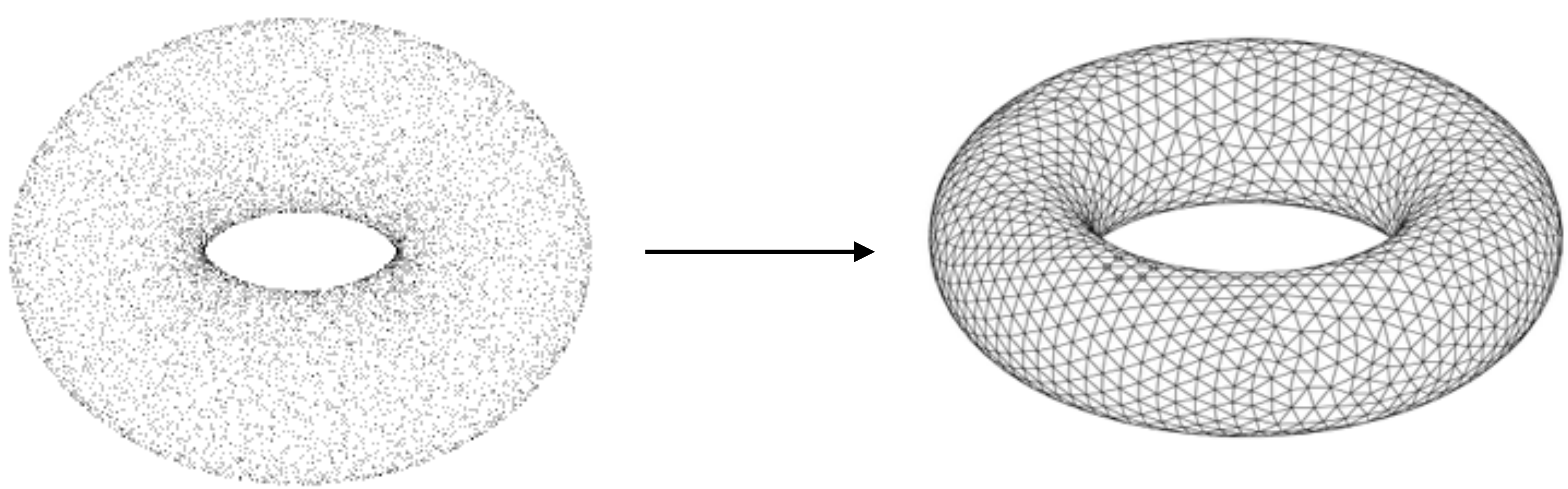
Reconstructing surfaces from point scans

Talk by Erin Chambers

Representing shapes

- A fundamental problem: given a set of points scanned from some input, reconstruct the underlying shape they represent



Images courtesy of Wikipedia

Reconstructing shape

- However, sometimes it isn't so clear what shape we want:

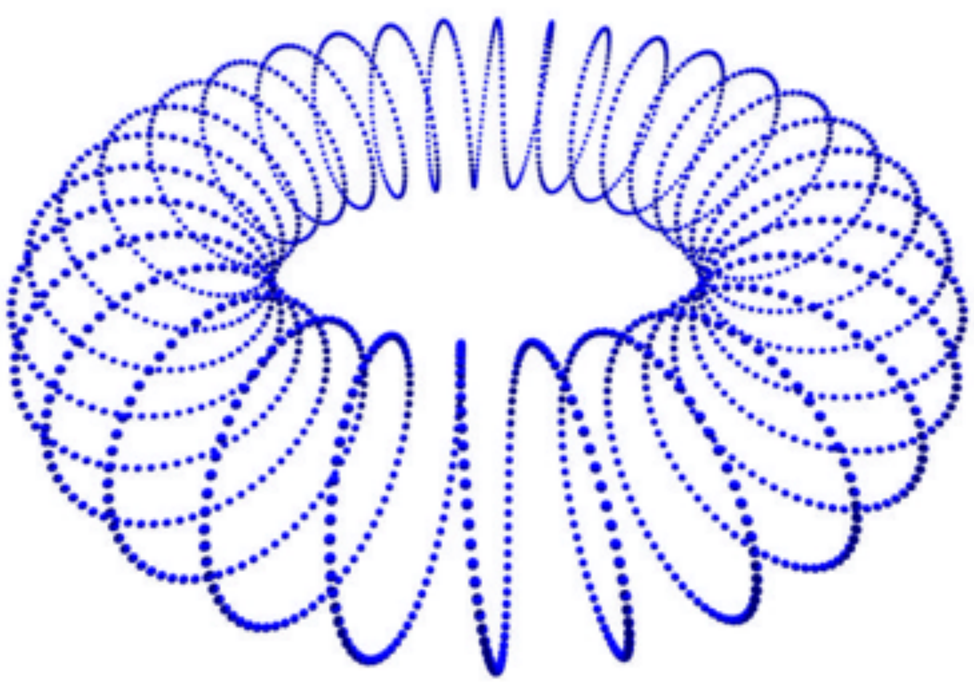


Image courtesy of the SIAM Journal of Applied Algebra and Geometry

Algorithms for shape reconstruction

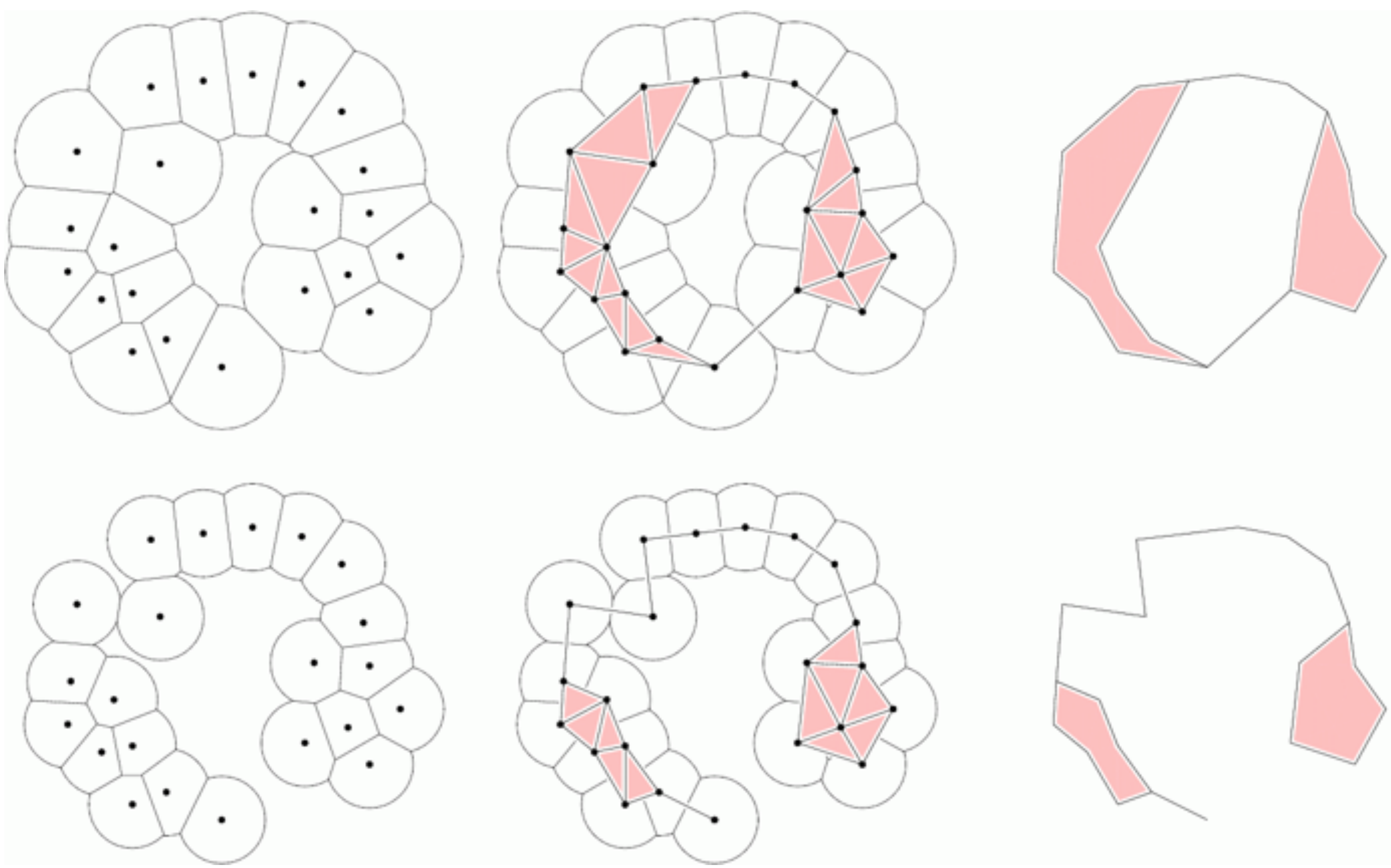
- Goal today: Survey some classical shape reconstruction algorithms
- Note that this is a very active area of research
  - Methods vary widely
- I'll focus on computational geometry and graphics algorithms, many of which build on the complexes we discussed last time.

Goals for any method

- Output a triangulation which is:
  - Homeomorphic to original shape
  - Close geometrically to original shape
  - Approximates the normals

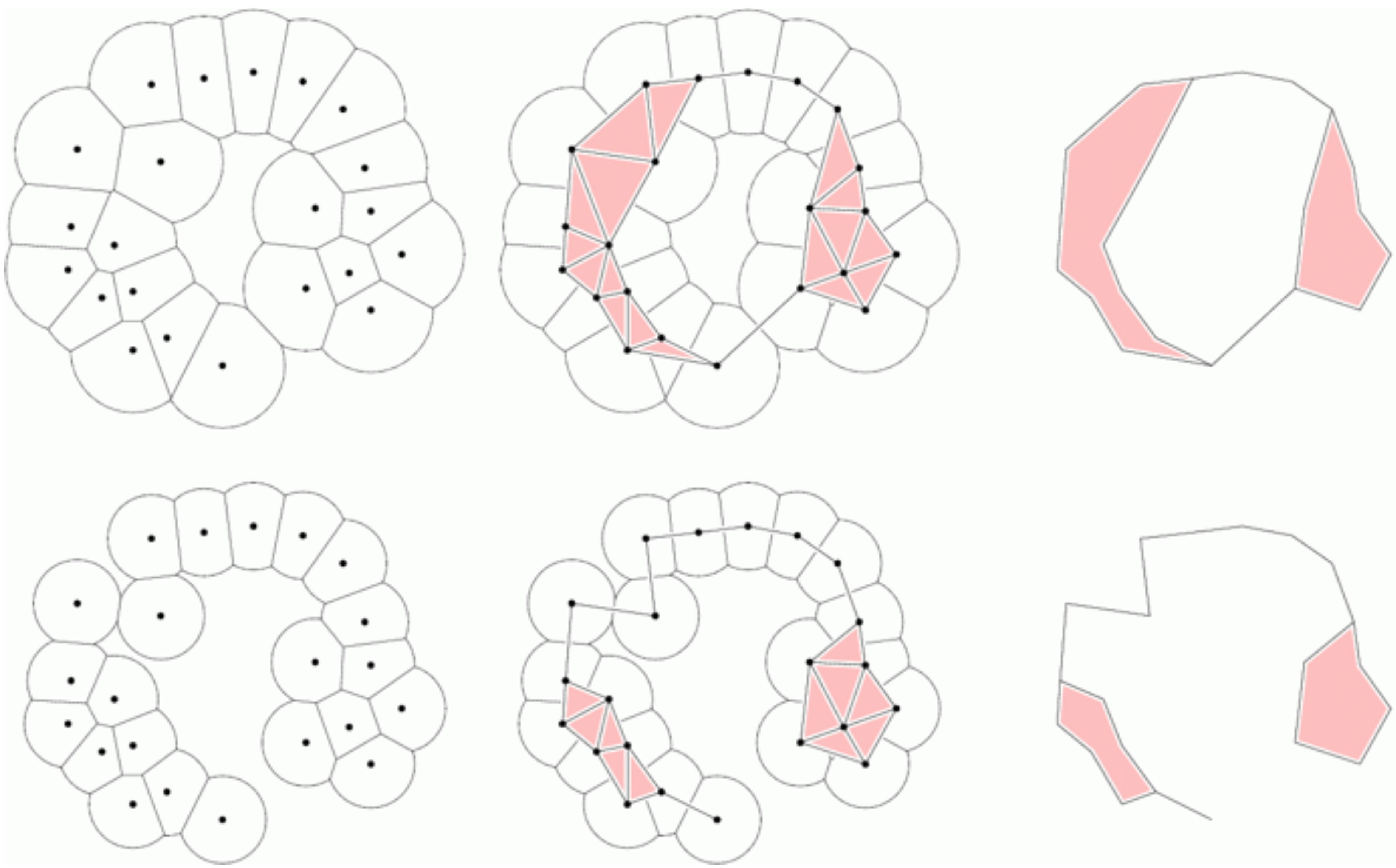
Recall: alpha shapes

- Given a radius α and a set of points, we take the union of all radius α balls at those points.



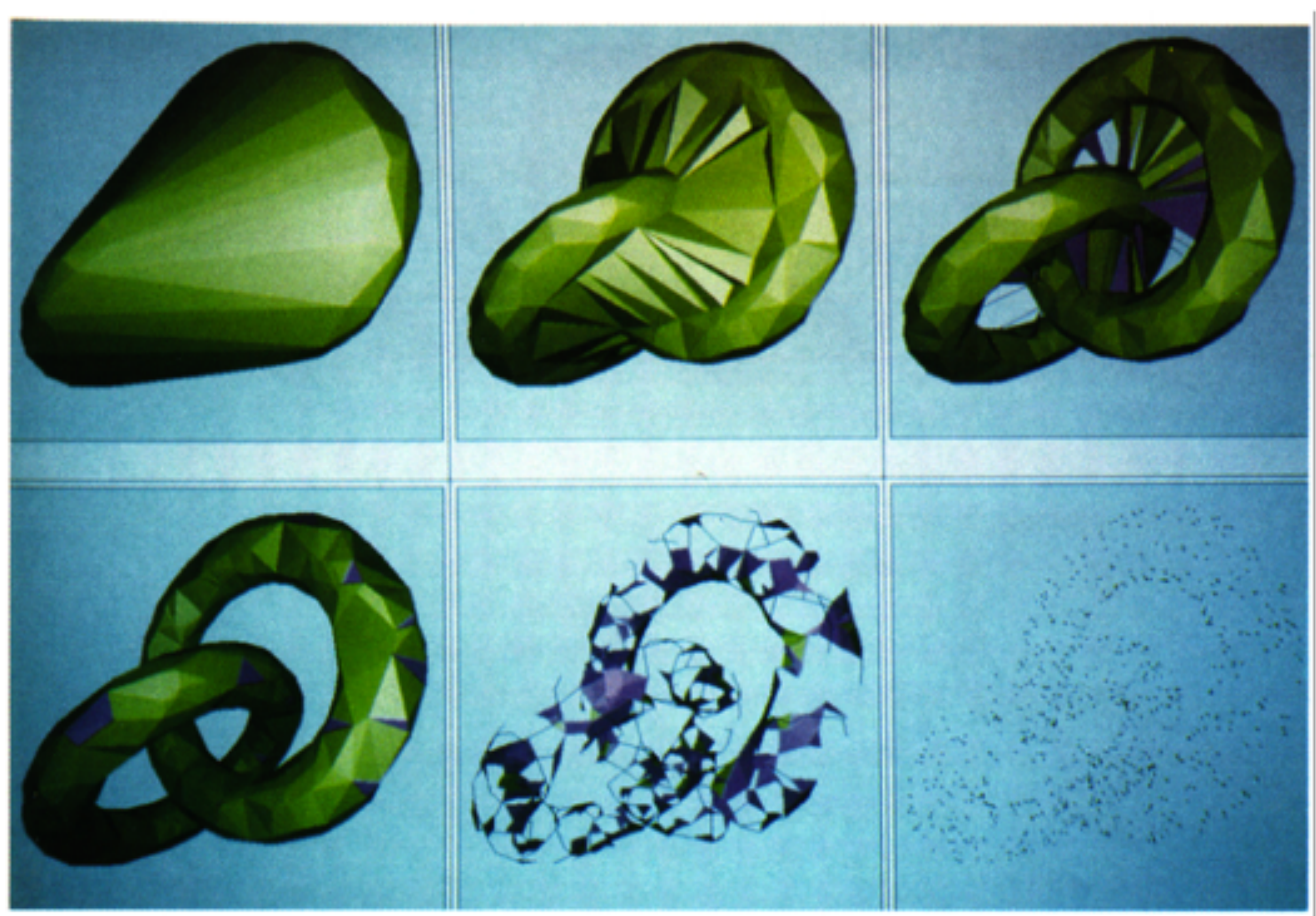
Recall: alpha complex

- The α-complex is then the nerve of this set of balls:



3d α-shapes

- In fact, one early reconstruction algorithm was just based on using α-shapes directly [Edelsbrunner-Mucke 1994]

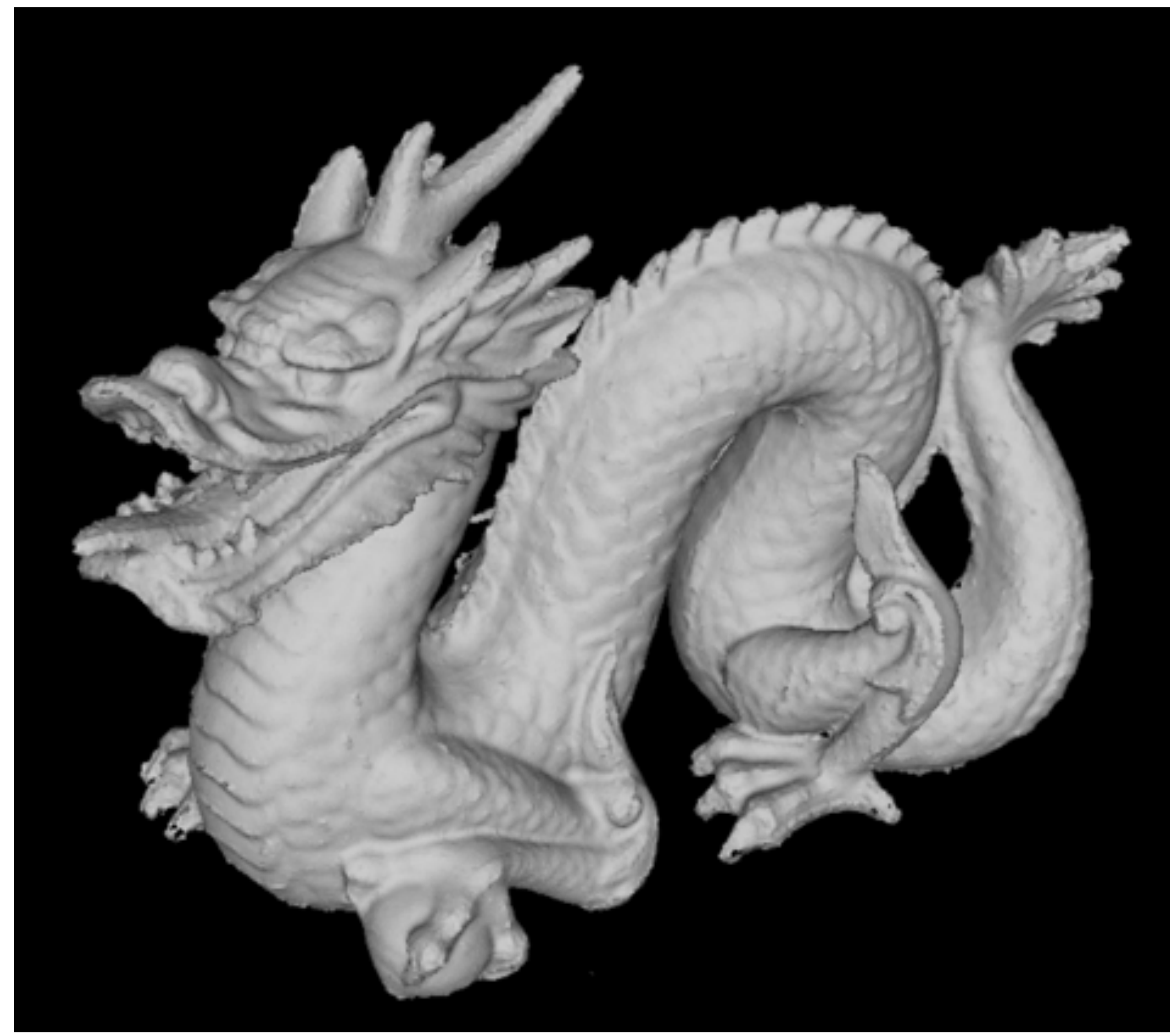


Ball-rolling algorithm

- One early extension which used the α-shape was the ball pivot algorithm [Bernardini et al]:
  - Starting at a seed triangle, pivot a ball around each edge of the triangle until a new sample point is hit.
  - Add that triangle to the mesh and continue.

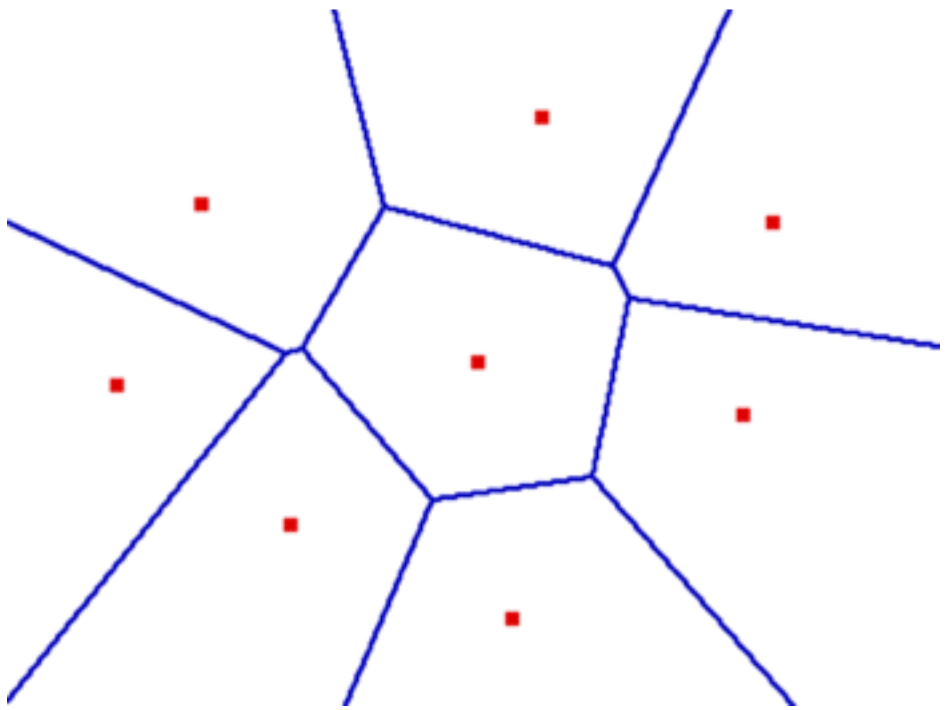
Ball rolling algorithm (cont.)

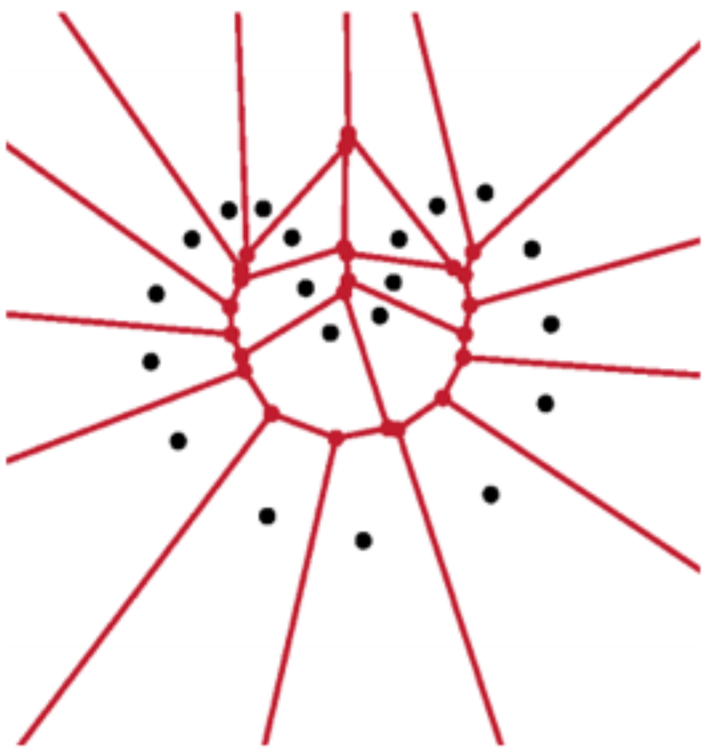
- Pros: Conceptually simple, very fast to implement
- Cons:
  - No theoretical guarantee of quality in terms of the topology
  - Not even always a surface



The crust algorithm: 2d

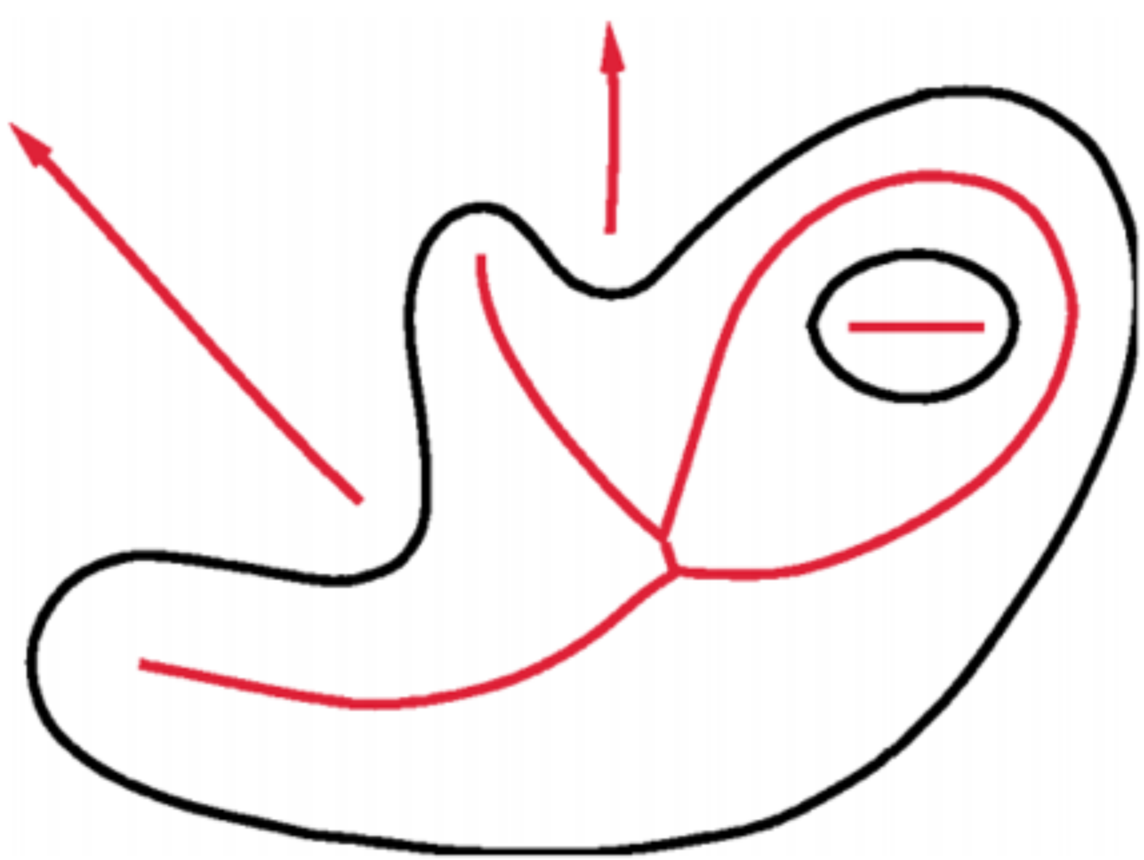
- If we go back to a 2d idea:
    - The Voronoi diagram is the division of the plane into cells where each cell consists of points closest to one of the input points:

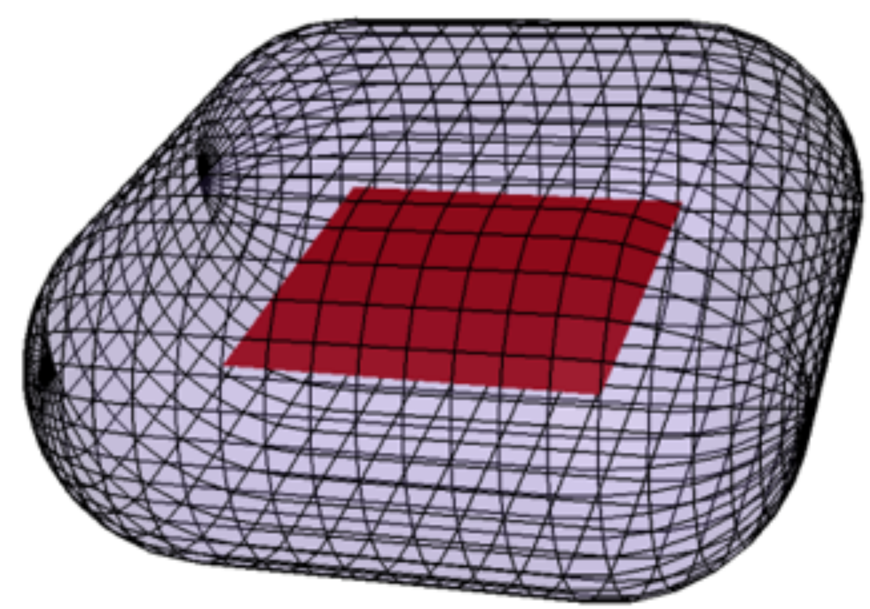




Related: medial axis

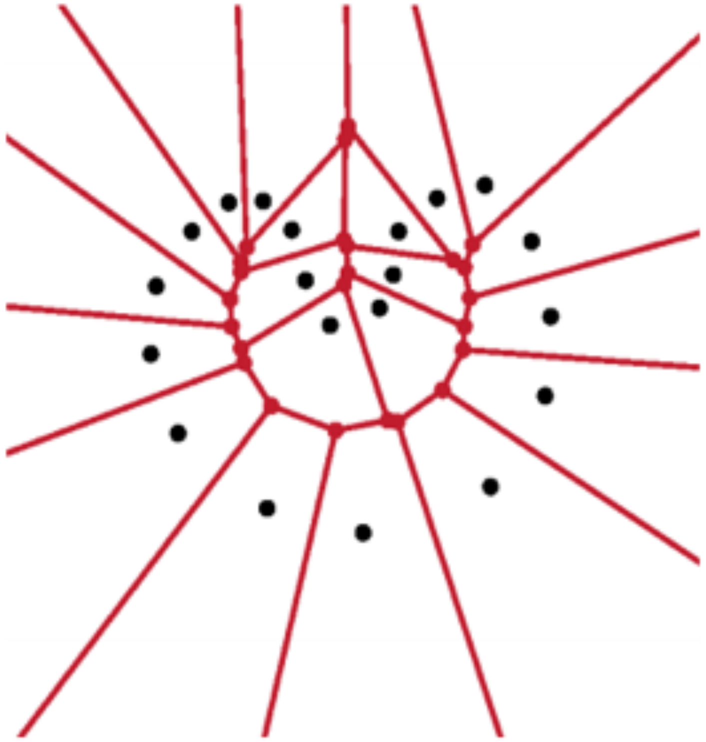
- The medial axis of a shape is the set of points with more than one closet point on the shape:





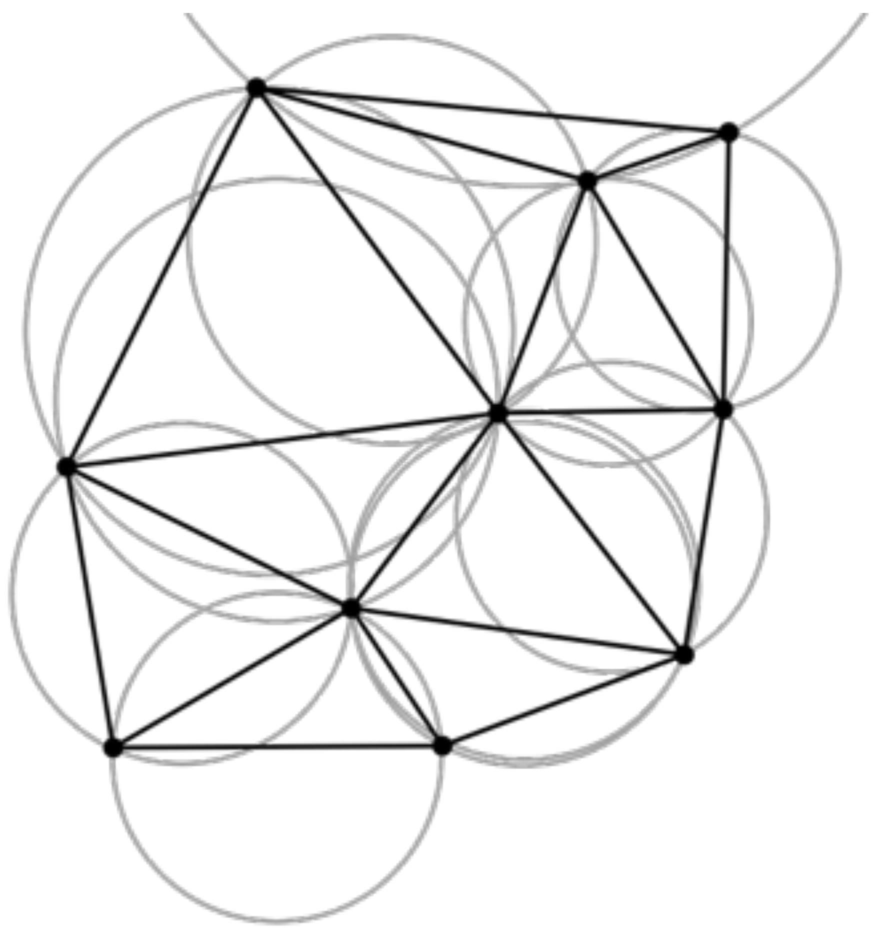
The connection

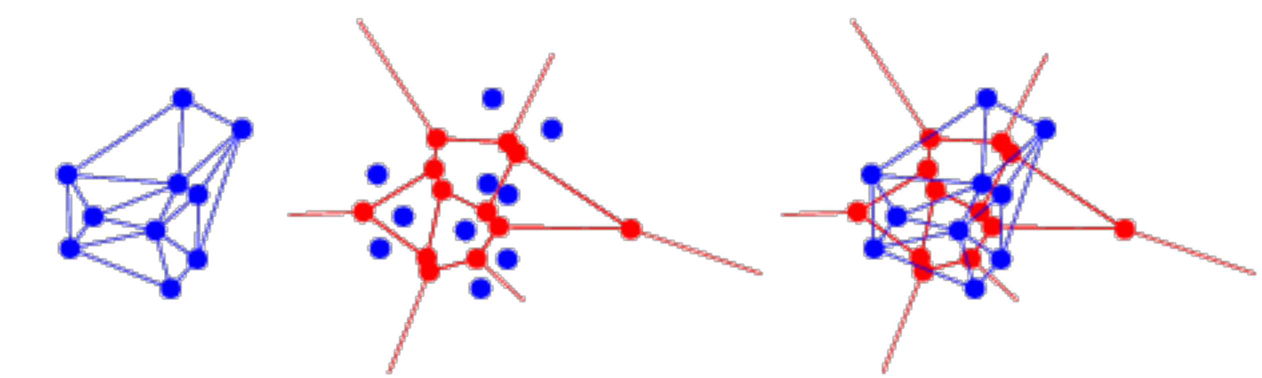
- In 2d, the Voronoi diagram of a point set that closely samples an underlying shape will contain an approximate medial axis of the shape:



Back to curve reconstruction

- Recall the dual to the Voronoi diagram: the Delaunay triangulation is the set of simplicies where the circumcircle of those simplicies is empty of other sites

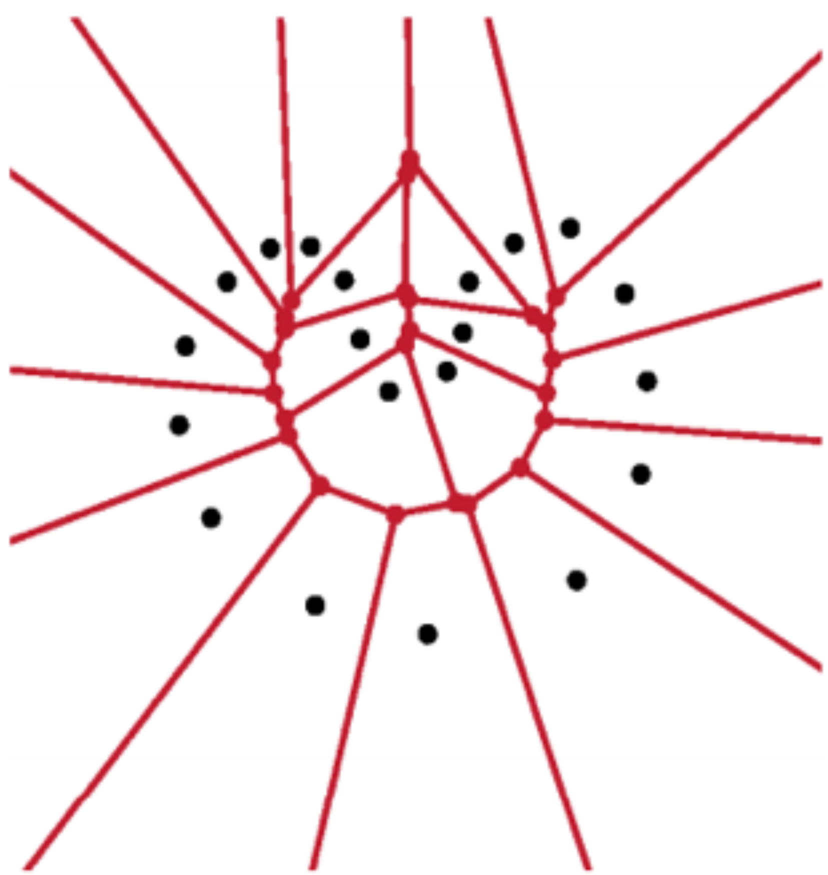




Delaunay triangulation *Voronoi diagram* *Delaunay and Voronoi*

2d crust algorithm

- In 2d, we want to select any edge of the Delaunay triangulation whose circumcircle is empty not only of sample points, but also of the Voronoi vertices:



Why?

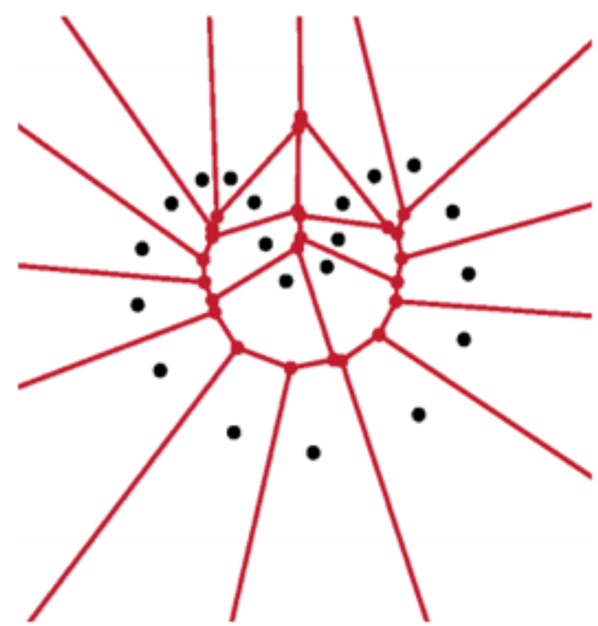
- Key lemma: Any Voronoi disk of a set of points sampled from a curve in the plane must contain a medial axis point of the curve.

    - Sketch: Essentially, the Voronoi disk's center is equidistant from more than 1 point on the curve, so it should be on the medial axis.

Why?

- Key lemma: For a fine enough sample S of a curve, an edge between two non-adjacent samples cannot be circumscribed by a circle that is empty of both Voronoi vertices and sample points.

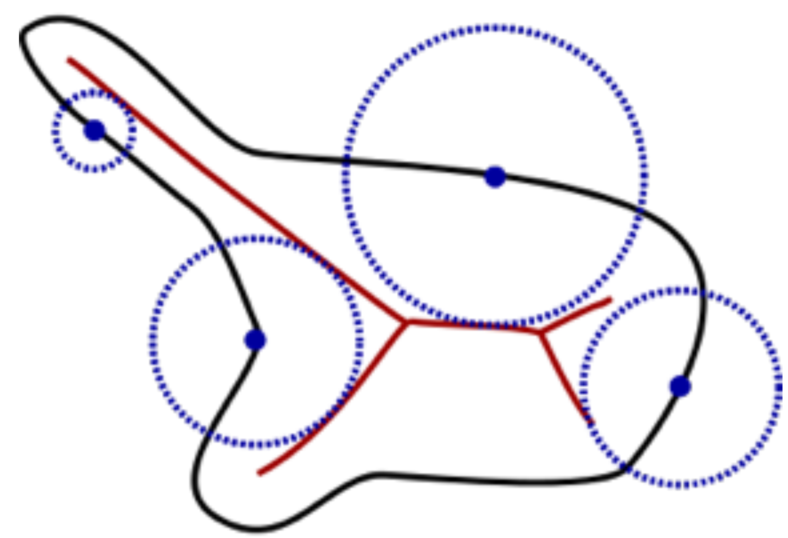
  - Proof by picture:



“Fine enough” sample

- More precisely: we must sample based on local feature size, lfs

    - For any x from the curve F, lfs(x) is the distance from x to the nearest medial axis point



- We say it is ε-sampled if every point p on the underlying curve is within ε×lfs(p) of a sample point

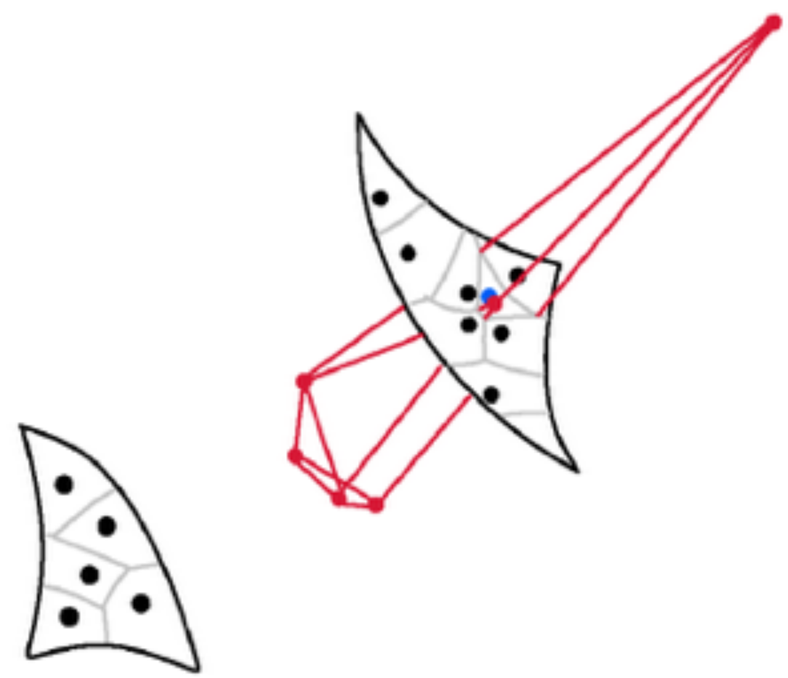
Algorithm for 2d:

- Compute the Delaunay triangulation and the Voronoi diagram of the point set. Include an edge from the triangulation if its circumcircle is empty of all Voronoi vertices.

  - Theorem: The crust of an ε-sample of a smooth (twice differentiable) curve, for ε≤.25, will connect only adjacent sample points.

Moving to 3d

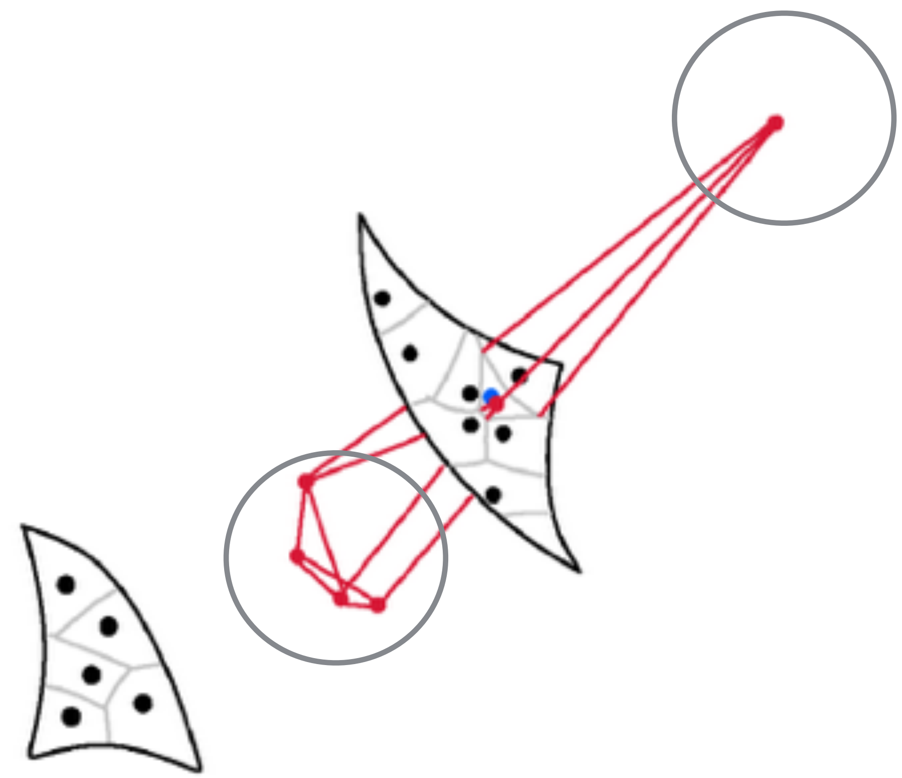
- Unfortunately, this simple filtering will NOT work for surfaces in 3d, because Voronoi vertices do not have to lie near the medial axis, no matter how dense the sample.



Finding a good subset

- However, some of the points are good!

Intuitively, we want to
take cells that exclude
the points of the cell that
are farthest away;
these are the ones
near the medial axis.



Poles

- To formalize this, in [Amenta-Bern] they define the poles of a sample point to be the two farthest vertices of its Voronoi cell, one on each side of the surface.

    - Of course, the algorithm doesn’t know the surface!

- Instead, it chooses the point furthest away as the first pole, and then the second is chosen to be the farthest in the opposite half space.

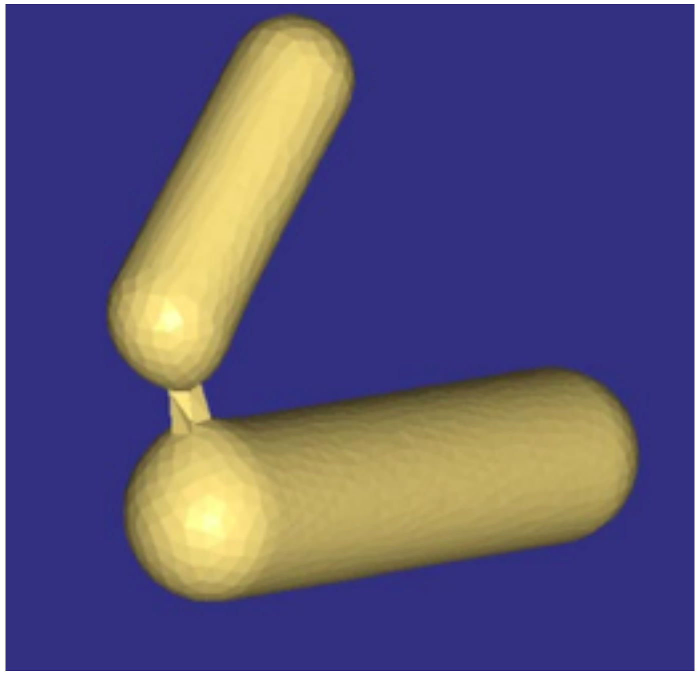
How do to this:

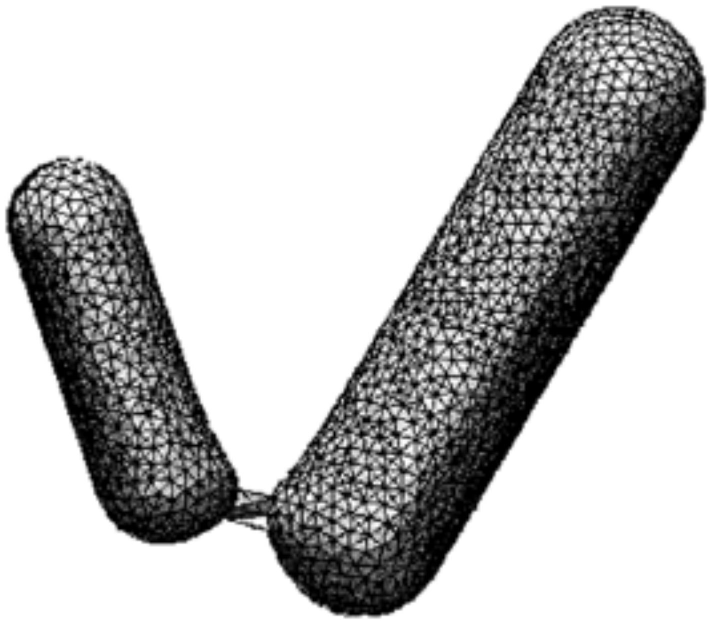
- More formally: if s is the sample point and p the first pole chosen, among all vertices q of the Voronoi cell with the angle $\angle psq > \pi/2$, choose the furthest one

- **Lemma**: Given an ε-sample of a surface, with ε<1/4, and a sample point s with farthest pole p. Then the second pole v will be the farthest Voronoi vertex where the vector sv has negative dot product with sp.

The crust

- We then take the Delaunay triangulation of the input points and their poles.
- The **crust** is the set of Delaunay triangles from this triangulation where all three vertices are sample points.





Quality

- At this point we have a fairly weak theoretical guarantee: it is pointwise convergent to the underlying surface as the sampling density increases.

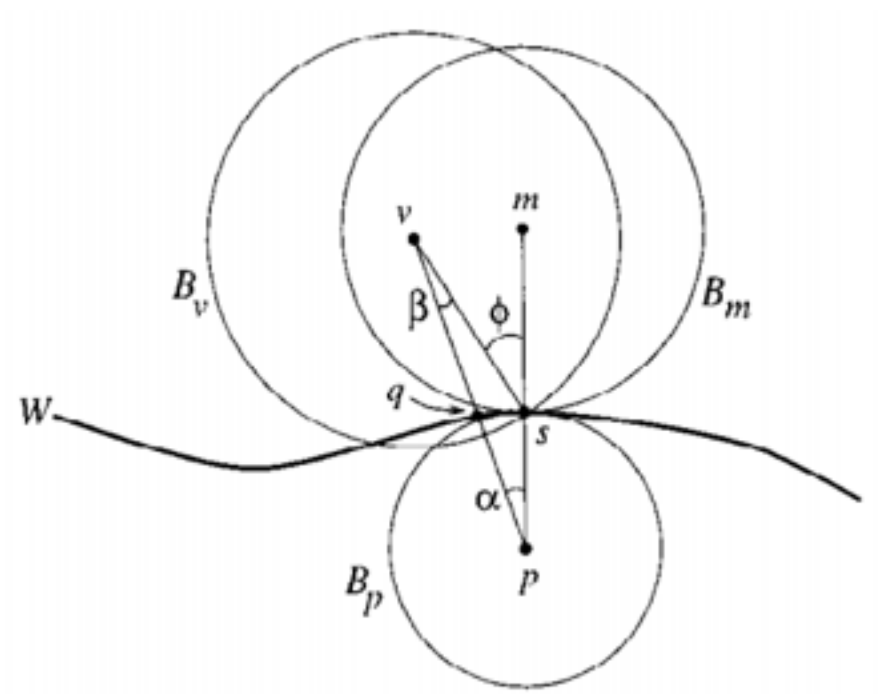
- However, we can still clearly have extra triangles in the result, as there is no guarantee that the normals at each triangle are close to the actual surface normals.

Additional filtering

- The next step in the algorithm is to filter:
  - The bad triangles we want to remove are nearly perpendicular to the underlying surface.
  - However, we don't know the underlying surface!

Using the poles

- Instead, we go back to the poles: we can prove that the line from a sample point to each of its pole is nearly orthogonal to the surface, given a sufficiently dense sample.

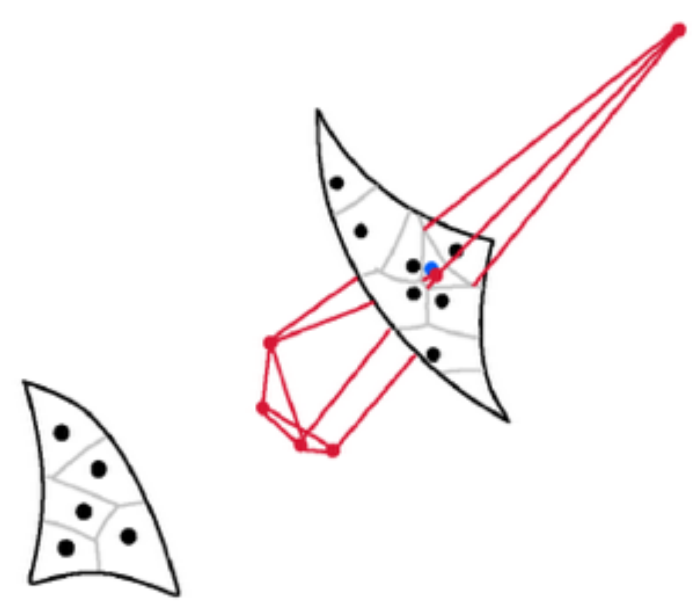


Next step in the algorithm:

- Remove any triangle T for which the normal to T and the vector to the pole at a vertex of the triangle are too large.
  - Greater than θ for the largest angle vertex of T, and greater than $3\theta/2$ for all others.
  - θ is another input parameter, which they set to be 4ε to get good practical results, but this can also be varied to find a “nice” output.

Theoretical guarantee

- More precisely: Take an ε-sample, and set θ=4ε. Let T be a triangle of the crust, trimmed as described on last slide, and take any point t∈T.
  Then the angle between T's normal and the normal to the actual underlying surface at the point closet to t measures O(√ε).



Final cleanup

- After filtering by normals, remaining triangles are roughly parallel to the original surface.
  - Can prove that this set of triangles still contains a piece-wise linear surface homeomorphic to F.
- However, we don’t necessarily have a surface, since there could be small remaining triangles that enclose pockets:
  - All 4 faces of a very flat tetrahedra may make it past the filtering step.

Sharp edges

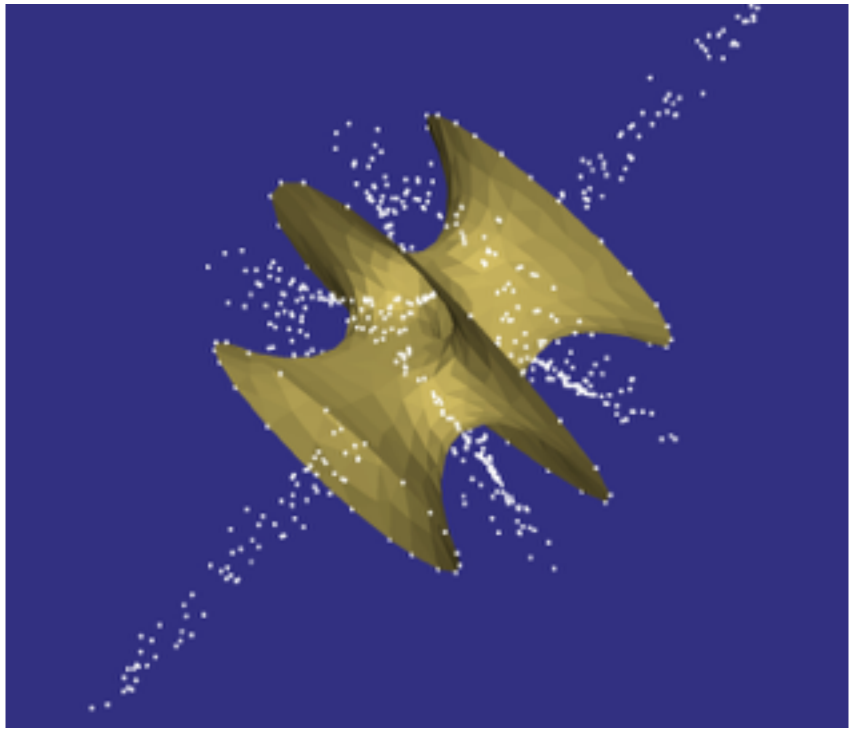
- Define a sharp edge as one which has a dihedral angle greater than $3\pi/2$ between a successive pair of incident triangles in the cyclic order around the edge.
  - In other words, an edge is sharp if all incident triangles are in a small wedge.
  - If only one incident triangle, then automatically sharp.

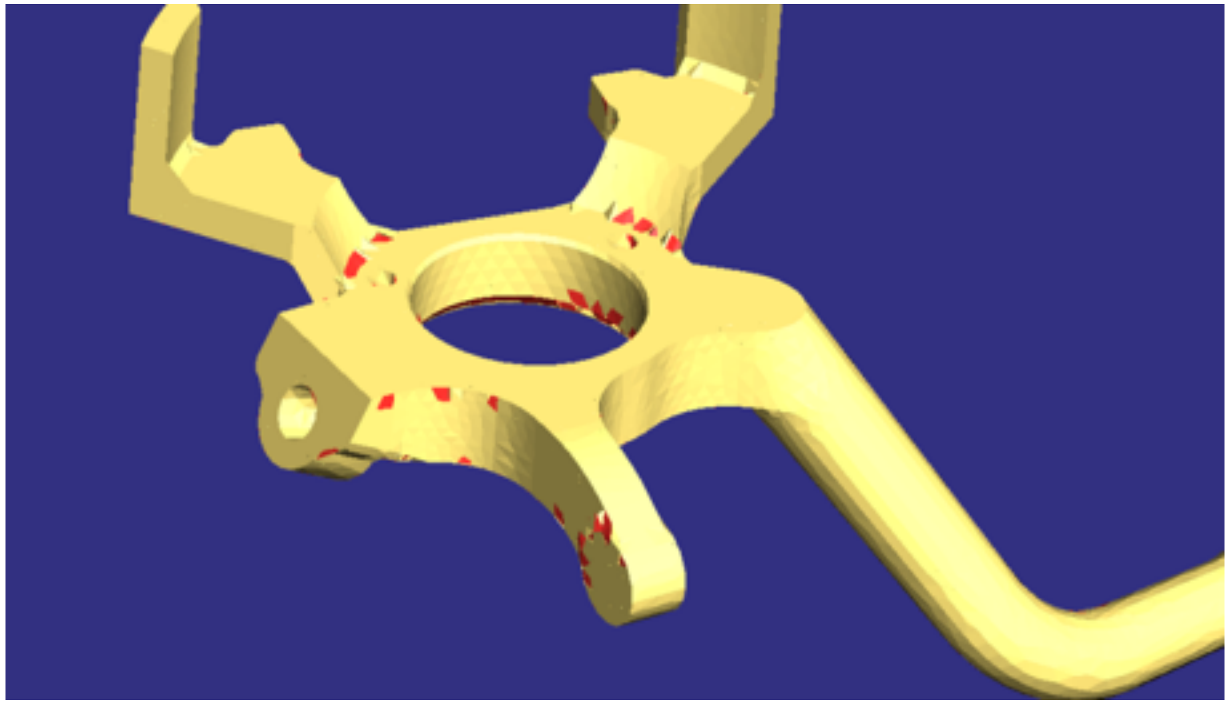
Final trimming

- The final step:
  - orients triangles and poles consistently
  - greedily remove triangles with sharp edges
  - take the “outside” of remaining triangles (which makes sense since we oriented things)

Crust: takeaway

- This was the first algorithm with good, provable guarantees on the quality of the reconstruction.
- The main drawback is ε-samples: it's hard to guarantee a good enough approximation.
- It is also only good for smooth inputs: anything with sharp edges can have holes

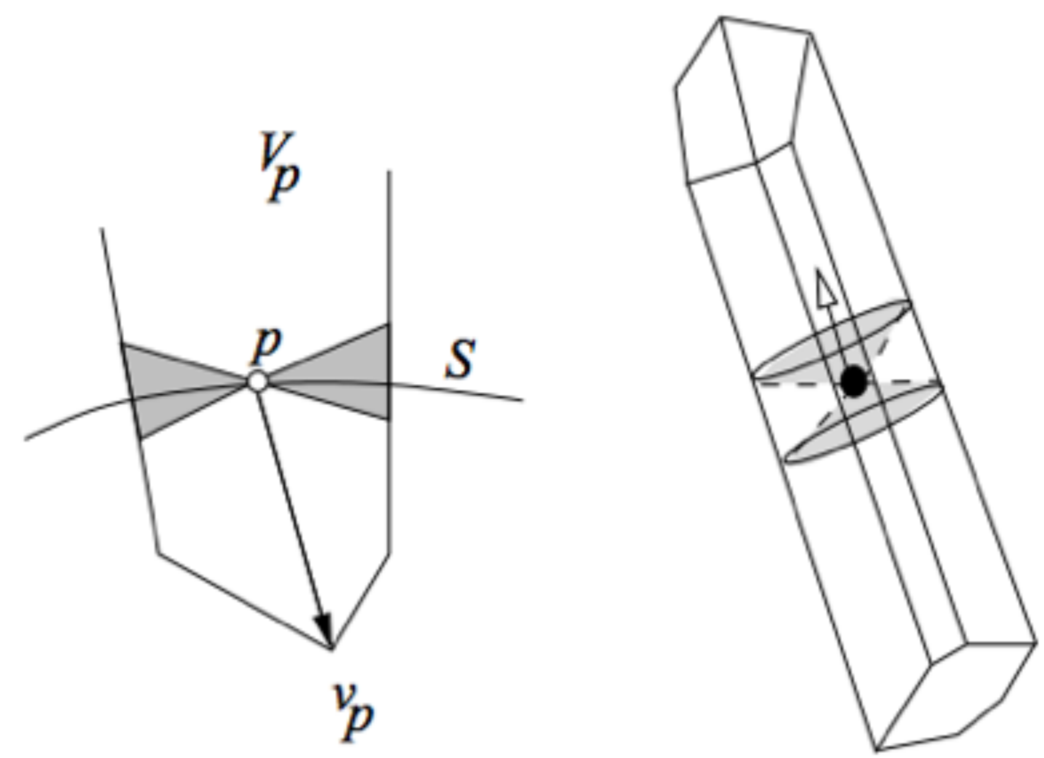




Extension: cocone

- The Cocone algorithm uses the poles from the crust algorithm in order to enumerate a set of triangles that will contain a good reconstruction:

We find any Voronoi edges that intersect the “cocone”, and take triangles from the Delaunay triangulation that are dual to one of these edges.



Cocone result

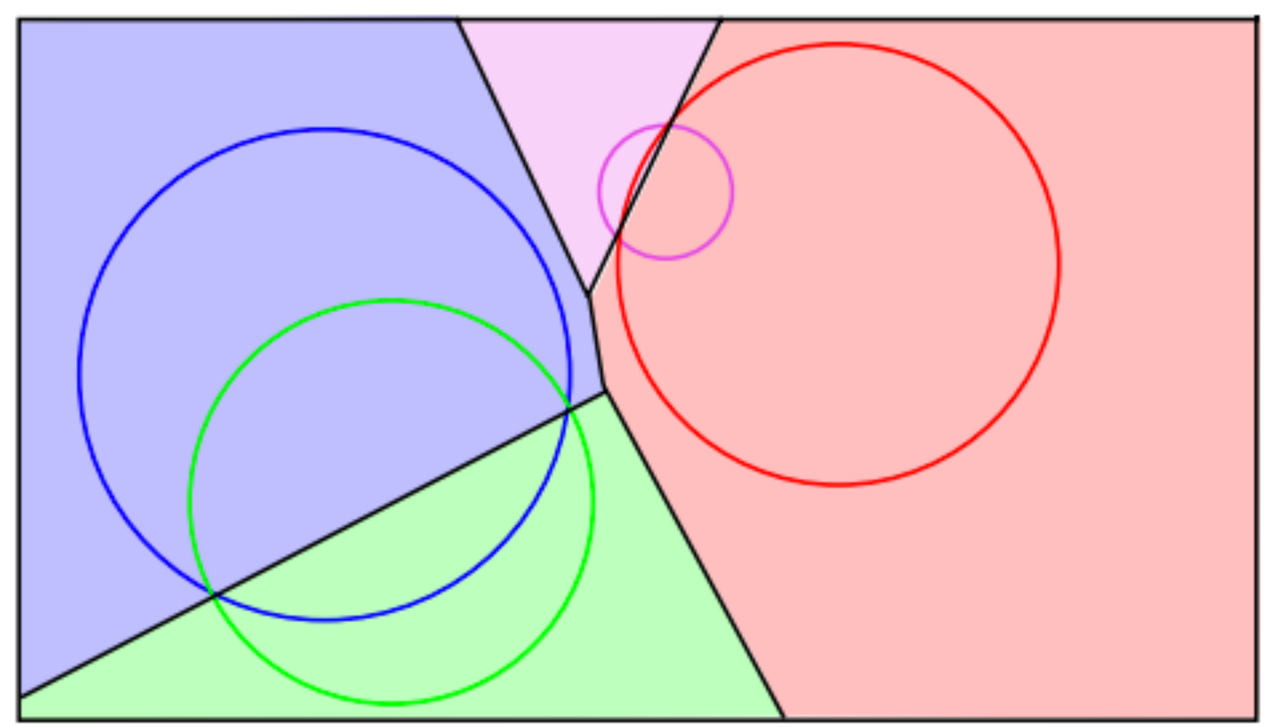
- In the end, the output of cocone is homeomorphic to the original surface, for $\varepsilon \leq .05$.
- In addition, they are also isotopic.
- (Really, same guarantees as in crust, but much simpler to prove and faster to implement.)

Extension: power crust

- The power crust algorithm computes a weighted Voronoi diagram:

- Think of a point c with weight ρ^2 as a ball $B_{c,\rho}$.

- Then the power distance between a point x and a ball $B_{c,\rho}$ as $d^2(c,x)-\rho^2$



Power crust

- The power crust algorithm then just uses the pole vertices (and their Voronoi balls)
  - It computes the power diagram of these polar balls, and does a similar filtering as the normal crust algorithm afterwards.
- It does do better on poorly sampled inputs and things with sharp corners, in practice.
  - The known theoretical guarantees are similar to crust.