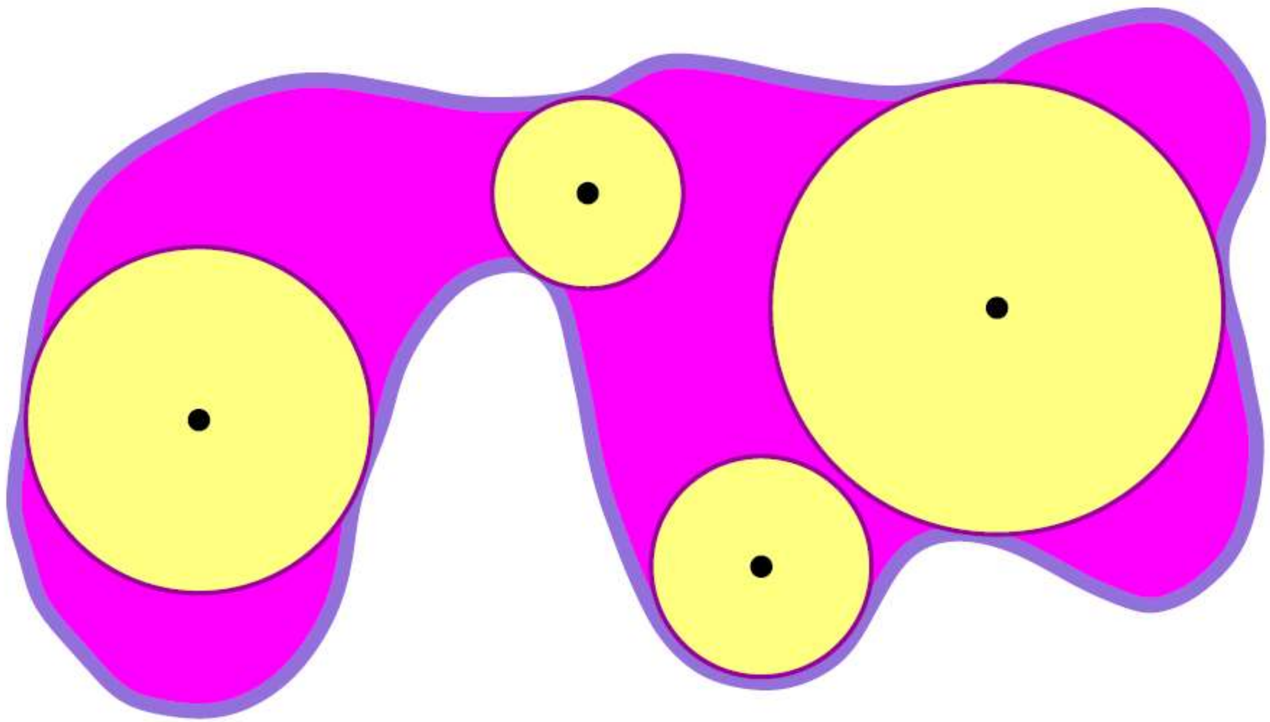


Medial Axis

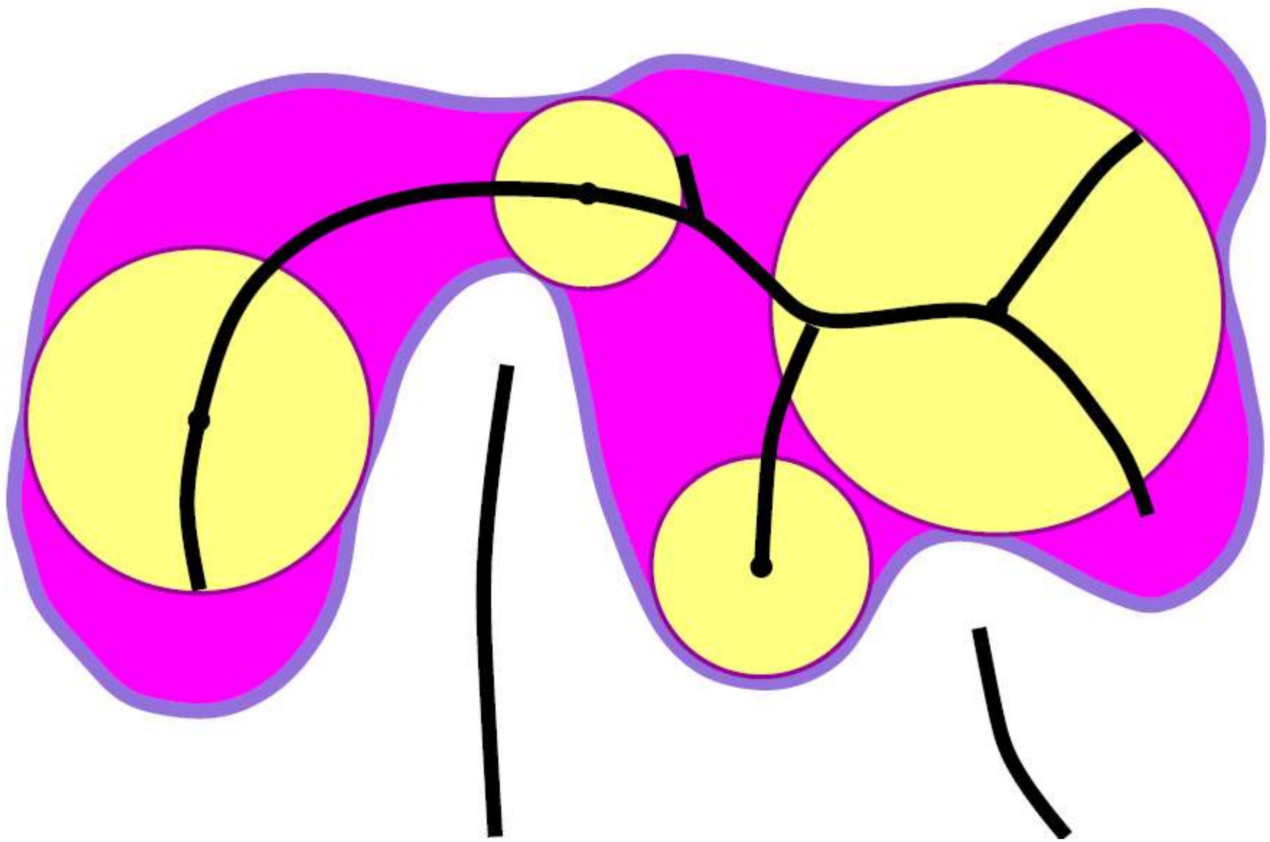


Figures from O. Devillers

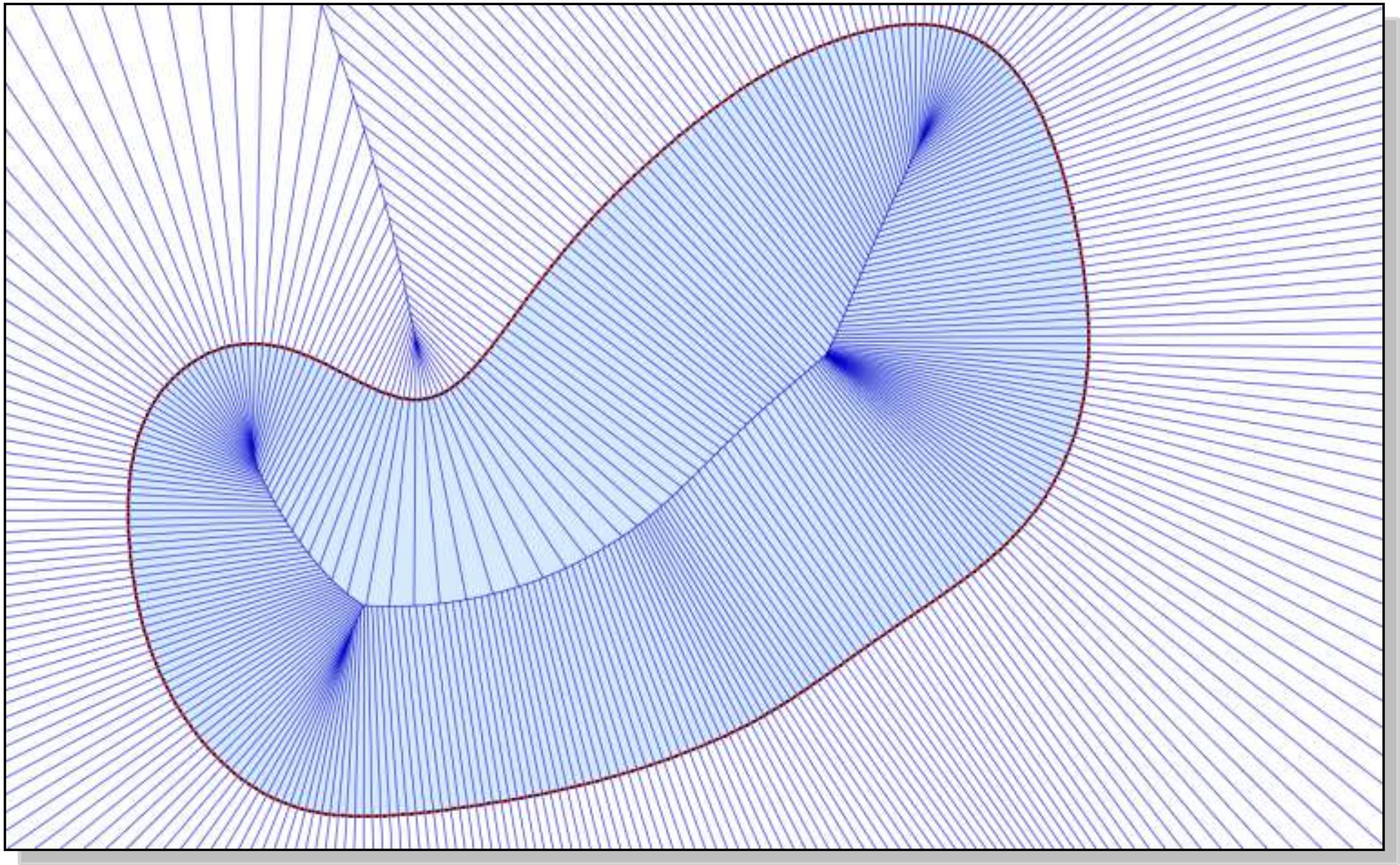
Medial Axis



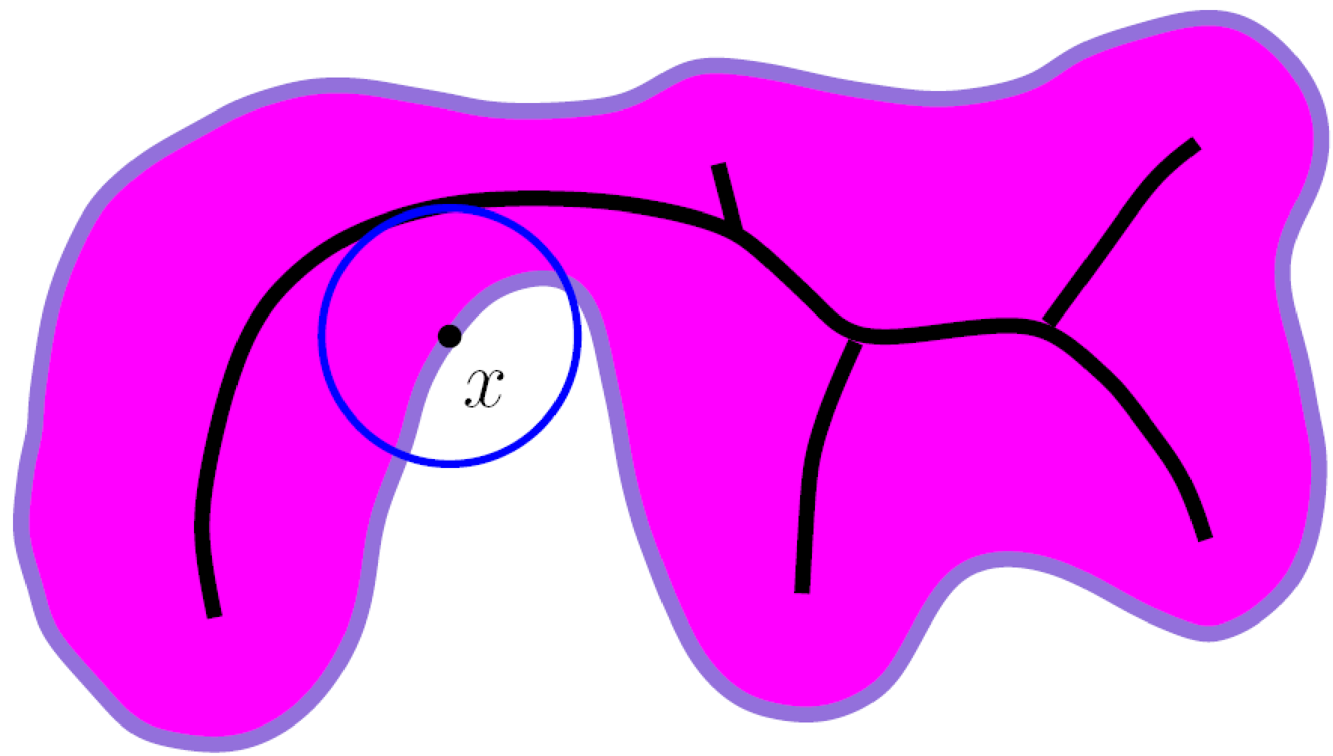
Medial Axis



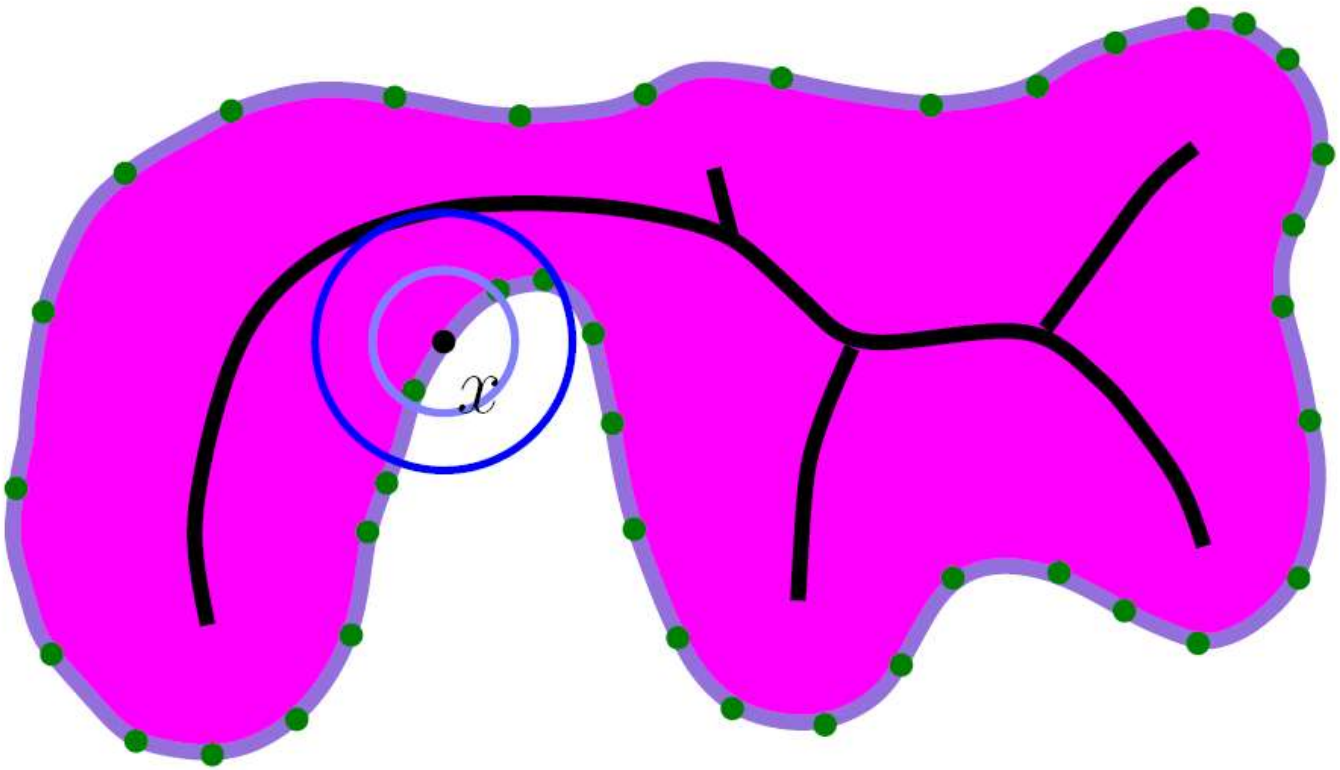
Voronoi & Medial Axis



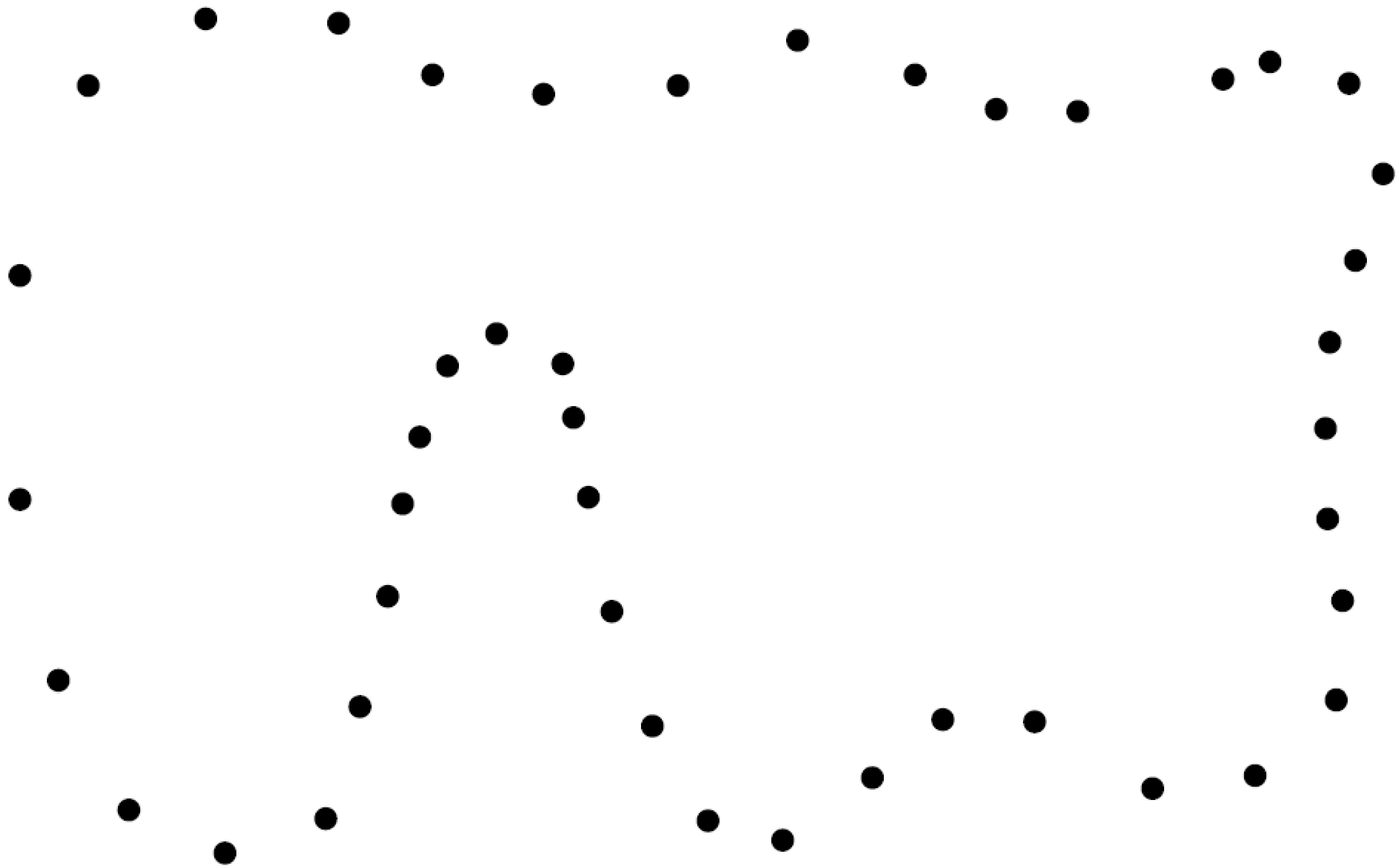
Local Feature Size



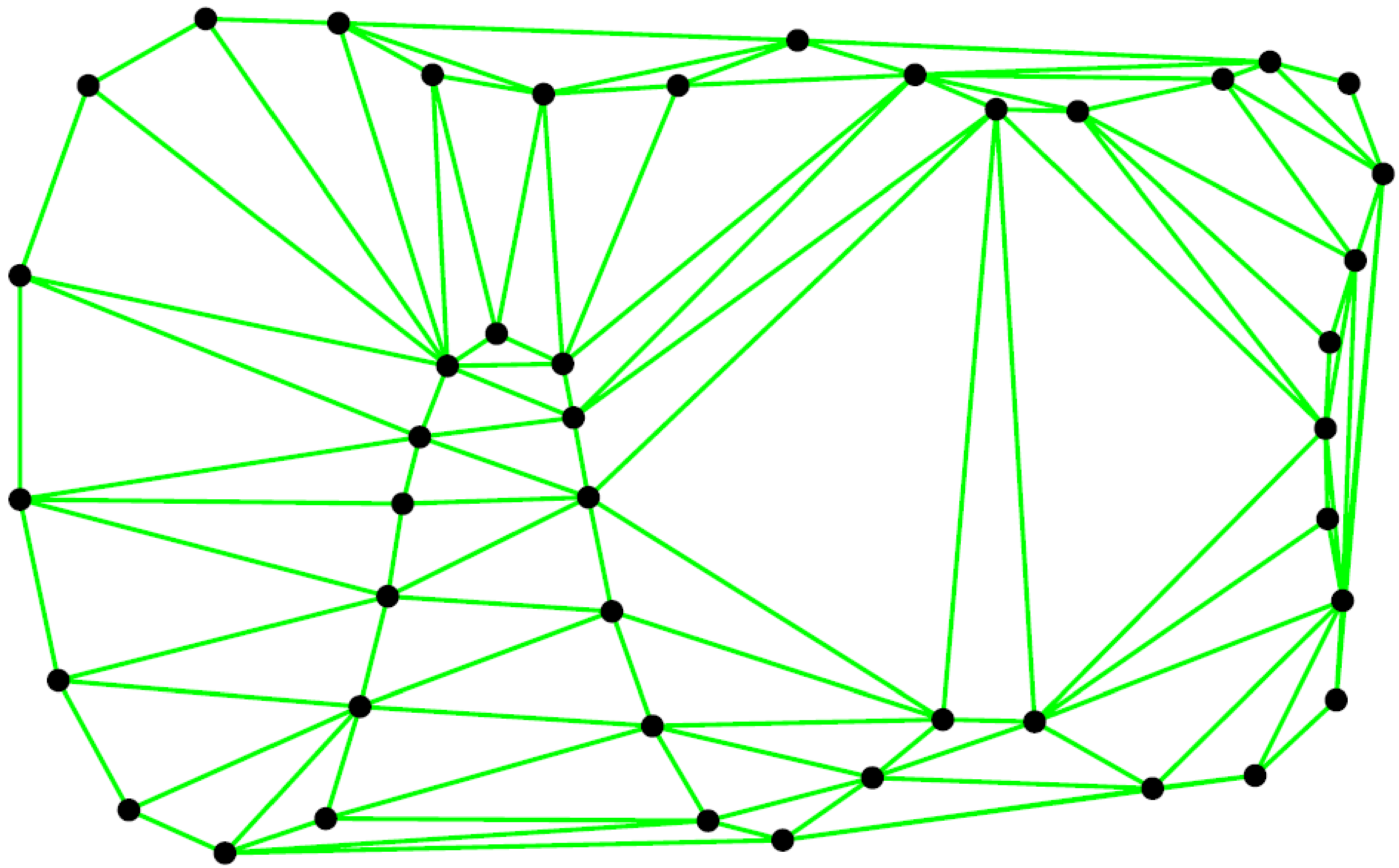
Epsilon-Sampling



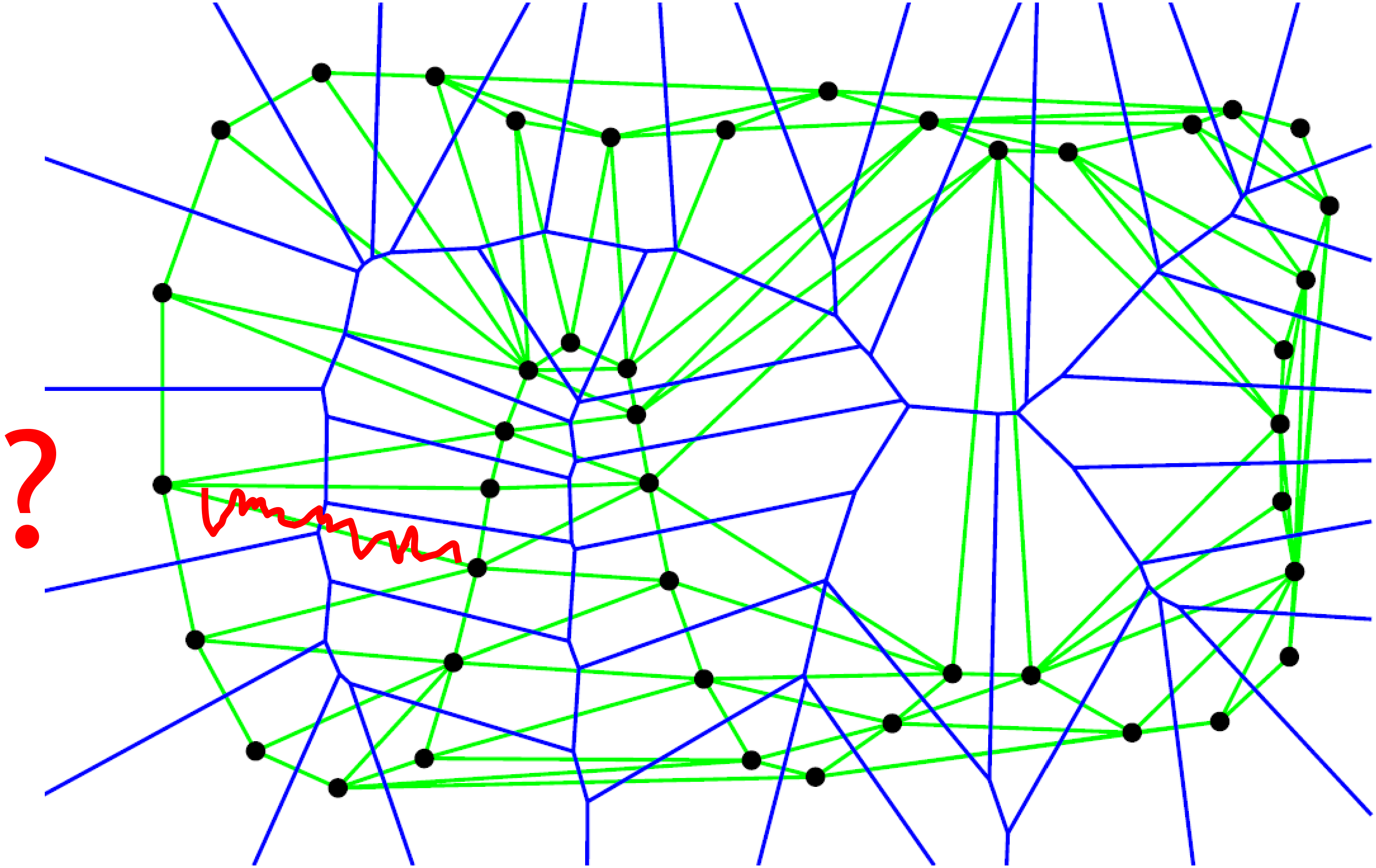
Crust [Amenta et al.]



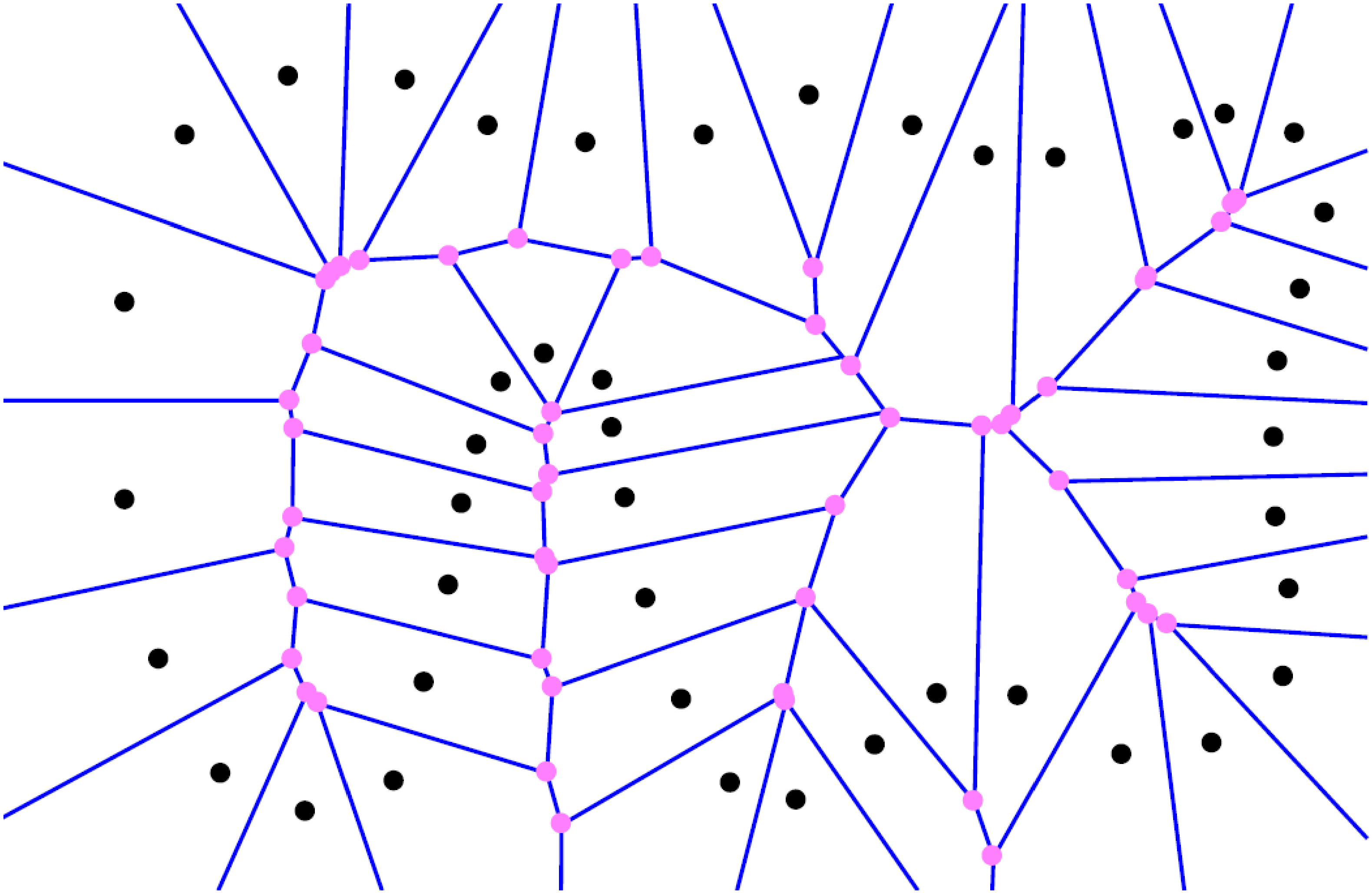
Delaunay Triangulation



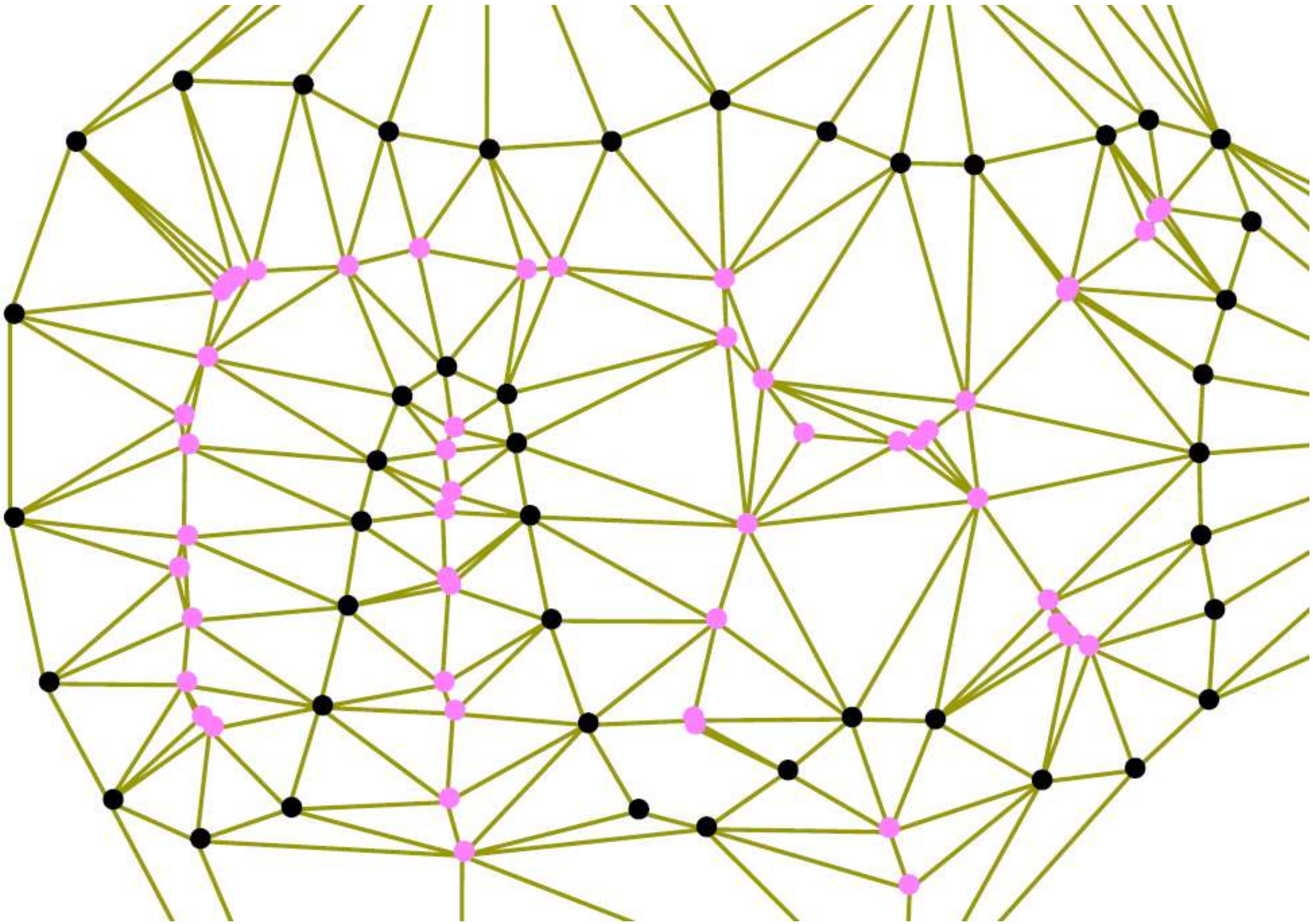
Delaunay Triangulation & Voronoi Diagram



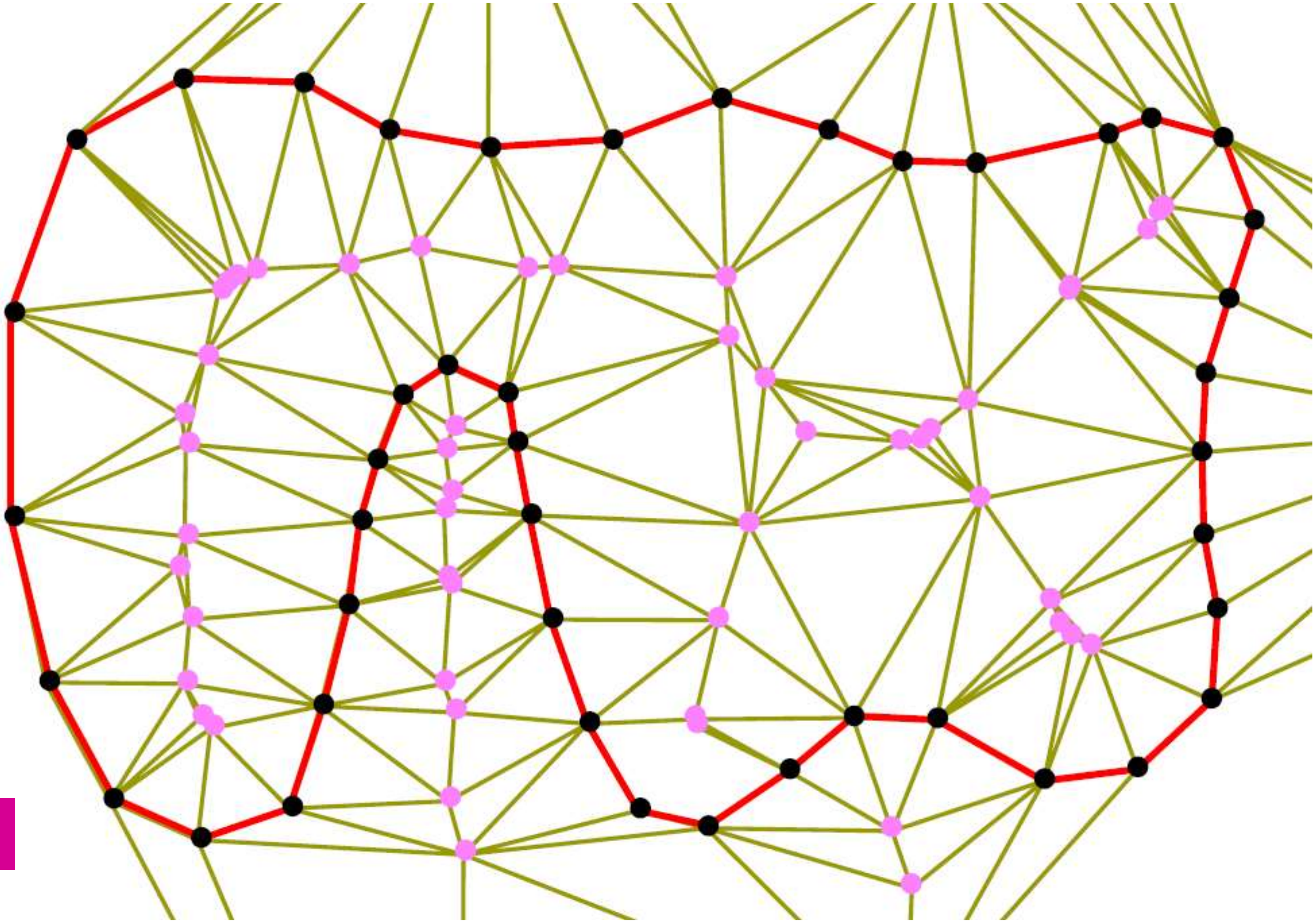
Voronoi Vertices



Refined Delaunay Triangulation

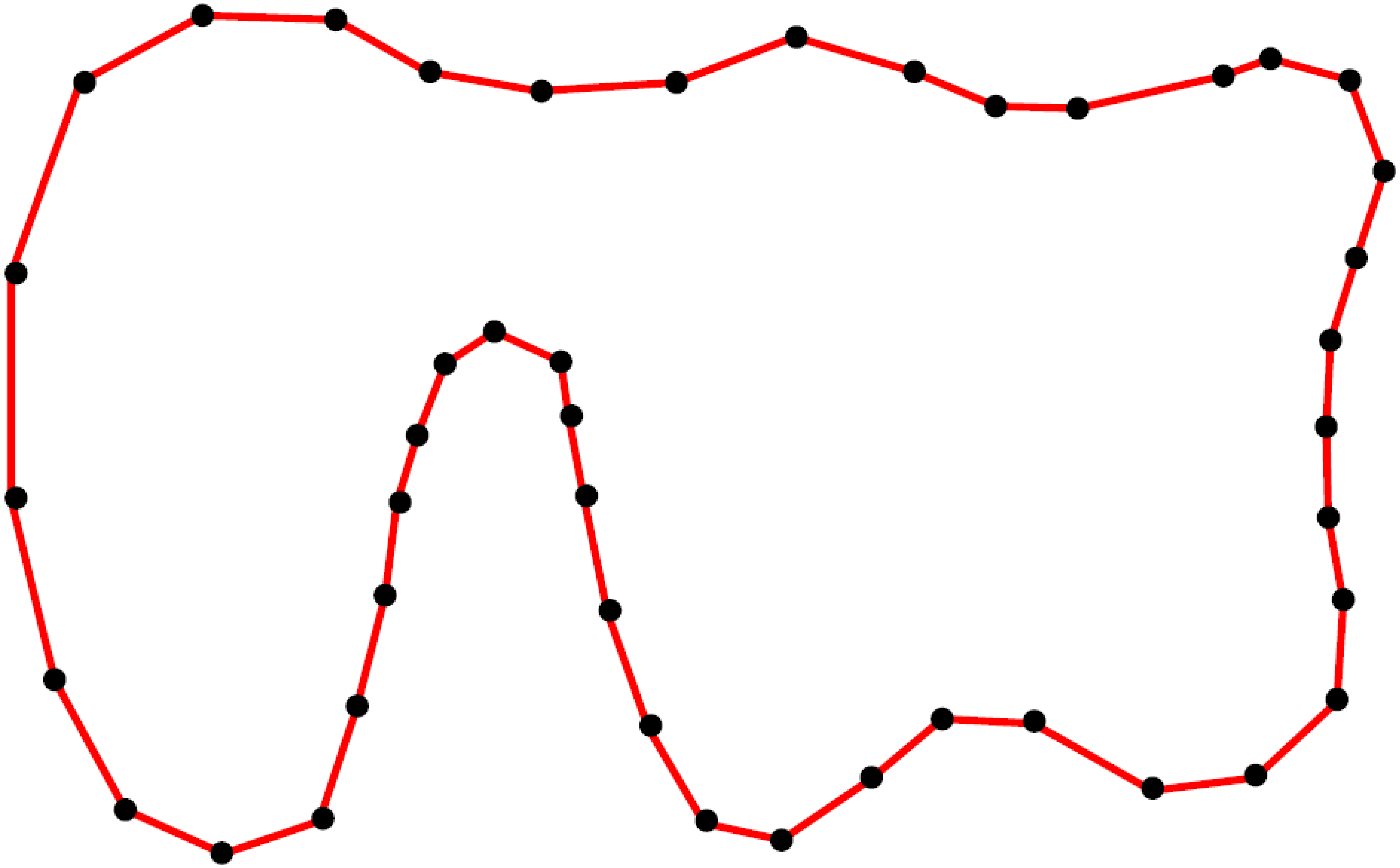


Crust



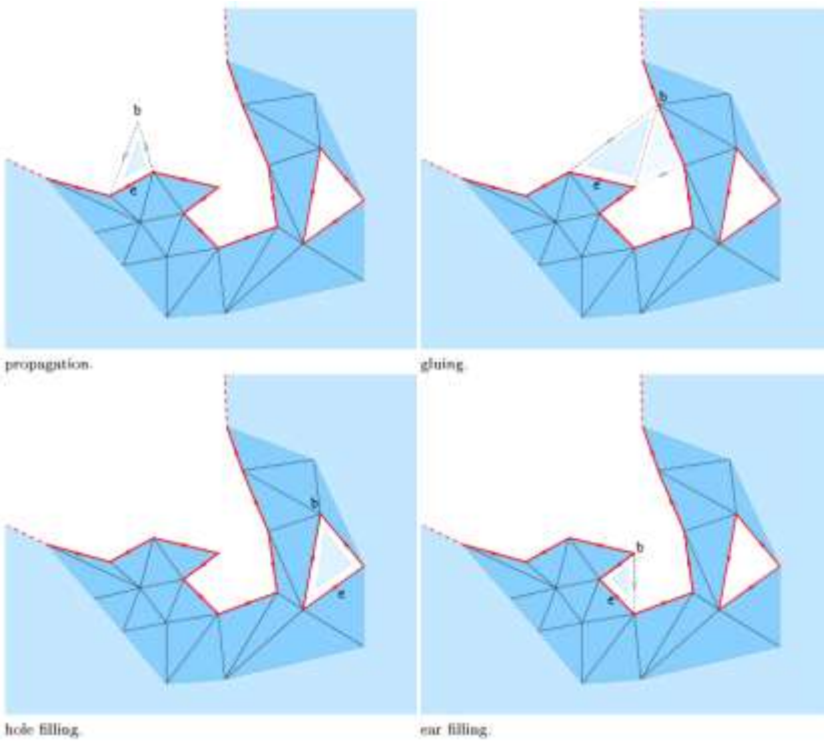
demo

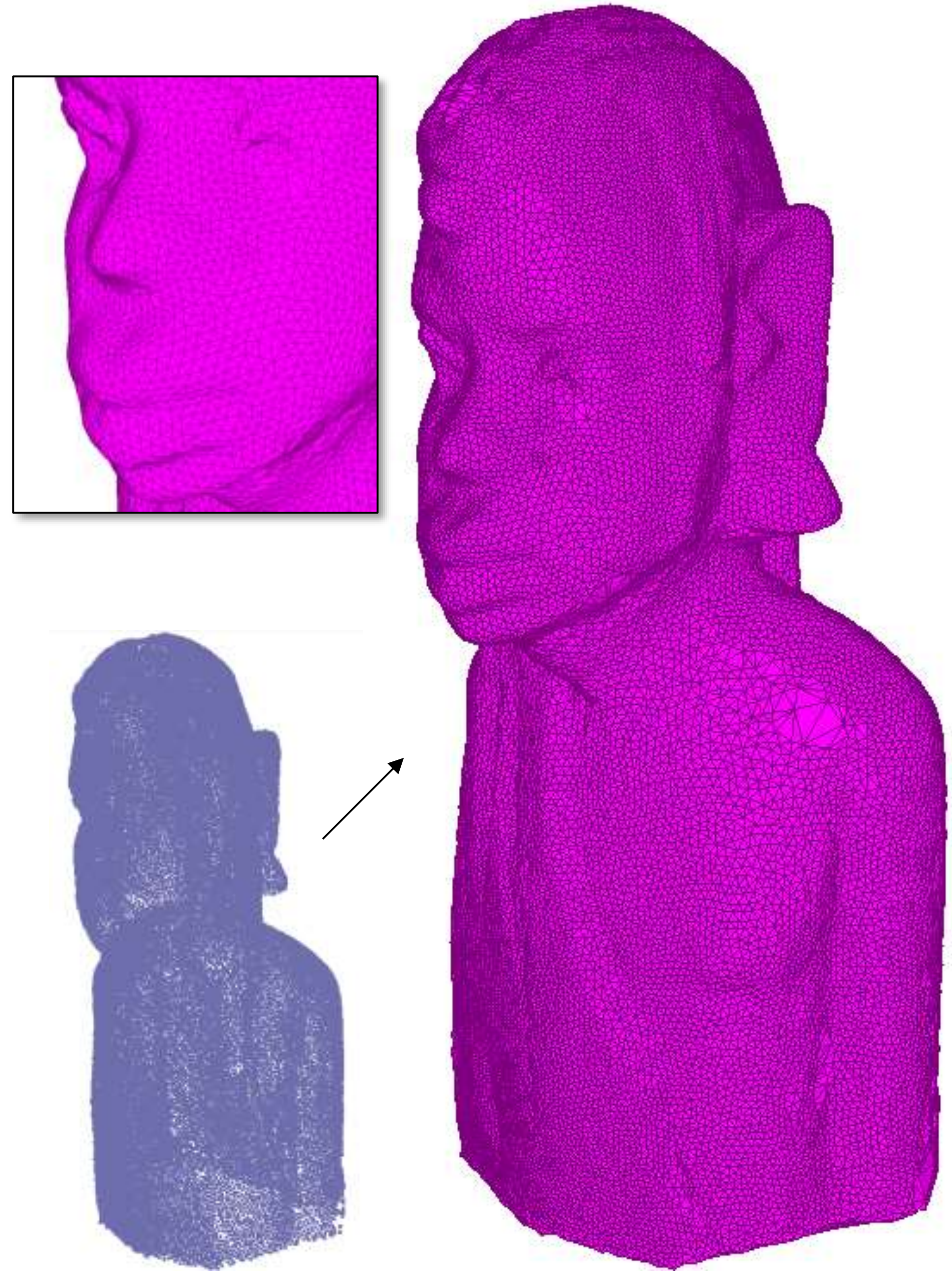
Crust



Advancing Front

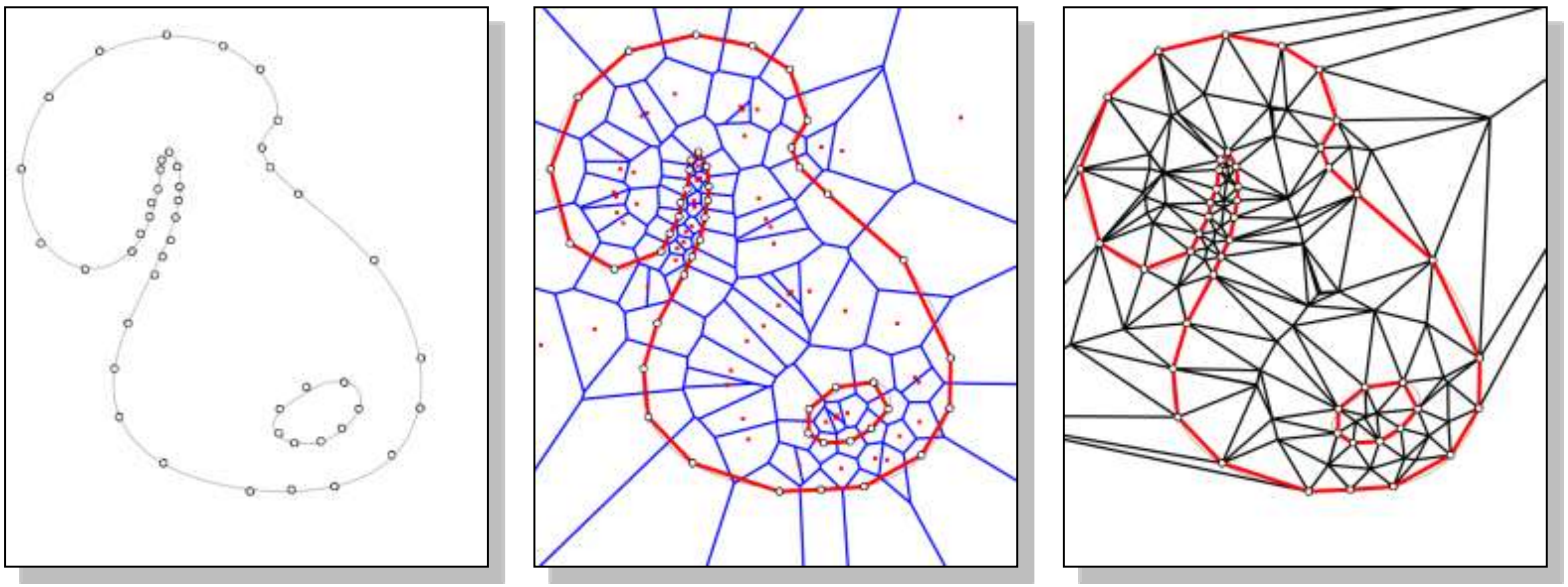
Advancing Front





Crust

- Several Delaunay algorithms provably correct

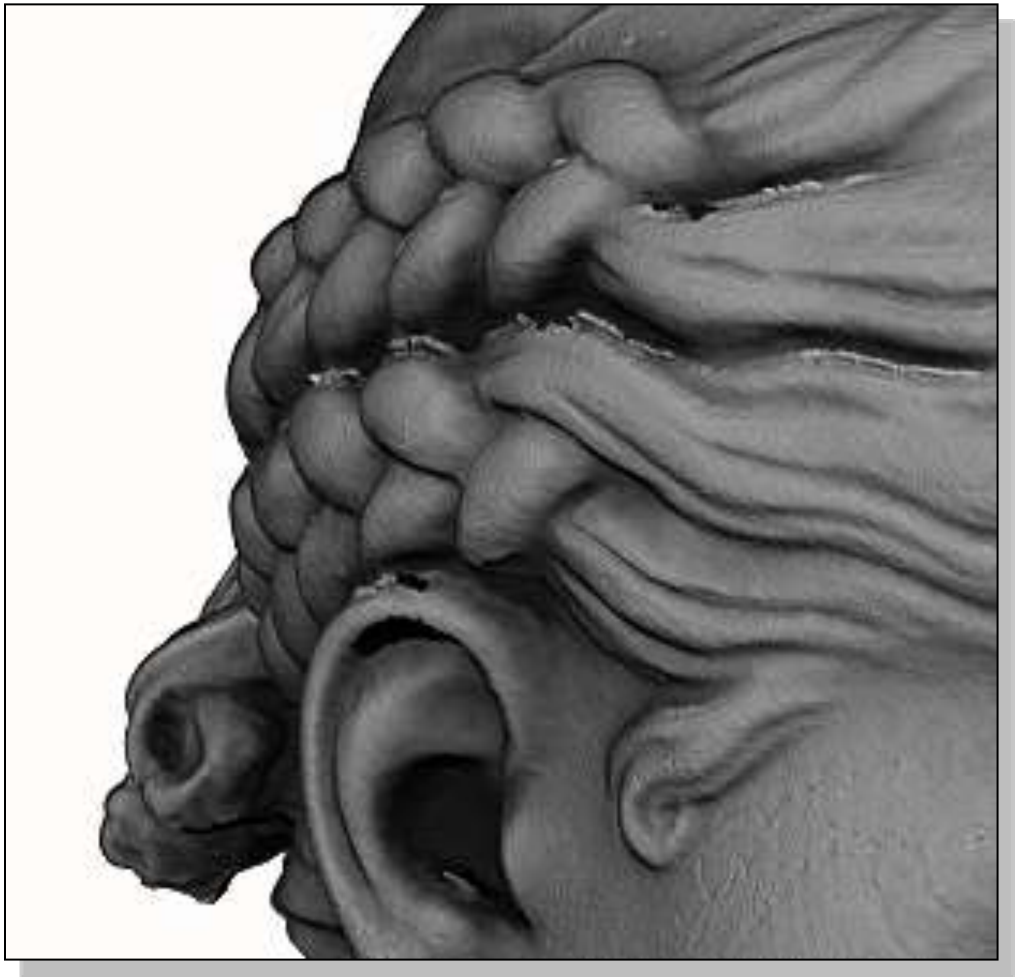


Delaunay-based

- Several Delaunay algorithms are **provably correct**... in the absence of noise and undersampling.

perfect data ?

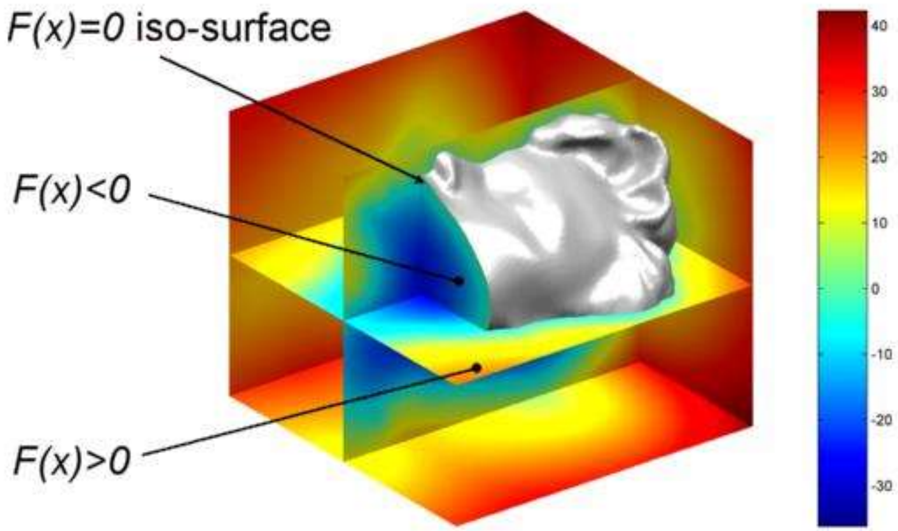
Noise & Undersampling



Delaunay-based

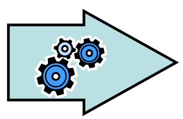
- Several Delaunay algorithms are **provably correct**... in the absence of noise and undersampling.

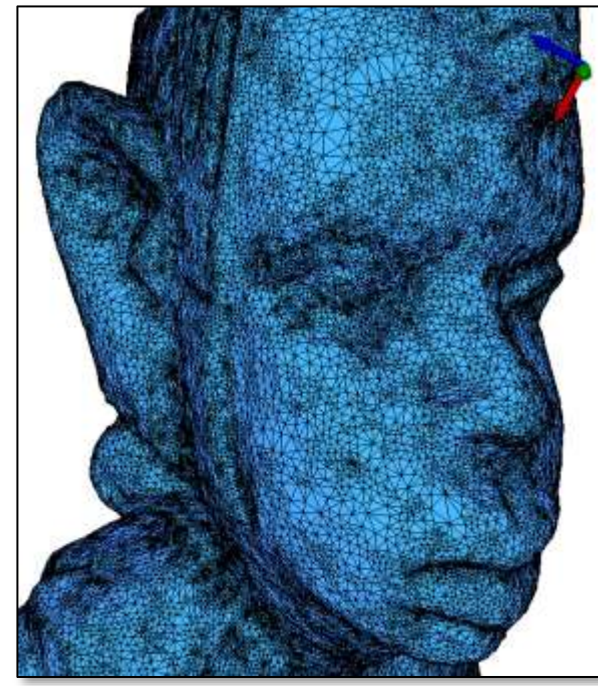
- Motivates reconstruction by fitting **approximating** implicit surfaces



Poisson Surface Reconstruction

Oriented point set

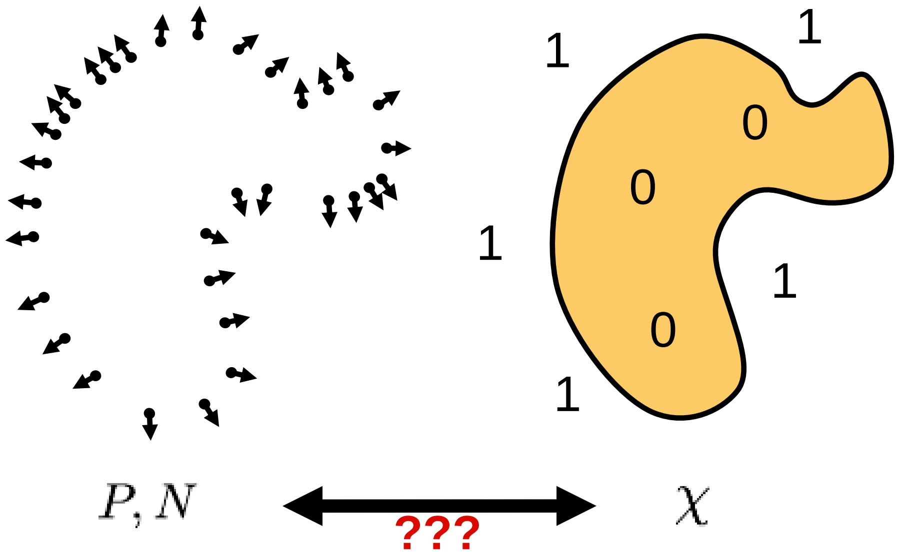




[Kazhdan et al. 06]

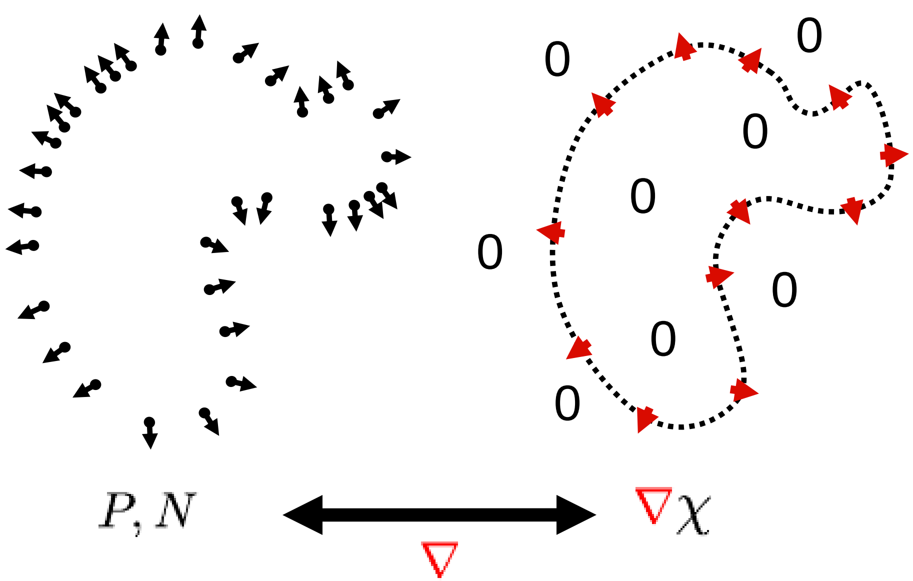
Indicator Function

Construct indicator function from point samples



Indicator Function

Construct indicator function from point samples



splatted normals

$$\min_{\chi} \int \|\nabla\chi(\mathbf{x}) - \boxed{\mathcal{N}(\mathbf{x})}\|_2^2 \, dx$$

variational calculus

$$\boxed{\Delta\chi} = \nabla \cdot \mathcal{N}$$

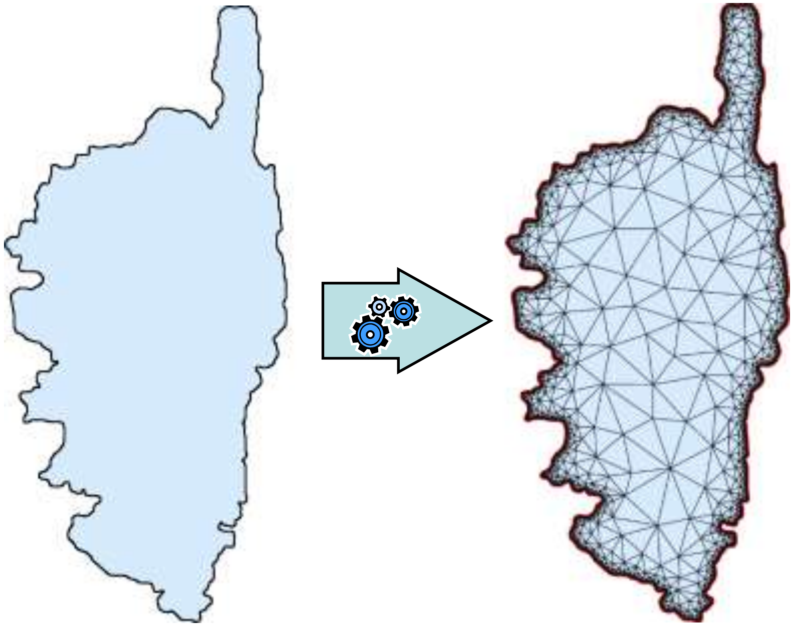
sparse linear system

Mesh Generation

2D Triangle Mesh Generation

Input:

- PSLG C (planar straight line graph)
- Domain Ω bounded by edges of C



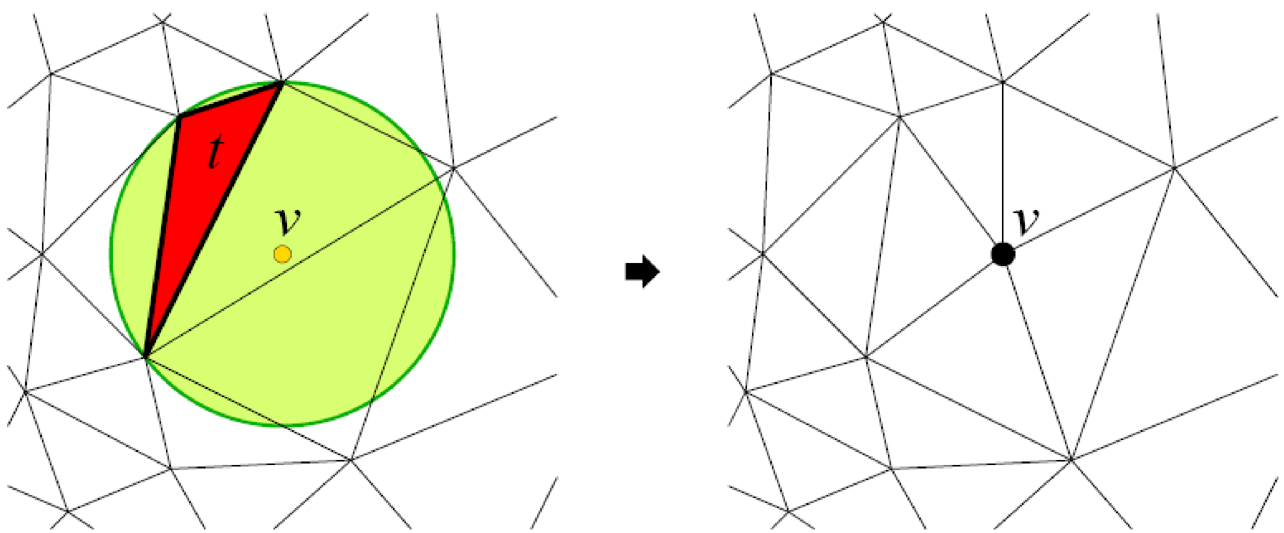
Output:

- Triangle mesh T of Ω such that
  - Vertices of C are vertices of T
  - Edges of C are union of edges in T
  - Triangles of T inside Ω have controlled size and quality

Key Idea

Break bad elements by inserting circumcenters (Voronoi vertices) [Chew, Ruppert, Shewchuk, Boissonnat...]

"bad" in terms of size or shape



Basic Notions

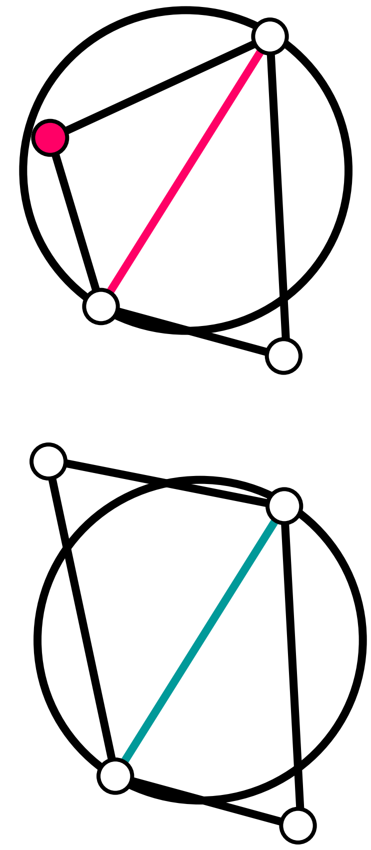
C: PSLG describing the constraints

T: Triangulation to be refined

Respect of the PSLG

- Edges a C are split until constrained subedges are edges of T
- Constrained subedges are required to be Gabriel edges

- An edge of a triangulation is a **Gabriel** edge if its smallest circumcircle encloses no vertex of T
- An edge e is **encroached** by point p if the smallest circumcircle of e encloses p.



Refinement Algorithm

C: PSLG bounding the domain to be meshed.

T: Delaunay triangulation of the current set of vertices

$T_{|\Omega}$: $T \cap \Omega$

Constrained subedges: subedges of edges of C

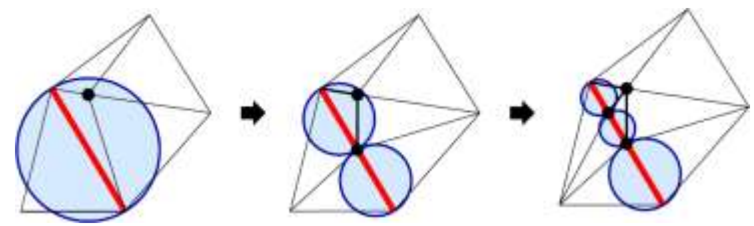
Initialize with T = Delaunay triangulation of vertices of C

Refine until no rule apply

- **Rule 1**

  if there is an encroached constrained subedge e

  insert c = midpoint(e) in T (refine-edge)



- **Rule 2**

  if there is a bad facet f in $T_{|\Omega}$

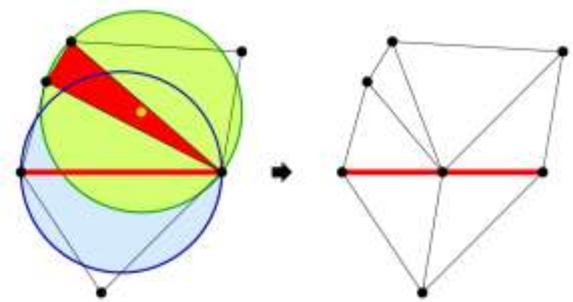
  c = circumcenter(f)

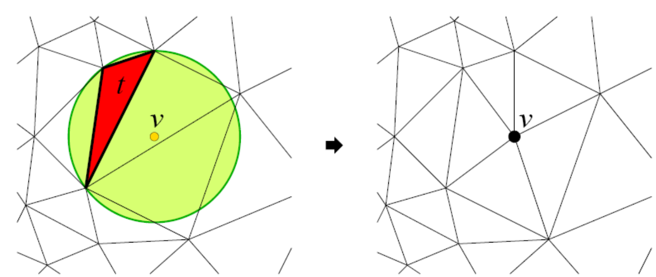
  if c encroaches a constrained subedge e

  refine-edge(e).

  else

  insert(c) in T



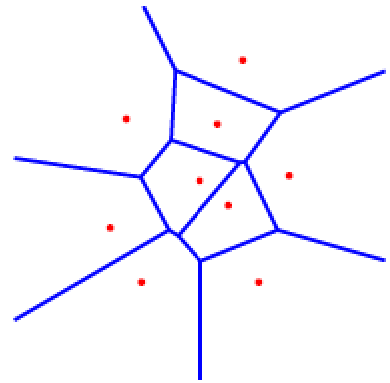


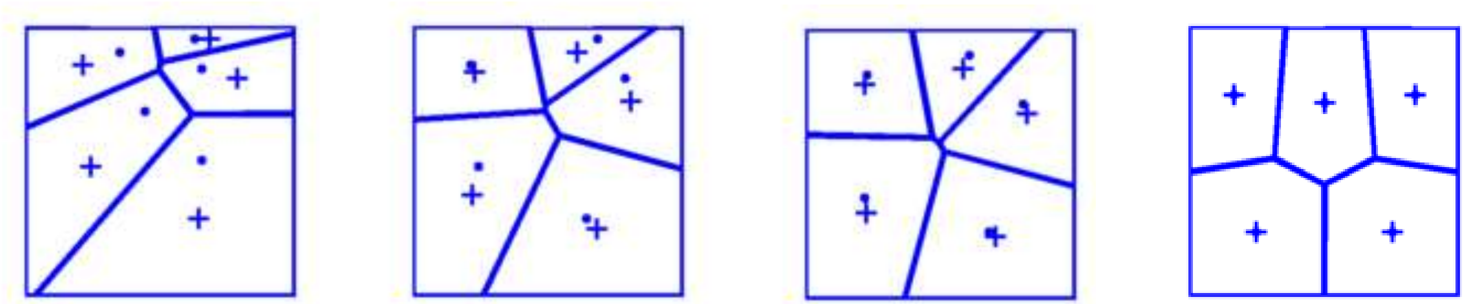
Pictures from [Shewchuk]

Mesh Optimization?

- Minimize error functional

$$E = \sum_{j=1..k} \int_{x \in R_j} \| x - x_j \|^2 \, dx$$

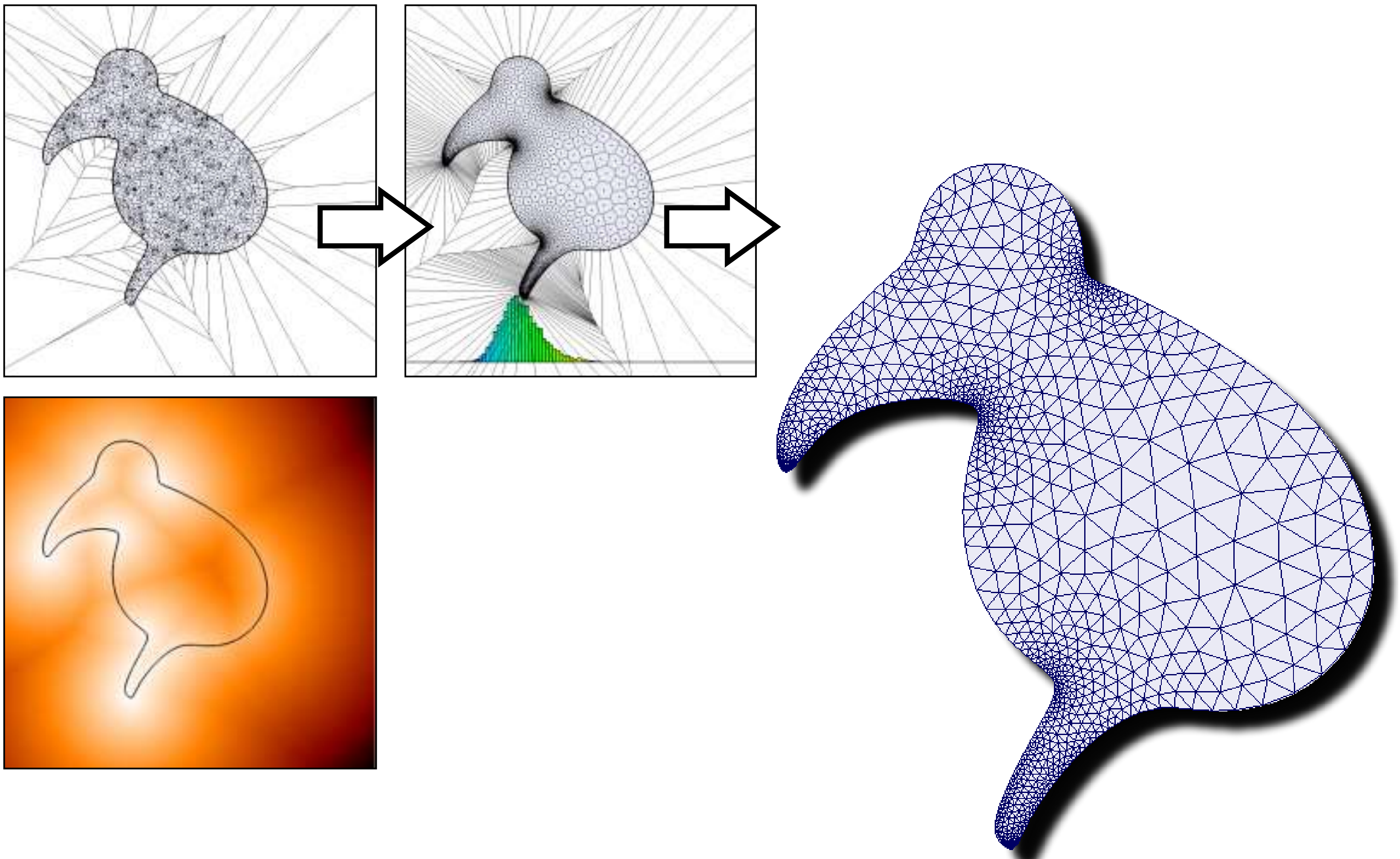




demo

Centroidal Voronoi Tessellation

Mesh Optimization



Surface Mesh Generation

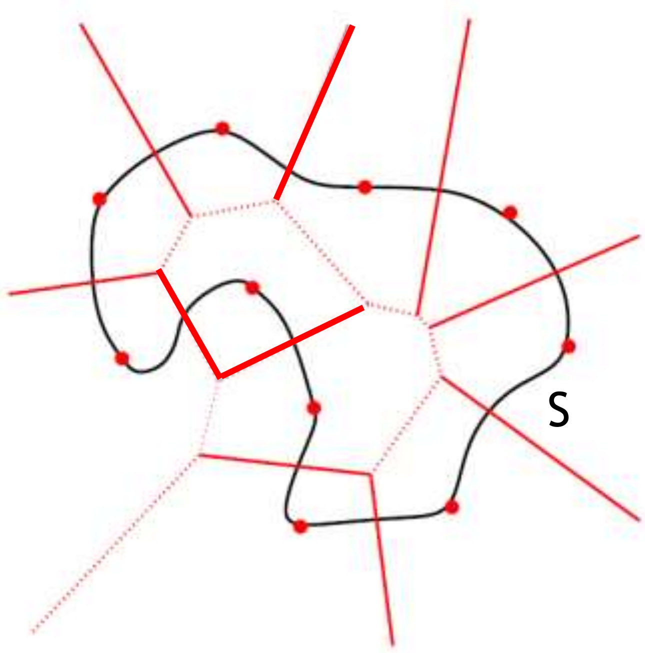
Mesh Generation

Key concepts:

- Voronoi/Delaunay filtering
- Delaunay refinement

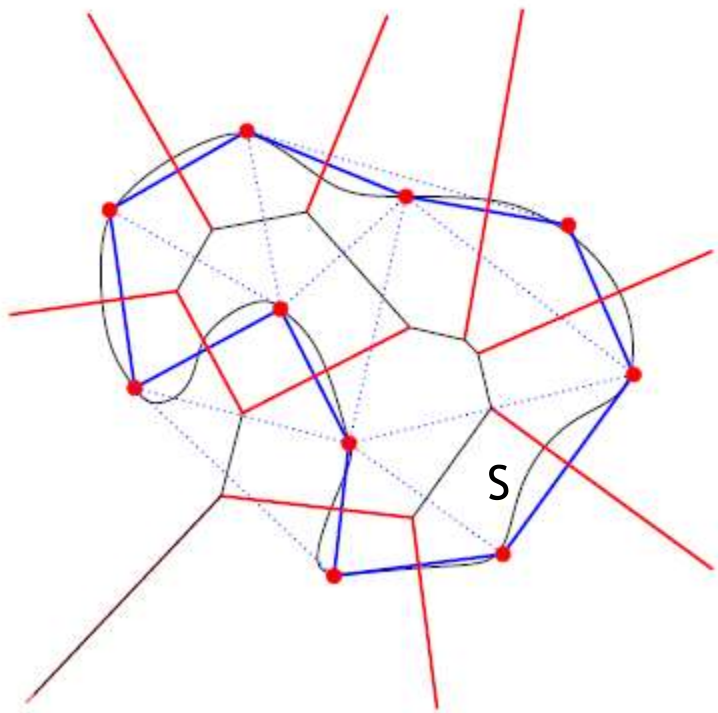
Voronoi Filtering

The Voronoi diagram **restricted** to a curve S, $Vor_{|S}(E)$, is the set of edges of Vor(E) that intersect S.

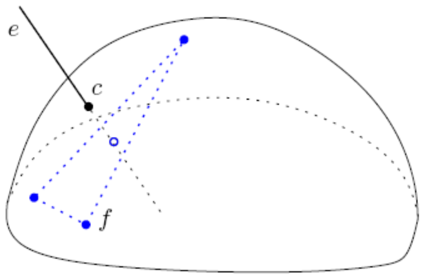


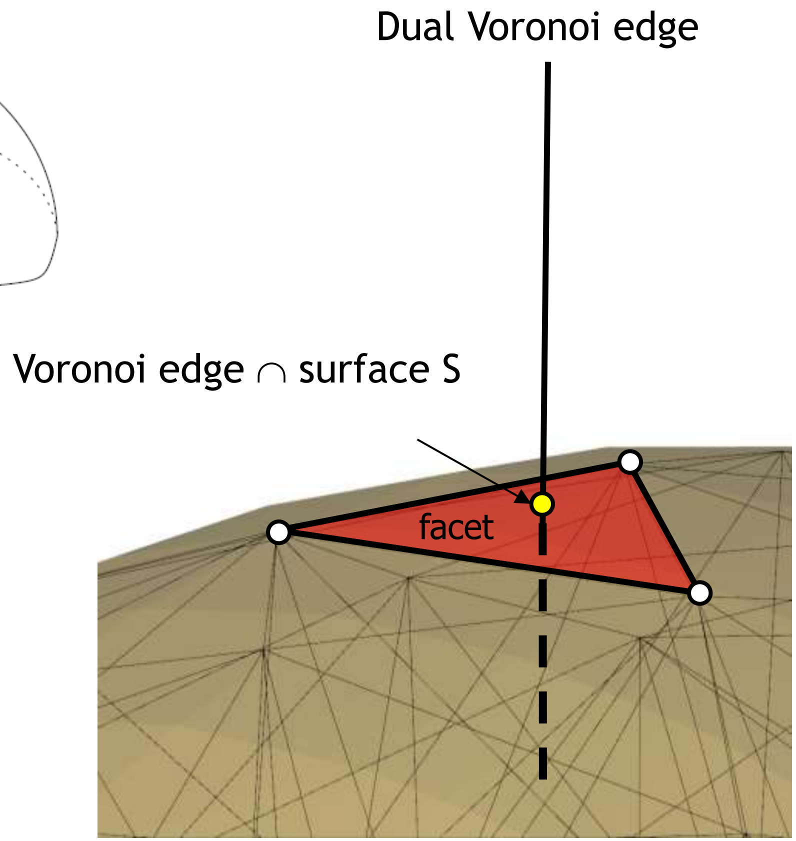
Delaunay Filtering

The restricted Delaunay triangulation restricted to a curve S is the set of **edges** of the Delaunay triangulation whose dual edges intersect S.

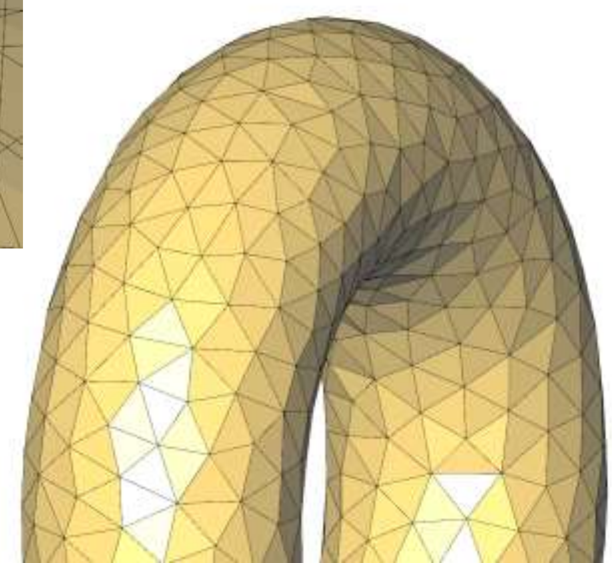


Delaunay Filtering

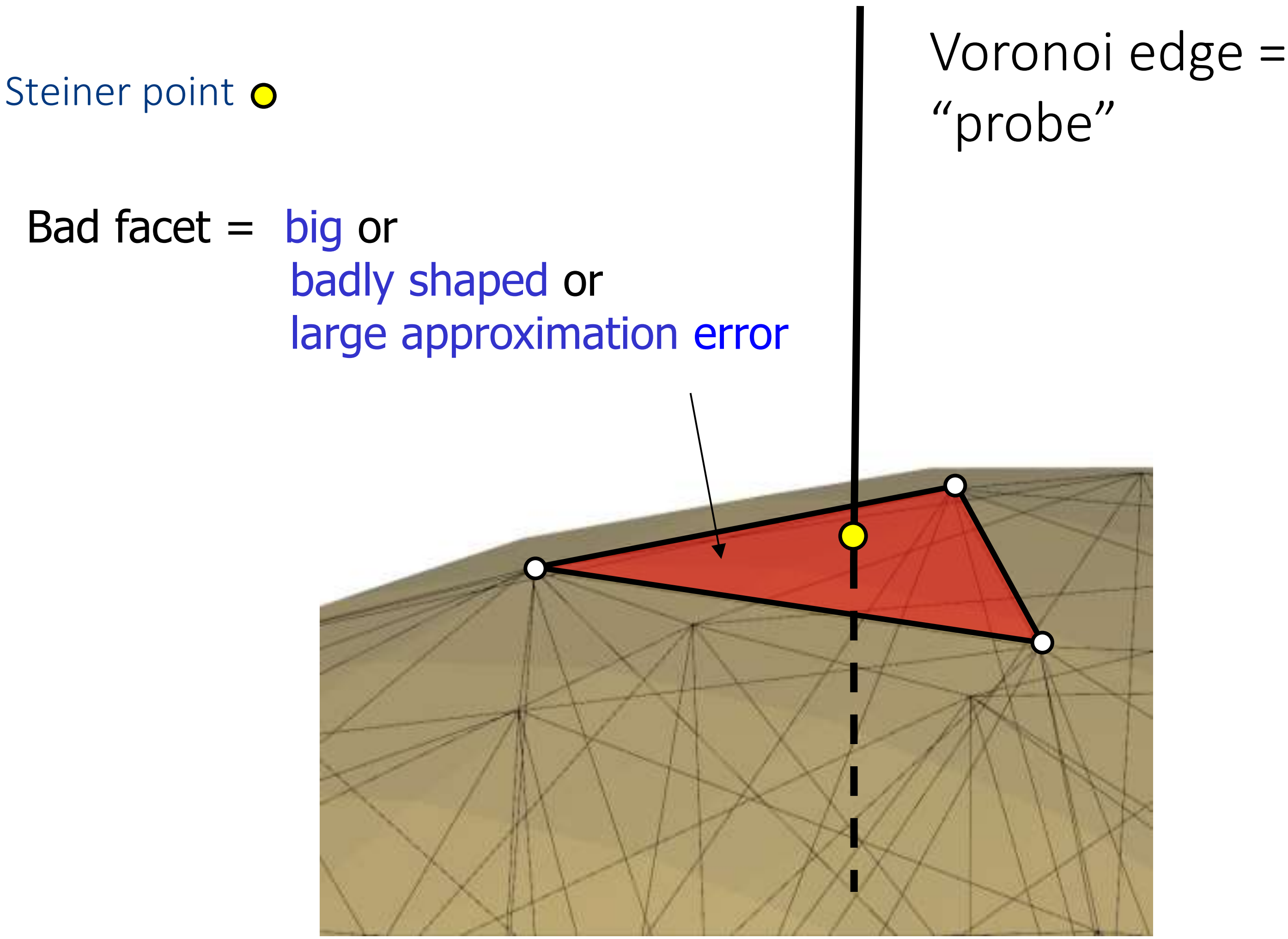




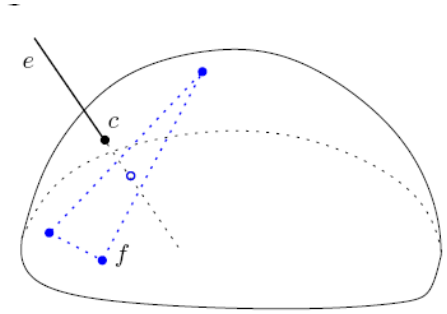
Delaunay triangulation restricted to surface S



Delaunay Refinement



Surface Mesh Generation Algorithm



repeat

{

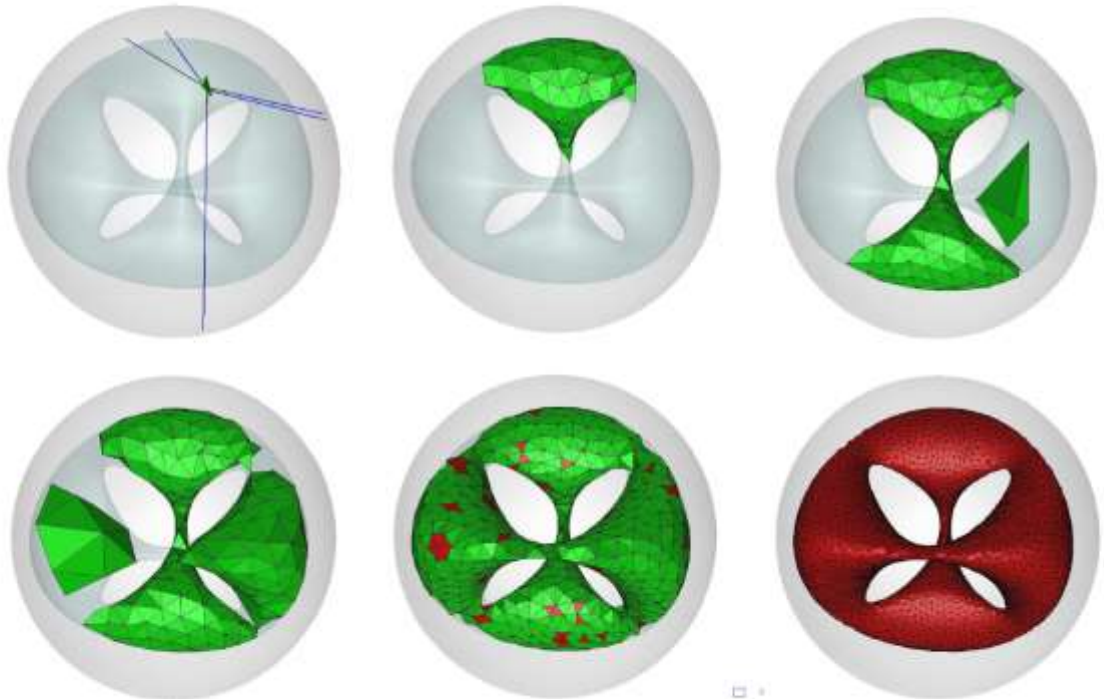
pick bad facet **f**

insert furthest (dual(**f**) $\cap$ **S**) in Delaunay triangulation

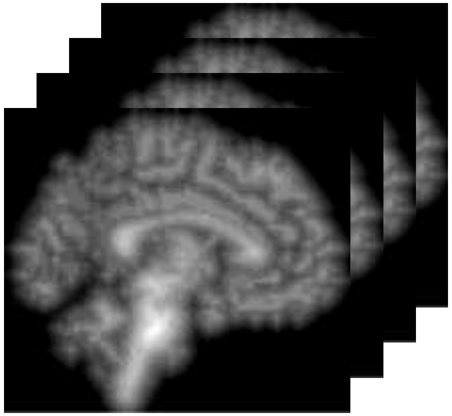
update Delaunay triangulation restricted to **S**

}

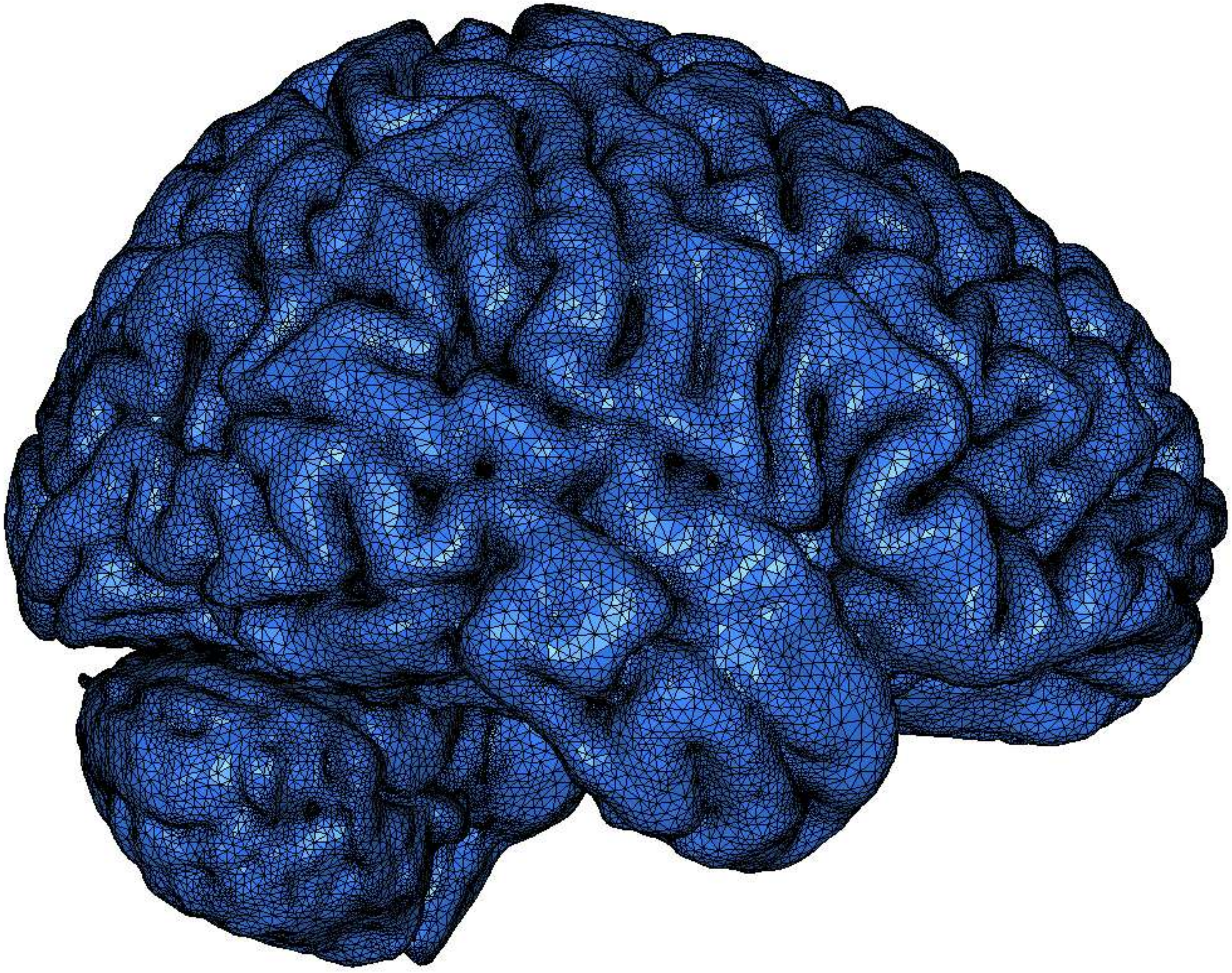
until all facets are good



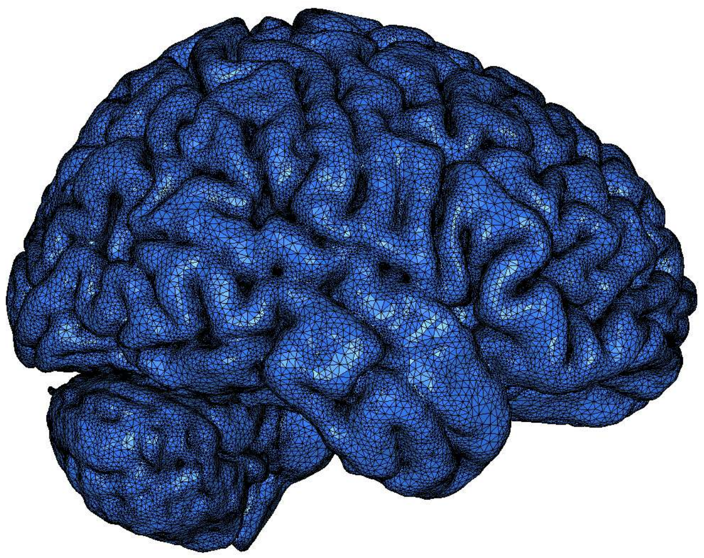
Isosurface from 3D Grey Level Image



input



Guarantees



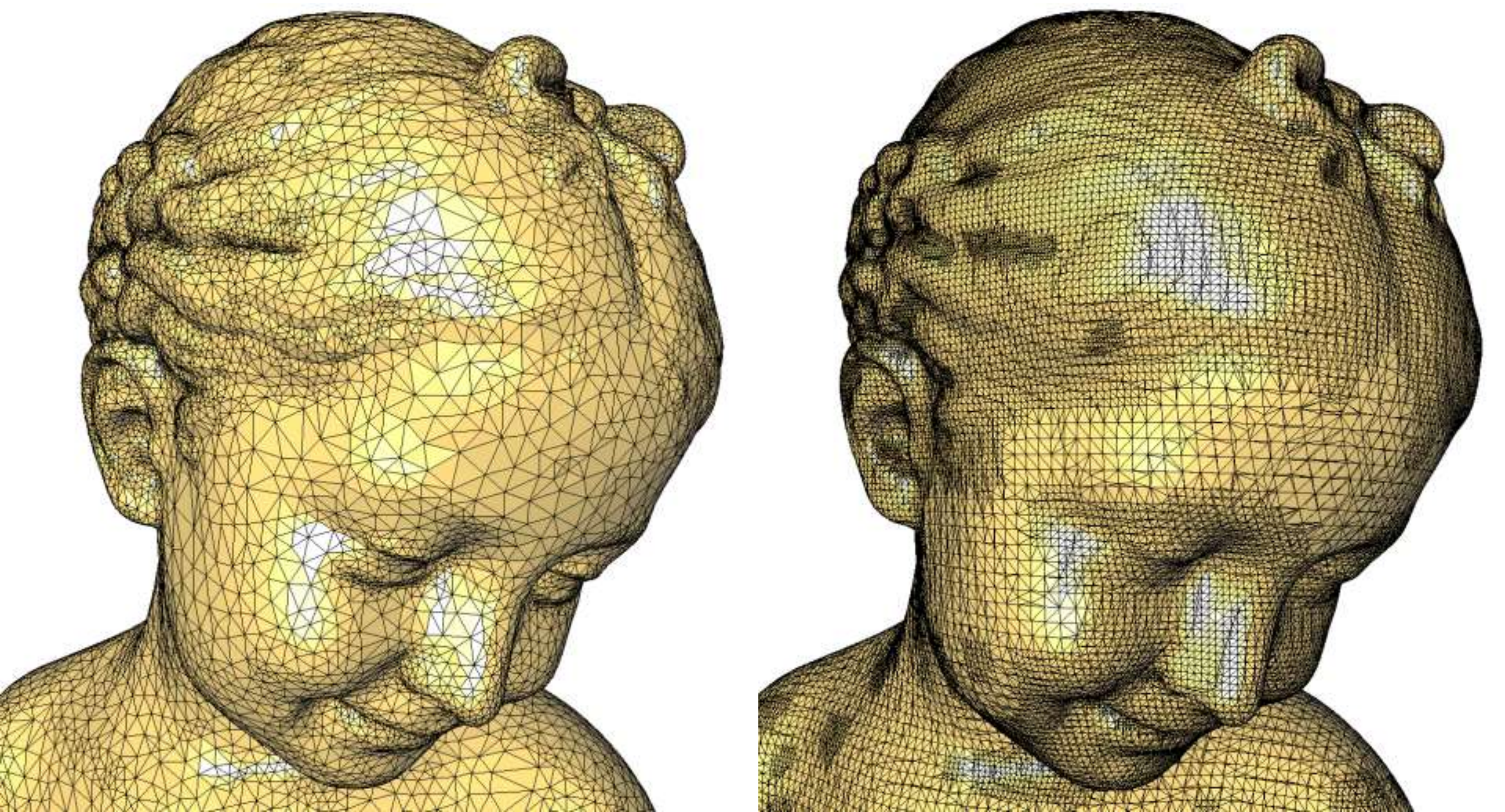
Termination

Parsimony

Output mesh properties:

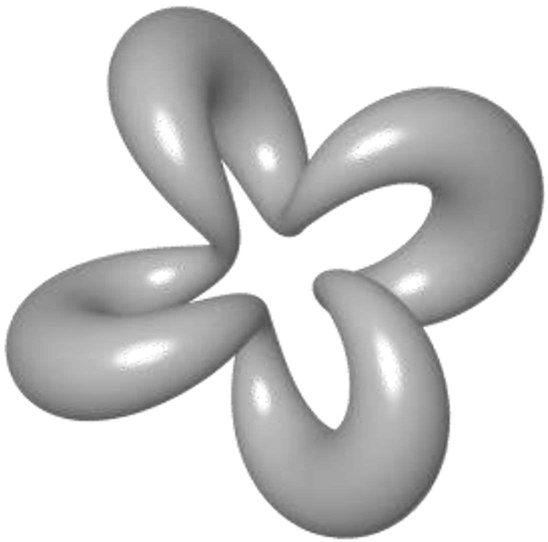
- Well shaped triangles
  - Lower bound on triangle angles
- Homeomorphic to input surface
- Manifold
  - not only combinatorially, i.e., no self-intersection
- Faithful approximation of input surface
  - Hausdorff distance
  - Normals

Delaunay Refinement vs Marching Cubes

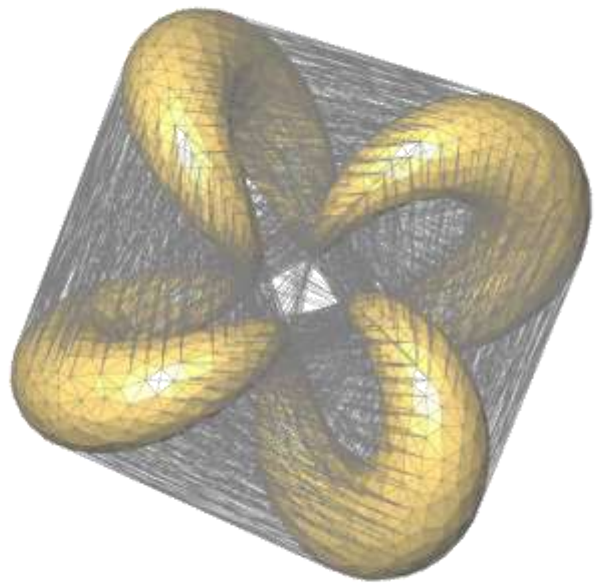


Volume Mesh Generation

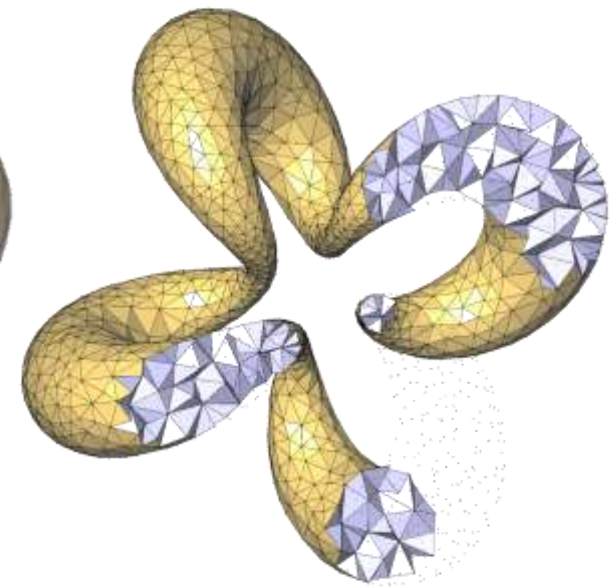
Delaunay Filtering



Domain boundary



3D Delaunay triangulation

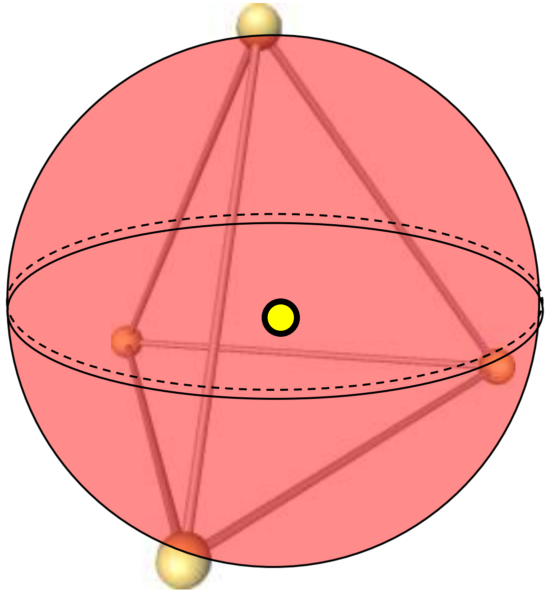


Restricted
Delaunay triangulation

Delaunay Refinement

Steiner point

Bad cell = big or
badly shaped



Volume Mesh Generation Algorithm

repeat

{

  pick bad simplex

  if(Steiner point encroaches a facet)

    refine facet

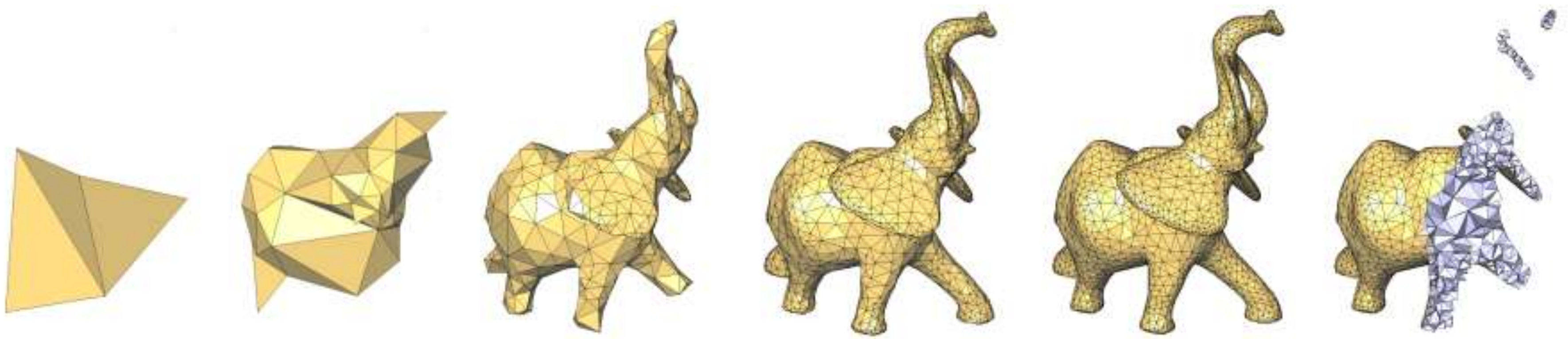
  else

    refine simplex

  update Delaunay triangulation restricted to domain

}

until all simplices are good



Thank you.