600.657: Mesh Processing

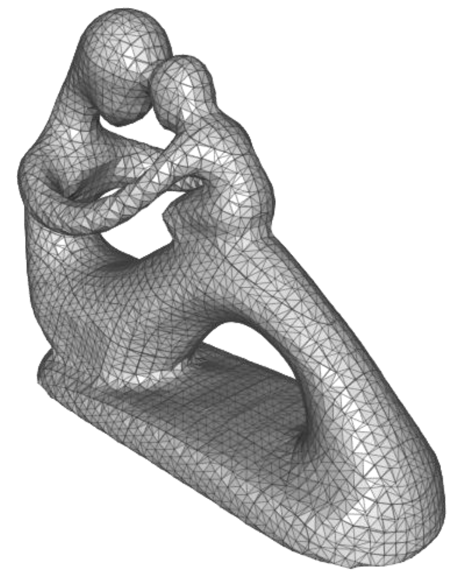
Chapter 6

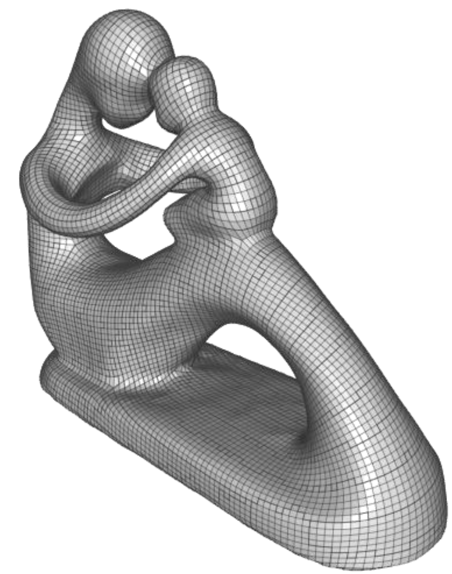
Outline

- Properties of a mesh
- Voronoi Diagrams & Delaunay Triangulations
- Triangle-Based Remeshing
  - Restricted Delaunay
  - Isotropic Remeshing

Properties of a mesh

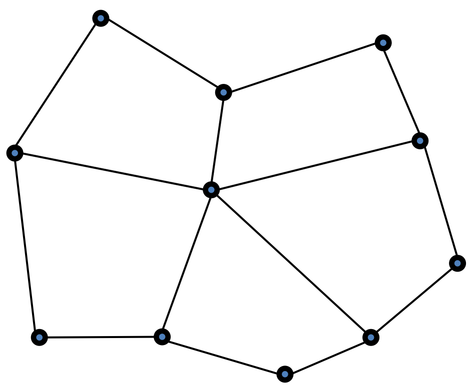
- Element Type
  - Triangles: define a simple function space for FEM simulations
  - Quads: can correspond to a local frame / symmetric matrix

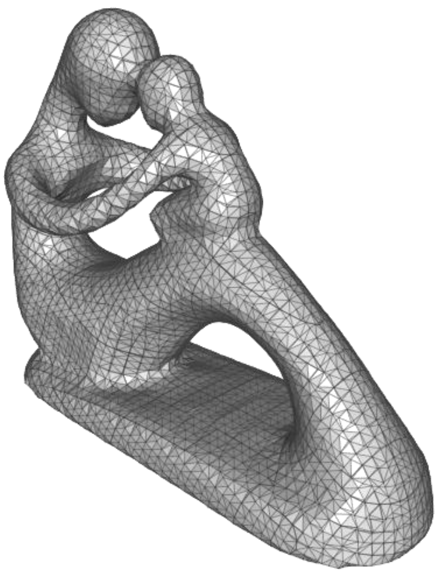


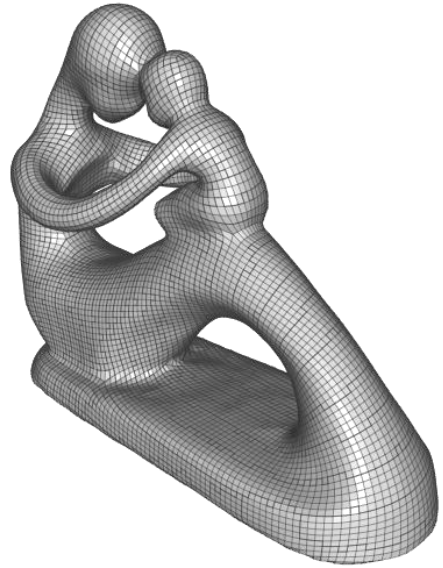


Properties of a mesh

- Element Type
  - Quad to triangles

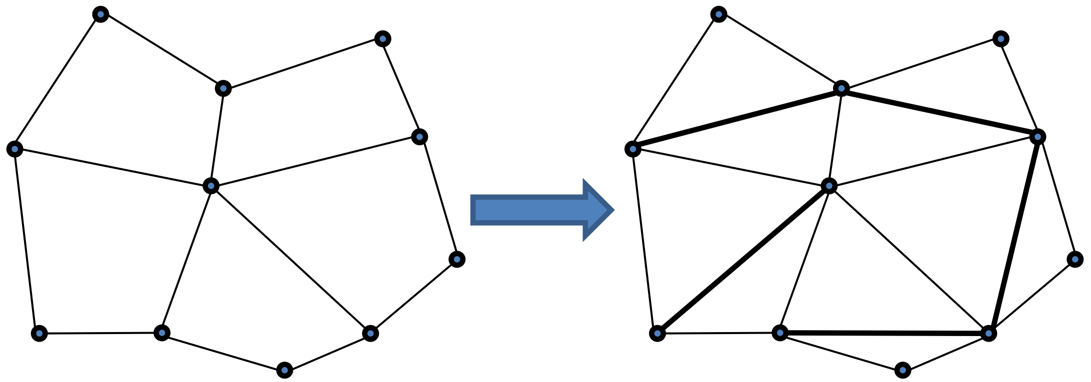


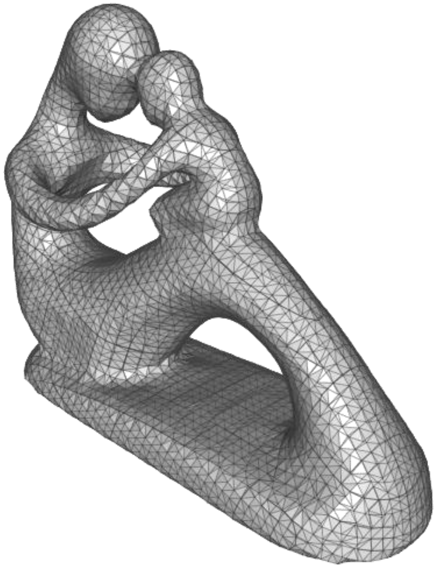


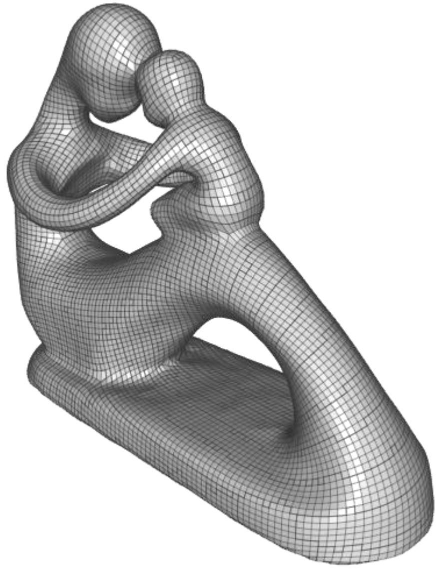


Properties of a mesh

- Element Type
  - Quad to triangles

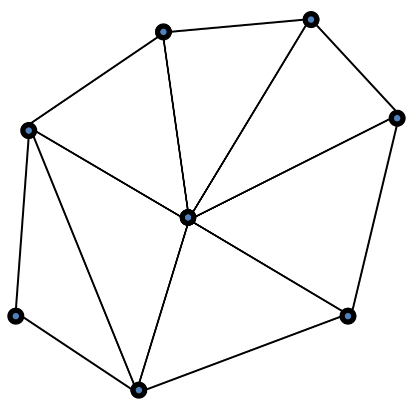


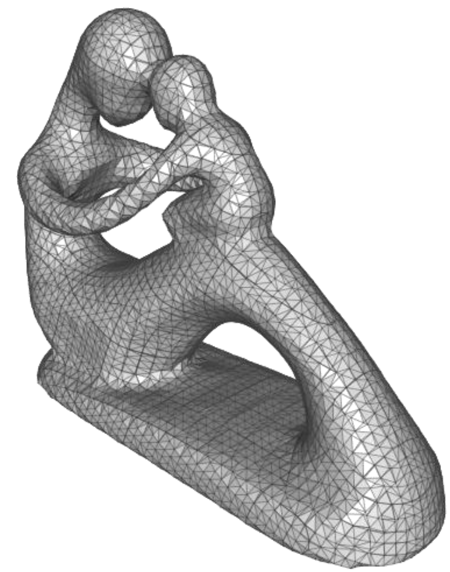


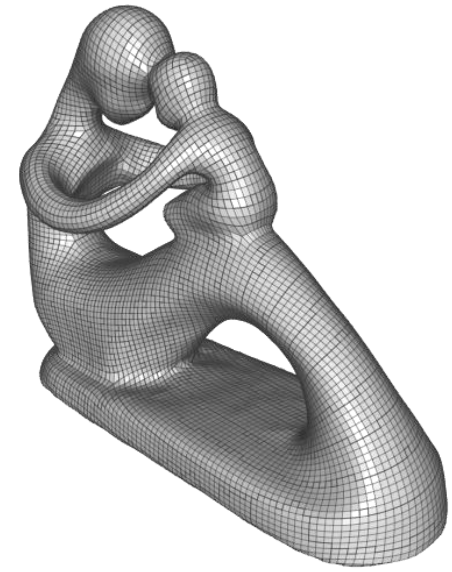


Properties of a mesh

- Element Type
  - Quad to triangles

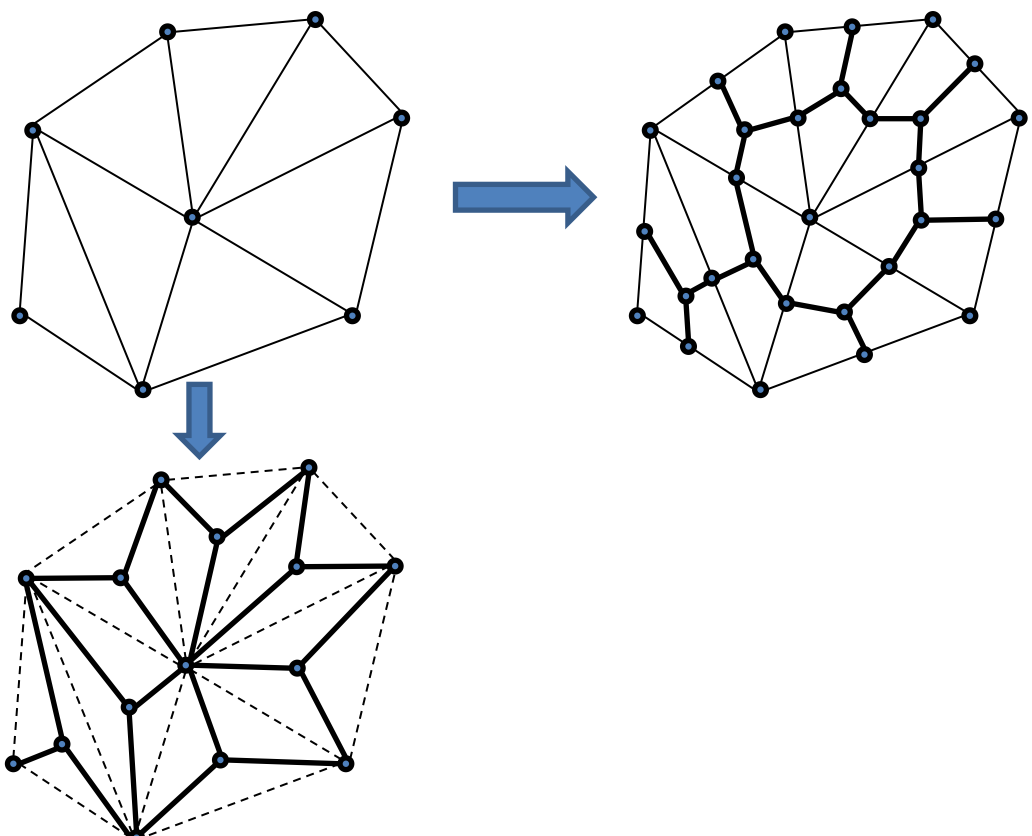


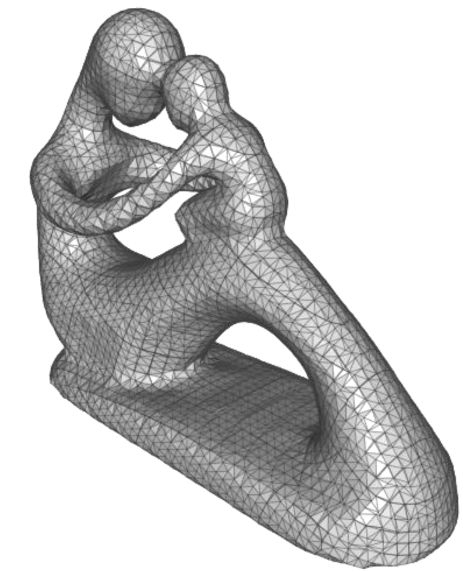


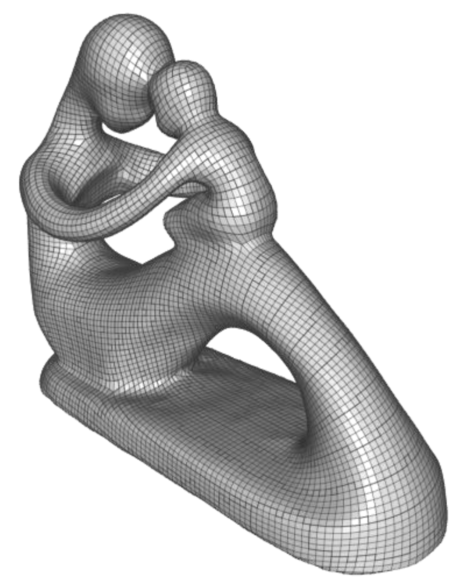


Properties of a mesh

- Element Type
  - Quad to triangles

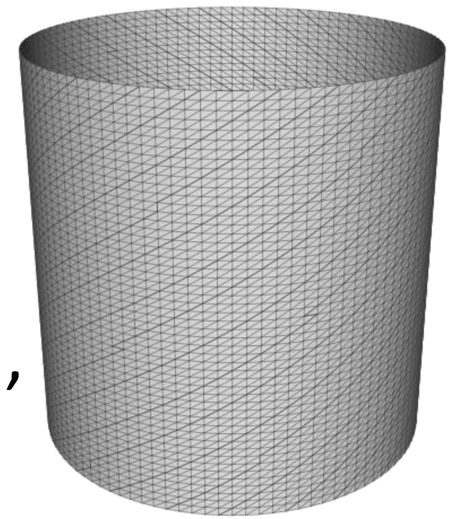


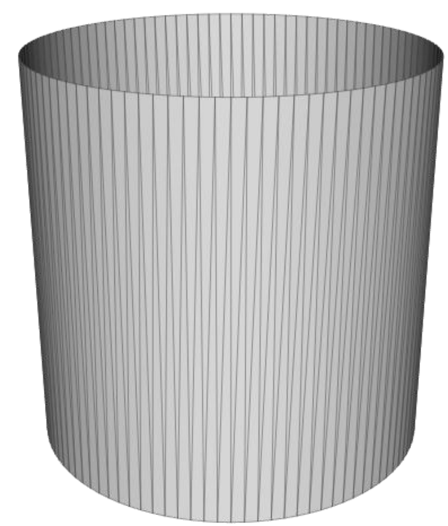




Properties of a mesh

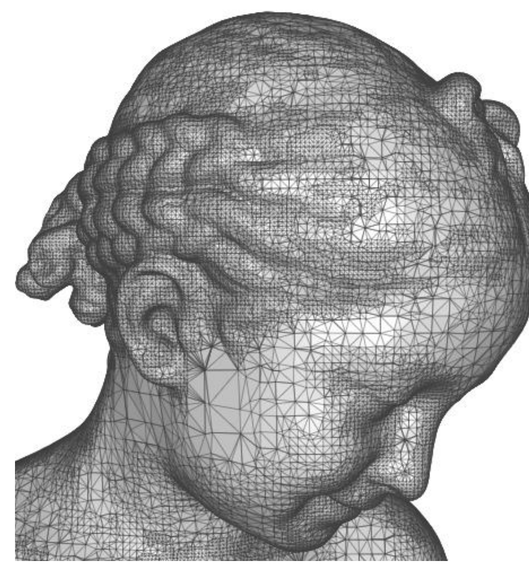
- Element Shape
  - Isotropic: better for FEM simulations
  - Anisotropic: supports an equally good, but cheaper, fit to the surface.





Properties of a mesh

- Element Density
  – Uniform: better for FEM
  – Adaptive: requires fewer elements to represent a surface of the same geometry complexity.



Outline

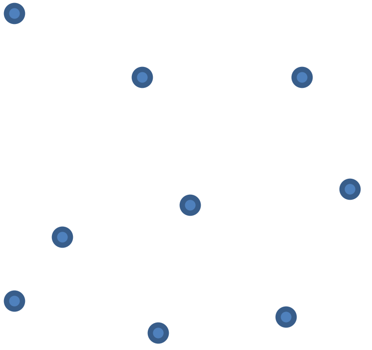
- Properties of a mesh
- Voronoi Diagrams & Delaunay Triangulations
- Triangle-Based Remeshing

Convex Hulls

Definition:

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

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

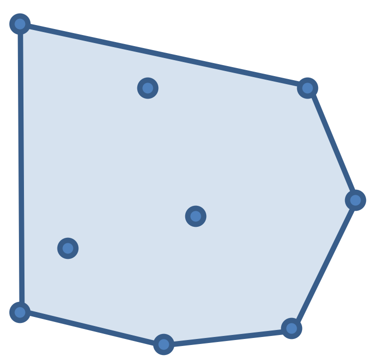


Convex Hulls

Definition:

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

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

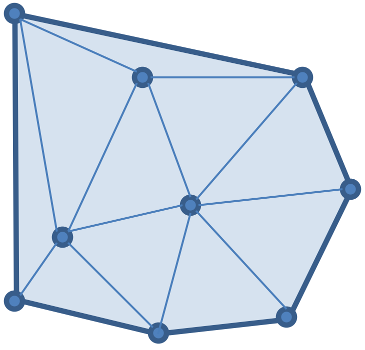


Planar Triangulations

Definition:

A *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:

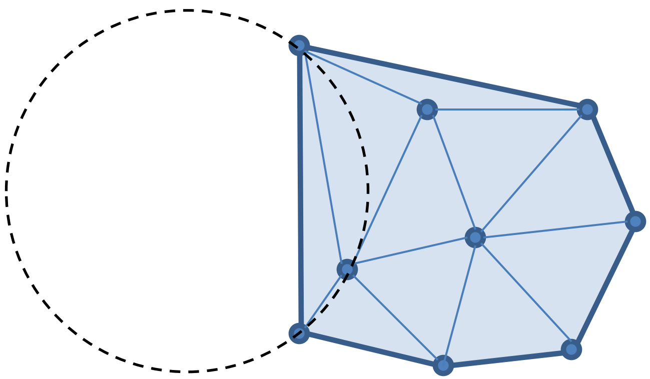
- The set of triangle vertices equals P
- The intersections of two triangles is either empty or is a common edge or vertex.



Delaunay Triangulations

<u>Definition</u>:

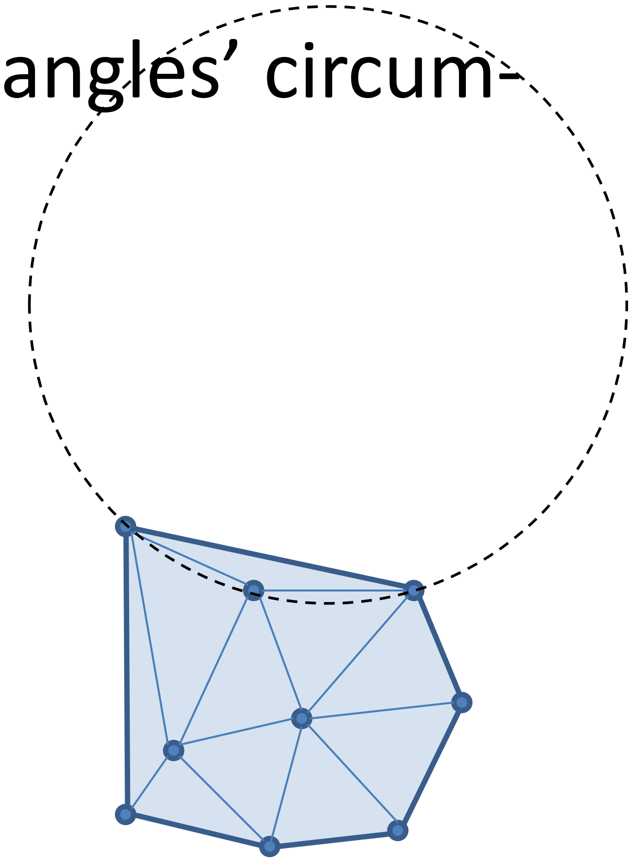
A triangulation of the set *P* is said to be *Delaunay* if the interior of the triangles' circum-circles are empty.



Delaunay Triangulations

Definition:

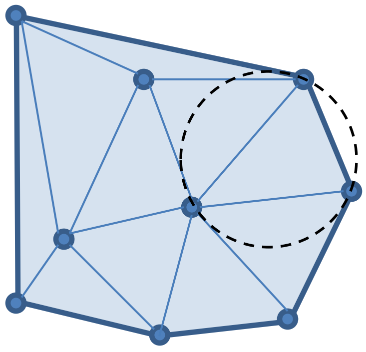
A triangulation of the set *P* is said to be *Delaunay* if the interior of the triangles' circum-circles are empty.



Delaunay Triangulations

Definition:

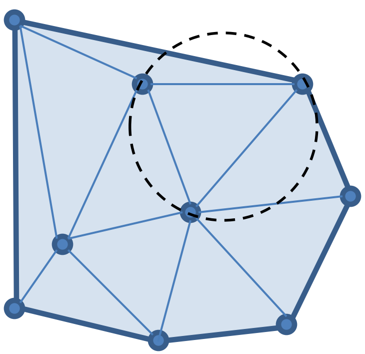
A triangulation of the set P is said to be *Delaunay* if the interior of the triangles' circum-circles are empty.



Delaunay Triangulations

Definition:

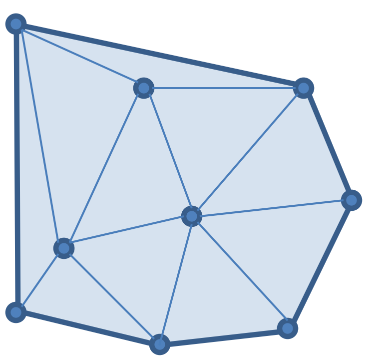
A triangulation of the set *P* is said to be *Delaunay* if the interior of the triangles' circum-circles are empty.



Delaunay Triangulations

Computing the Delaunay Triangulation:

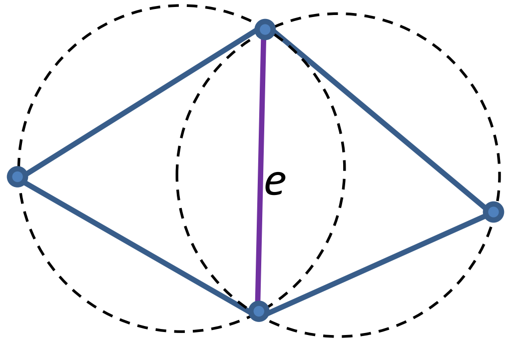
- Incremental
- Divide and Conquer
- Sweepline (planar)
- Convex hulls of paraboloids



Delaunay Edges

<u>Definition</u>:

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



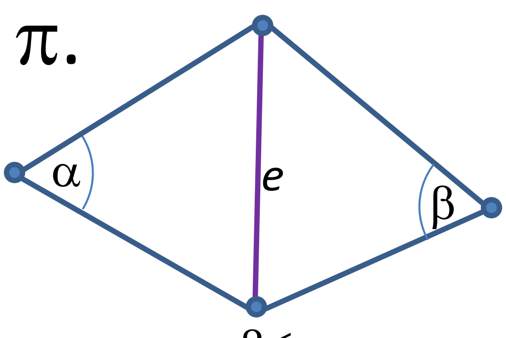
Delaunay Edges

<u>Definition</u>:

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

<u>Property</u>:

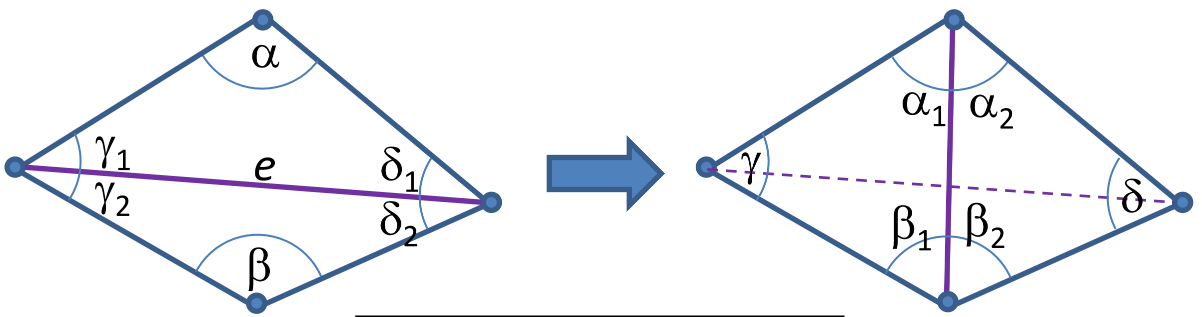
An interior edge is Delaunay iff. the sum of the opposite angles is not greater than π.



Delaunay Edges

<u>Note</u>:

If the sum of the opposite angles is greater than π, then flipping the edge will give a sum that is less than π.



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

Delaunay Triangulations

Property:

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

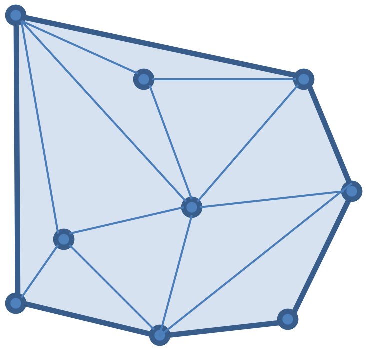
Delaunay Triangulations

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.



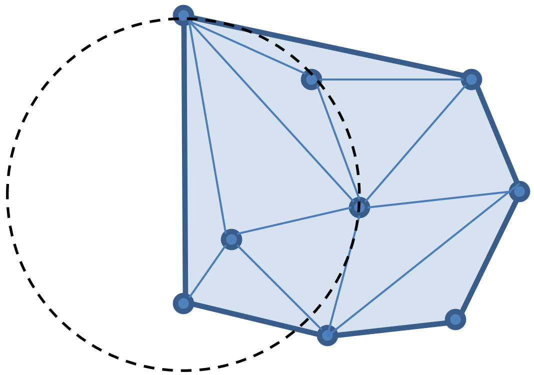
Delaunay Triangulations

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.



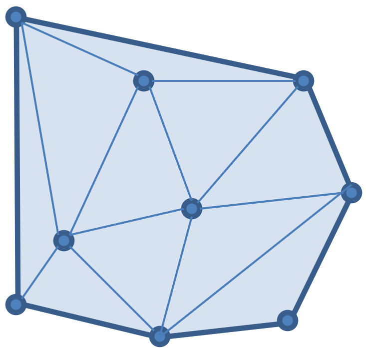
Delaunay Triangulations

<u>Property</u>:

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

<u>Edge Flipping Algorithm</u>:

Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.



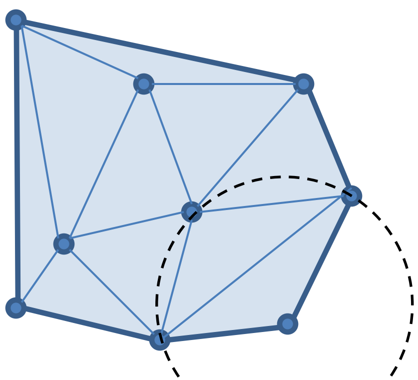
Delaunay Triangulations

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.



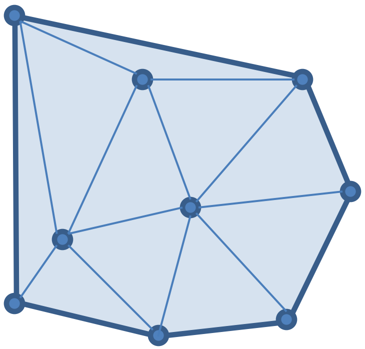
Delaunay Triangulations

<u>Property</u>:

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

<u>Edge Flipping Algorithm</u>:

Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.

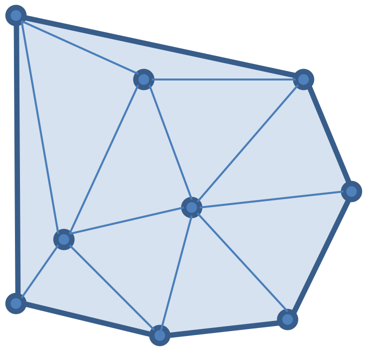


Delaunay Triangulations

Edge Flipping Algorithm:

Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.

Is this algorithm guaranteed to terminate?



Delaunay Triangulations

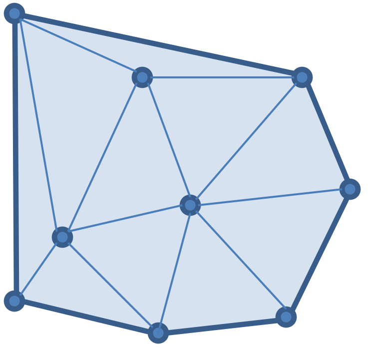
<u>Edge Flipping Algorithm</u>:

Starting with an arbitrary triangulation, flip edges until each edge is locally Delaunay.

<u>Is this algorithm guaranteed to terminate</u>?

Termination is proved by:

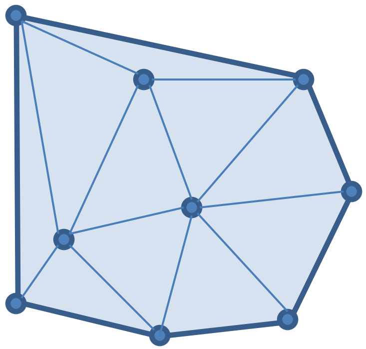
- Showing that there finitely many different triangulations.
- Defining a global "energy" that is reduced with each flip (e.g. sum of squared circum-radii.)



Delaunay Triangulations

<u>Why Should we Care</u>:

Of all possible triangulations, the Delaunay triangulation maximizes the minimal angle.

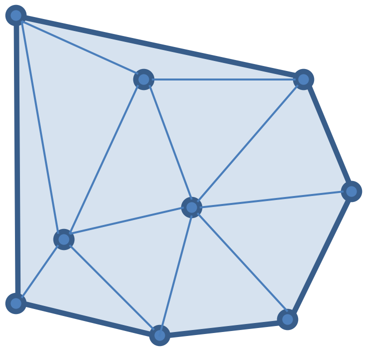


Delaunay Triangulations

Why Should we Care:

Of all possible triangulations, the Delaunay triangulation maximizes the minimal angle.

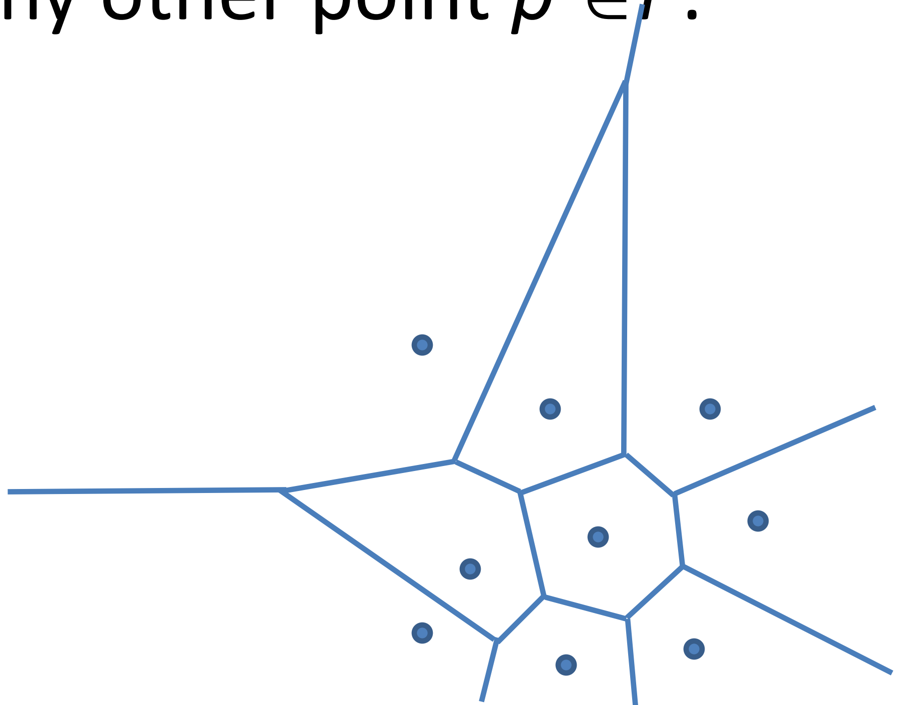
This results in a triangulation with “well-formed” triangles, facilitating numerical processing over the triangulation.



Voronoi Diagram

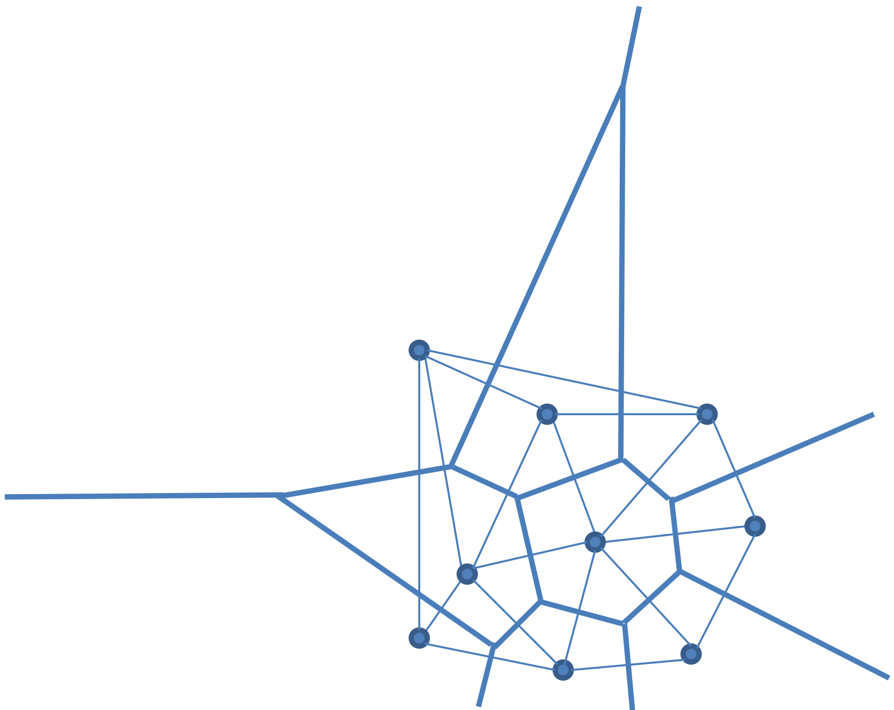
<u>Definition</u>:

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.

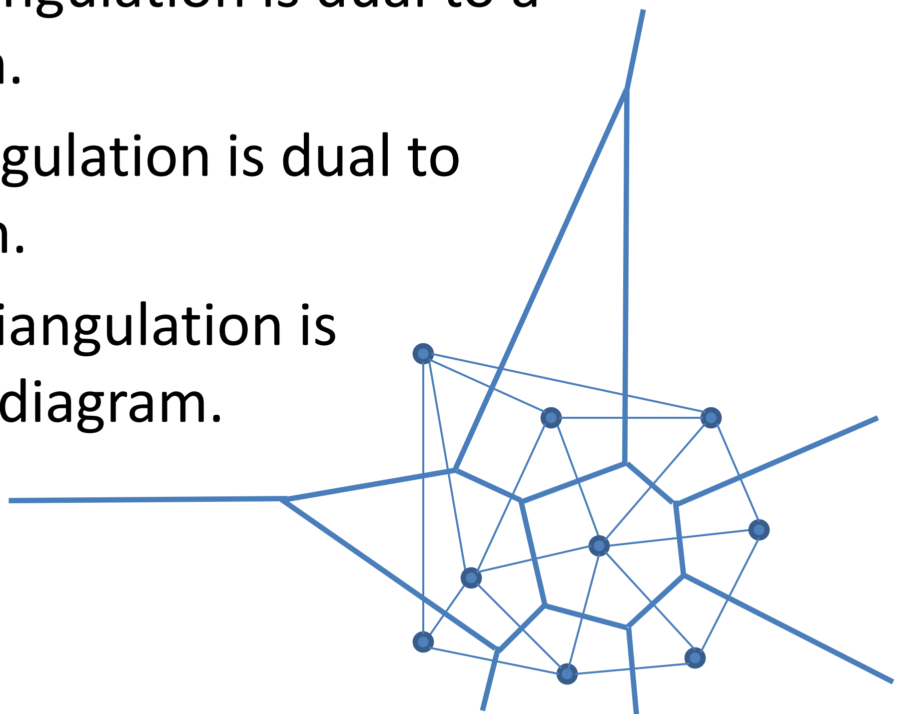


Voronoi Diagram

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

– <u>2D</u>:

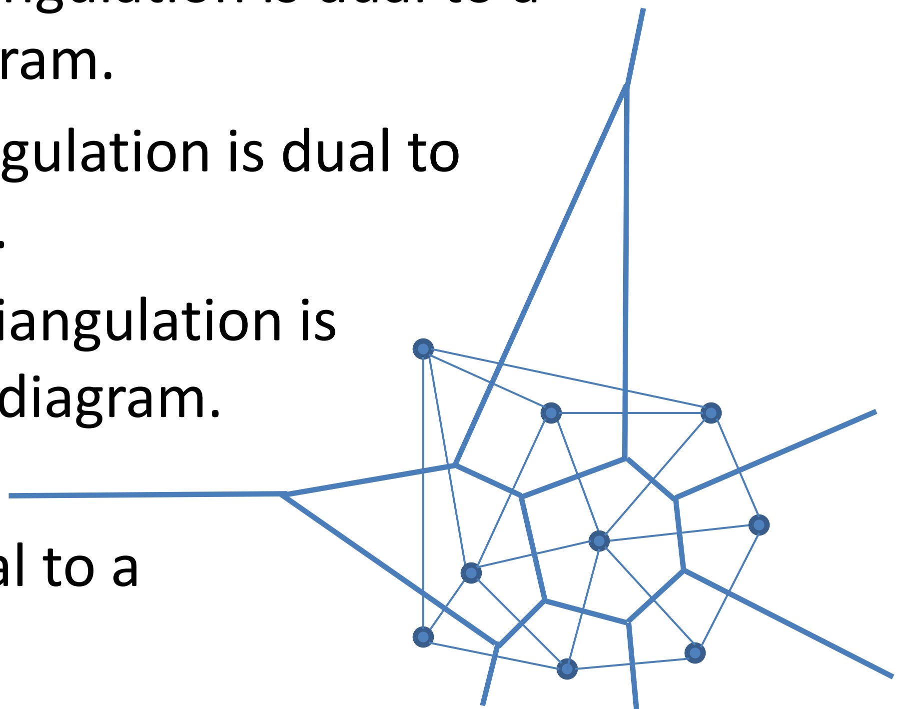
- Every vertex of the triangulation is dual to a polygon in the diagram.
- Every edge of the triangulation is dual to an edge of the diagram.
- Every triangle of the triangulation is dual to a vertex of the diagram.



Voronoi 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.



Restricted Delaunay Triangulation

Goal:

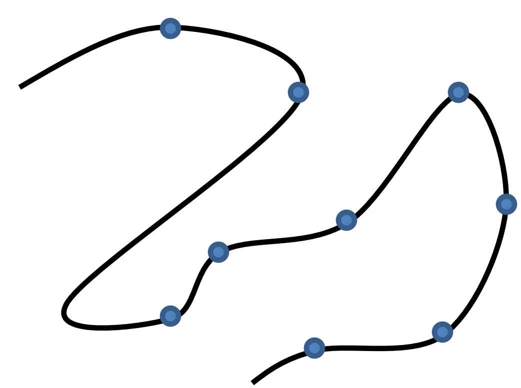
Given a surface *S* and a set of points *P* in *S*, we would like to compute a good triangulation of *P* that is true* to the surface.

*Note that not every point set *P* has to admit a true triangulation.

Restricted Delaunay Triangulation

<u>Approach (Take 1)</u>:

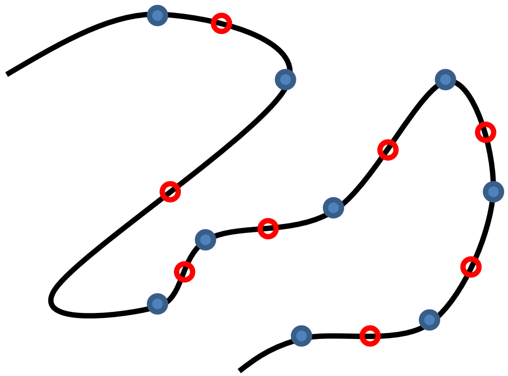
We could compute a Voronoi Diagram on *S* using the notion of distances on the surface, and then take the dual to get a Delaunay Triangulation.



Restricted Delaunay Triangulation

Approach (Take 1):

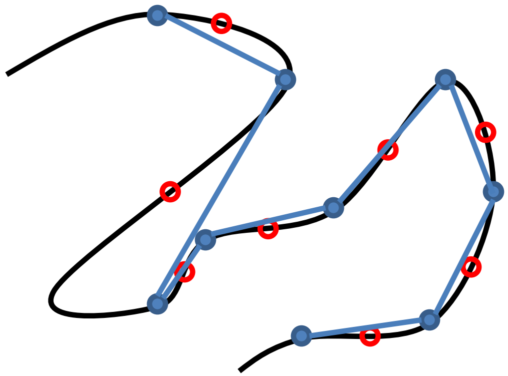
We could compute a Voronoi Diagram on *S* using the notion of distances on the surface, and then take the dual to get a Delaunay Triangulation.



Restricted 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.



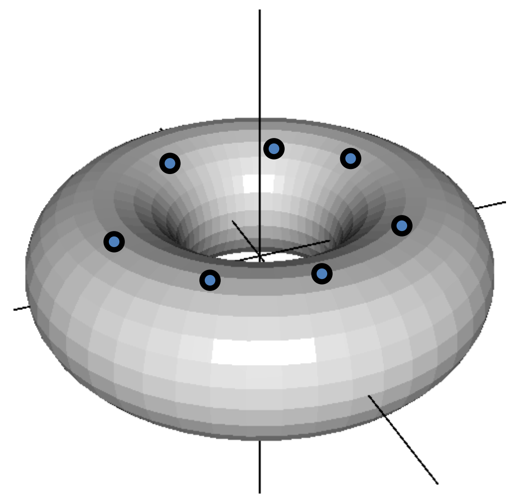
Restricted Delaunay Triangulation

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.

Challenges:

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



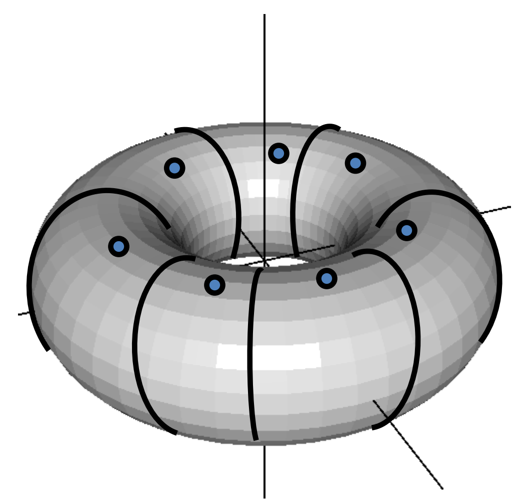
Restricted Delaunay Triangulation

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).



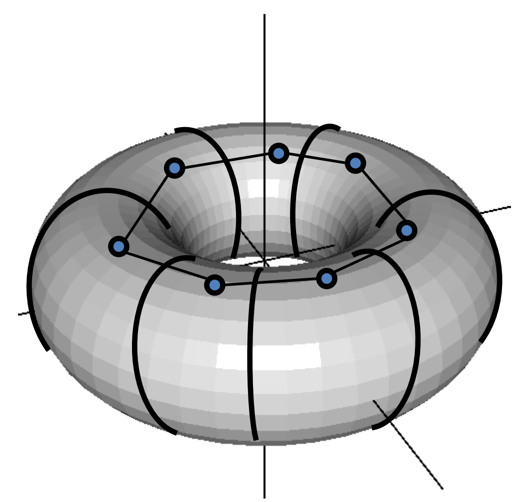
Restricted Delaunay Triangulation

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:

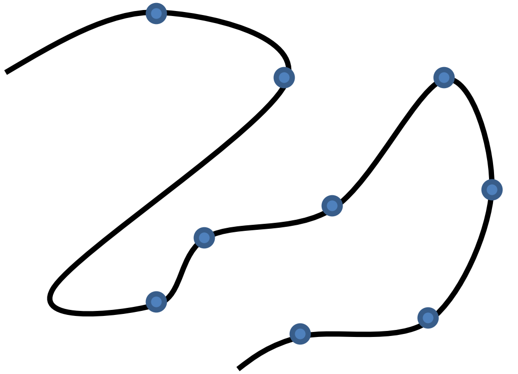
1. Measuring distances on a surface can be expensive.
2. The dual complex may not be a manifold (or even have any triangles).



Restricted Delaunay Triangulation

Approach (Take 2):

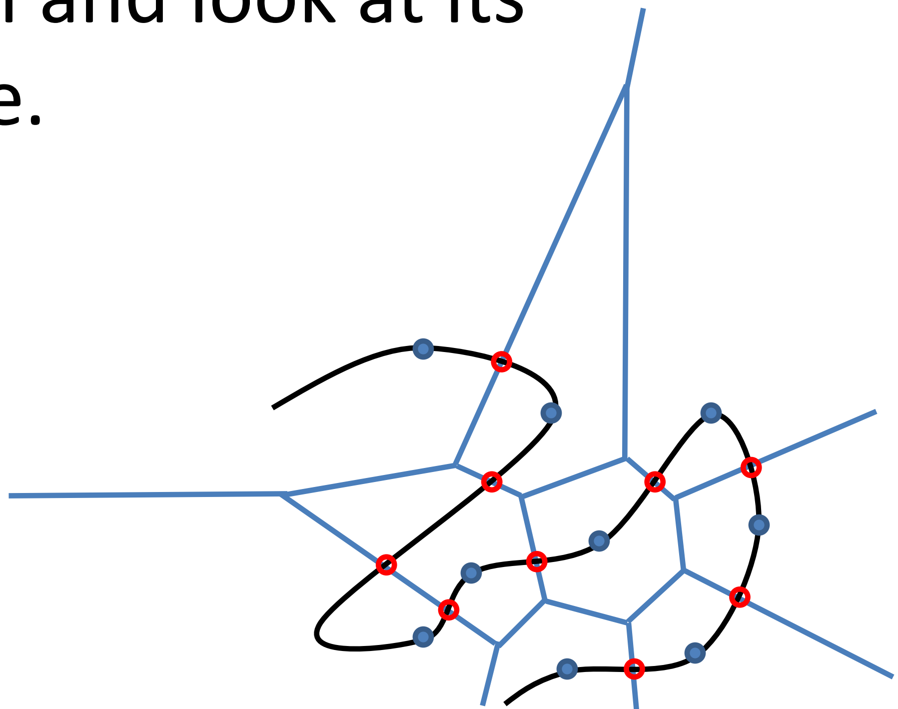
Instead of trying to compute a Voronoi Diagram using distances on the surface, compute a regular Voronoi Diagram and look at its restriction to the surface.



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.

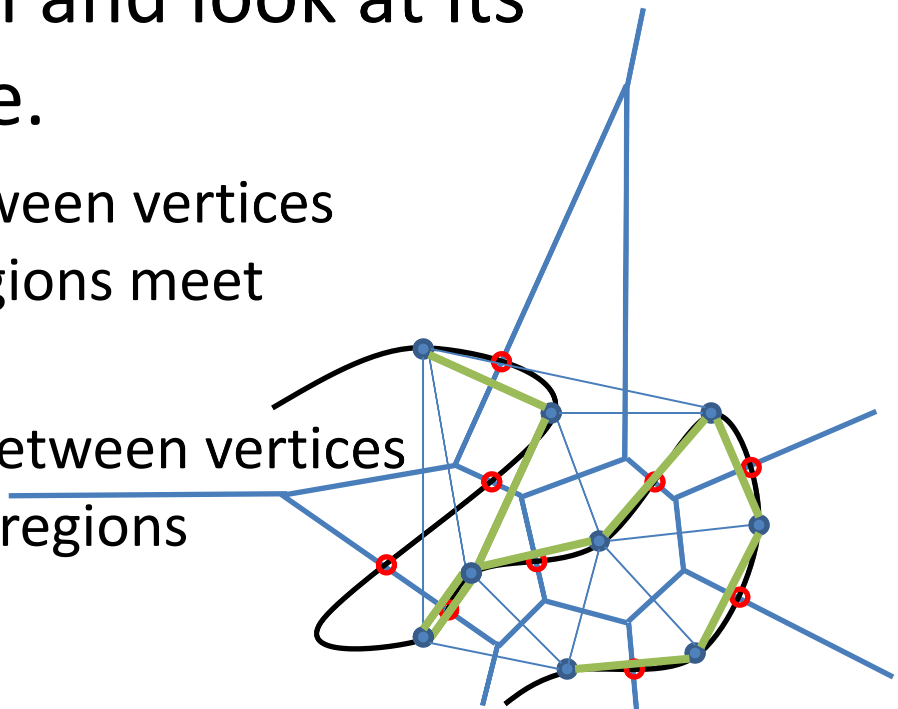


Restricted Delaunay Triangulation

<u>Approach (Take 2)</u>:

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 <u>on the surface</u>.
- Add a Delaunay triangle between vertices $p, p', p'' \in P$ if their Voronoi regions meet <u>on the surface</u>.



Restricted Delaunay Triangulation

Approach (Take 2):

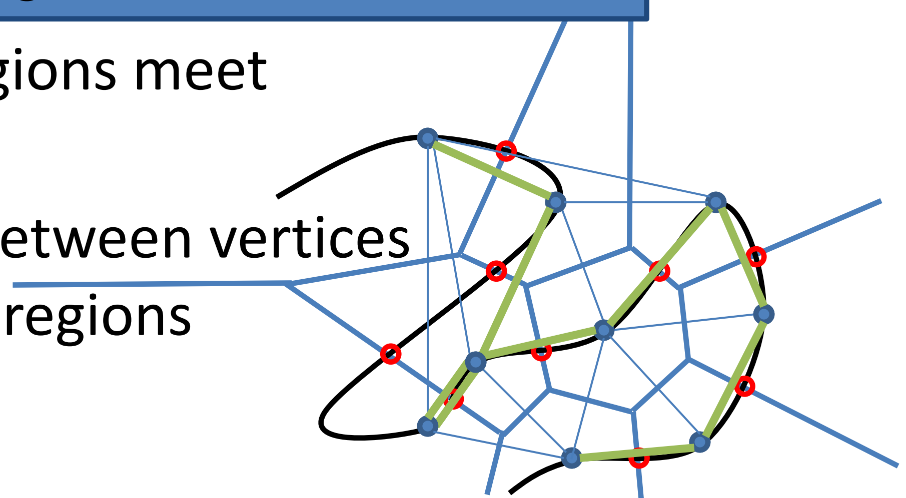
Instead of trying to compute a Voronoi Diagram

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 mee 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

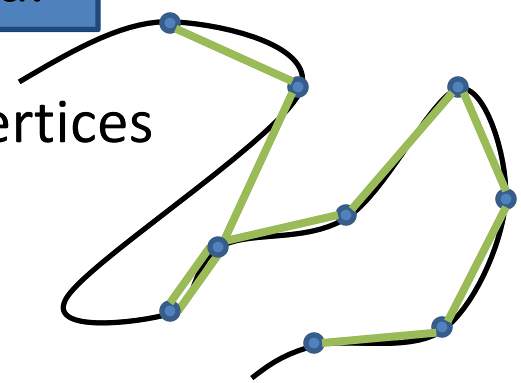
Note:

- 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 mee ... e surface.

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

... on the surface.

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



Outline

- Properties of a mesh
- Voronoi Diagrams & Delaunay Triangulations
- Triangle-Based Remeshing
  - Restricted Delaunay
  - Isotropic Remeshing

Restricted Delaunay [Boissonnat & Oudot '05]

<u>Goal</u>:

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

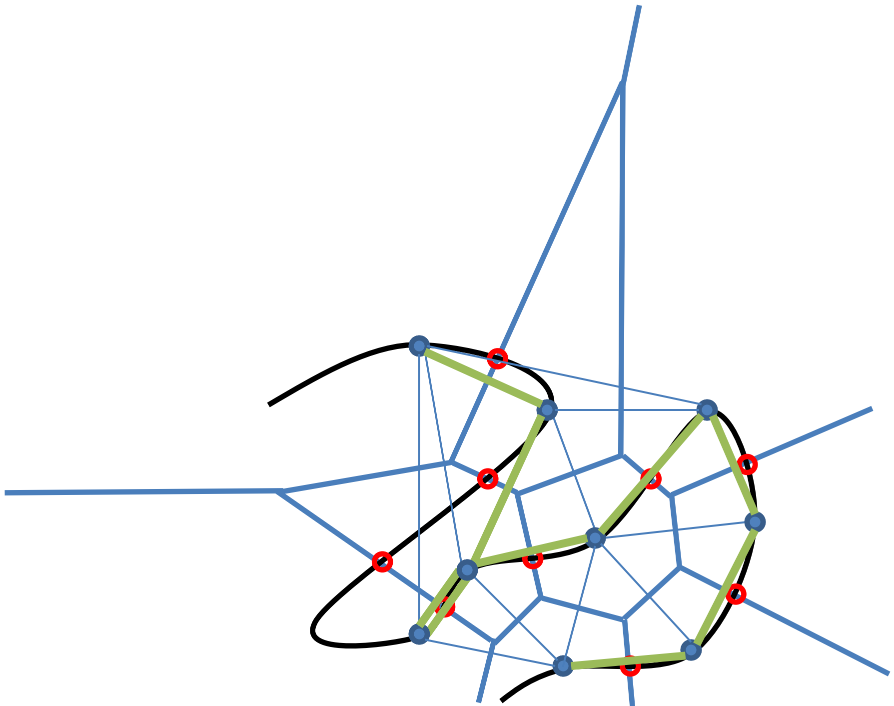
<u>Approach</u>:

Ensure that the complex is manifold by inserting a additional points when it is not.

Restricted Delaunay [Boissonnat & Oudot '05]

General Idea:

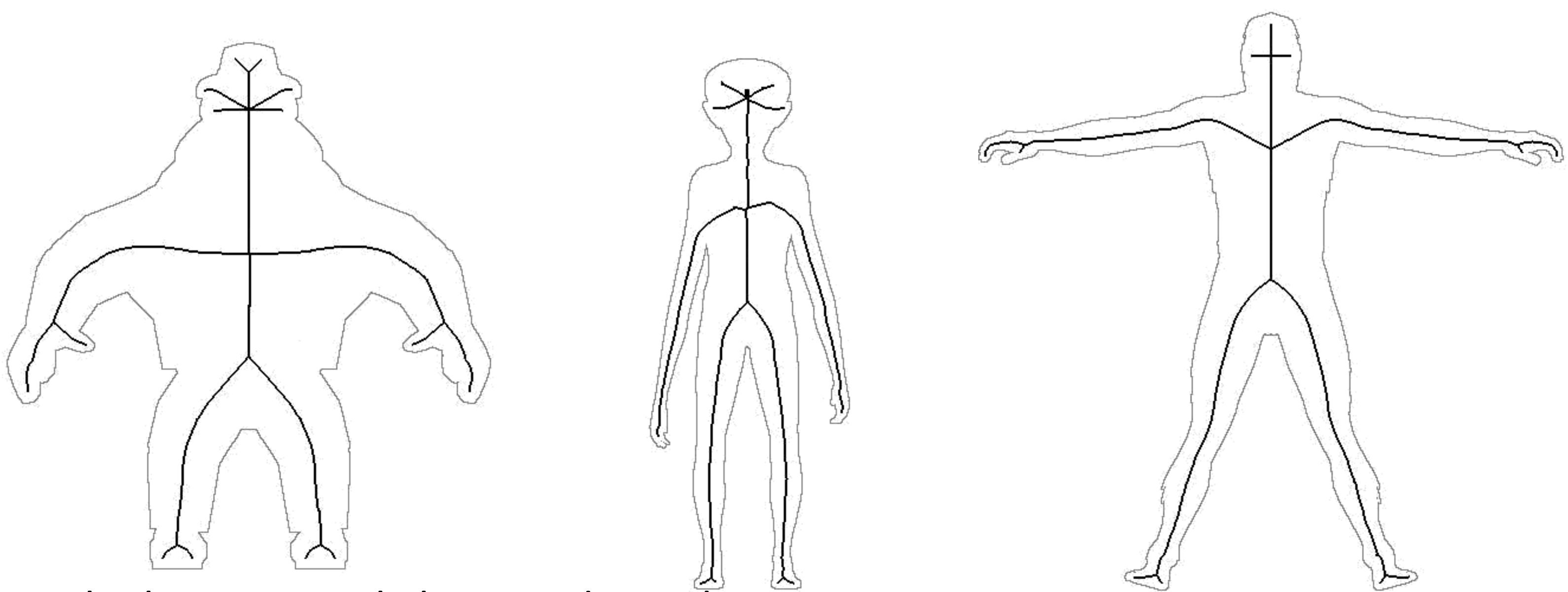
The restricted Delaunay Triangulation will fail to be manifold when the samples are not well-spaced.



Restricted Delaunay [Boissonnat & Oudot '05]

<u>Definition</u>:

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.

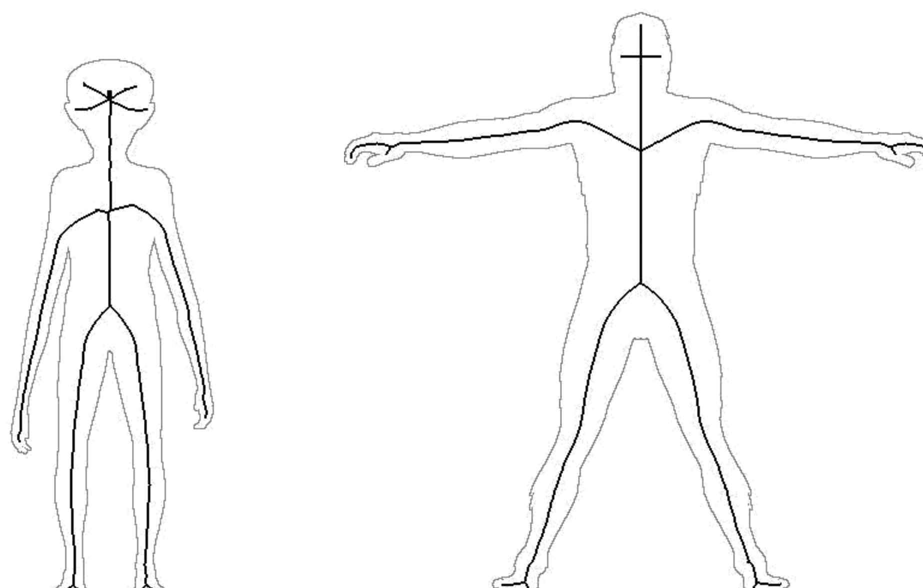
Restricted Delaunay [Boissonnat & Oudot '05]

<u>Definition</u>:

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

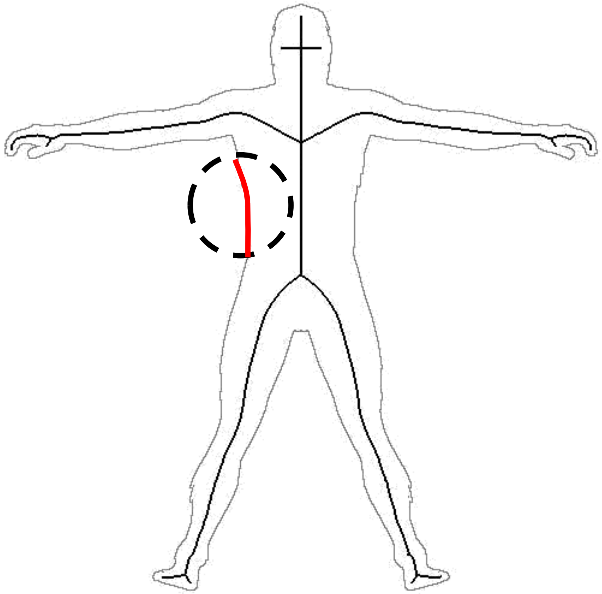


*Note that only the interior skeleton is drawn here.

Restricted Delaunay [Boissonnat & Oudot '05]

<u>Note</u>:

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]

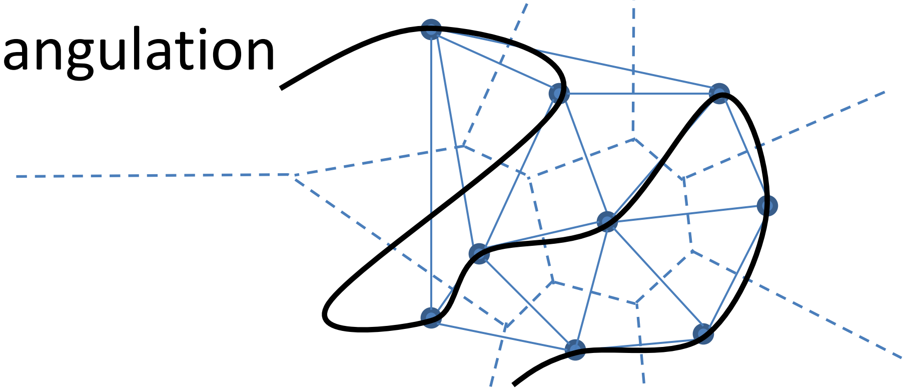
Algorithm:

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]

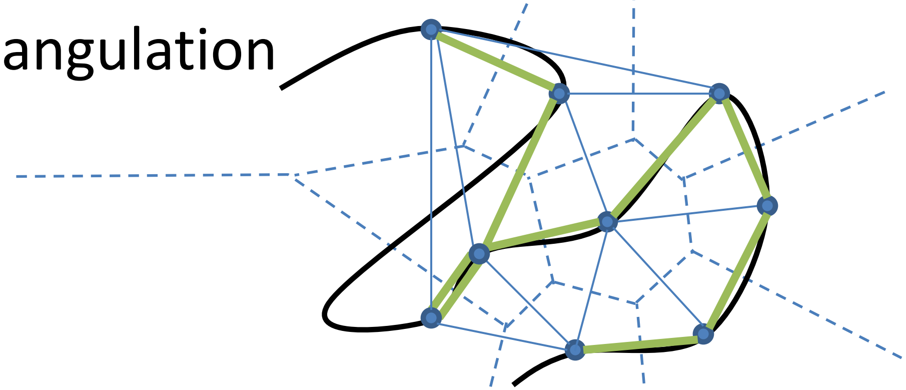
Algorithm:

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]

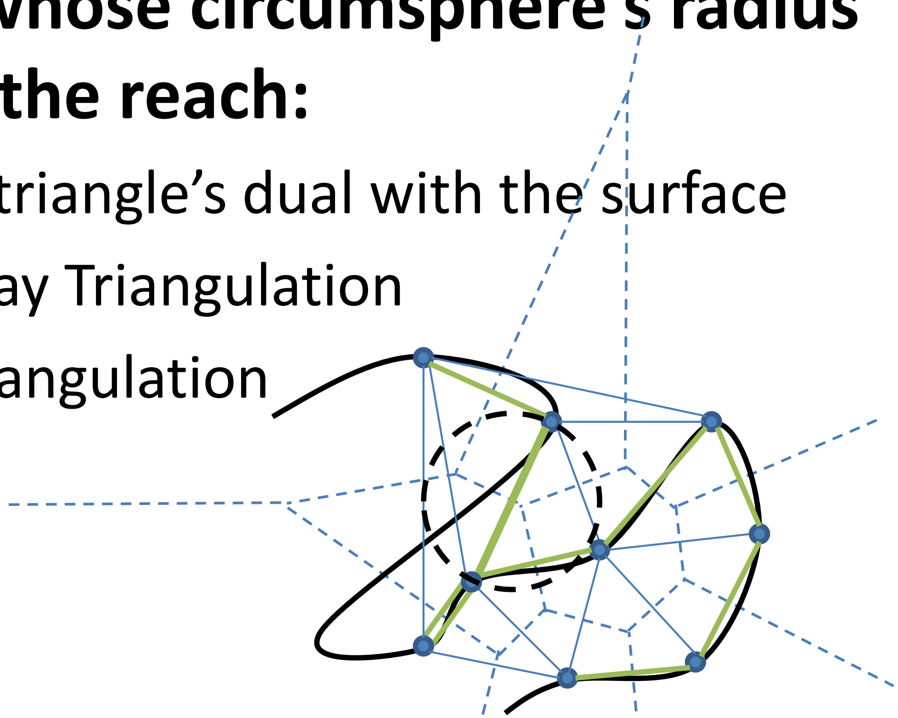
Algorithm:

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]

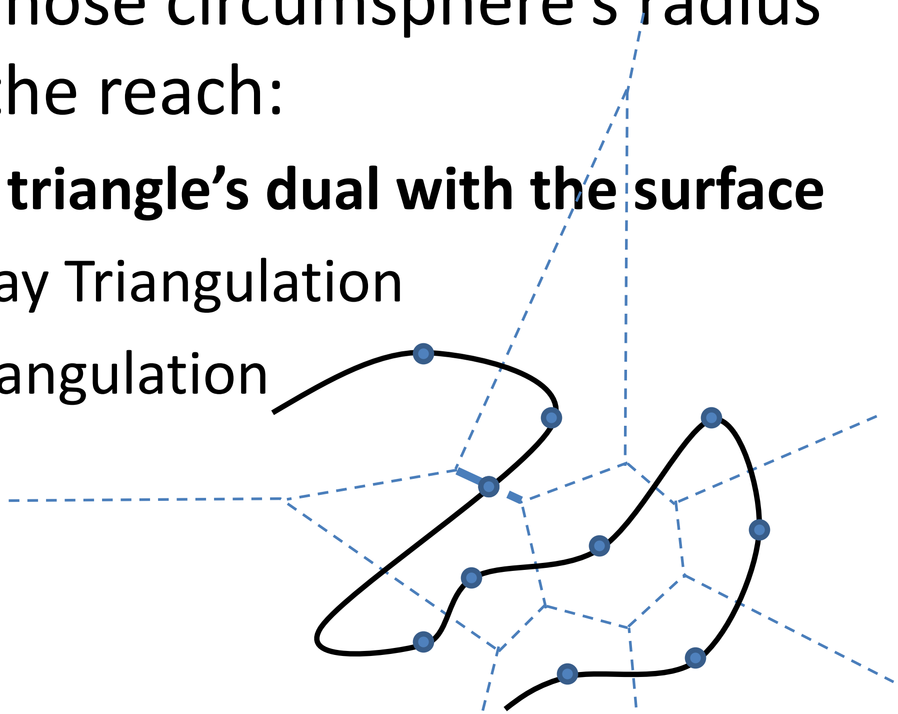
Algorithm:

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.

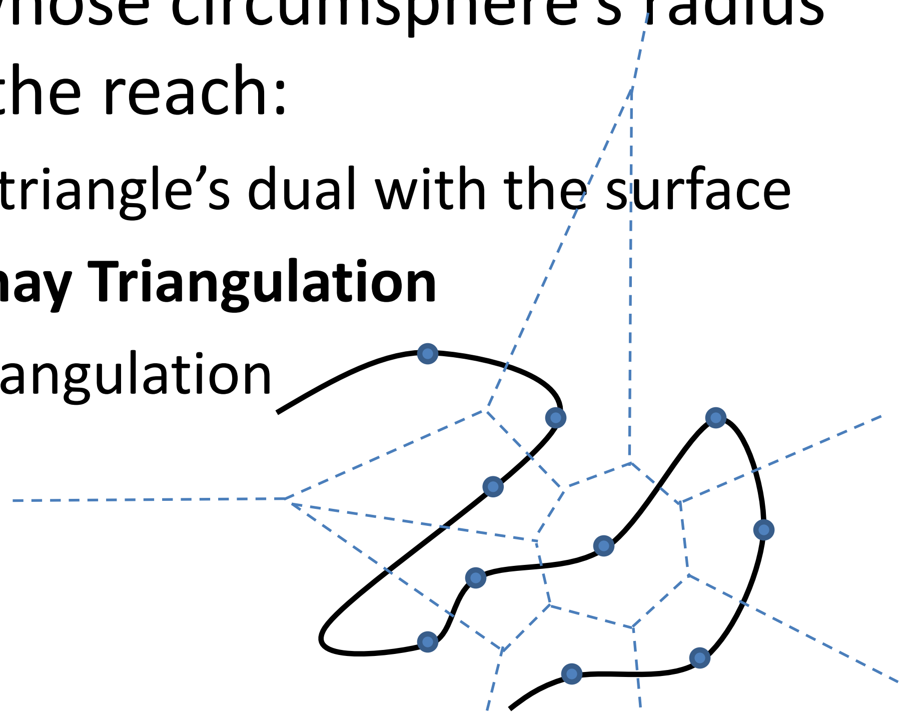
Compute the restricted D. Triangulation

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]

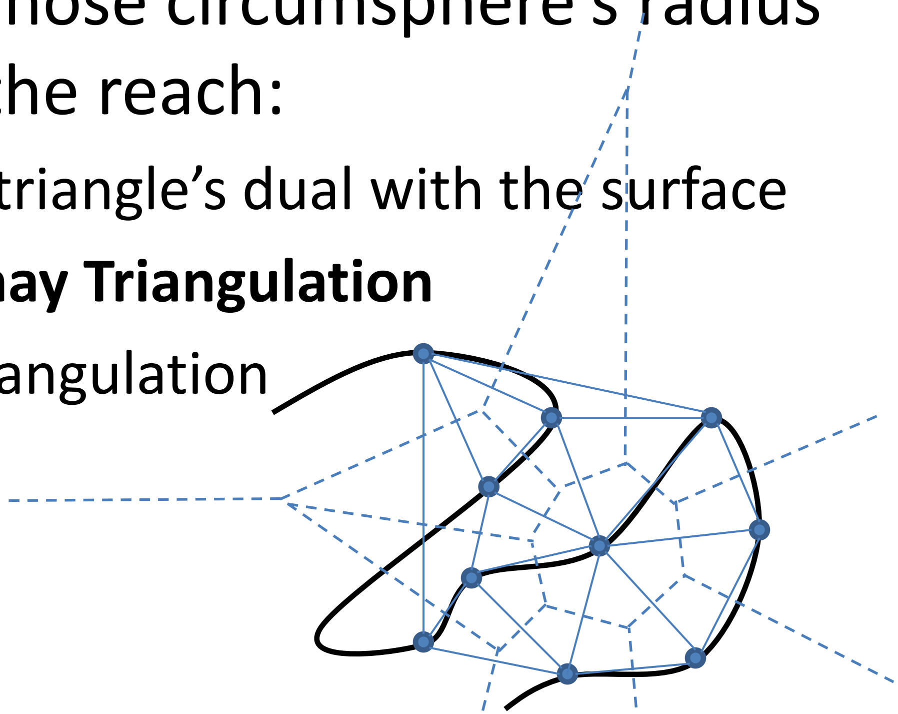
Algorithm:

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]

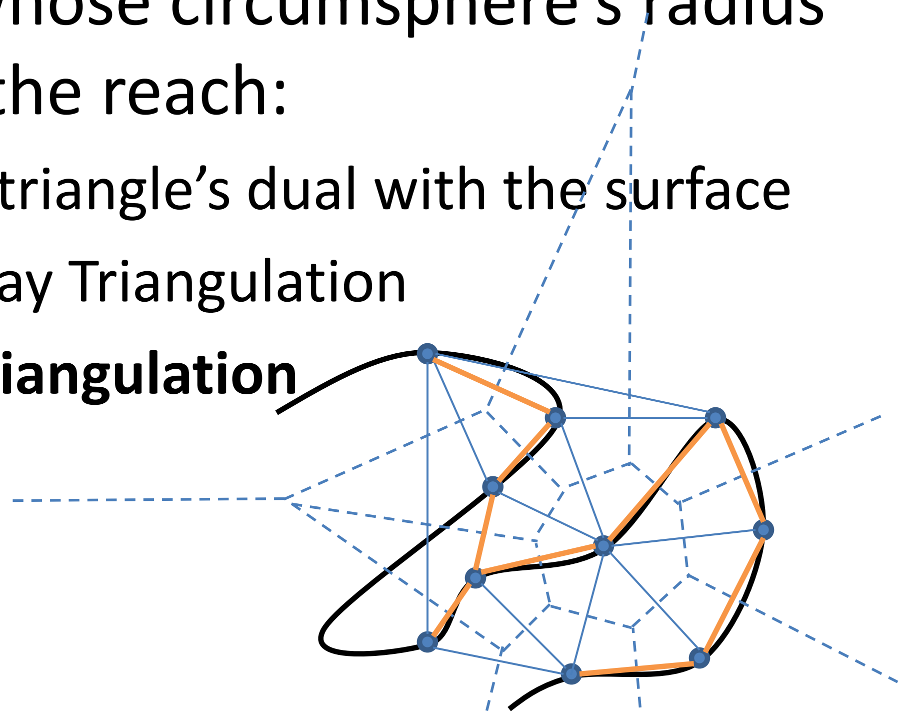
<u>Algorithm</u>:

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]

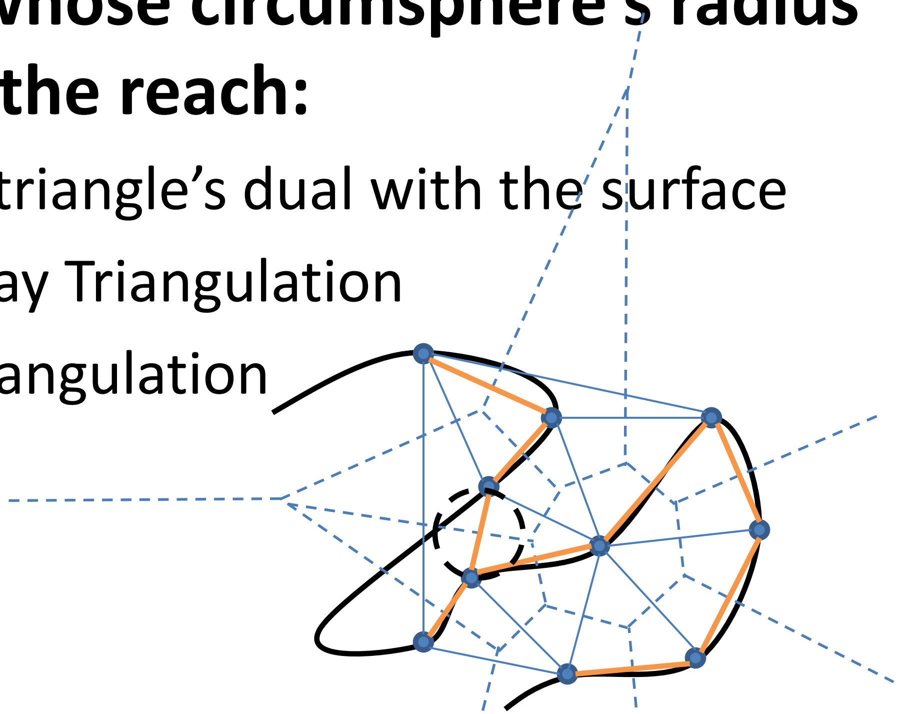
<u>Algorithm</u>:

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]

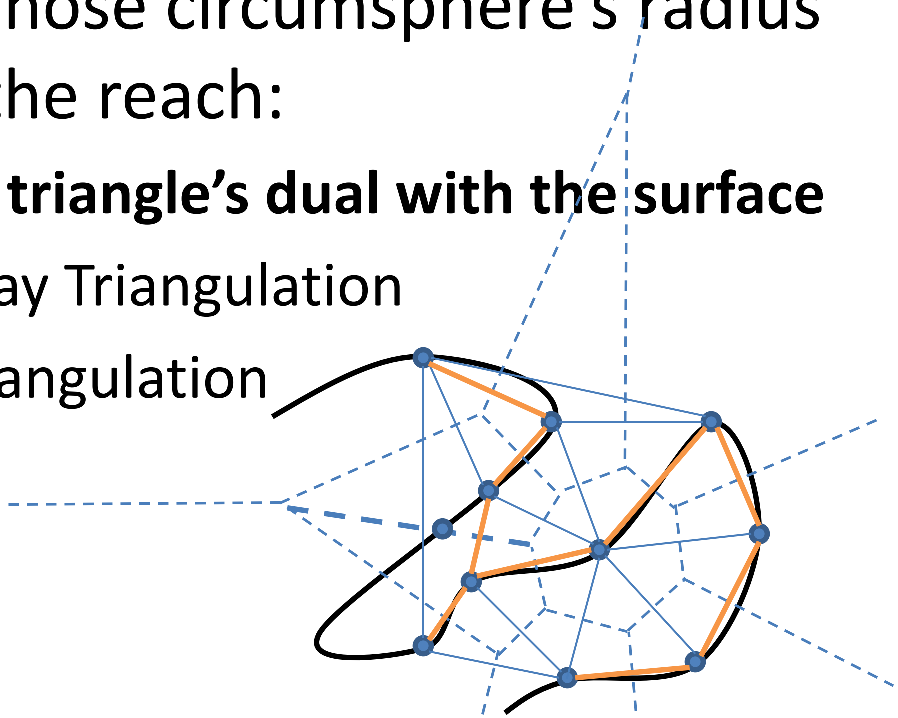
Algorithm:

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]

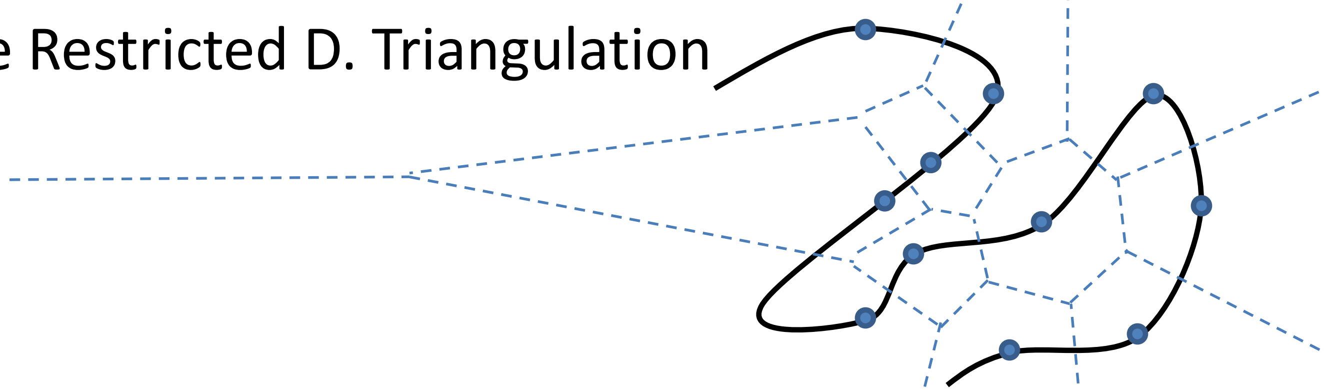
Algorithm:

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]

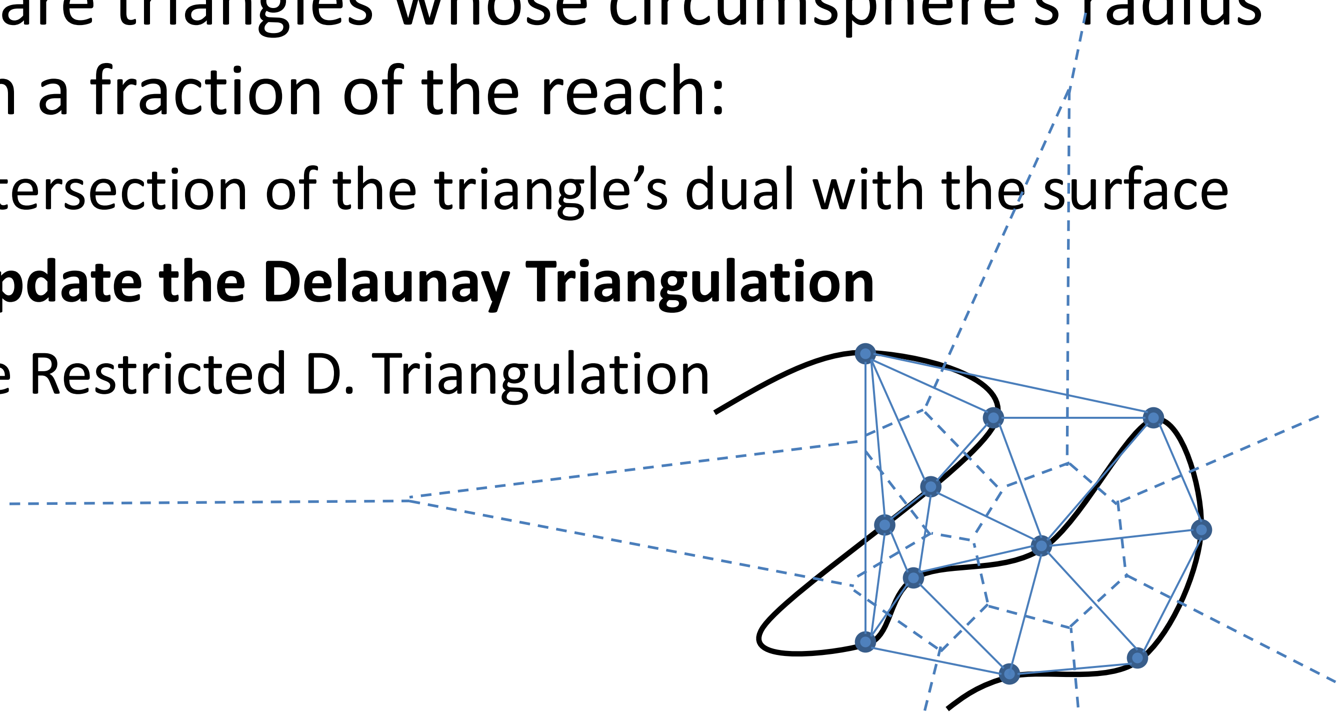
Algorithm:

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]

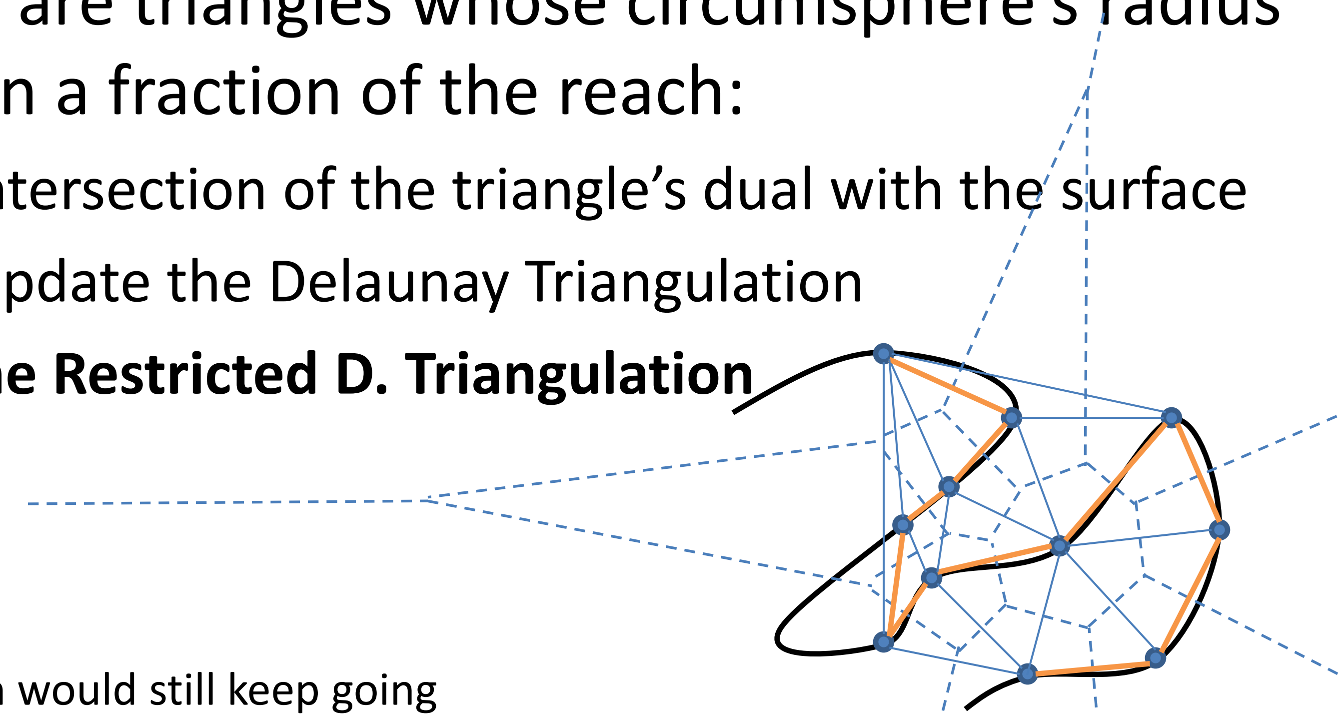
Algorithm:

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**

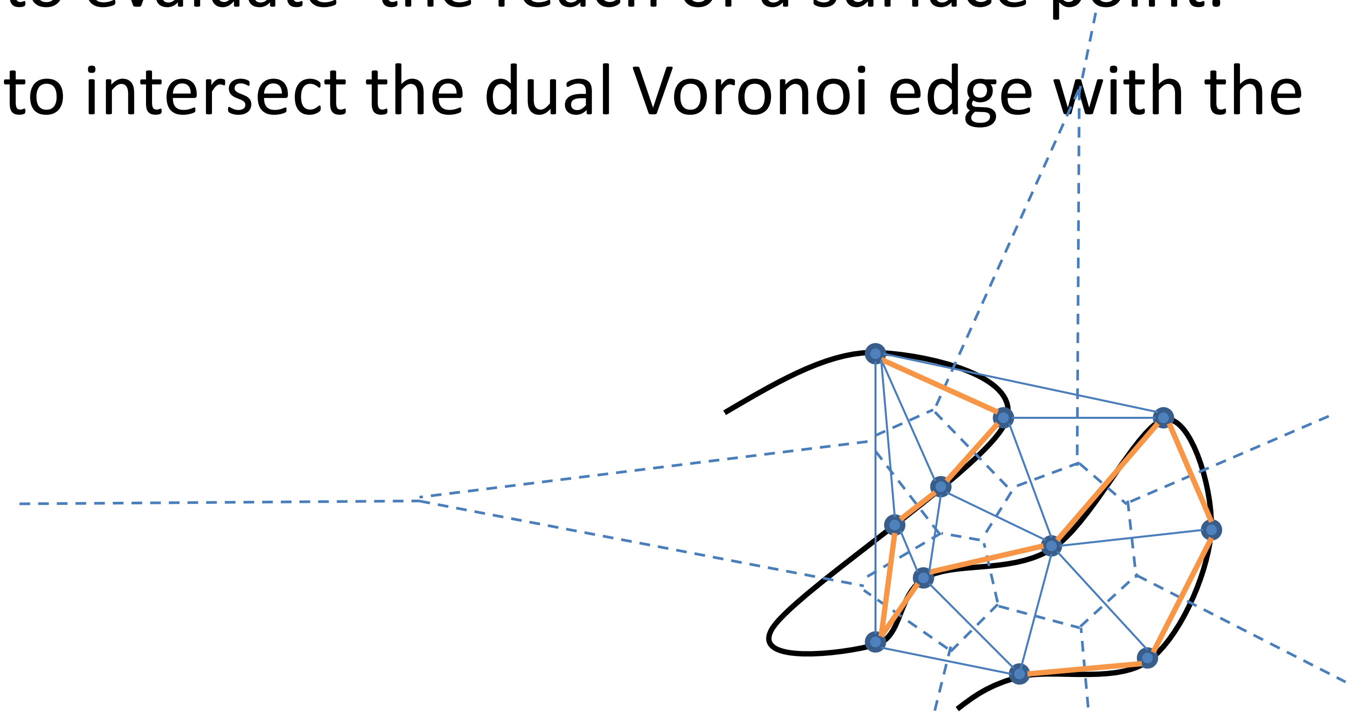


*Note that the algorithm would still keep going

Restricted Delaunay [Boissonnat & Oudot '05]

<u>Implementation Requirements</u>:

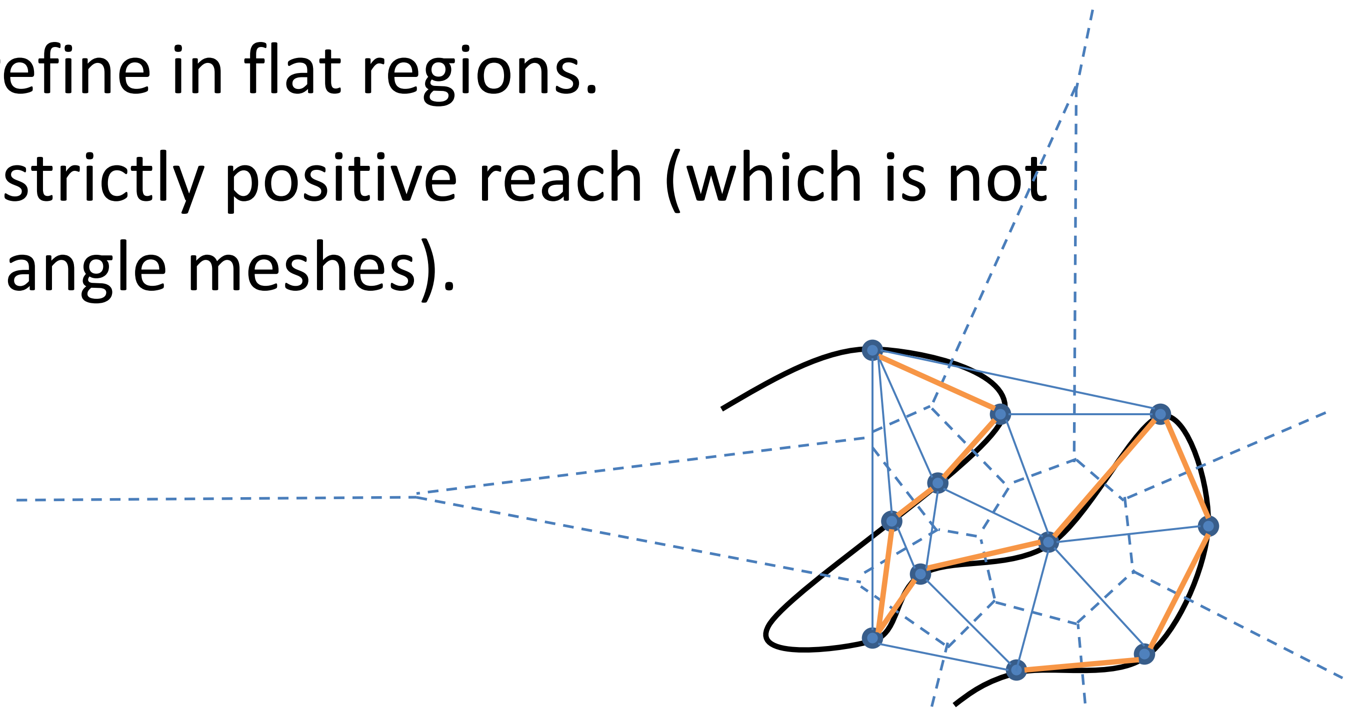
- A single computation of a (restricted) Delaunay Triangulation plus local updates.
- The ability to evaluate the reach of a surface point.
- The ability to intersect the dual Voronoi edge with the surface.



Restricted Delaunay [Boissonnat & Oudot '05]

Properties:

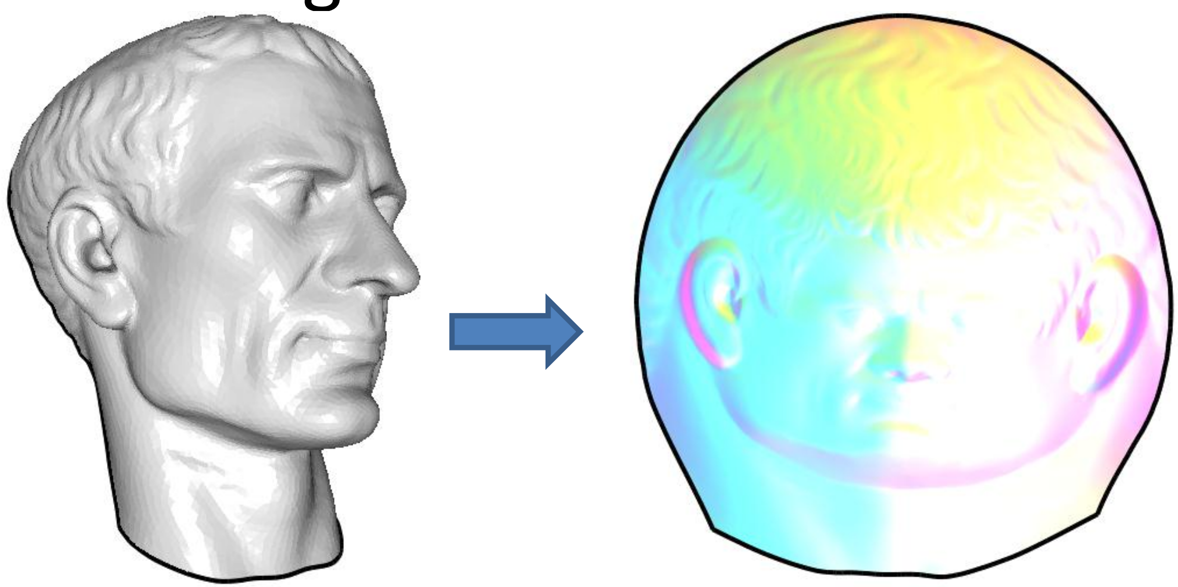
- With the appropriate scaling, the method returns a manifold, non-self-intersecting, triangulation with the same topology as *S*.
- May over-refine in flat regions.
- Requires a strictly positive reach (which is not satisfied triangle meshes).



Isotropic Remeshing [Alliez *et al*. '03]

<u>Observation</u>:

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.*

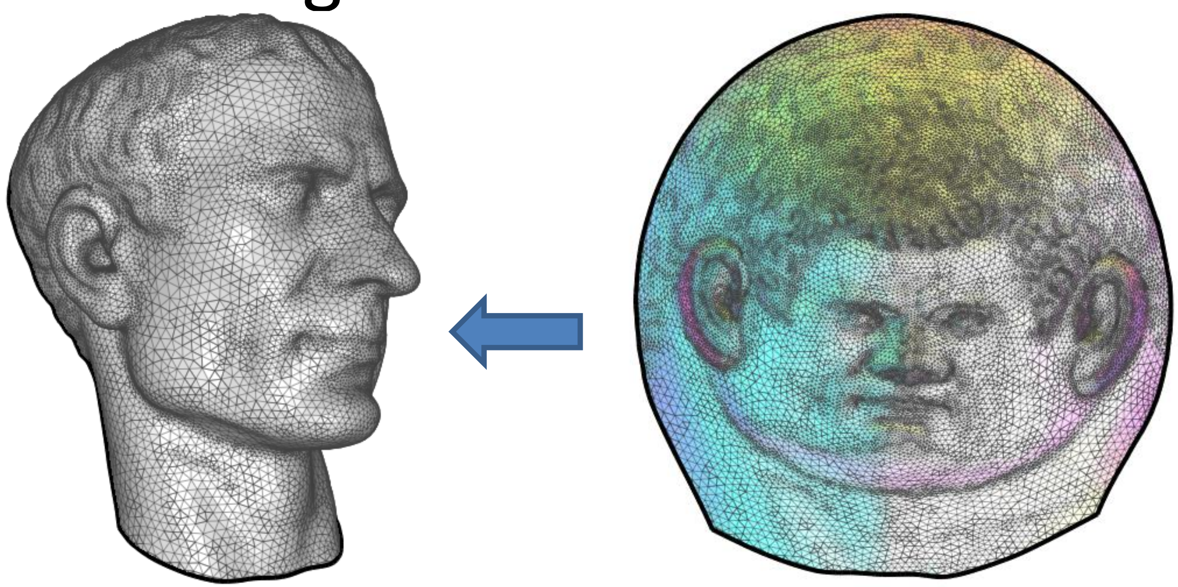


*May have intersecting triangles

Isotropic Remeshing [Alliez *et al*. ‘03]

<u>Observation</u>:

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.*

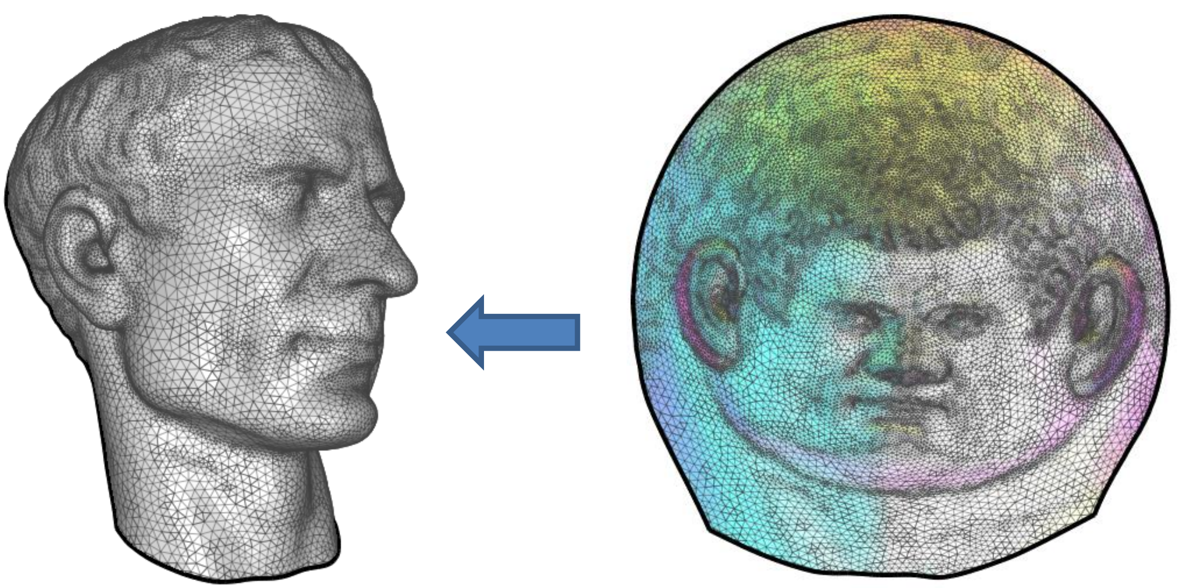


*May have intersecting triangles

Isotropic Remeshing [Alliez *et al*. ‘03]

<u>Questions</u>:

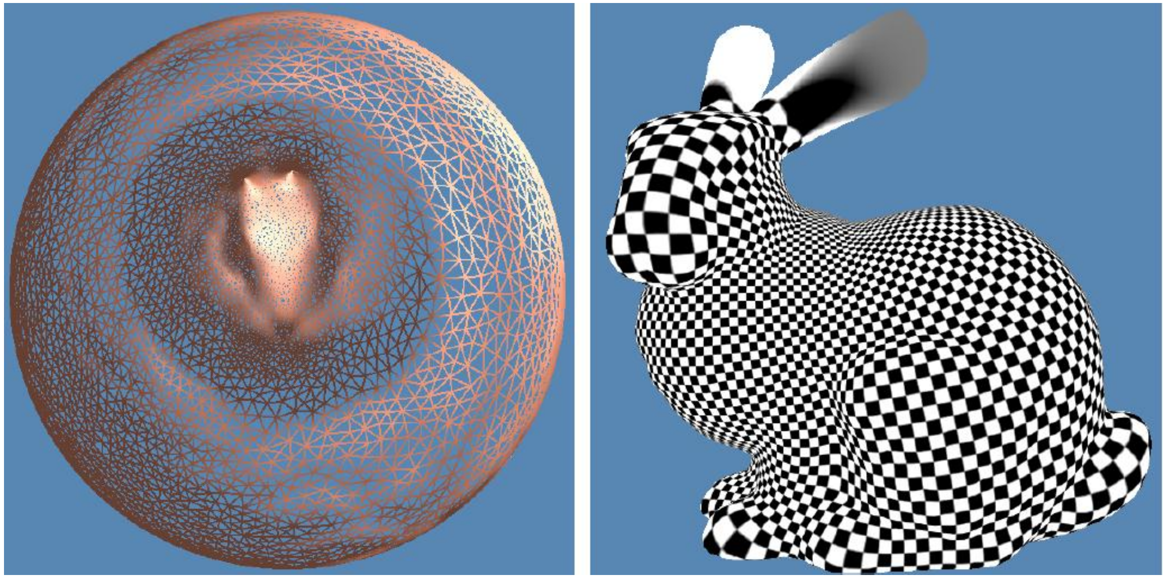
1. Which parameterization do we choose?
2. How do we triangulate the 2D domain?



Isotropic Remeshing [Alliez *et al*. ‘03]

1. <u>Which parameterization do we choose?</u>

Use a conformal parameterization.
The distortion is strictly due to scaling, so we can undue that by appropriately tesselating the 2D domain.



[*Global Conformal Surface Parameterization*, Gu and Yau]

Isotropic Remeshing [Alliez *et al*. ‘03]

2. <u>How do we triangulate the 2D domain?</u>

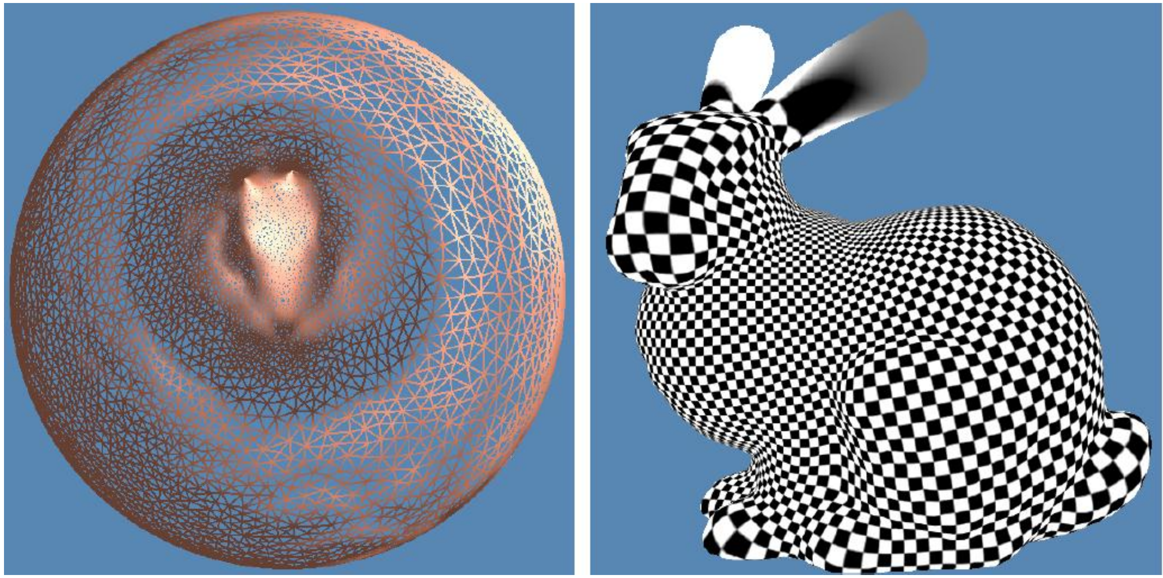
If we have a point sampling, we can compute the (constrained) Delaunay triangulation…

So how do we choose the point set?

Isotropic Remeshing [Alliez *et al*. ‘03]

<u>Goal</u>:

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



[*Global Conformal Surface Parameterization*, Gu and Yau]

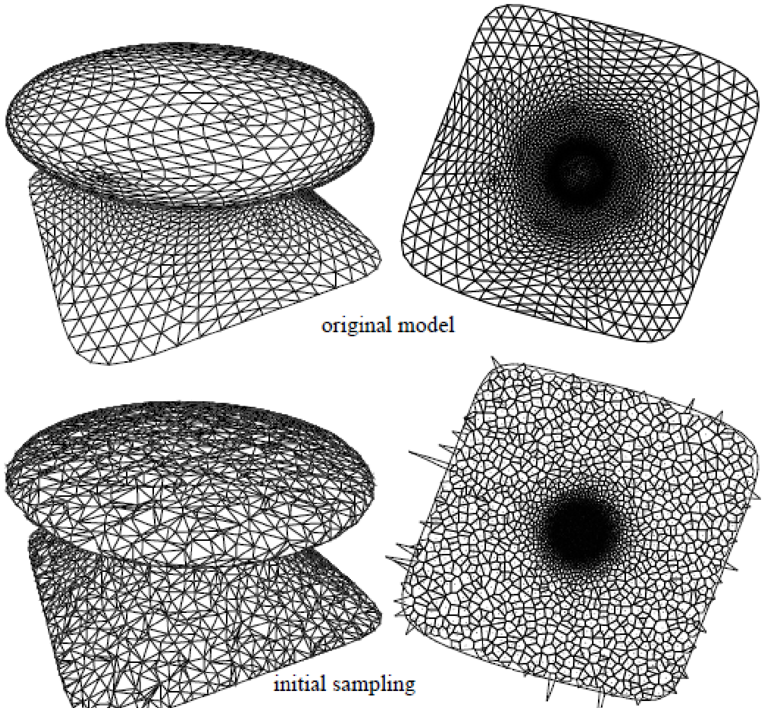
Isotropic Remeshing [Alliez *et al*. ‘03]

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 Surface Remeshing*, Alliez et al.]

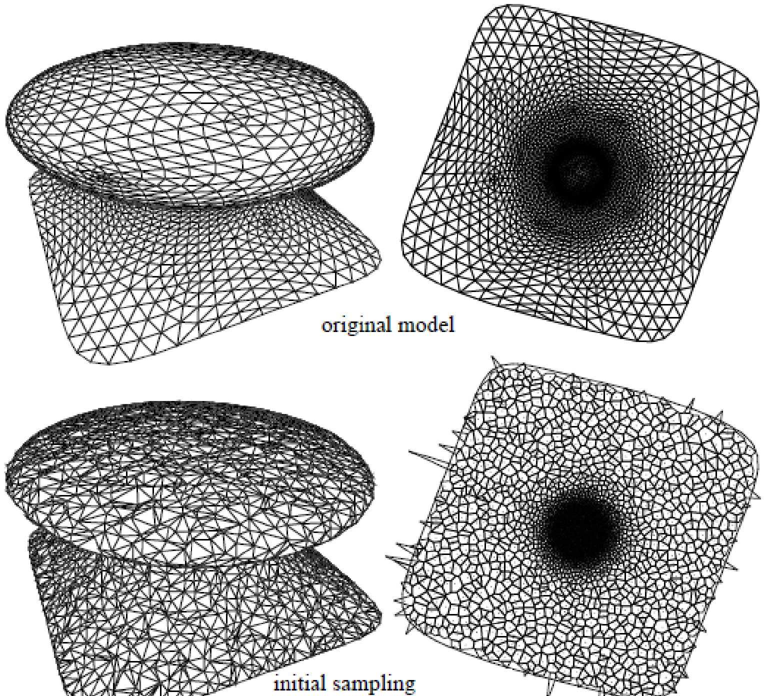
Isotropic Remeshing [Alliez *et al*. '03]

<u>Goal</u>:

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

<u>Approach</u>:

Use the distortion to sample the 2D domain adaptively.



[*Isotropic Surface Remeshing*, Alliez et al.]

<u>Challenge</u>:

Just because the points are randomly distributed, that doesn't make them uniform.

Isotropic Remeshing [Alliez *et al*. '03]

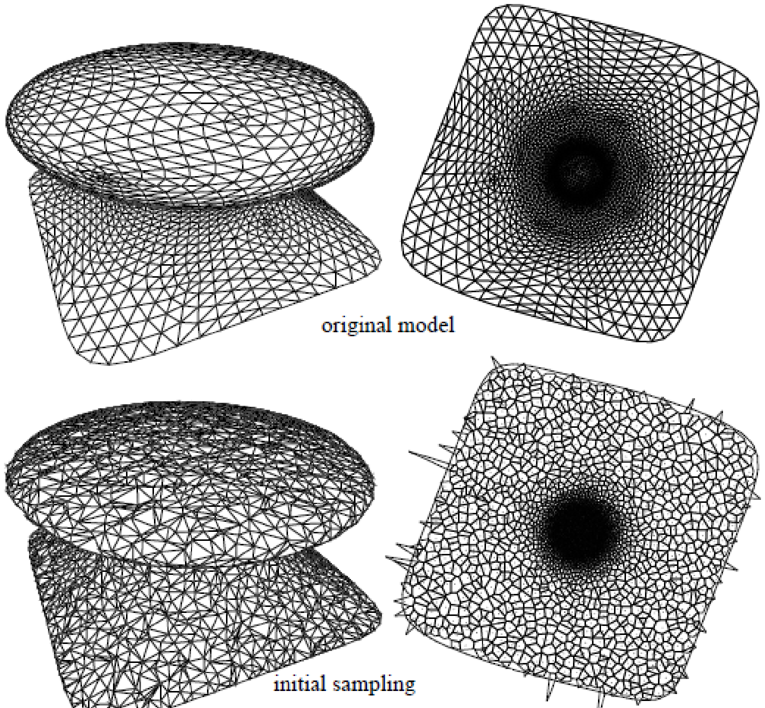
<u>Update/Solve for well-distributed positions</u>.

Given a density function ρ, solve for a point set *P* and a partition of the 2D domain:

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

that minimizes:

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



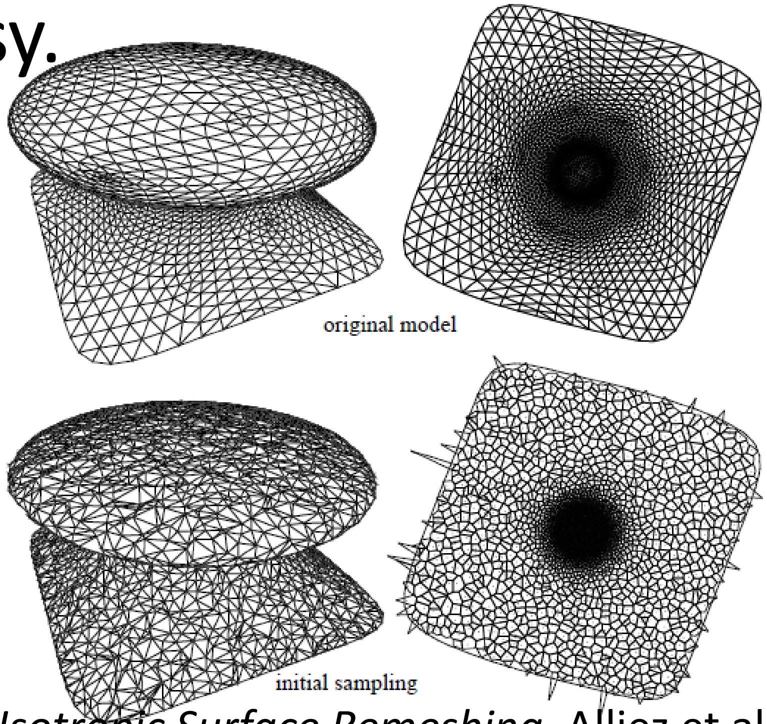
[*Isotropic Surface Remeshing*, Alliez et al.]

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.



[*Isotropic Surface Remeshing*, Alliez et al.]

Isotropic Remeshing [Alliez *et al*. '03]

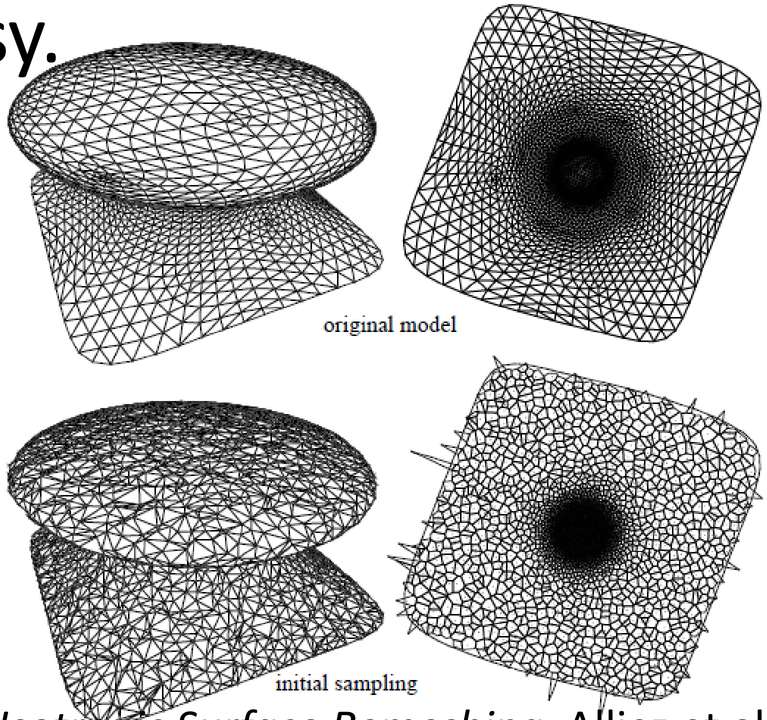
$$E(P,R)=\sum_{p\in P}\int_{x\in R_p}\rho(x)\|x-p\|^2\,dx$$

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 Surface Remeshing*, Alliez et al.]

Isotropic Remeshing [Alliez *et al*. '03]

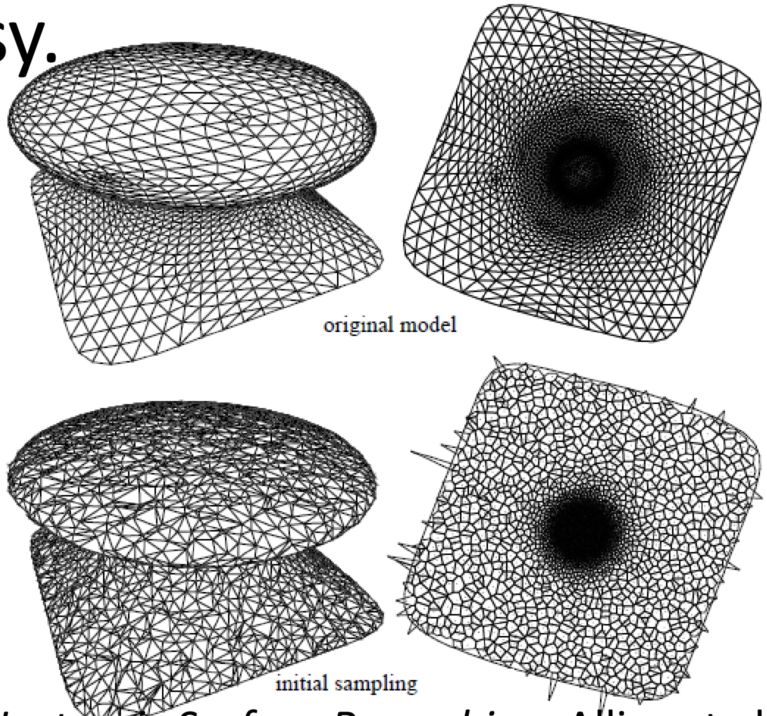
$$E(P,R)=\sum_{p\in P}\int_{x\in R_p}\rho(x)\|x-p\|^2\,dx$$

<u>Lloyd Relaxation</u>:

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

<u>Observations</u>:

- 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 ρ-weighted centers of R_p.



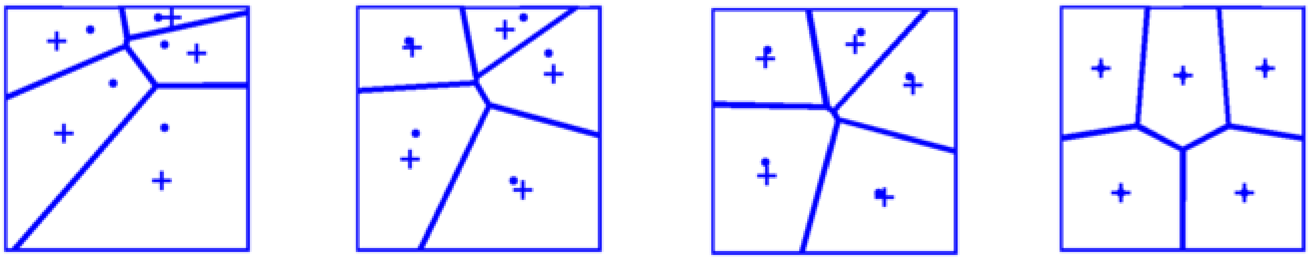
[*Isotropic Surface Remeshing*, Alliez et al.]

Isotropic Remeshing [Alliez *et al*. '03]

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

Implementation:

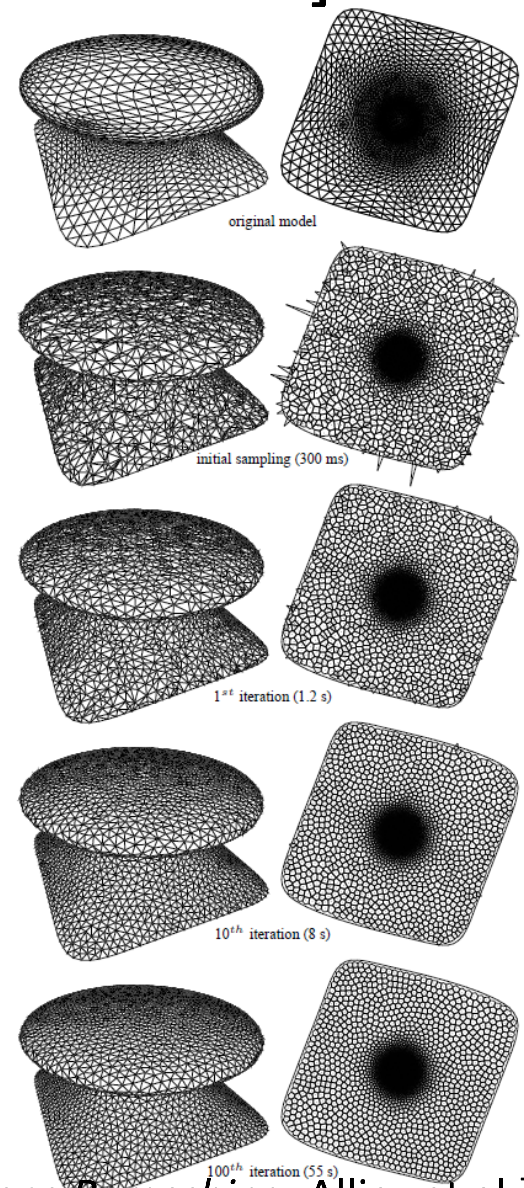
Iteratively alternate between computing the Voronoi regions of the points in *P*, and computing the centers of the regions.



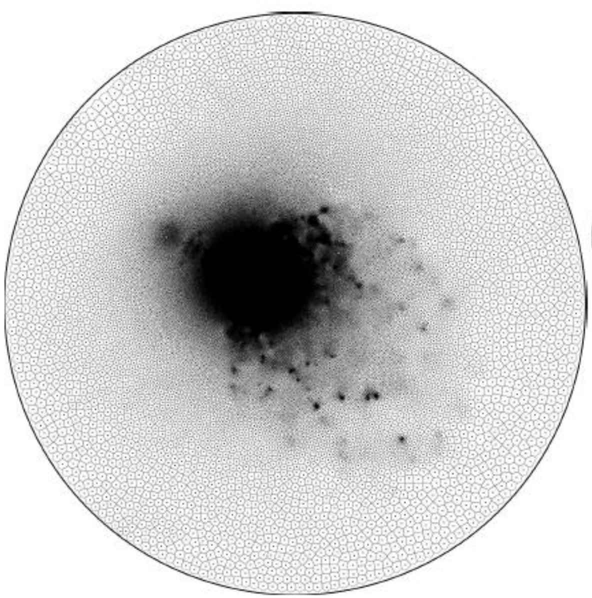
[*Wikipedia*]

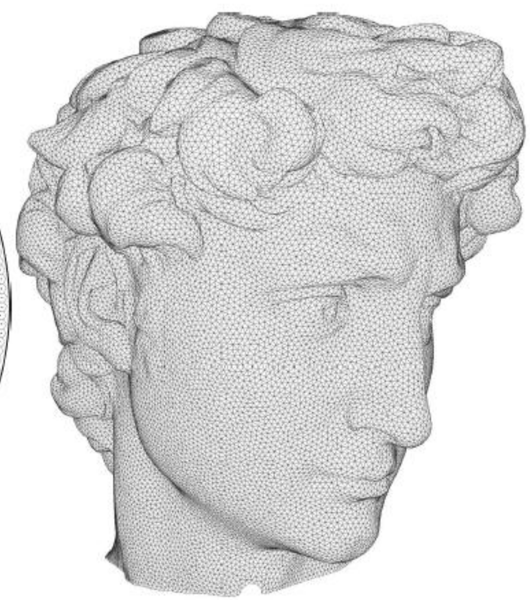
Isotropic Remeshing [Alliez *et al*. '03]

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



Applying this using the distortion weights from the conformal map, we get an isotropic tessellation.



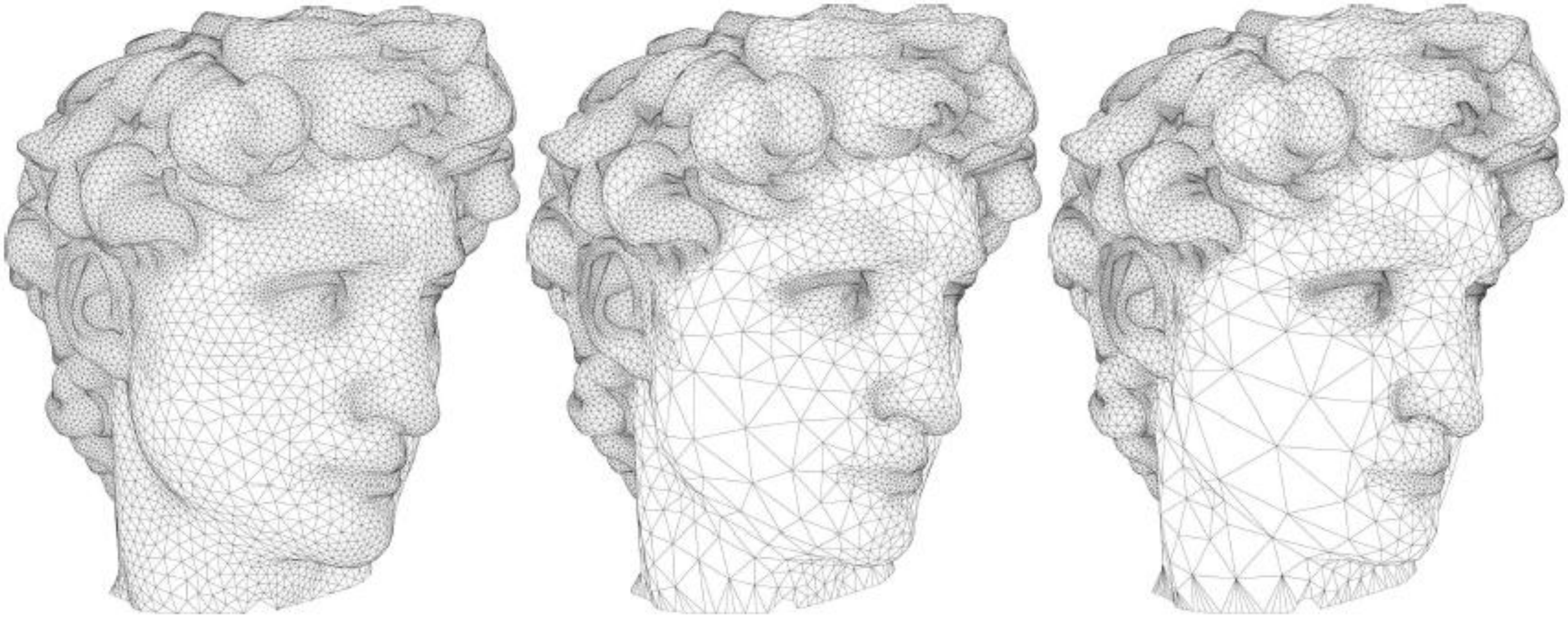


[*Isotropic Surface Remeshing*, Alliez et al.]

Isotropic Remeshing [Alliez *et al*. '03]

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

Adapting the weights to take into account, curvature, you can get curvature-adapted tessellations.

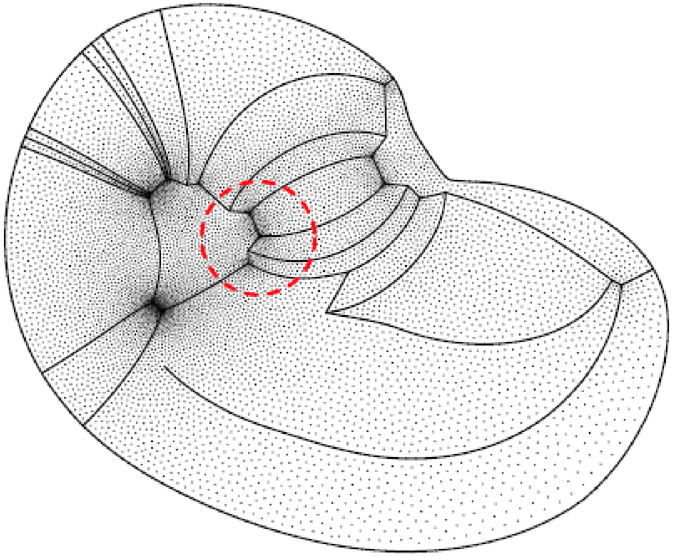


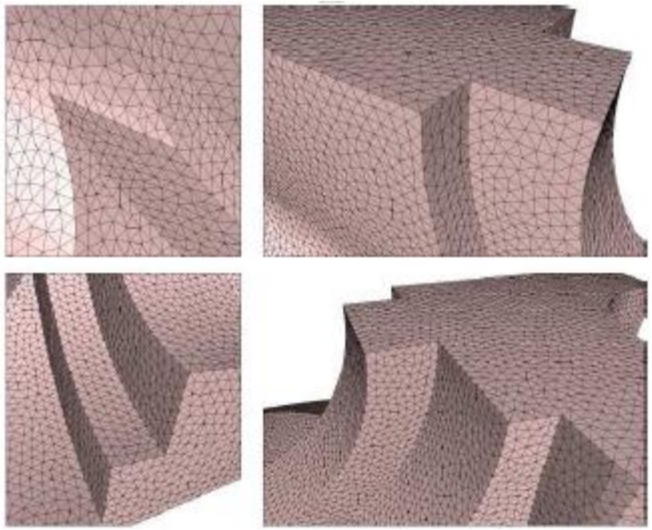
[*Isotropic Surface Remeshing*, Alliez et al.]

Isotropic Remeshing [Alliez *et al*. '03]

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

Constraining the Delaunay Triangulation, you can preserve edges in the triangulation.





[*Isotropic Surface Remeshing*, Alliez et al.]