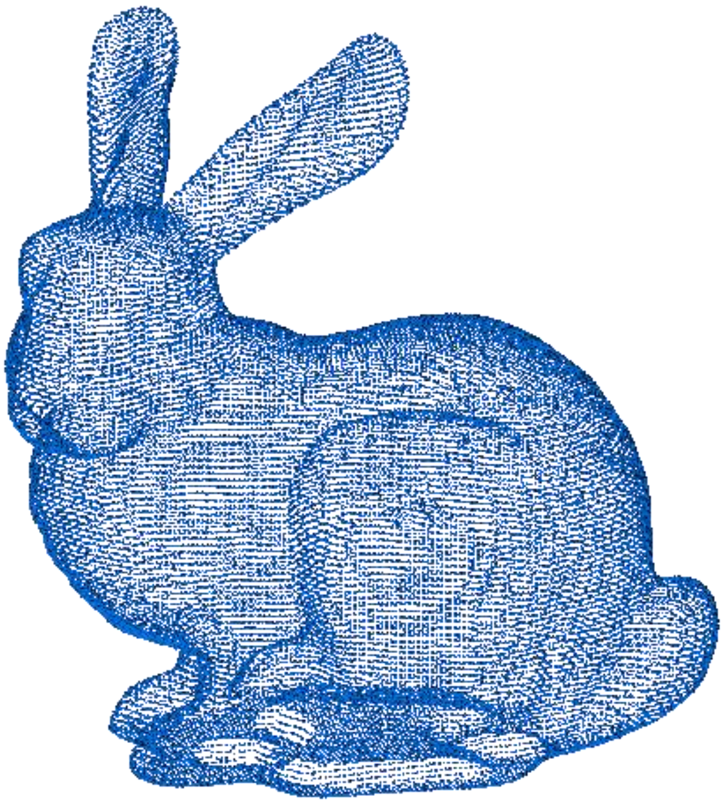
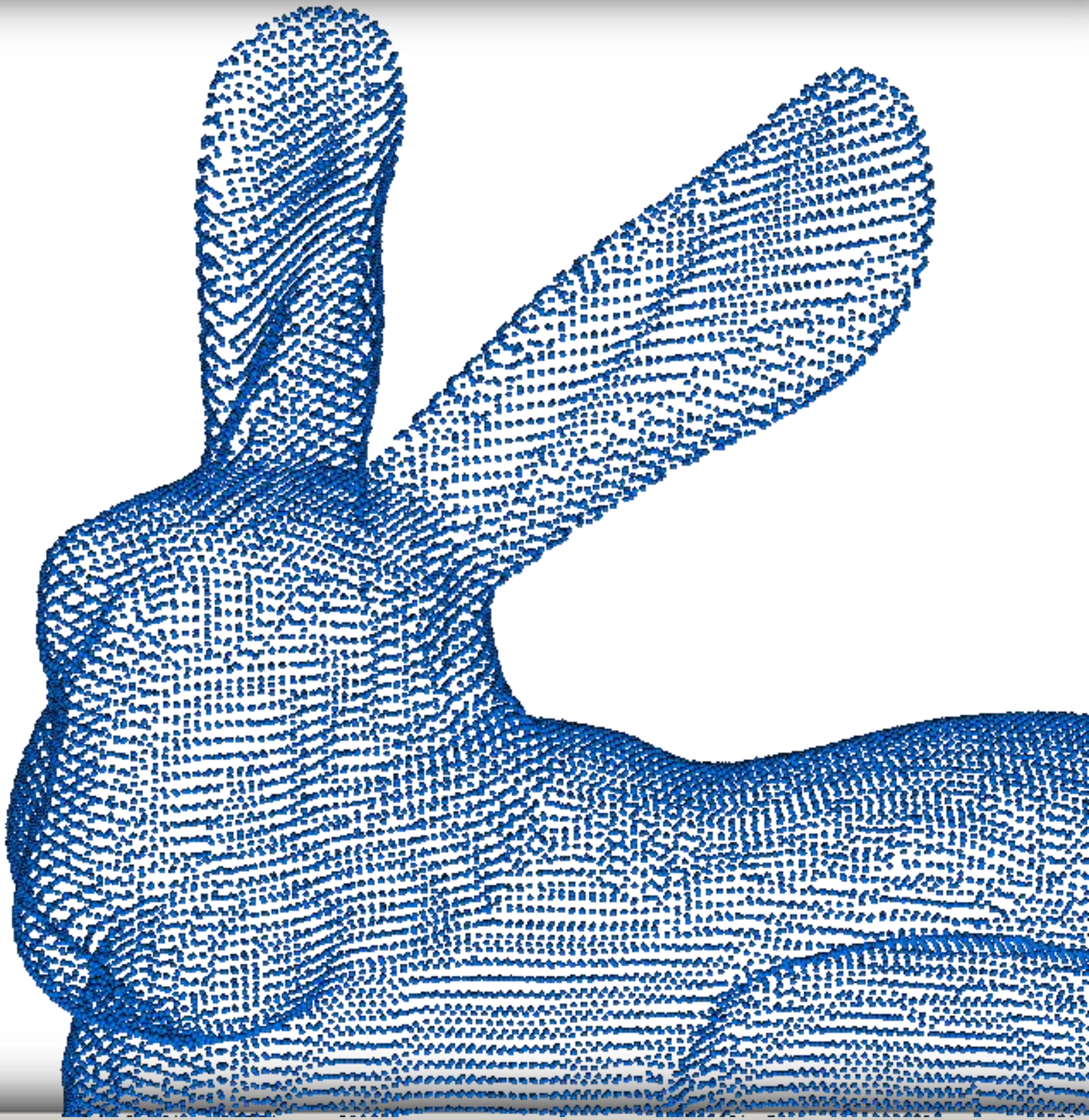
Lecture 1: The Low-Level

Reconstruction, Meshing and Sampling

Bruno Lévy

Introduction

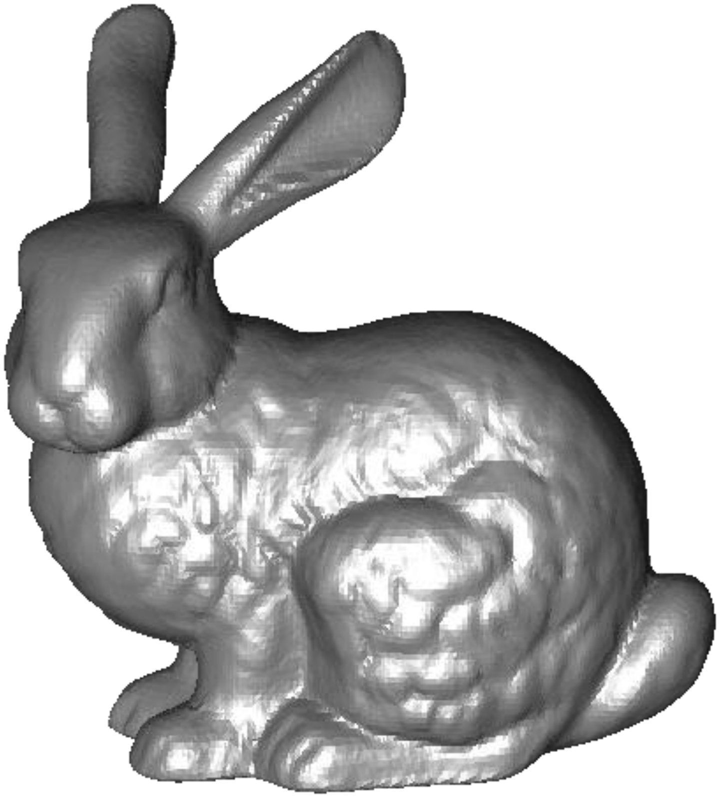


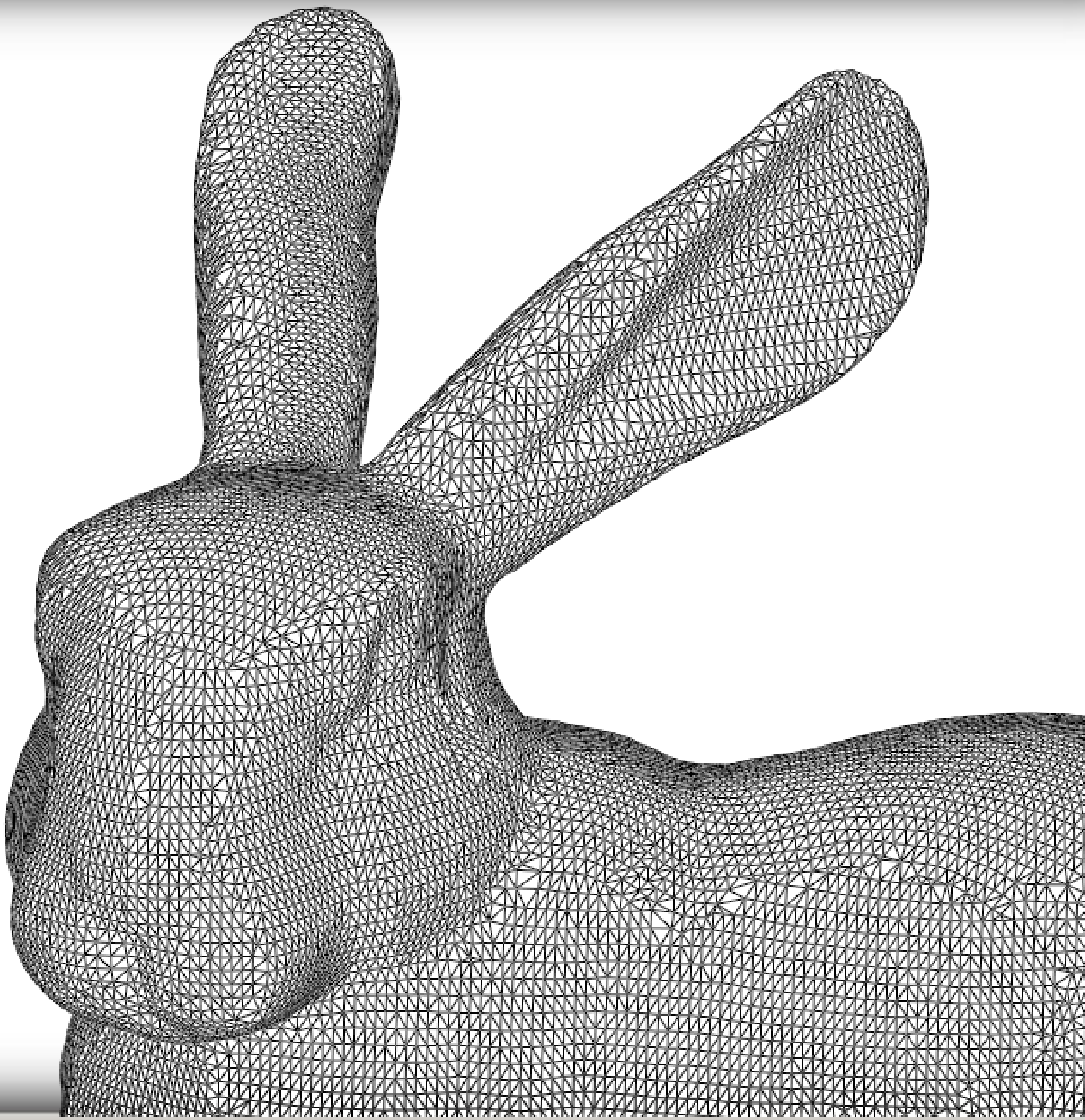


Output of the scanner
(after merging)

Introduction

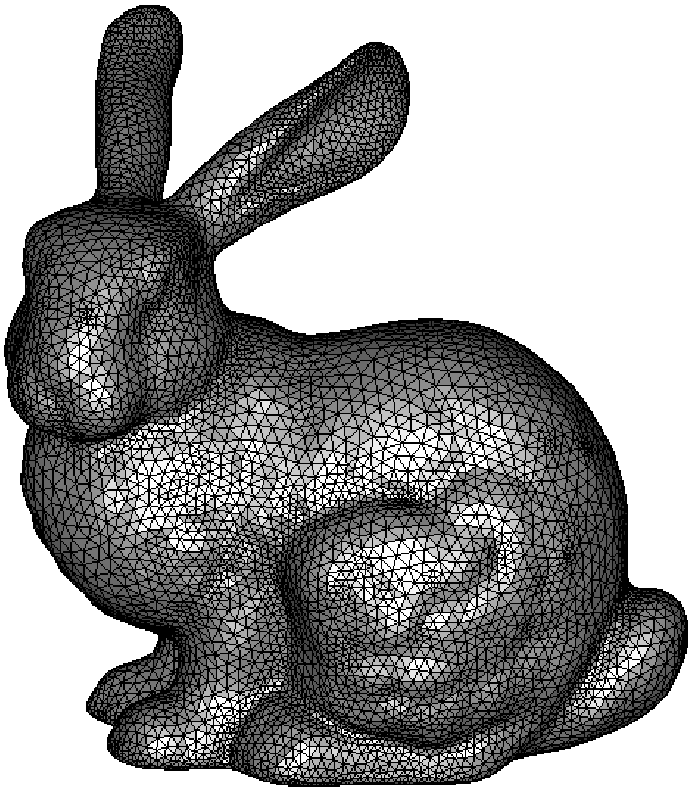
SIGGRAPH ASIA 2011 HONG KONG

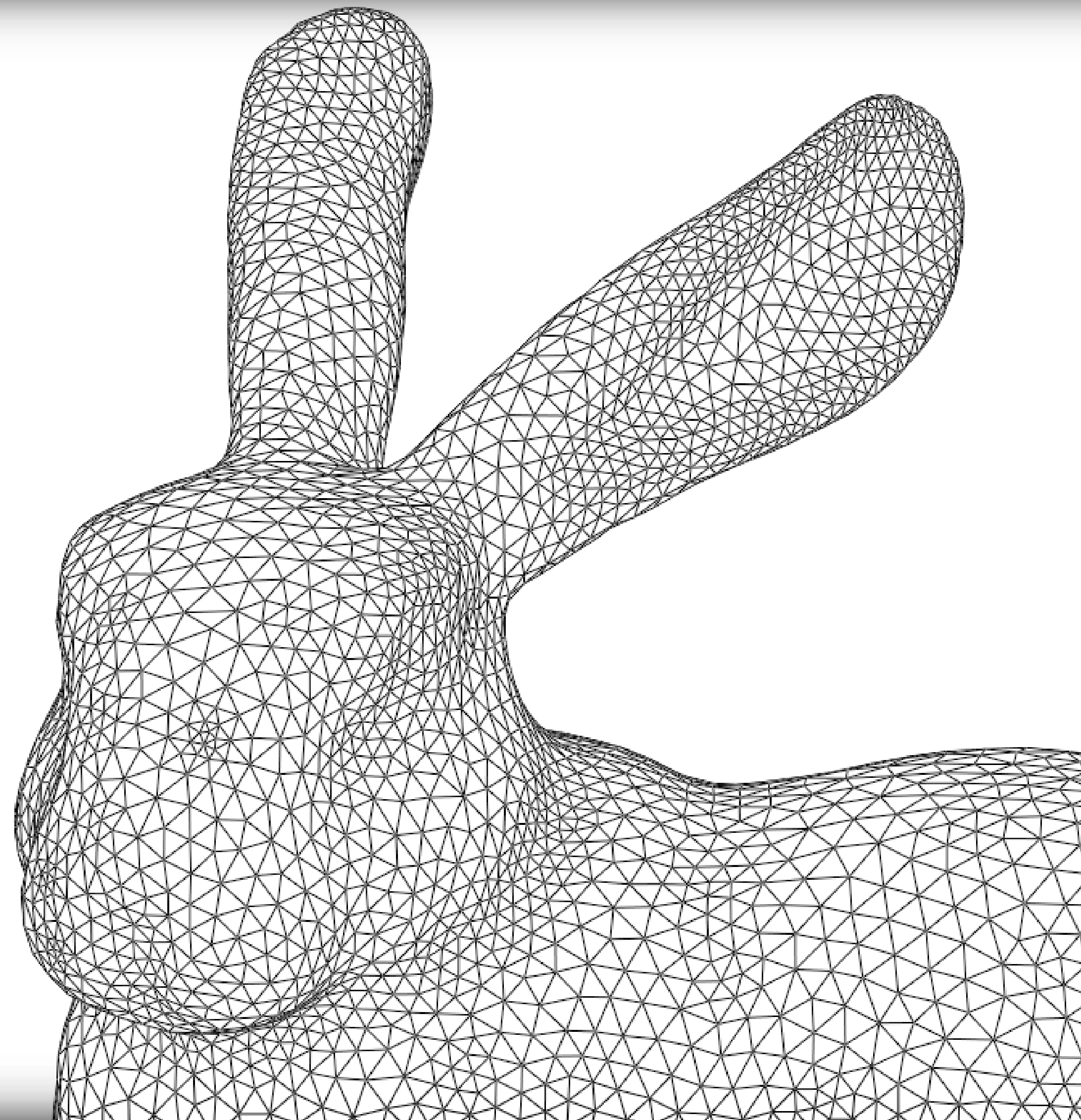




Q1: how to
connect the points ?

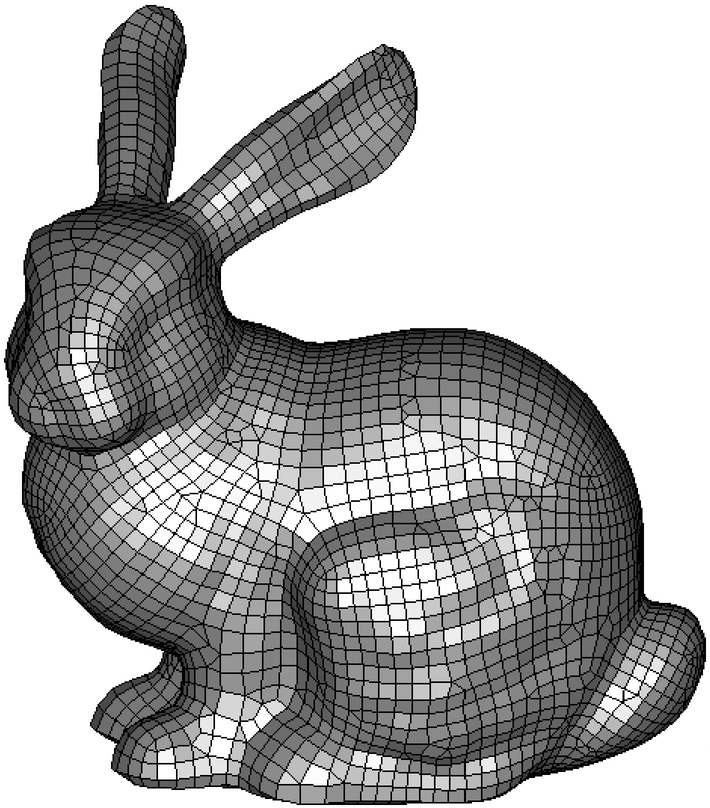
Introduction

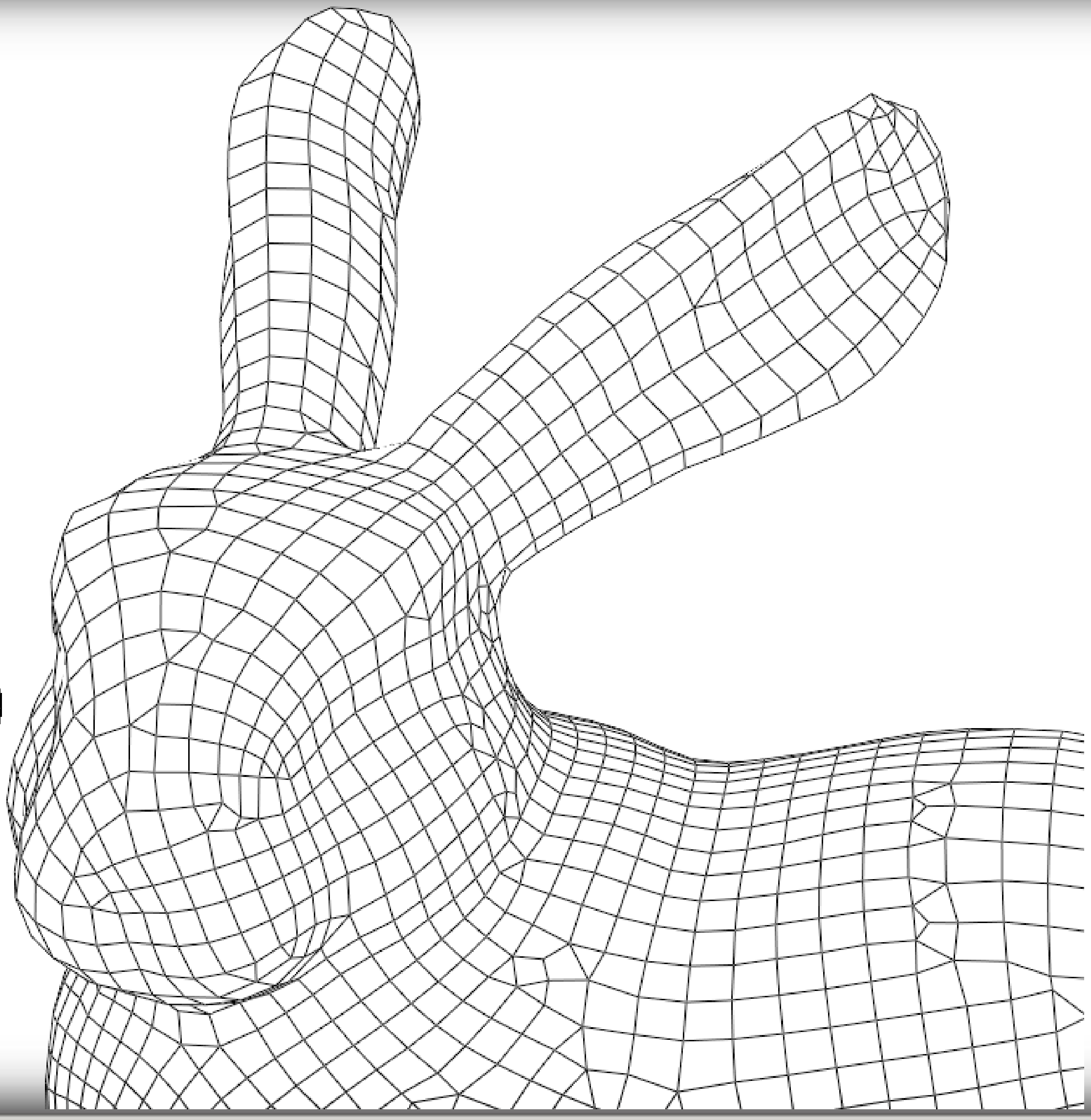




Q2: how to
improve the mesh ?

Introduction





Q2: how to
improve the mesh ?

Overview

- **1. Reconstruction :** *How to connect the points*
  - **Voronoi and Delaunay**
  - **The crust**

- **2. Re-meshing :** *How to improve the mesh*
  - **Centroidal Voronoi Tesselation**
  - **Restricted CVT**
  - **Lp CVT**

Overview

- **1. Reconstruction**

  Not covered here:

  Volumetric methods, model repair

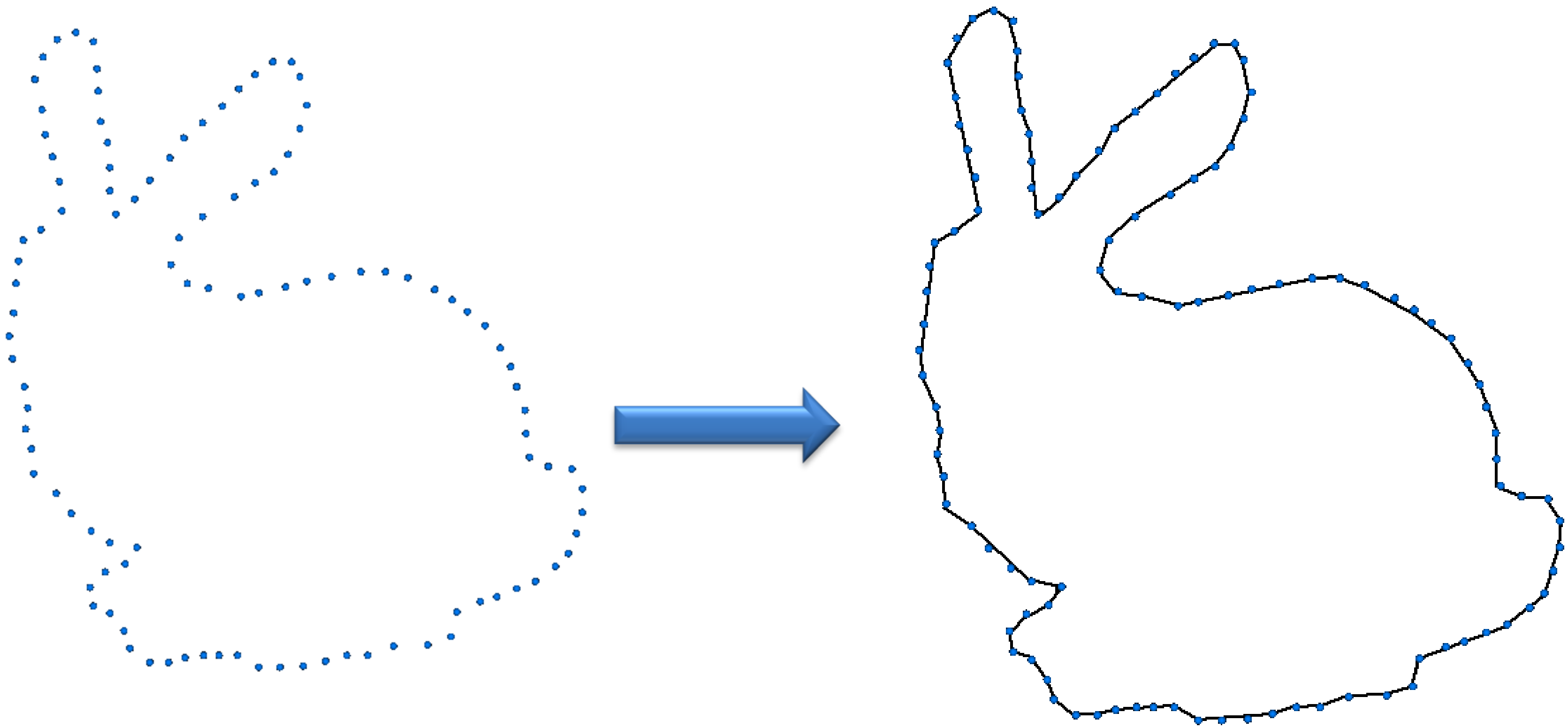
  See "Polygon Mesh Processing", Chapter 8

- **2. Re-meshing**

  More on this Chapter 6

1. Reconstruction

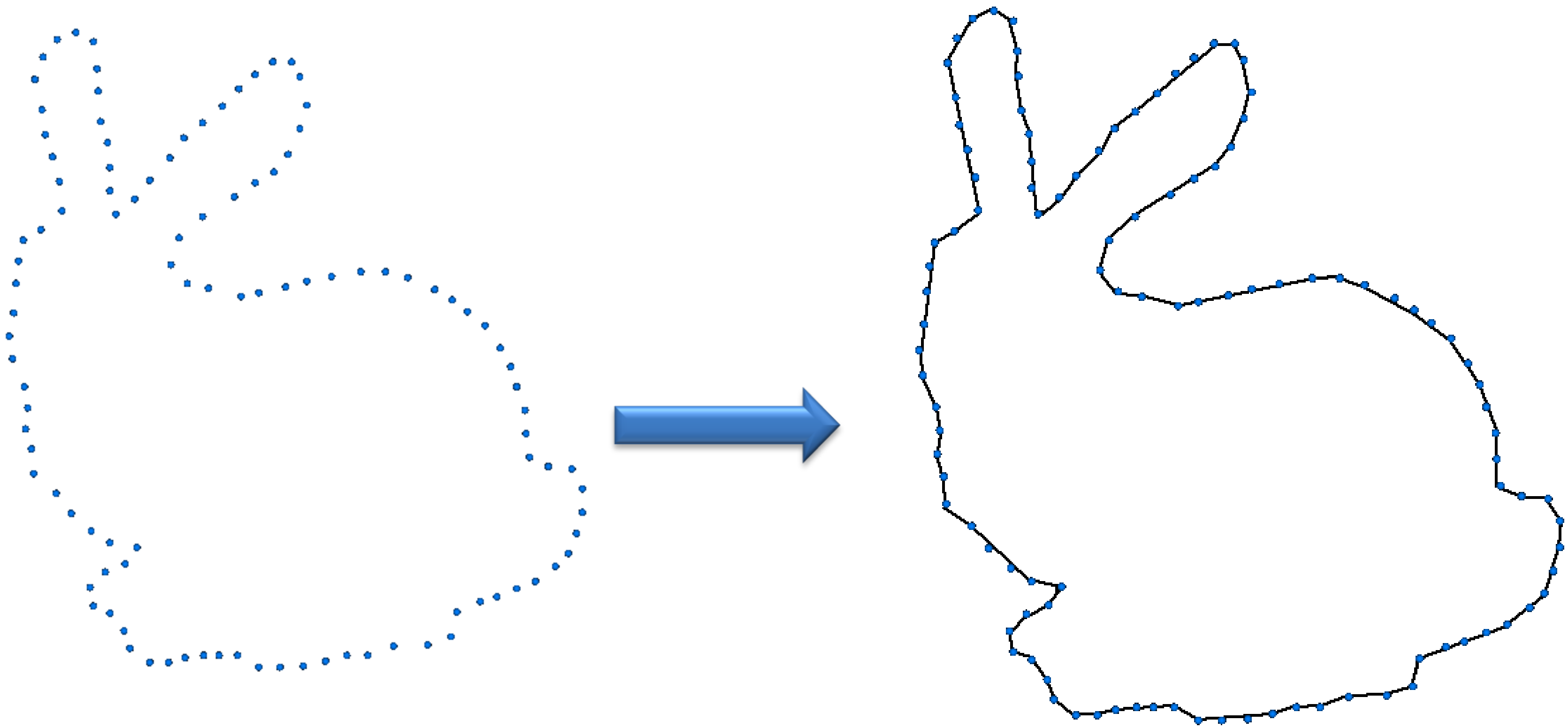
How to connect the points ?



1. Reconstruction

How to connect the points ?

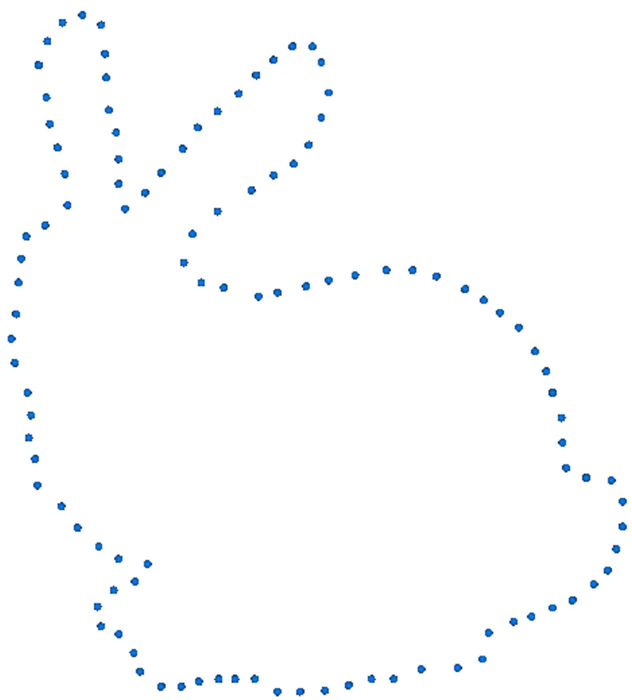
Intuitively: based on proximity



1. Reconstruction

Capturing all the "proximities" in a set of points

Voronoi diagram



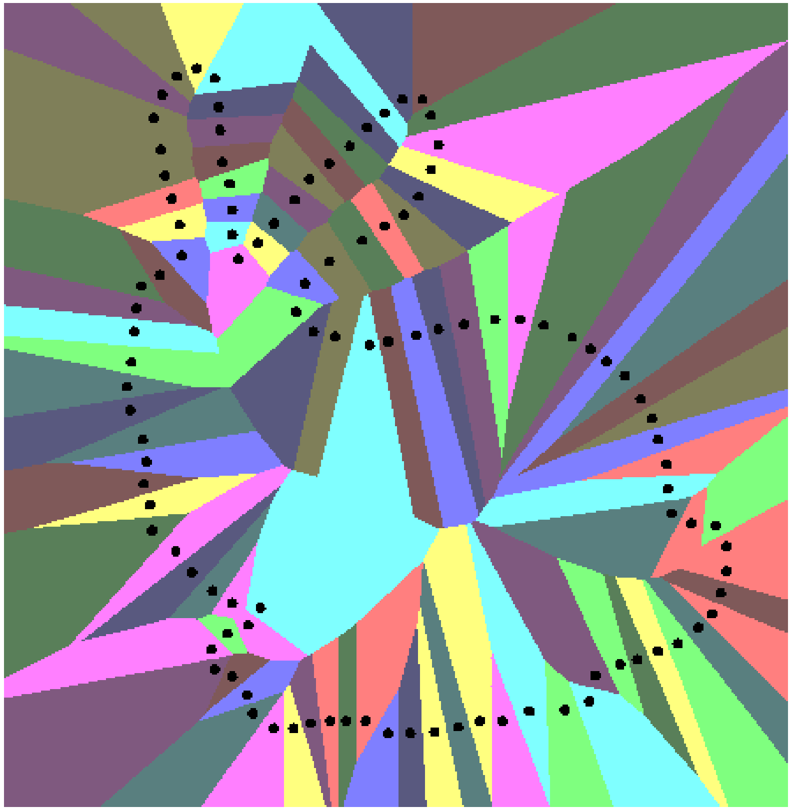
$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ set of points

$\mathbf{x}_i = (x_i, y_i)$ one of the points

1. Reconstruction

Capturing all the "proximities" in a set of points

Voronoi diagram



$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ set of points

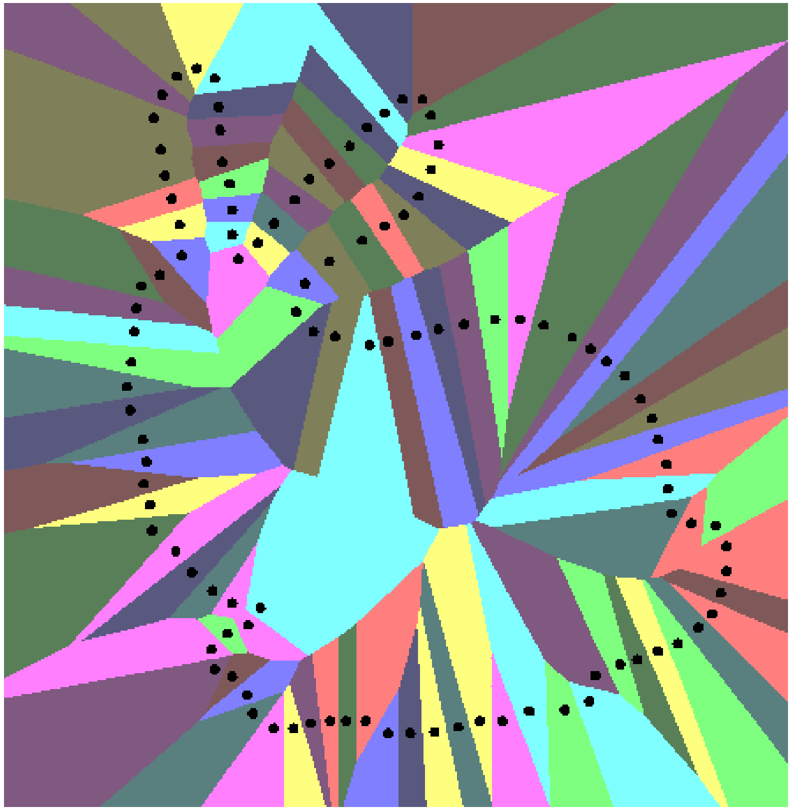
$\mathbf{x}_i = (x_i, y_i)$ one of the points

$Vor(i) = \{ \mathbf{x} \,/\, d(\mathbf{x}, \mathbf{x}_i) \leqslant d(\mathbf{x}, \mathbf{x}_j) \} \; \forall \; j$

1. Reconstruction

Capturing all the "proximities" in a set of points

Voronoi diagram



$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ set of points

$\mathbf{x}_i = (x_i, y_i)$ one of the points

$\mathrm{Vor}(i) = \{ \mathbf{x} \,/\, d(\mathbf{x},\mathbf{x}_i) \leqslant d(\mathbf{x},\mathbf{x}_j) \} \ \forall\ j$

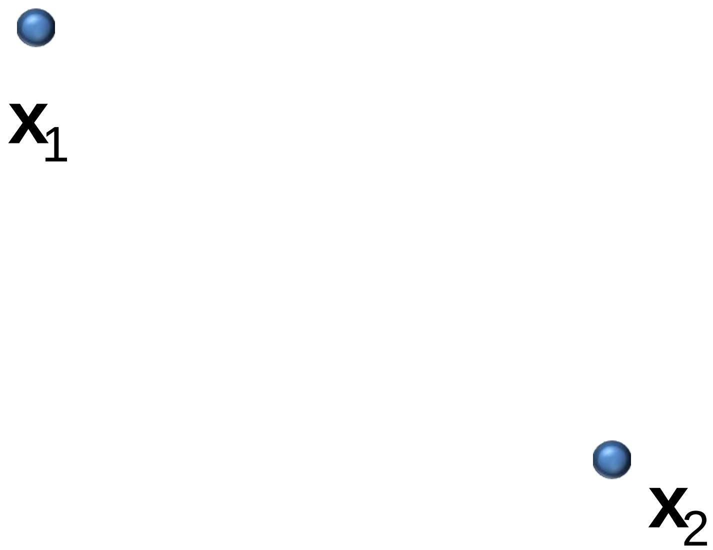
Euclidean distance

1. Reconstruction

Voronoi diagram

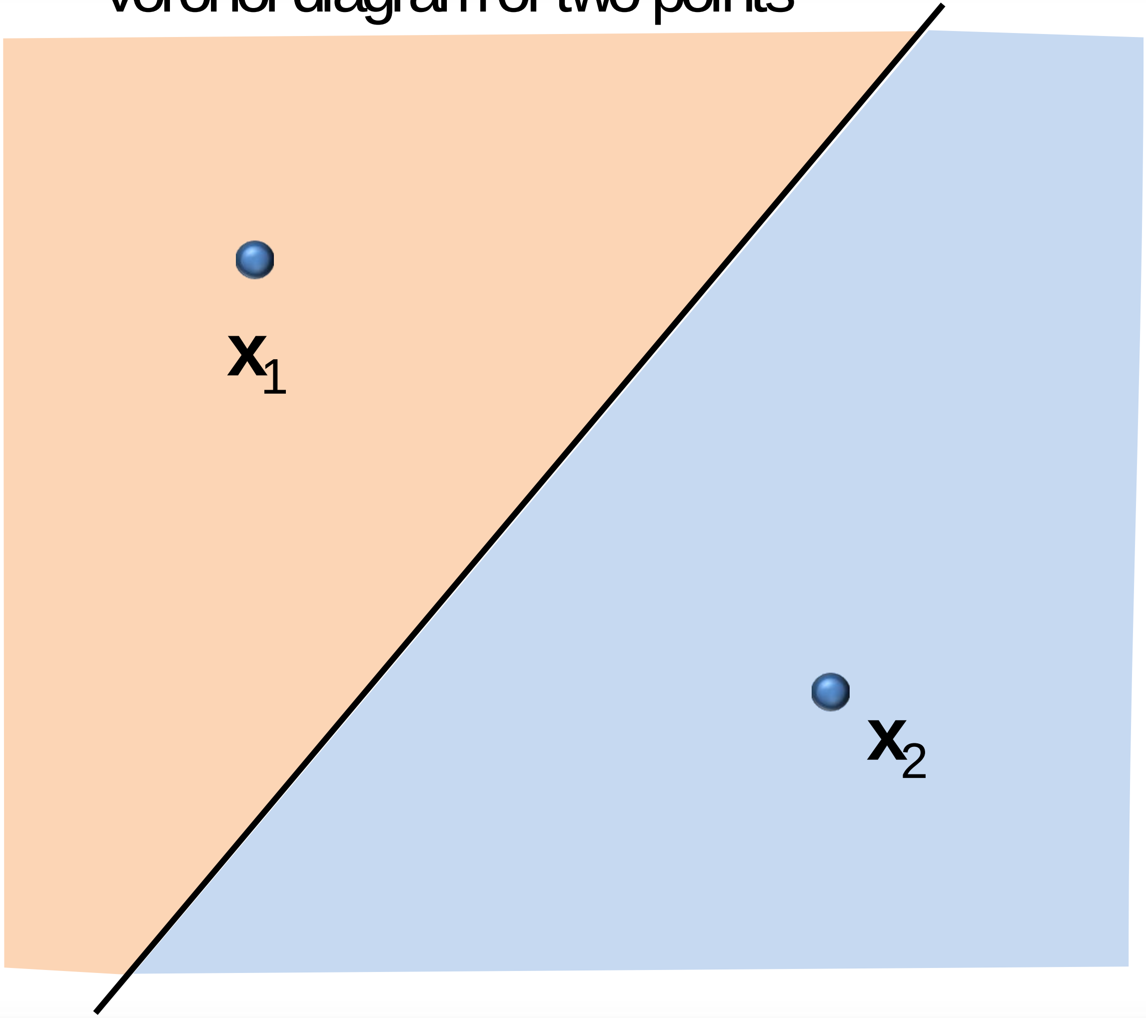
1. Reconstruction

Voronoi diagram of two points



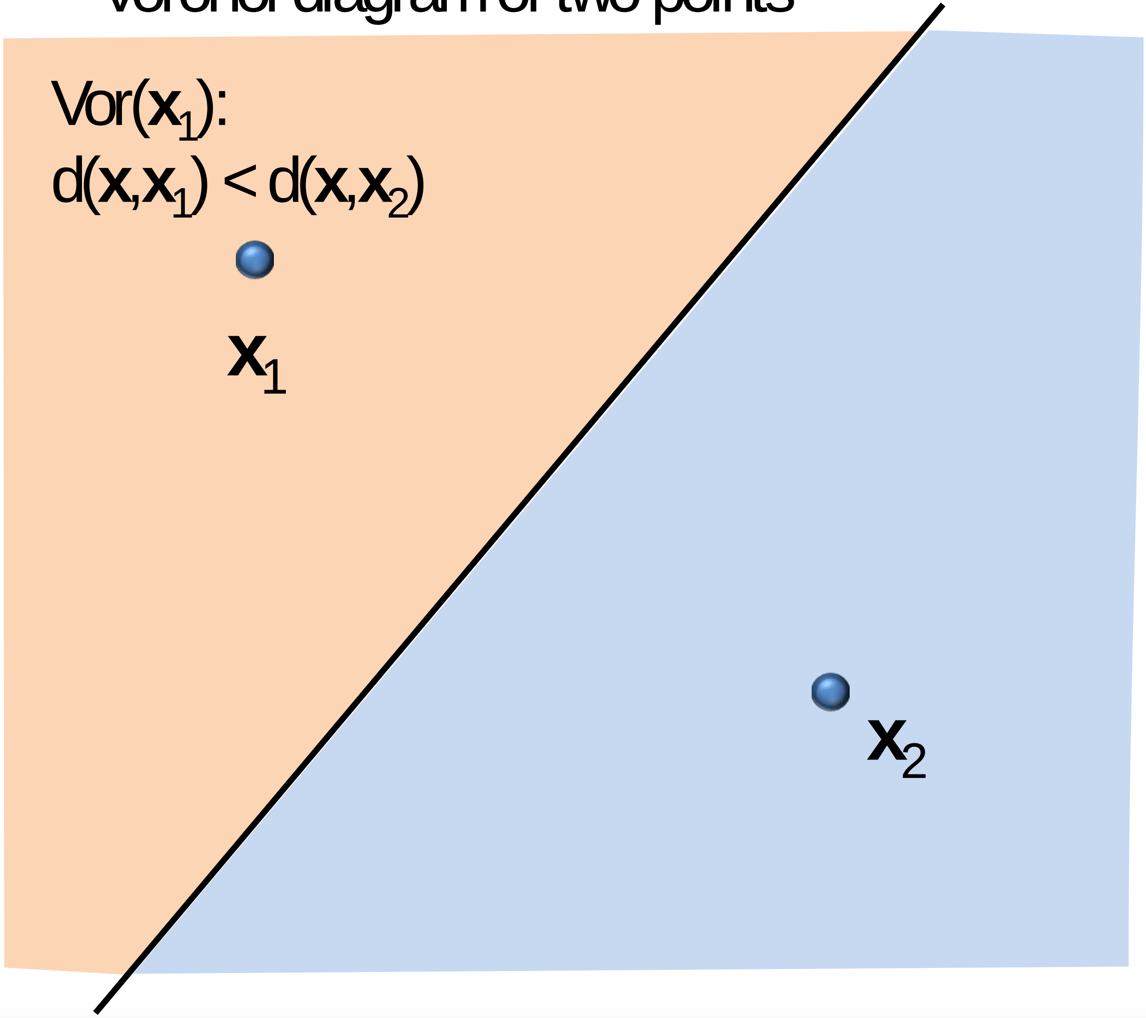
1. Reconstruction

Voronoi diagram of two points



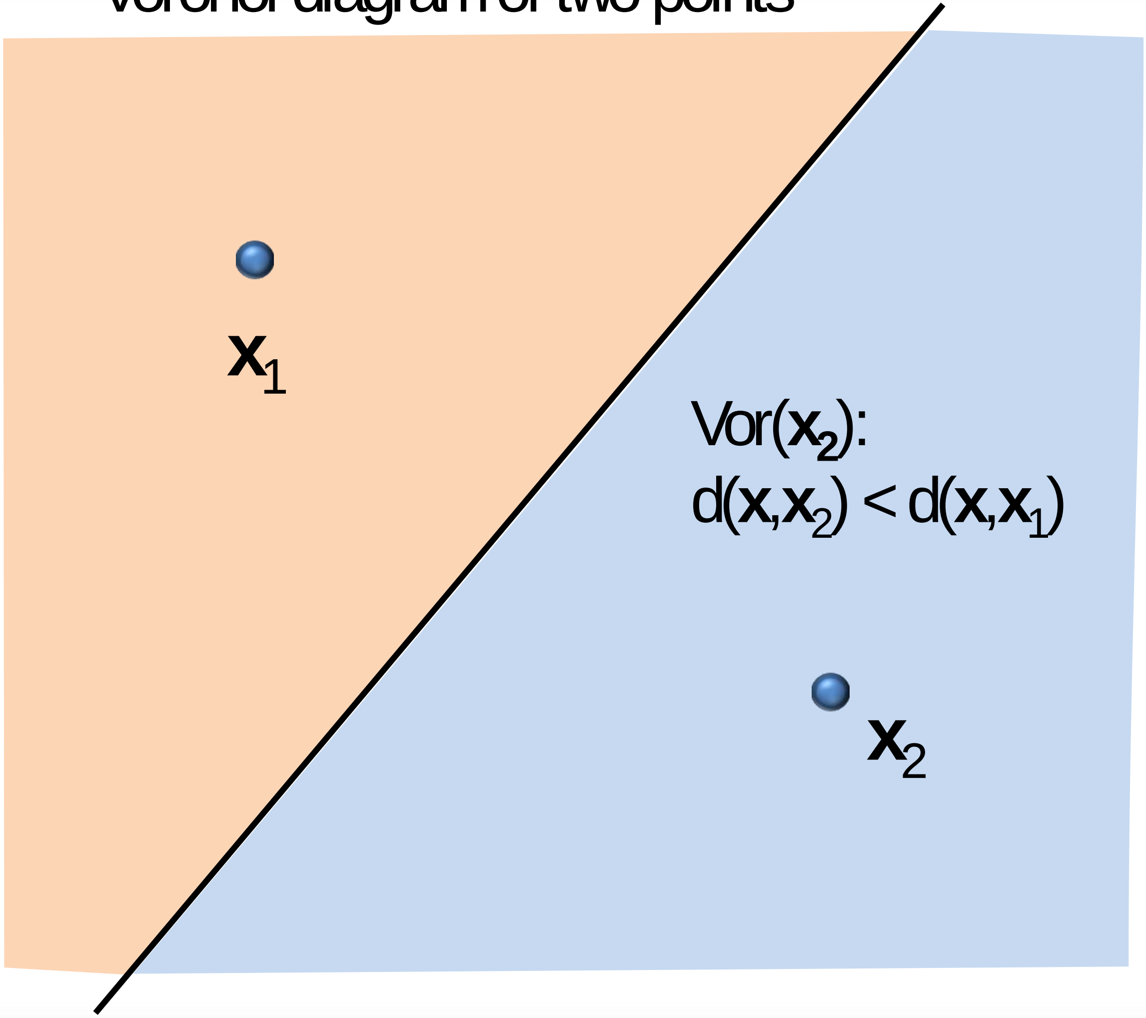
1. Reconstruction

Voronoi diagram of two points



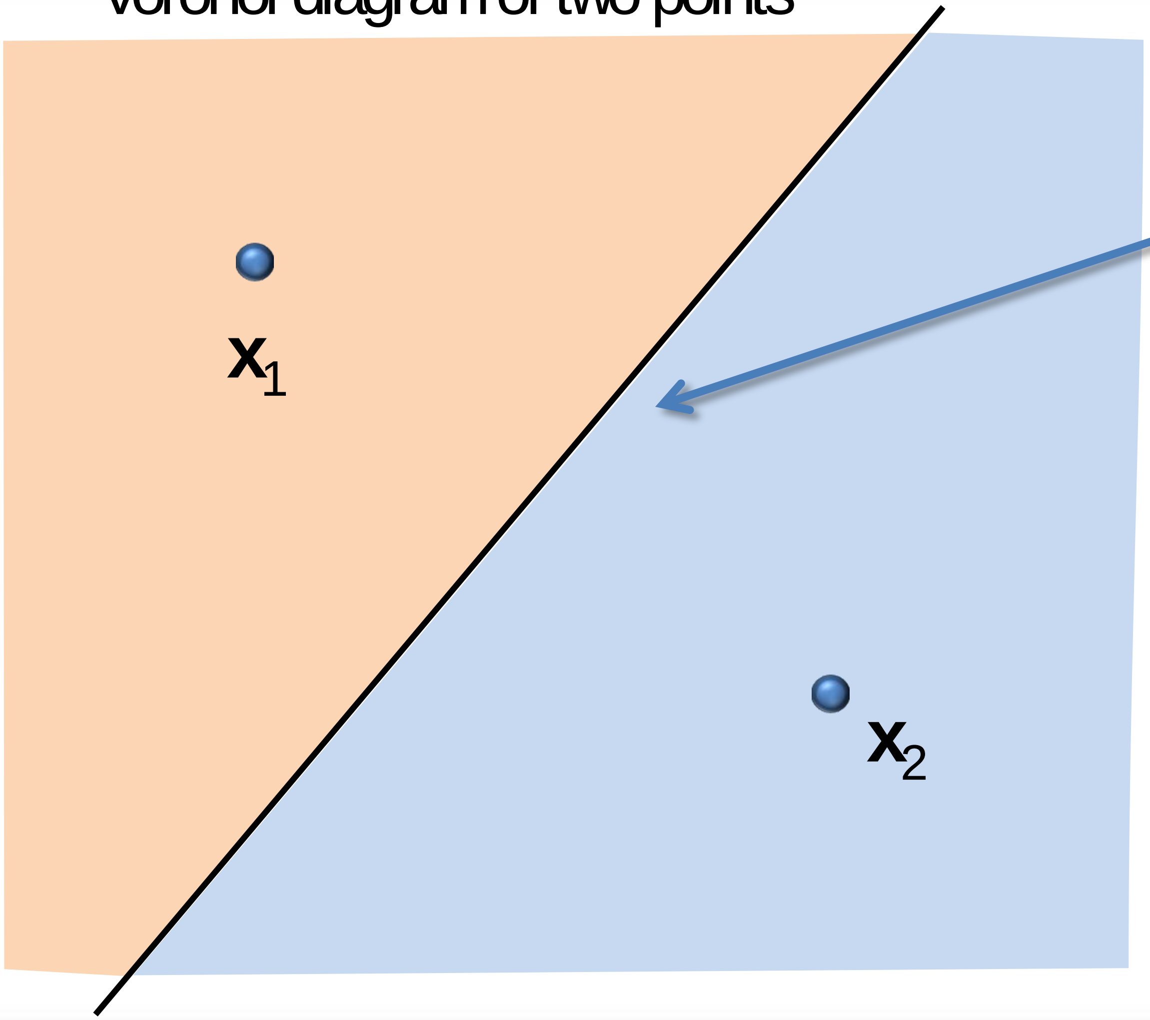
1. Reconstruction

Voronoi diagram of two points



1. Reconstruction

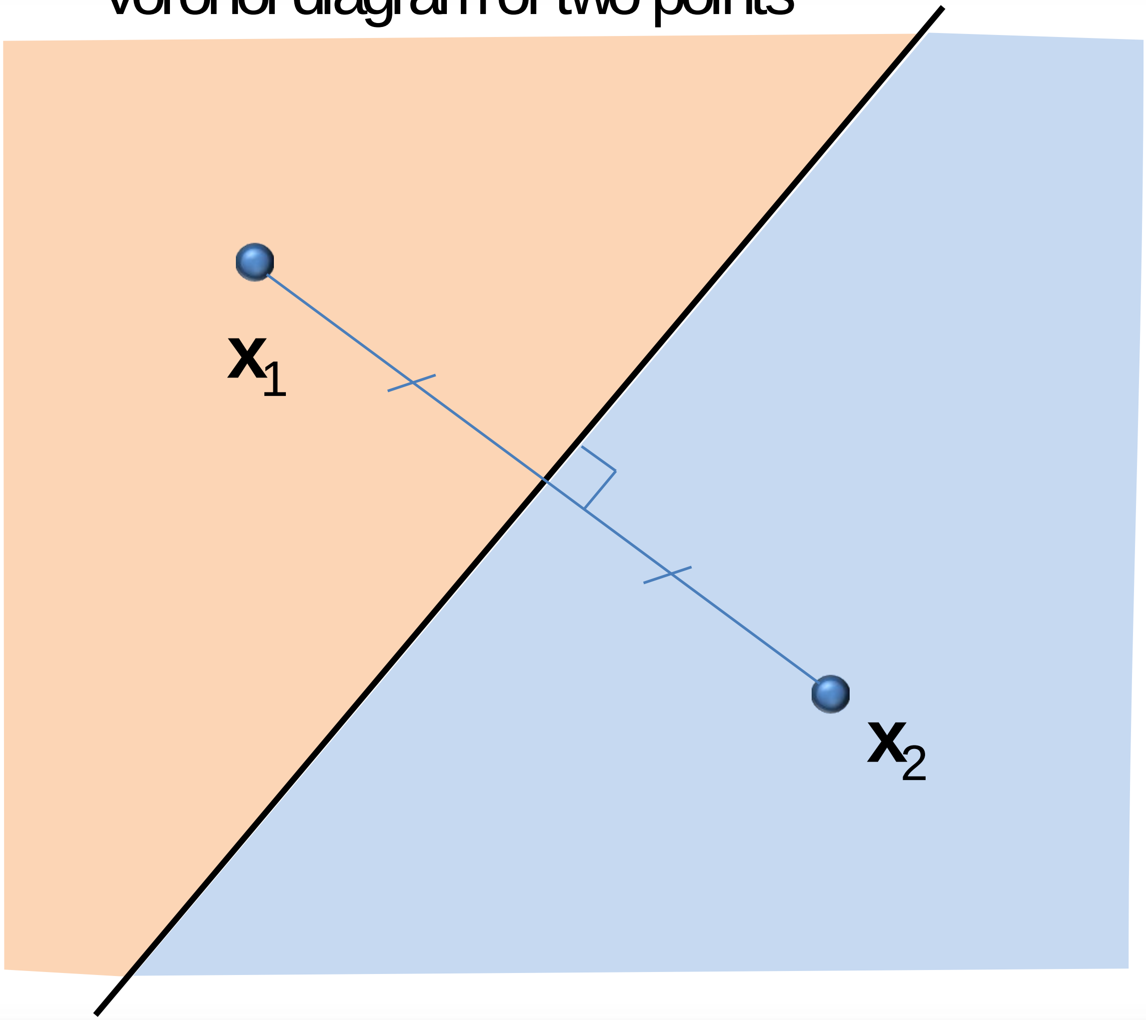
Voronoi diagram of two points



Bisector:
$d(\mathbf{x},\mathbf{x}_1) = d(\mathbf{x},\mathbf{x}_2)$

1. Reconstruction

Voronoi diagram of two points

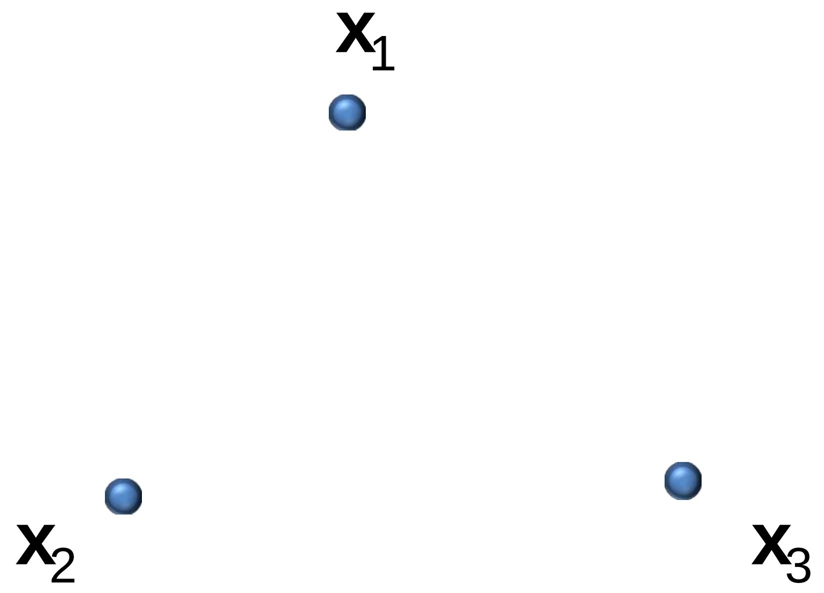


Bisector:

$d(\mathbf{x},\mathbf{x}_1) = d(\mathbf{x},\mathbf{x}_2)$

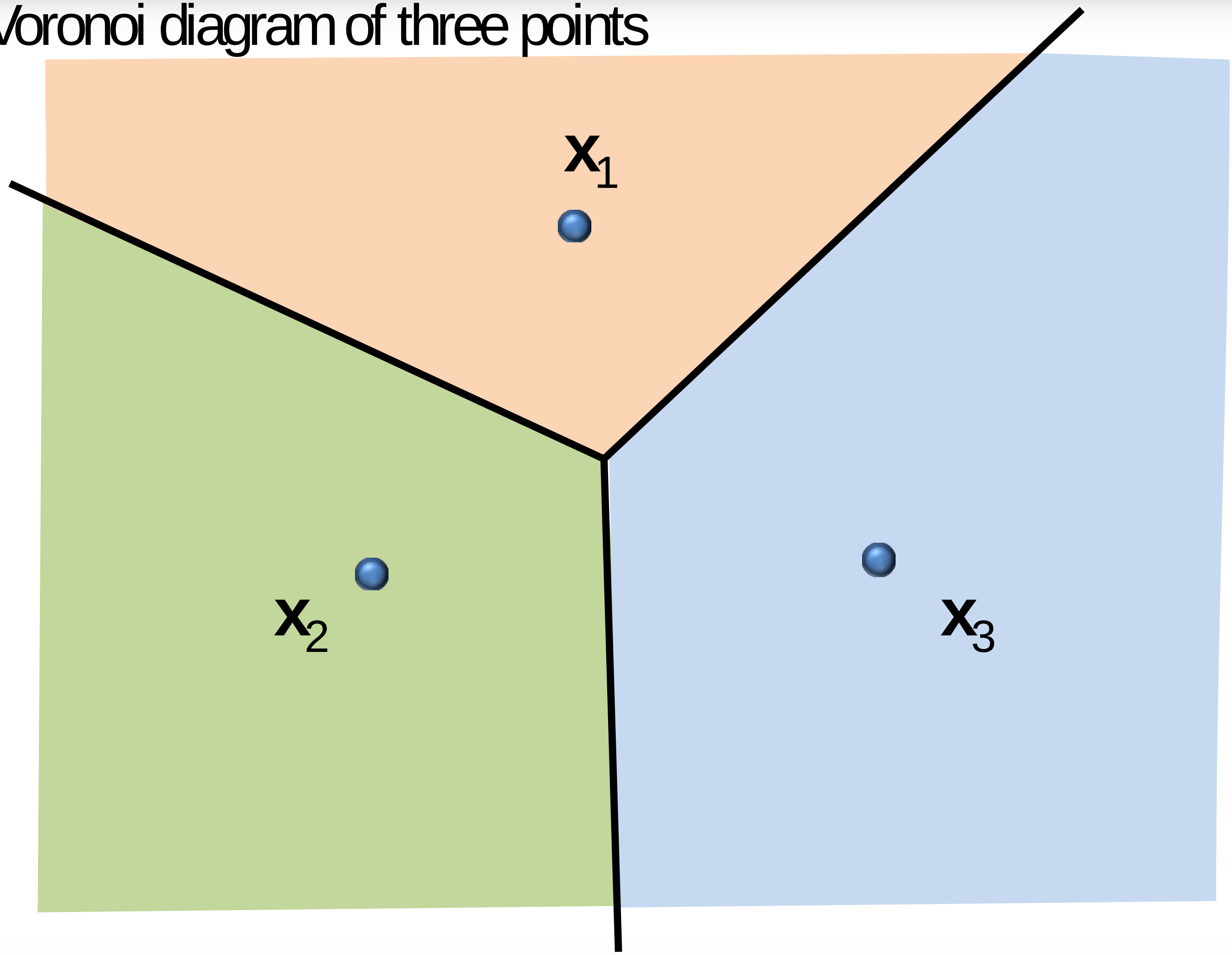
1. Reconstruction

Voronoi diagram of three points



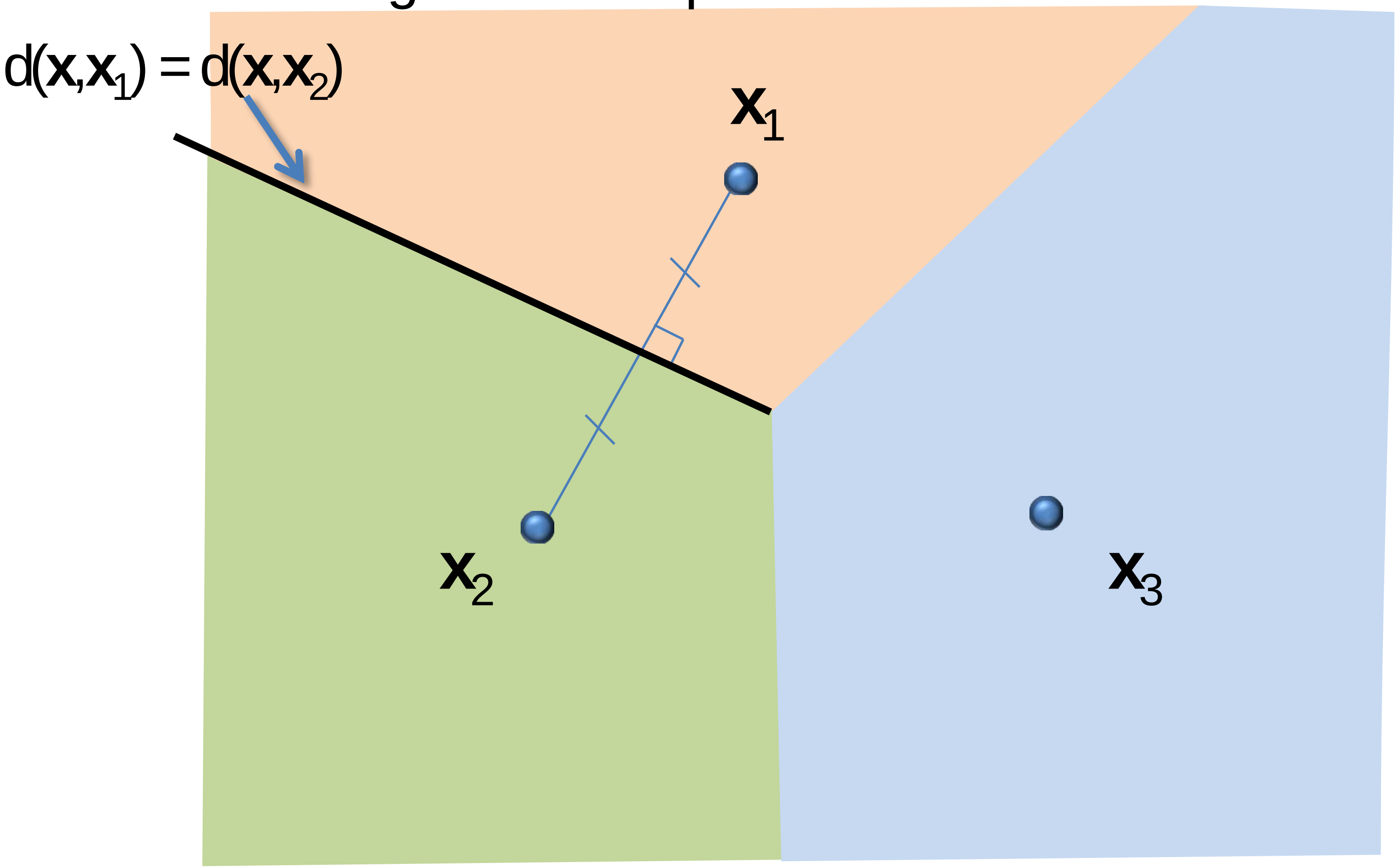
1. Reconstruction

Voronoi diagram of three points



1. Reconstruction

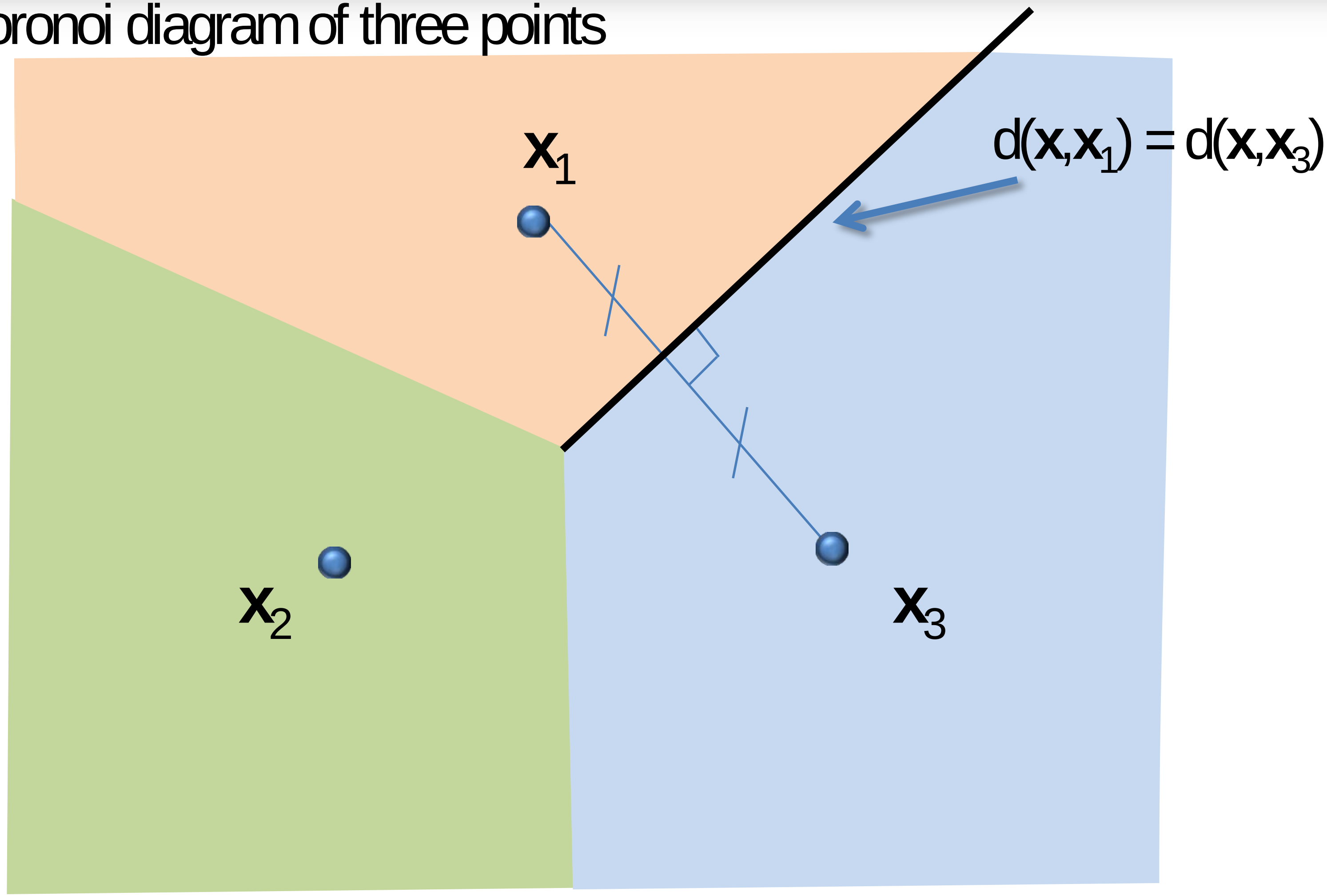
Voronoi diagram of three points



1. Reconstruction

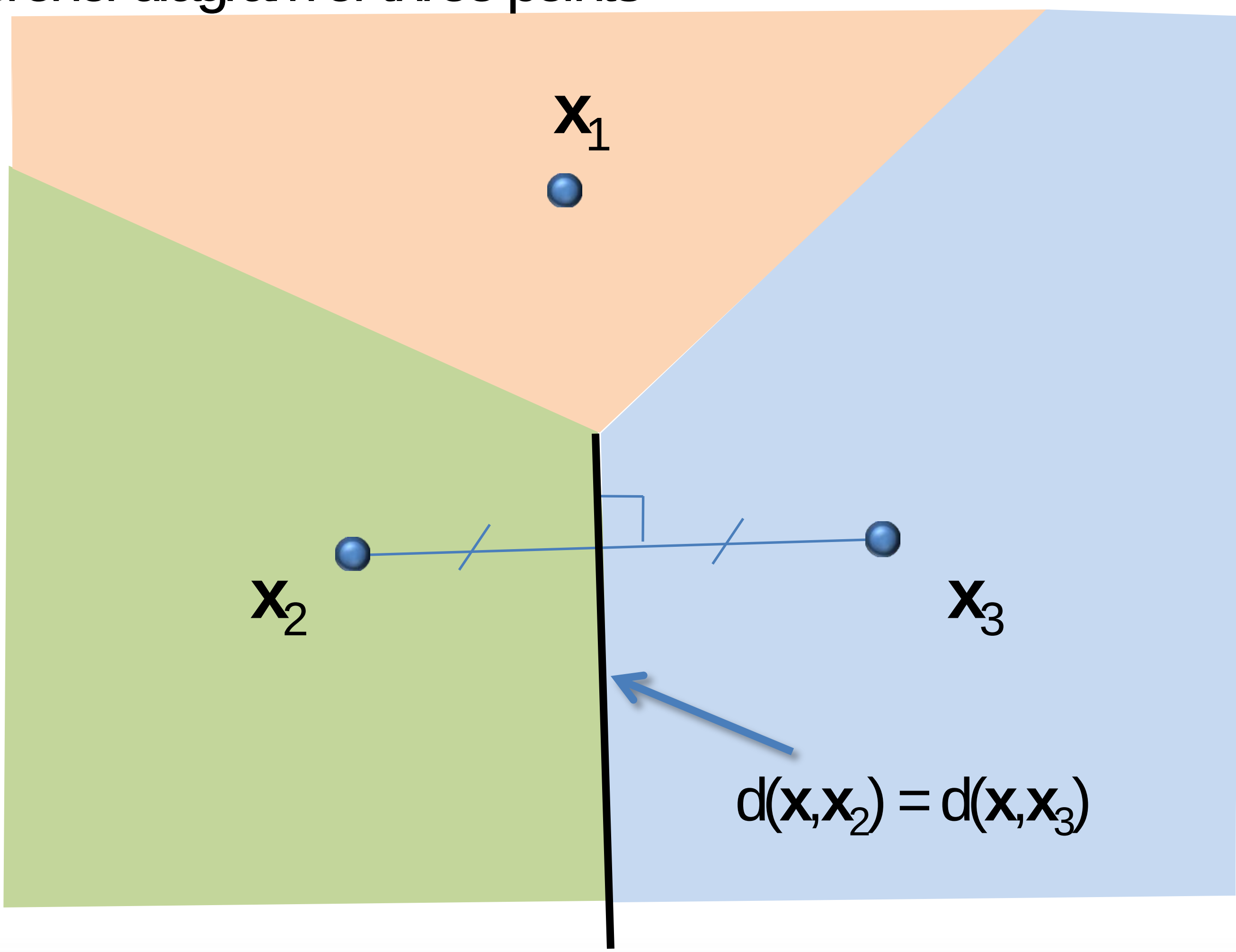
SIGGRAPH ASIA 2011 HONG KONG

Voronoi diagram of three points



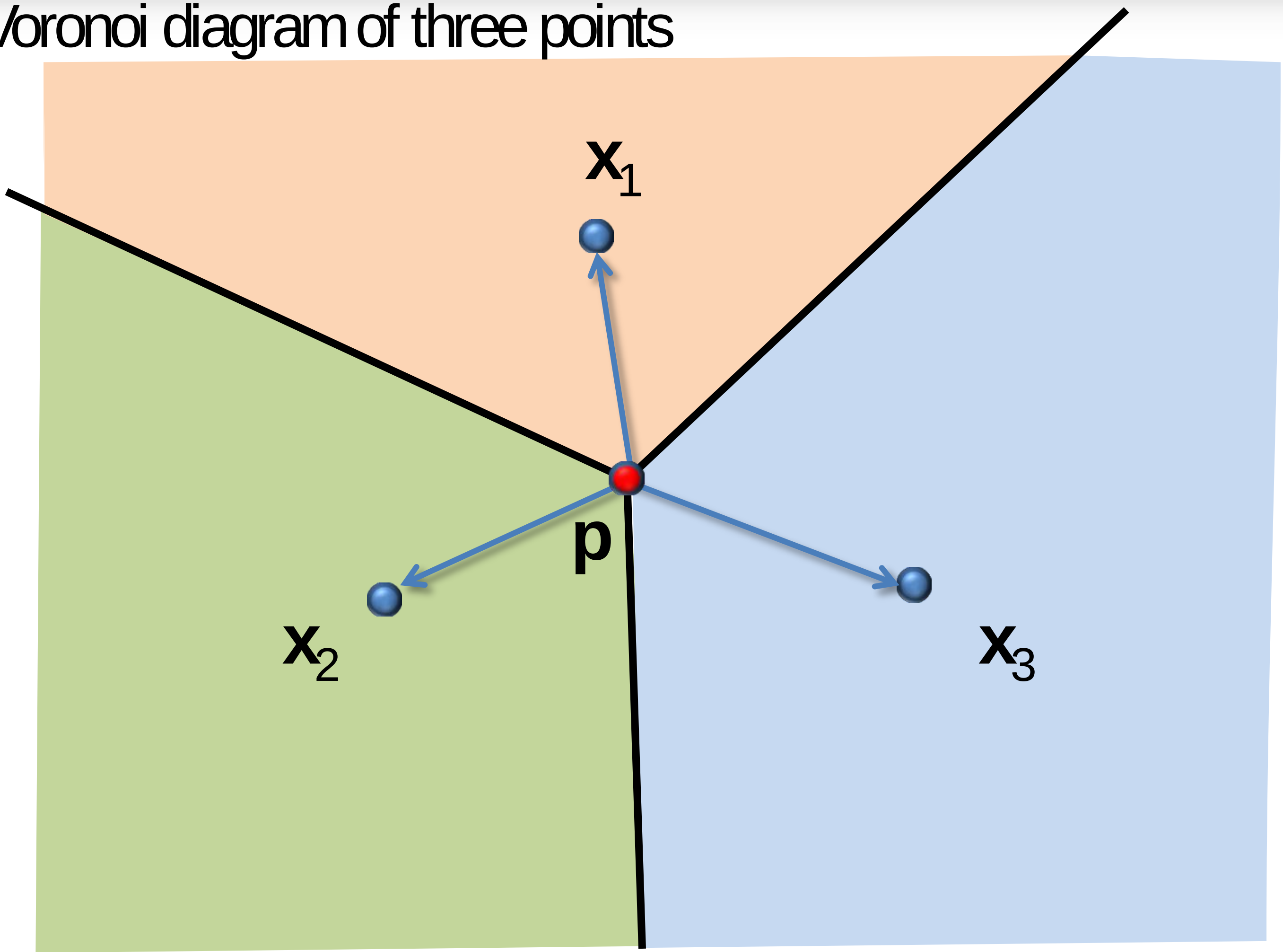
1. Reconstruction

Voronoi diagram of three points



1. Reconstruction

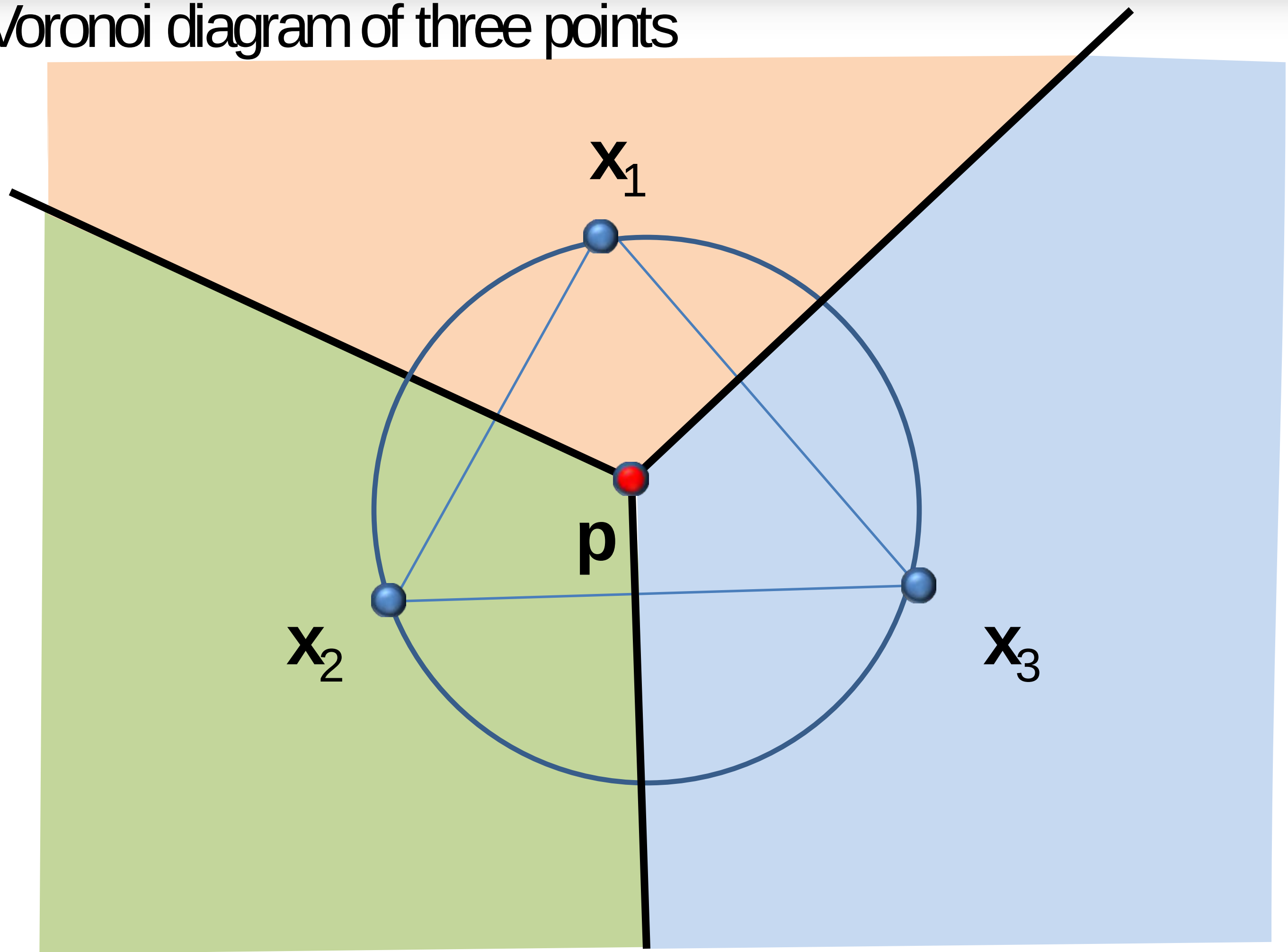
Voronoi diagram of three points



$$d(\mathbf{p},\mathbf{x}_1) = d(\mathbf{p},\mathbf{x}_2) = d(\mathbf{p},\mathbf{x}_3)$$

1. Reconstruction

Voronoi diagram of three points



$\mathbf{p}$ is the *circumcenter* of the triangle $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$

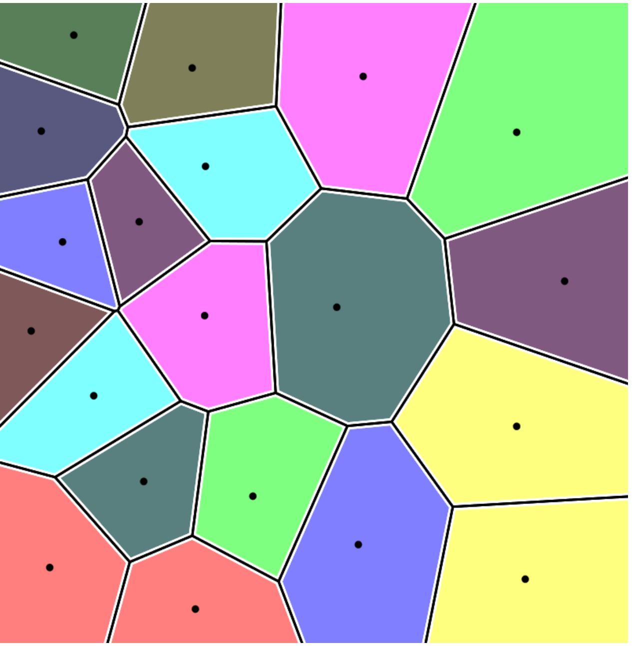
1. Reconstruction

Voronoi diagram
of N points

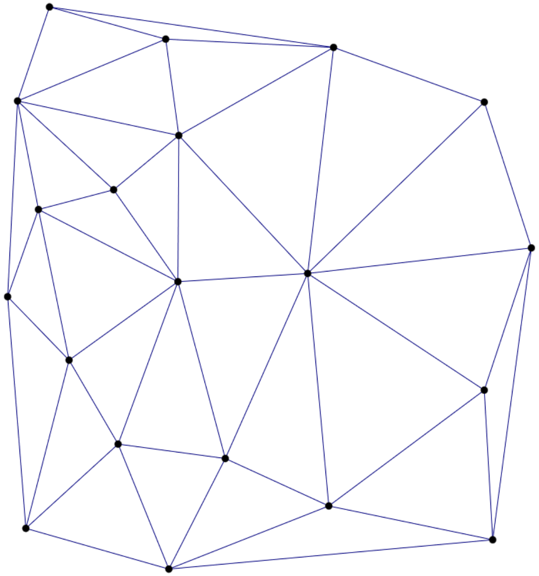
1. Reconstruction

Voronoi diagram

Delaunay triangulation



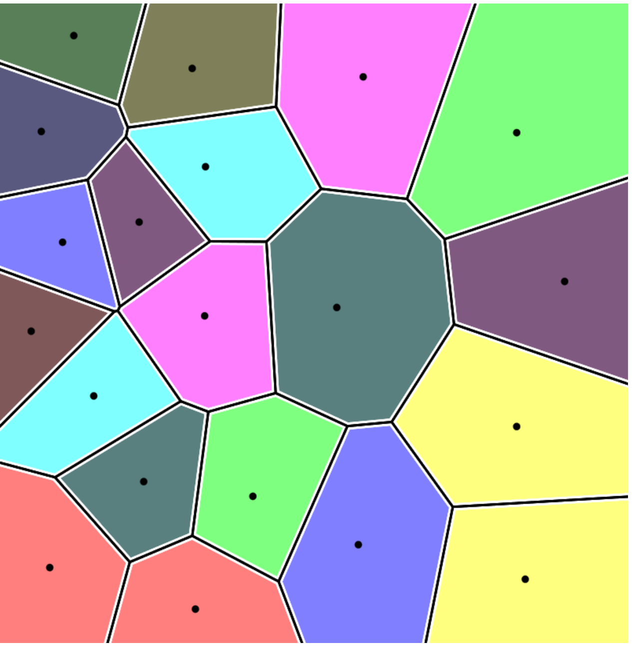
Cell



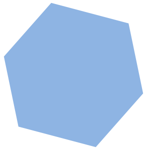
1. Reconstruction

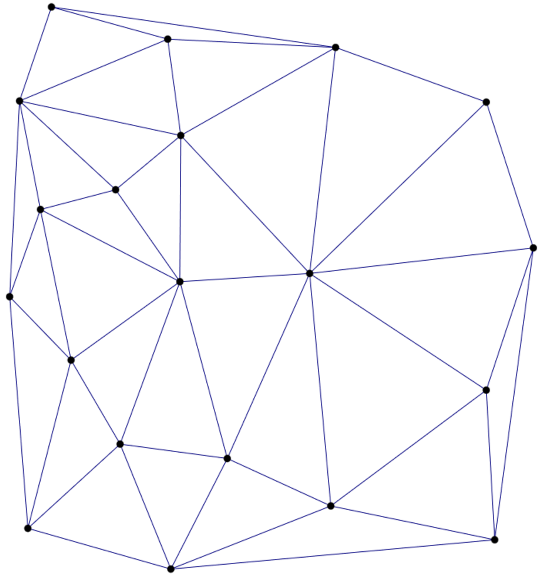
Voronoi diagram

Delaunay triangulation



Cell



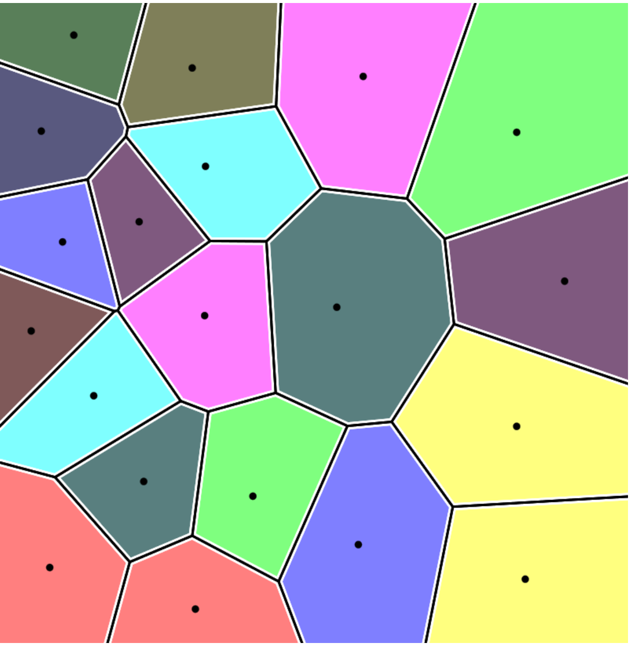


1. Reconstruction

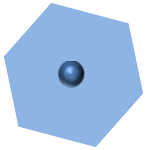
SIGGRAPH ASIA 2011 HONG KONG

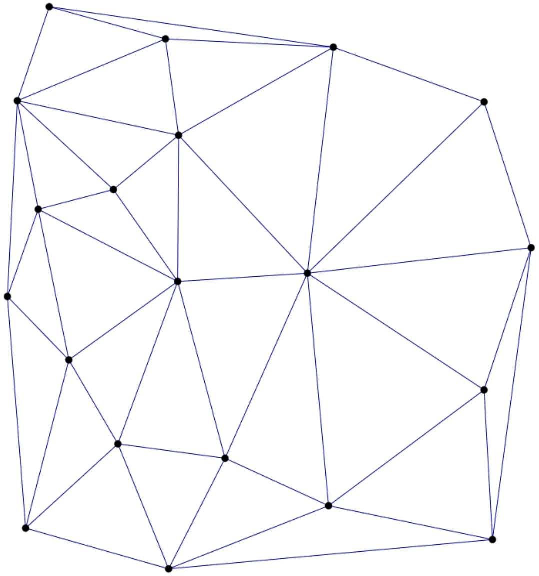
Voronoi diagram

Delaunay triangulation



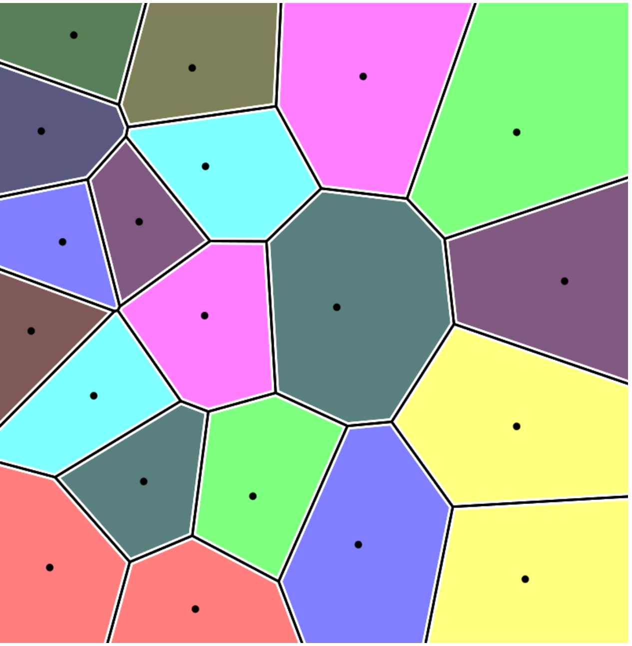
Cell



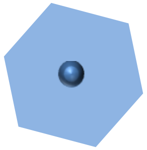


1. Reconstruction

Voronoi diagram

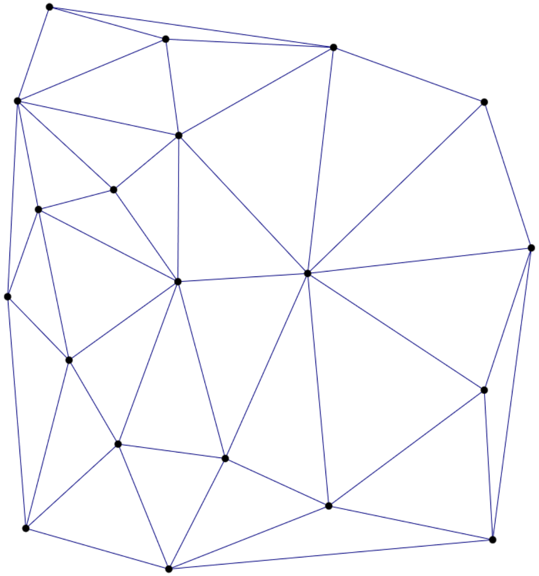


Cell



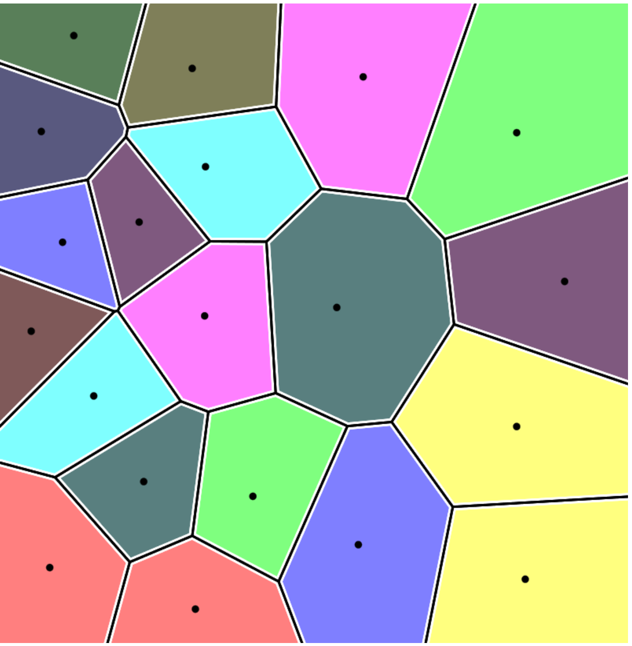
Delaunay triangulation

Vertex

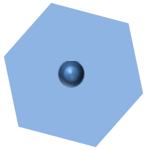


1. Reconstruction

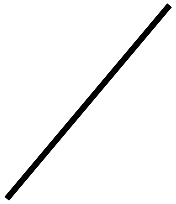
Voronoi diagram



Cell

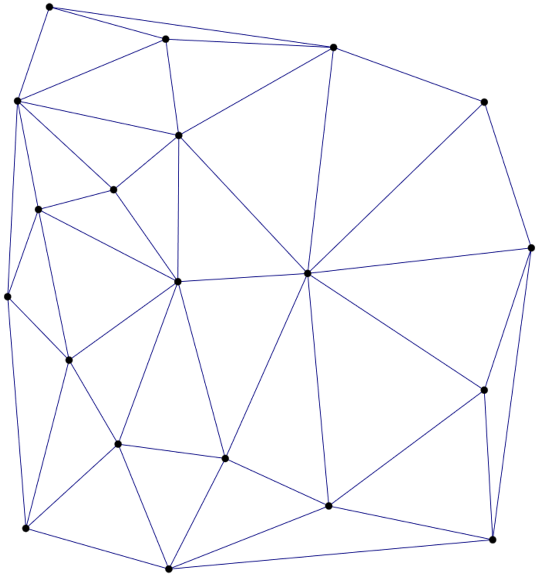


Edge



Delaunay triangulation

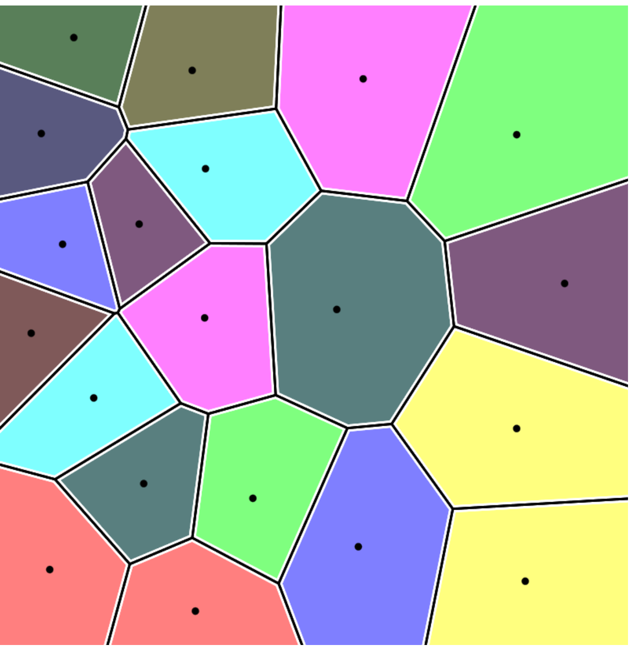
Vertex



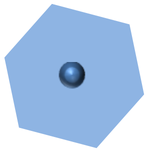
1. Reconstruction

Voronoi diagram

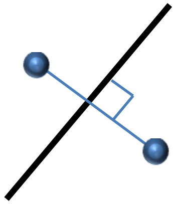
Delaunay triangulation



Cell

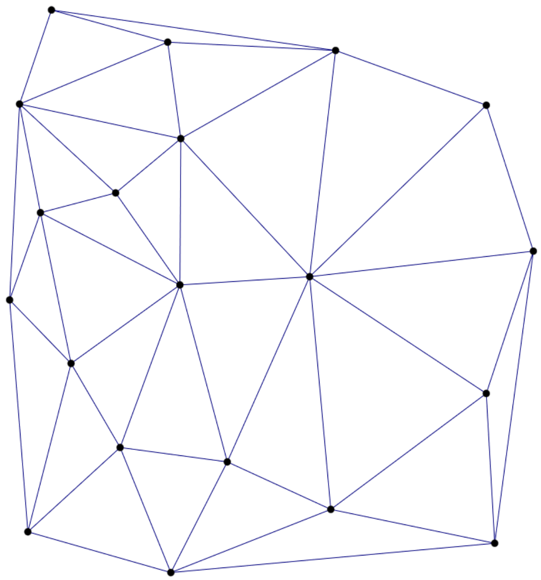


Edge



Vertex

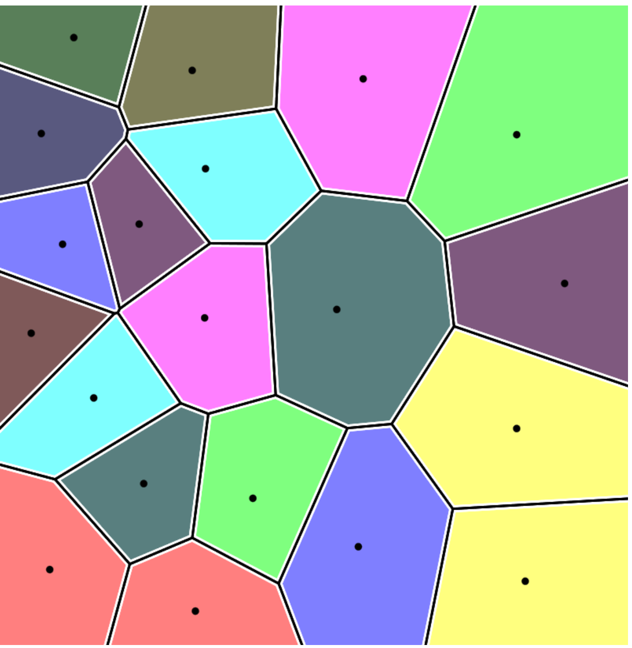
Edge



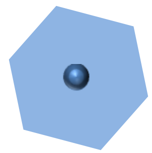
1. Reconstruction

Voronoi diagram

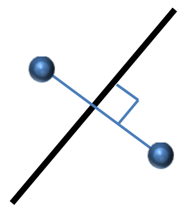
Delaunay triangulation



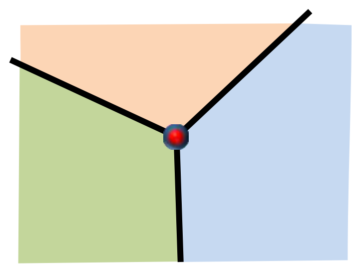
Cell



Edge

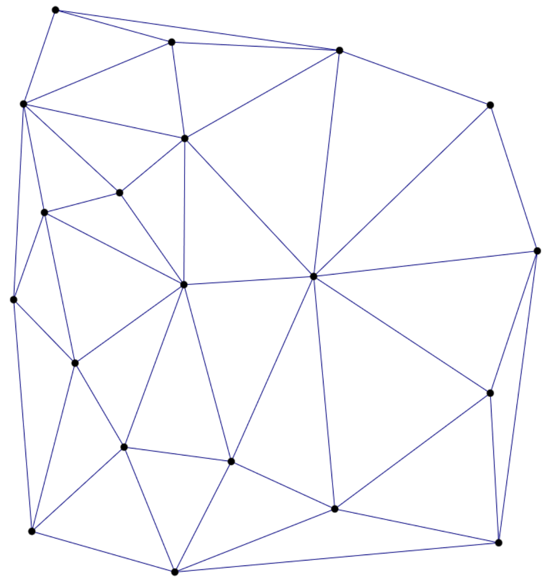


Vertex



Vertex

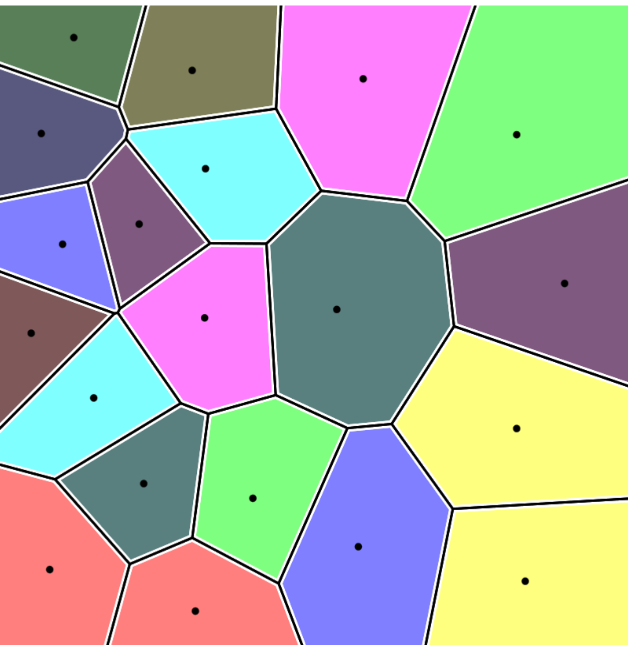
Edge



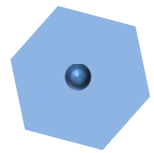
1. Reconstruction

Voronoi diagram

Delaunay triangulation

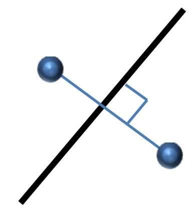


Cell



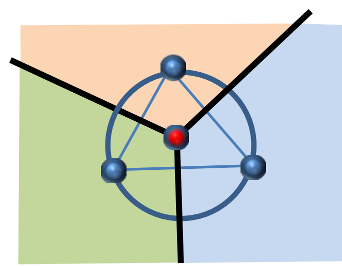
Vertex

Edge

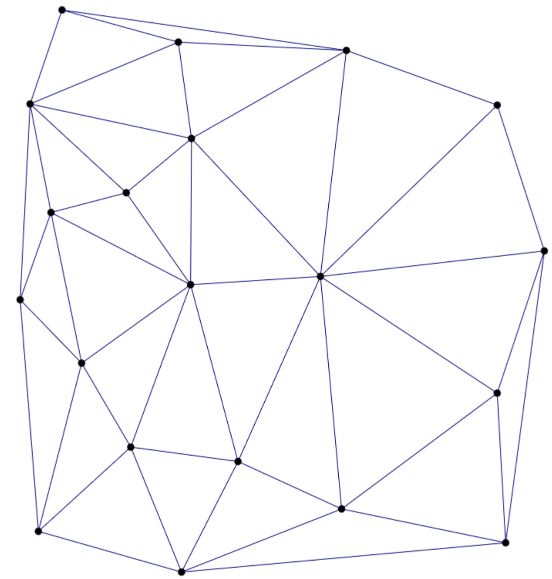


Edge

Vertex



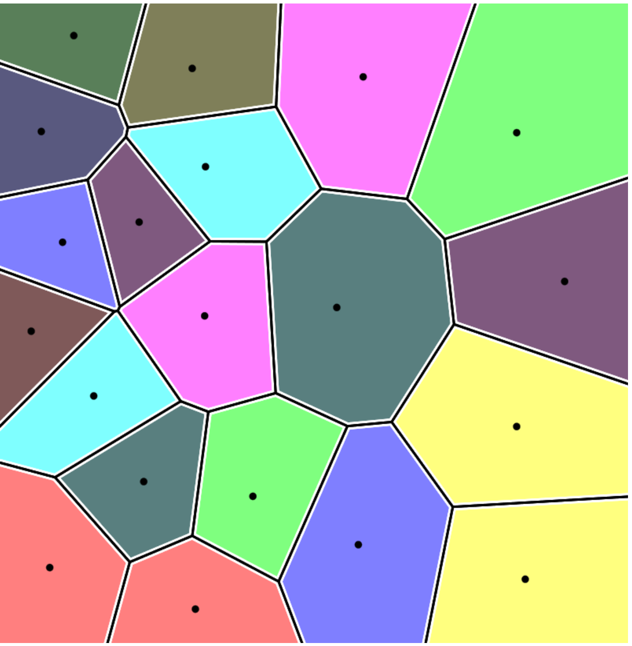
Cell (triangle)



1. Reconstruction

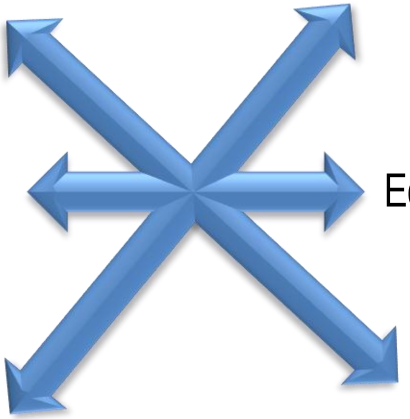
Voronoi diagram

Delaunay triangulation



Cell

Vertex

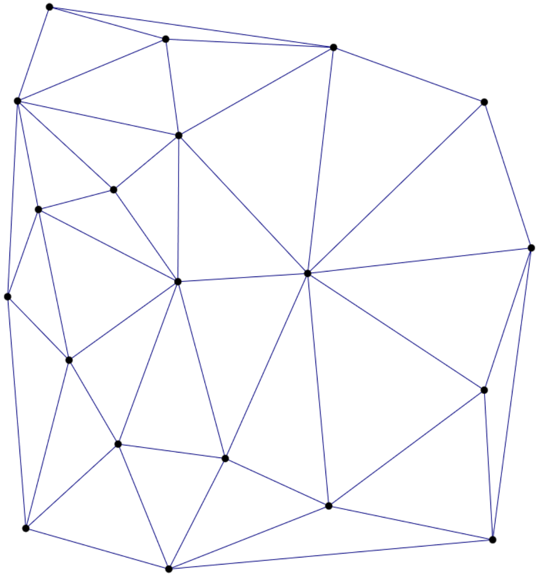


Edge

Edge

Vertex

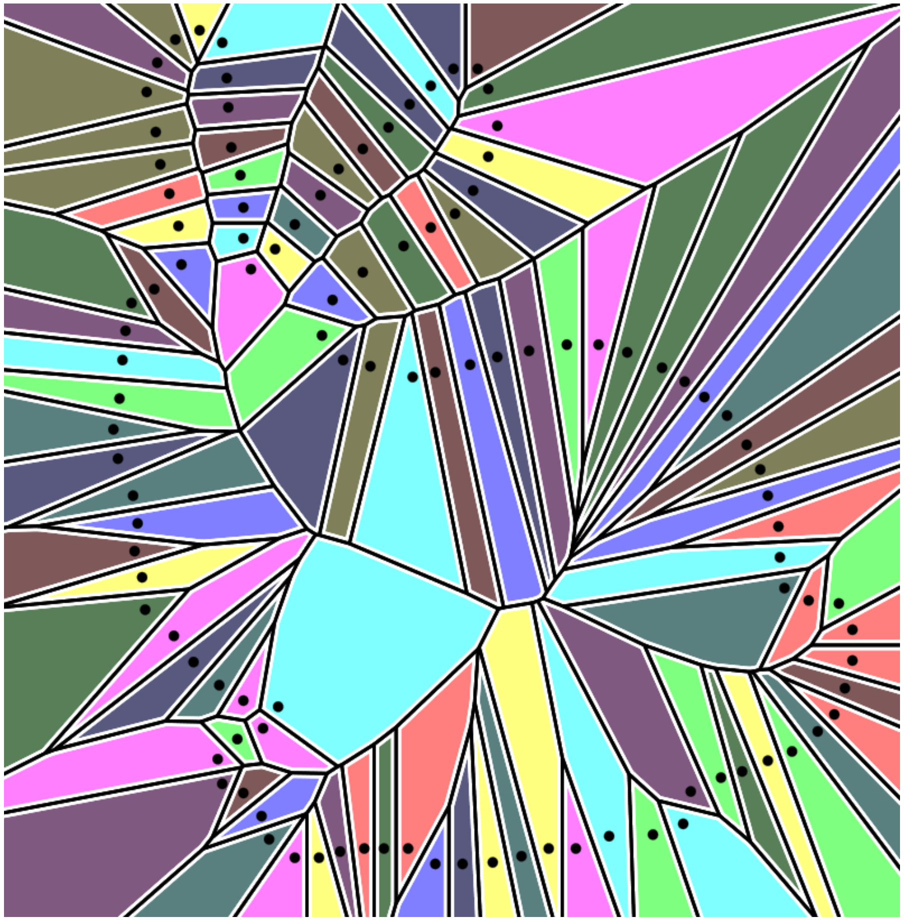
Cell
(triangle)



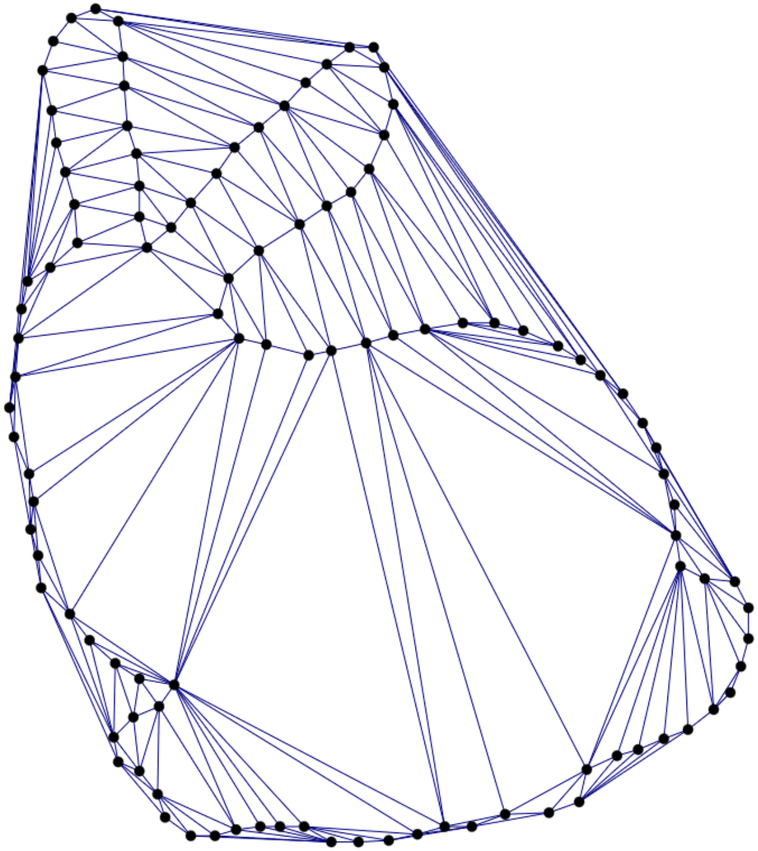
Duality

1. Reconstruction

Voronoi diagram

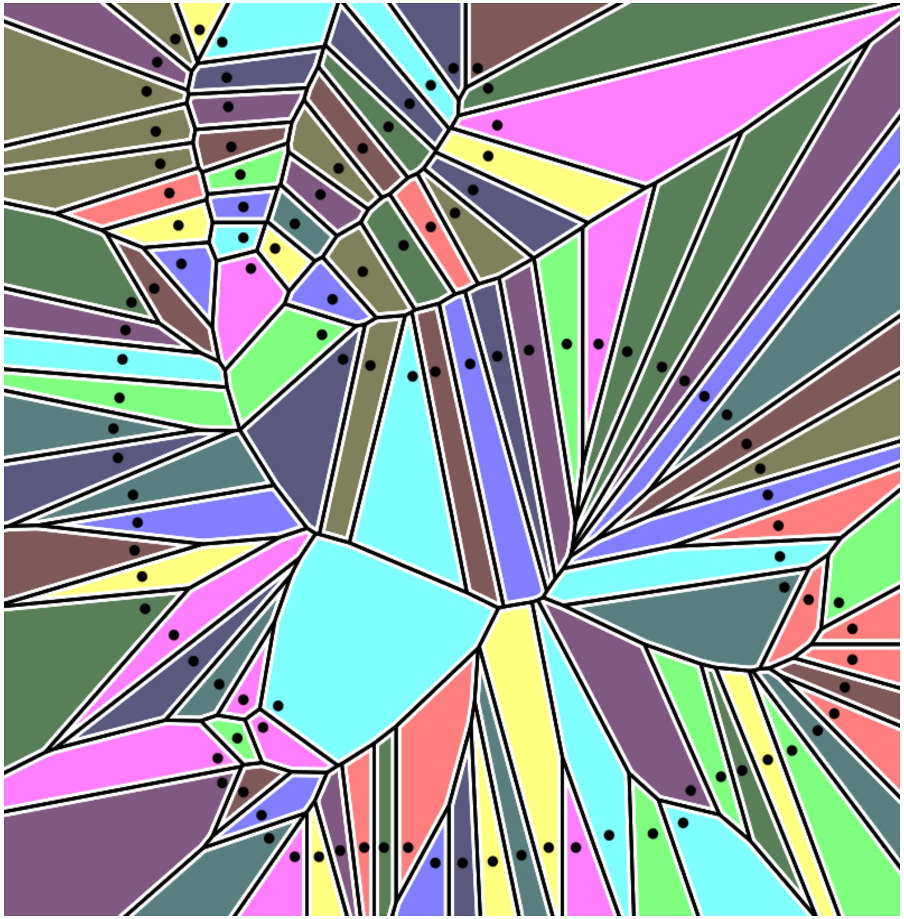


Delaunay triangulation

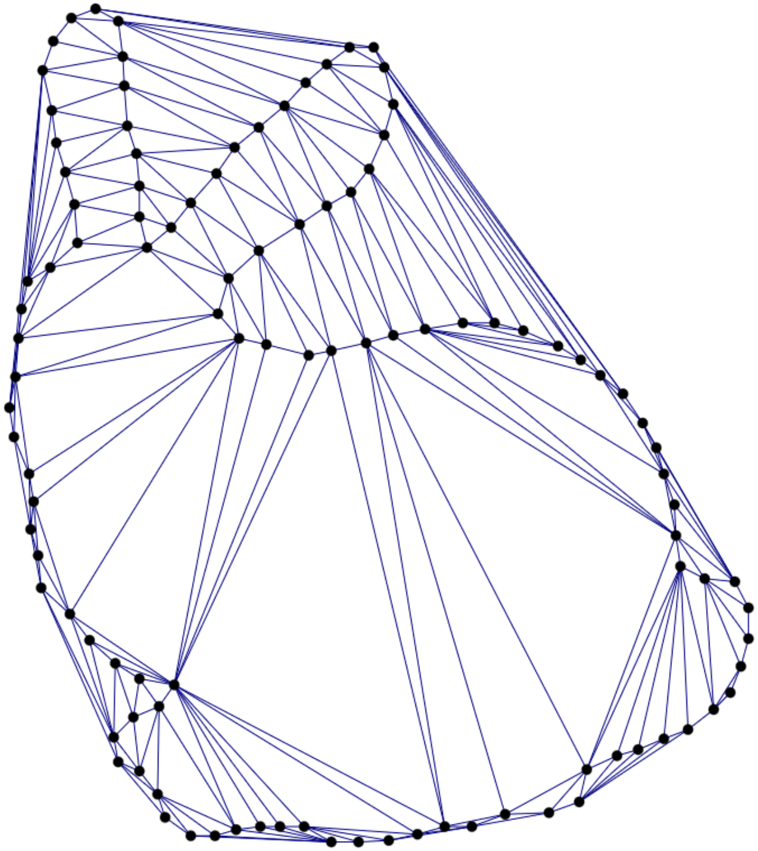


1. Reconstruction

Voronoi diagram



Delaunay triangulation

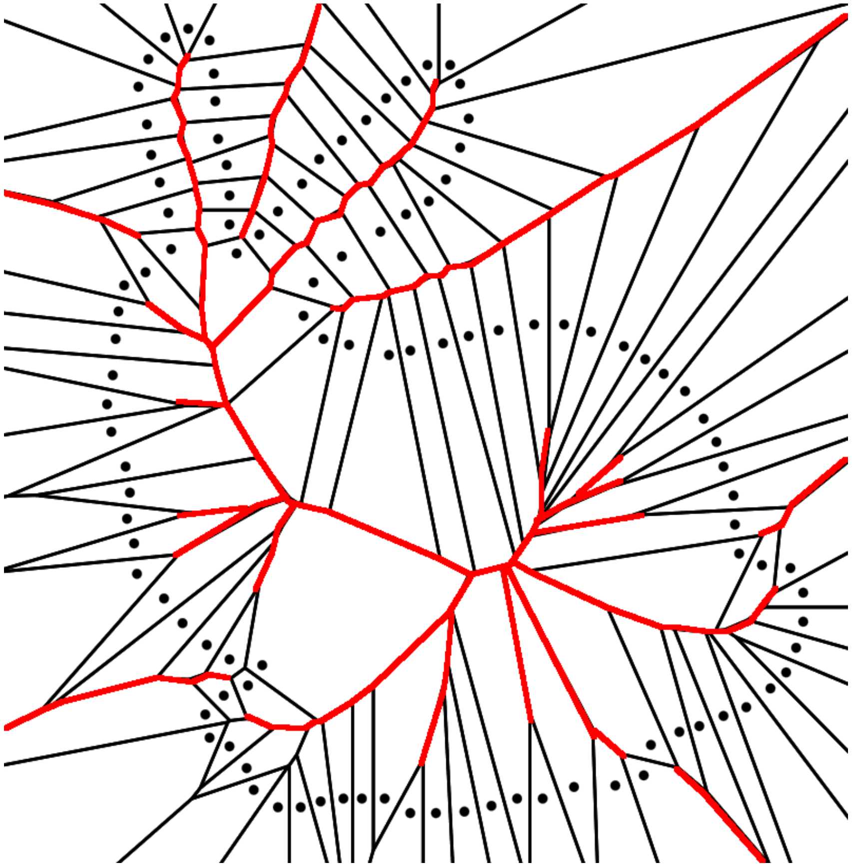


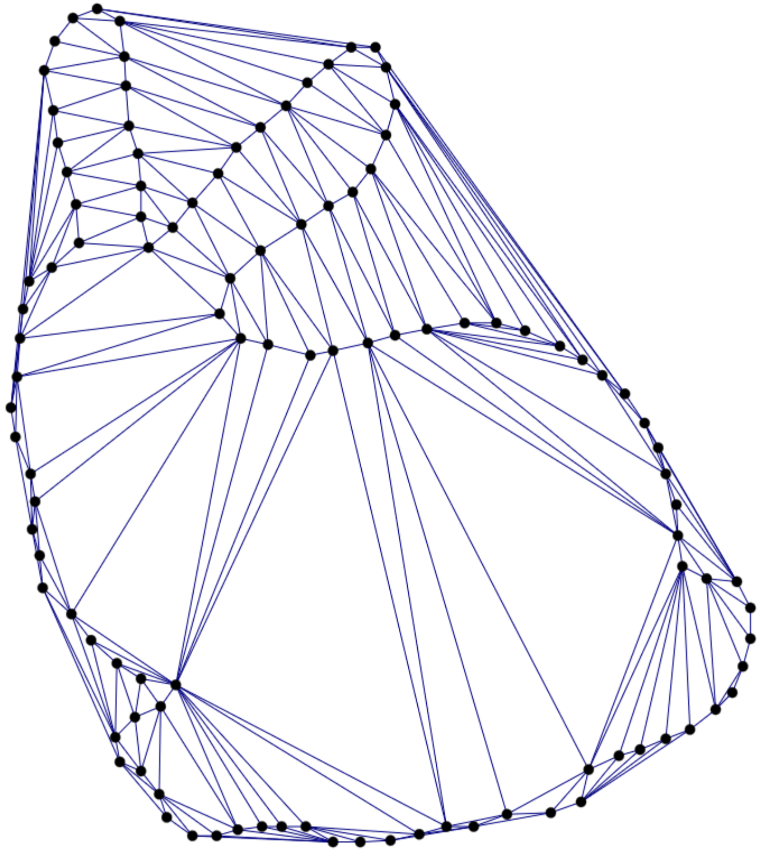
Q: how to select the “right” edges (triangles) ?

1. Reconstruction

Voronoi diagram

Delaunay triangulation

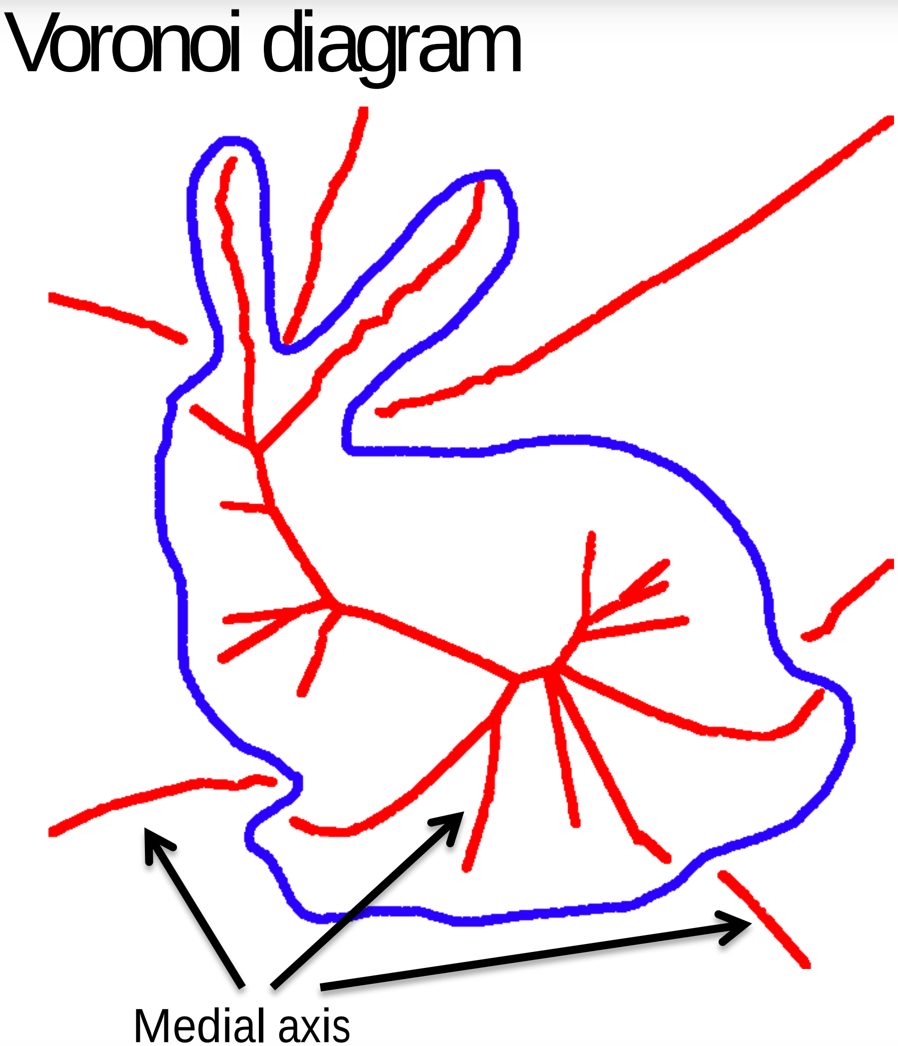


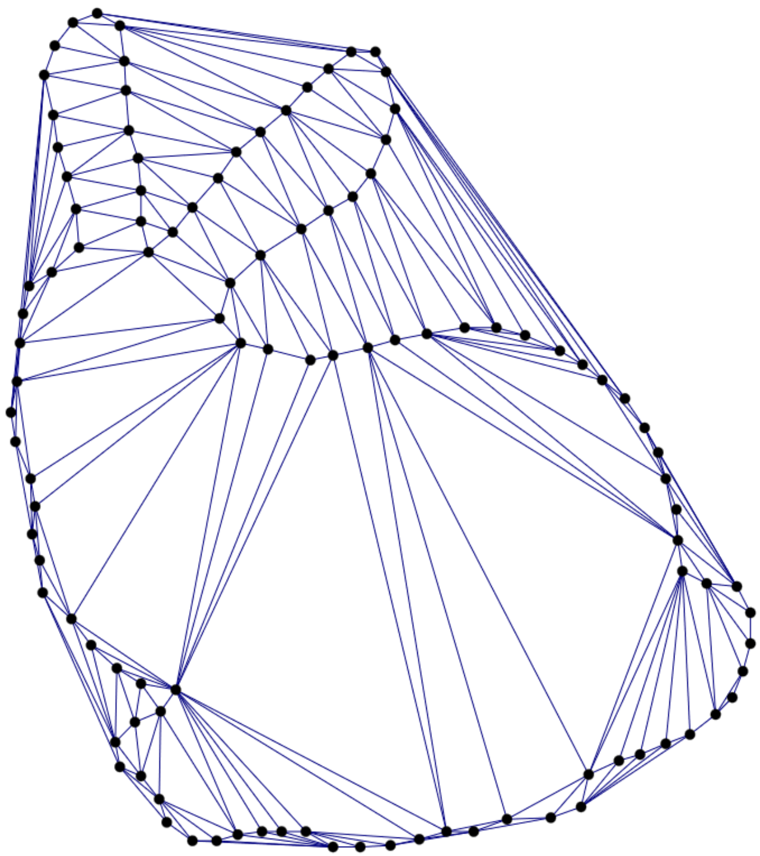


The "crust" [Amenta et.al, 1998]

"right" edges do not cross the medial axis

1. Reconstruction

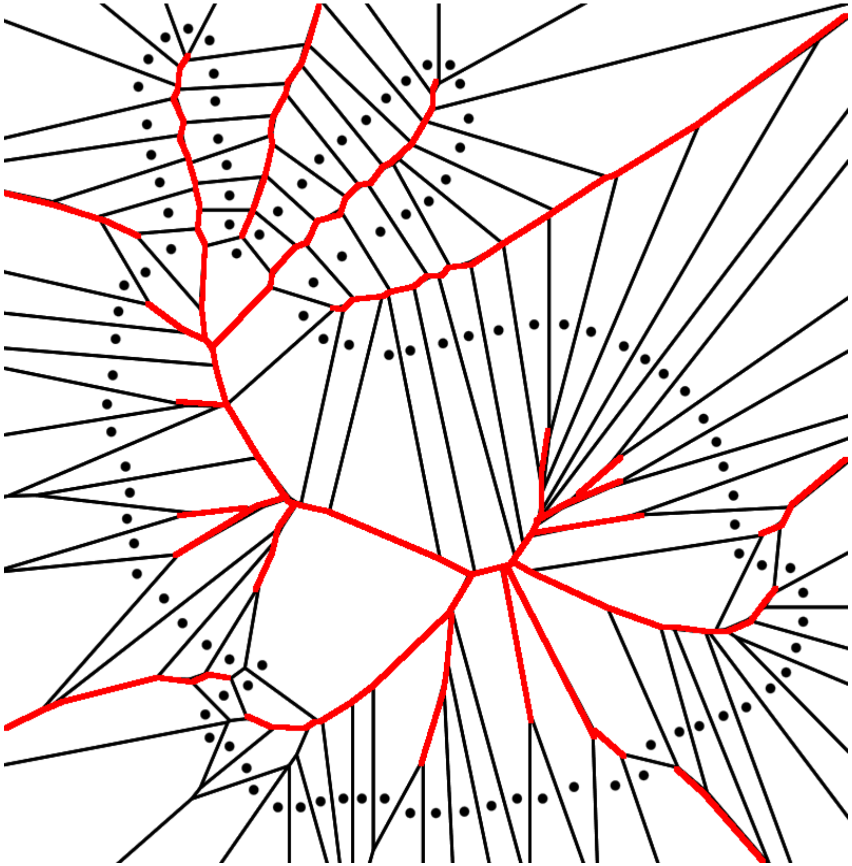




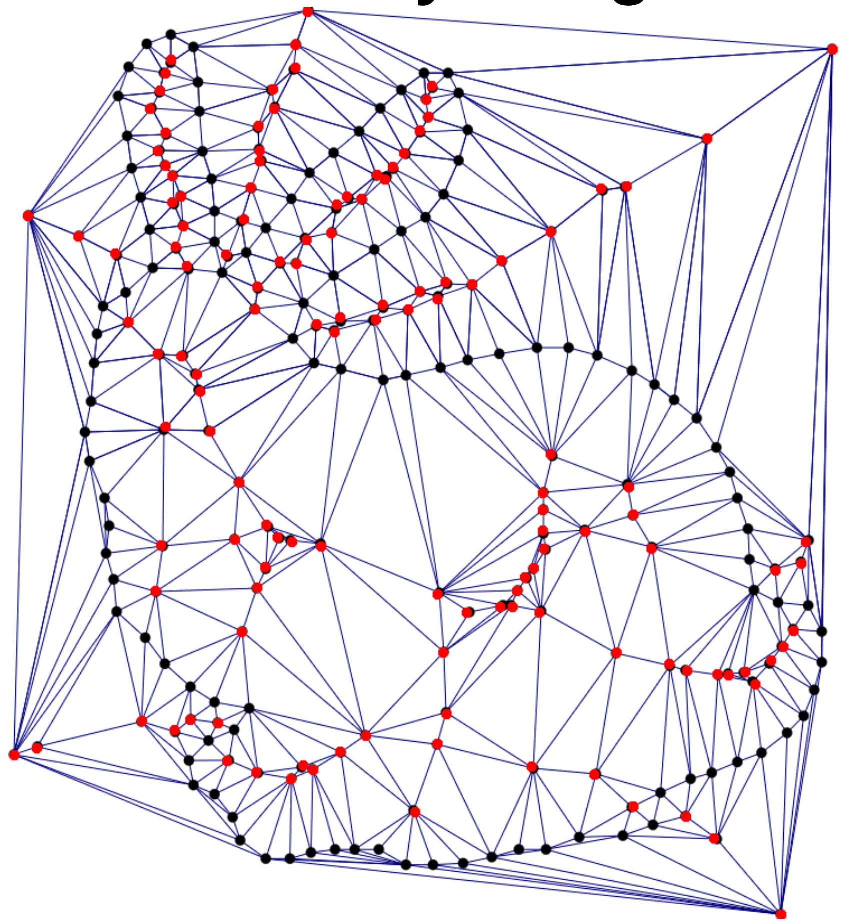
"right" edges do not cross the medial axis

1. Reconstruction

Voronoi diagram

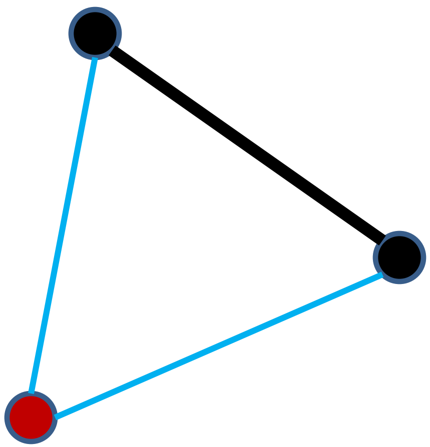


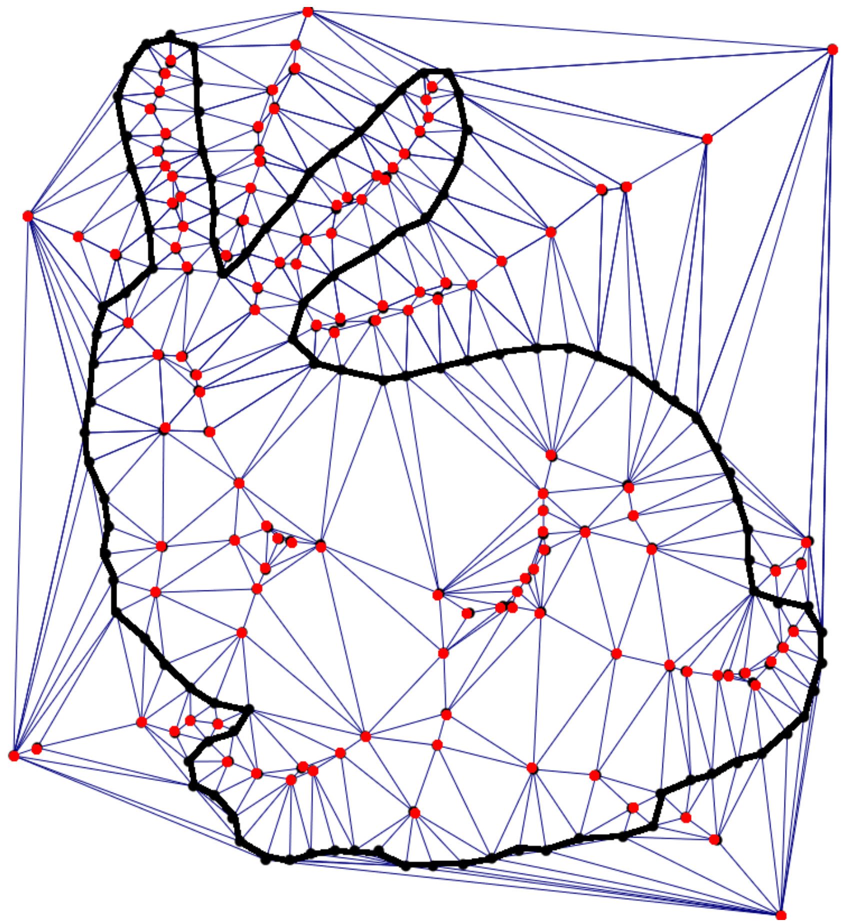
Delaunay triangulation



Insert the Voronoi vertices (red) into the triangulation

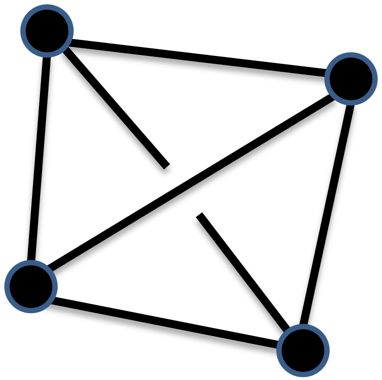
1. Reconstruction





1. Reconstruction

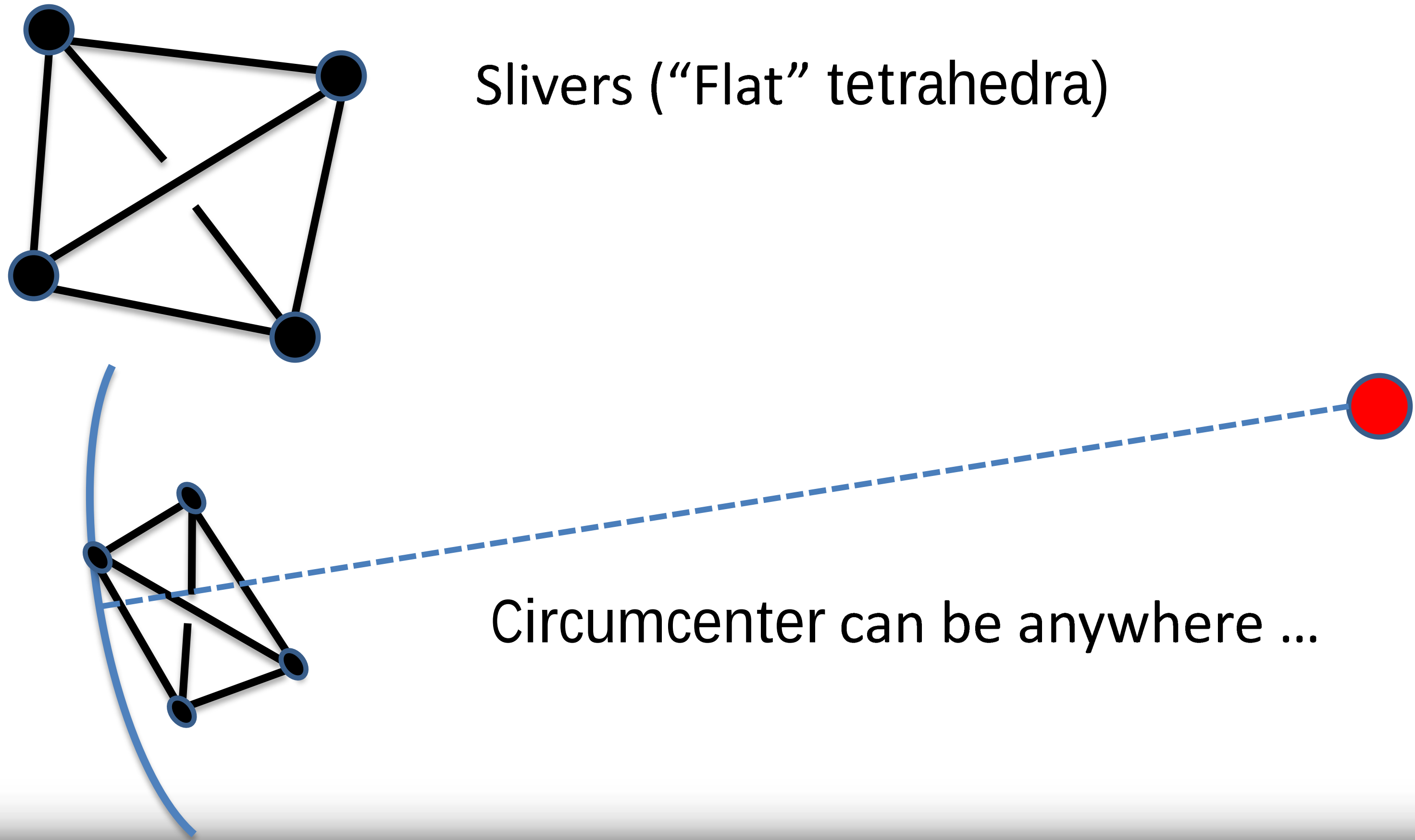
A problem: in 3D the set of Voronoi vertices does not converge to the medial axis !



Slivers ("Flat" tetrahedra)

1. Reconstruction

A problem: in 3D the set of Voronoi vertices does not converge to the medial axis !

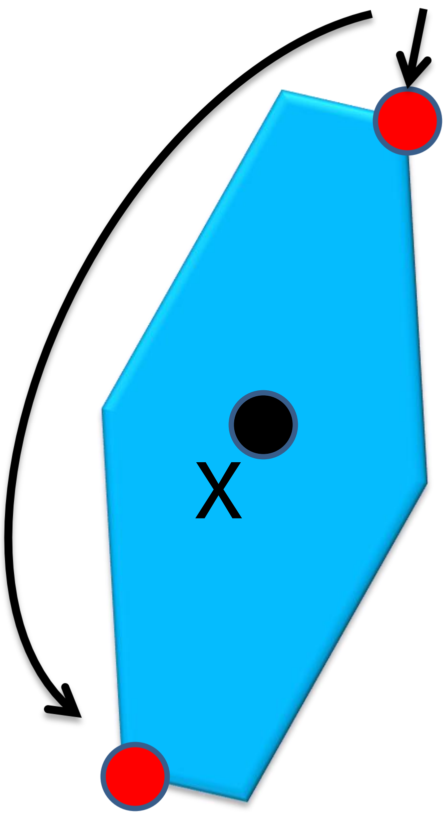


Slivers ("Flat" tetrahedra)

Circumcenter can be anywhere ...

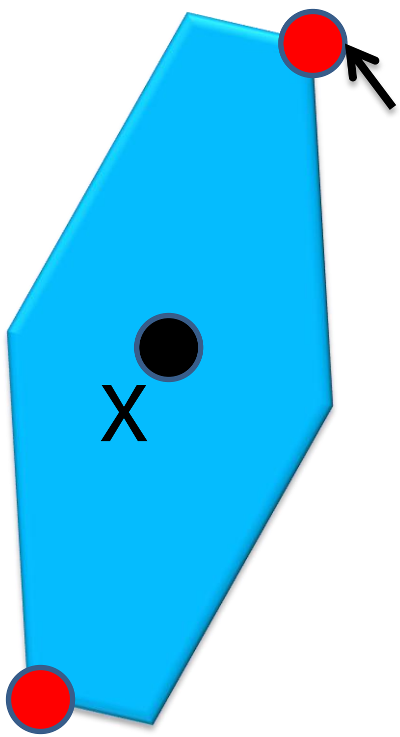
1. Reconstruction

Solution: the set of poles (subset of Voronoi vertices) converge to the medial axis !



1. Reconstruction

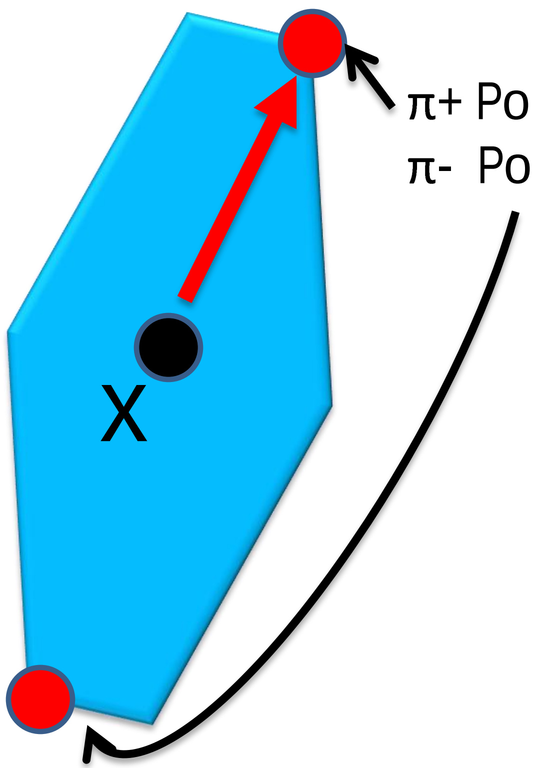
Solution: the set of poles (subset of Voronoi vertices) converge to the medial axis !



π+ Pole of x'sVoronoi cell = furthest Voronoi vertex

1. Reconstruction

Solution: the set of poles (subset of Voronoi vertices) converge to the medial axis !



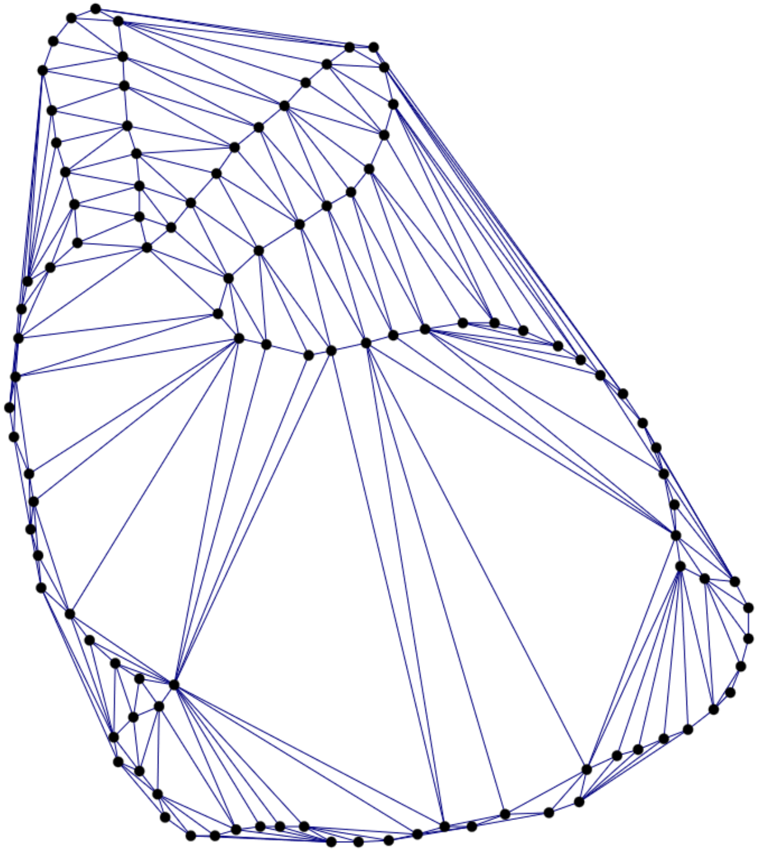
π+ Pole of x'sVoronoi cell = furthest Voronoi vertex

π- Pole = furthest Voronoi vertex in opposite direction

1. Reconstruction

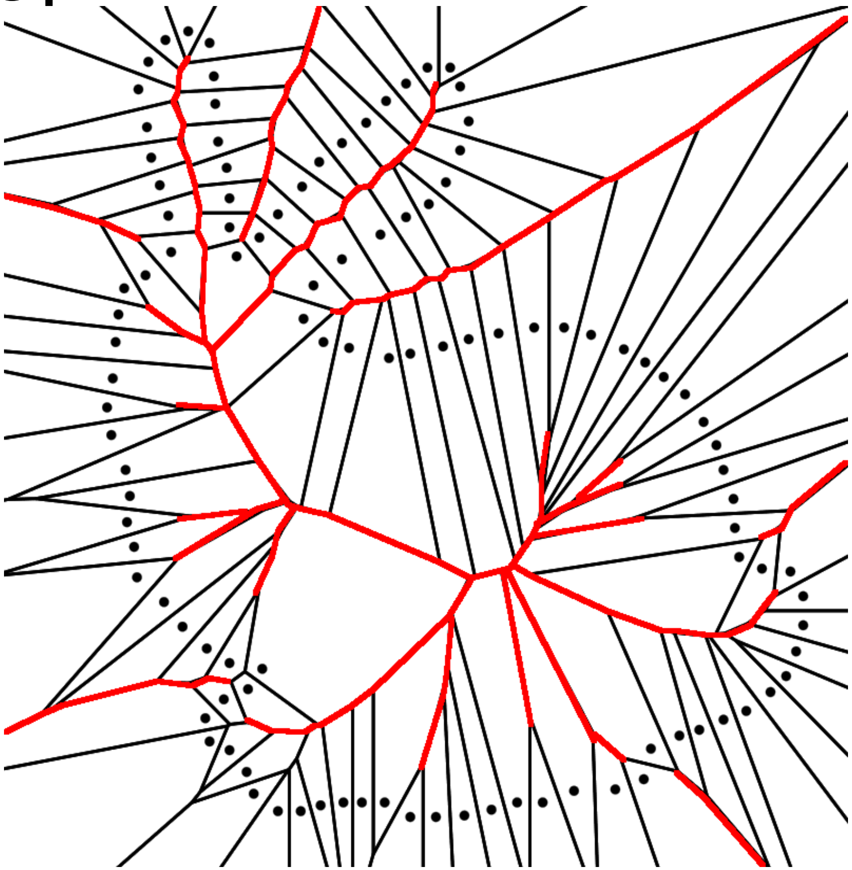
The “crust” [Amenta et.al 98]

1)Compute Del(X)



The “crust” [Amenta et.al 98]

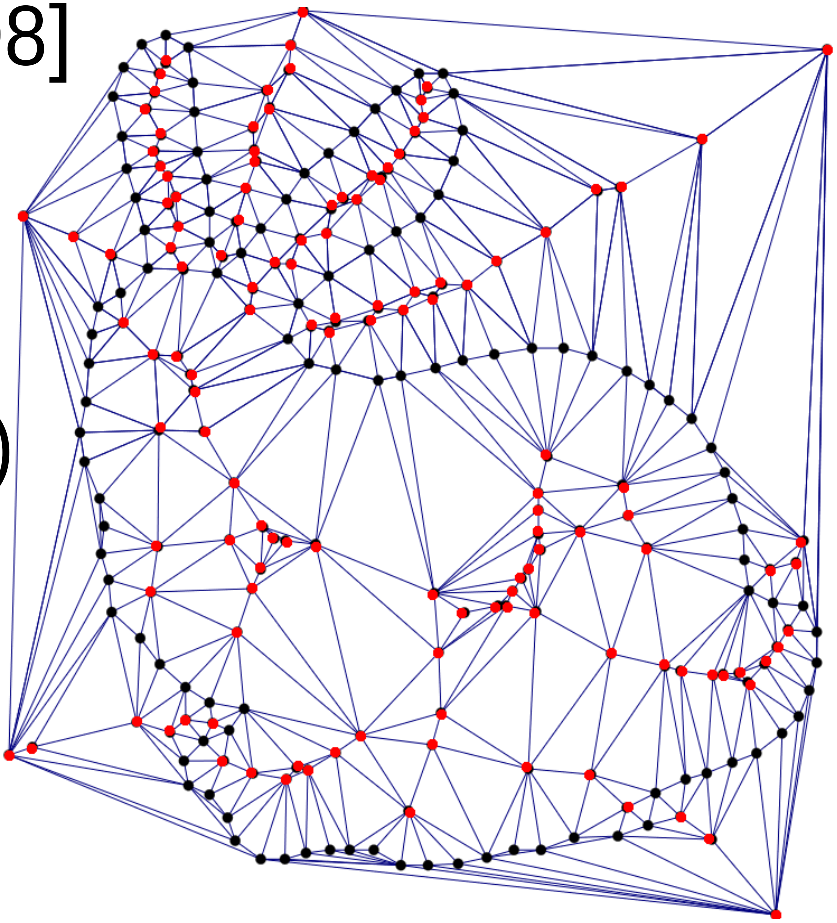
1)Compute Del(X)
2)Compute the poles



1. Reconstruction

The “crust” [Amenta et.al 98]

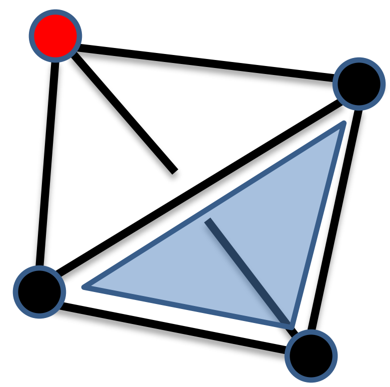
1)Compute Del(X)
2)Compute the poles
3)Compute Del(X U {poles})



1. Reconstruction

The “crust” [Amenta et.al 98]

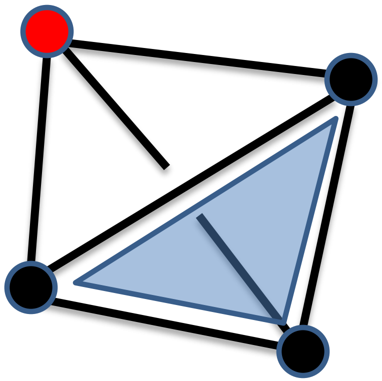
1)Compute Del(X)
2)Compute the poles
3)Compute Del(X U {poles})
4)Extract the triangles



1. Reconstruction

The “crust” [Amenta et.al 98]

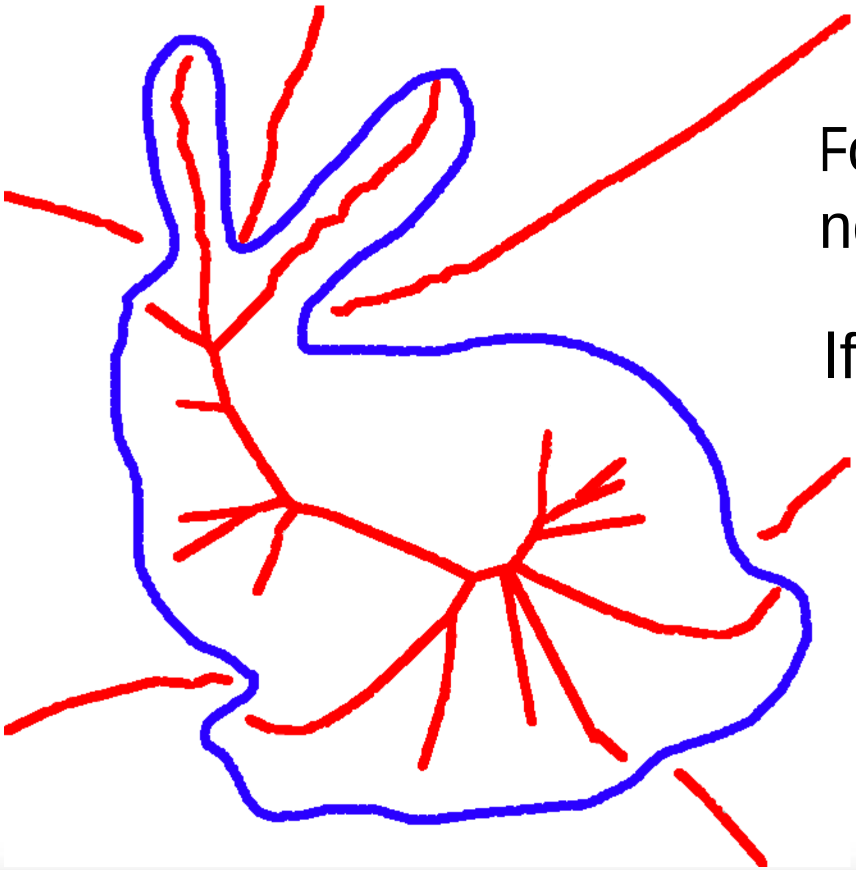
1)Compute Del(X)
2)Compute the poles
3)Compute Del(X U {poles})
4)Extract the triangles



See also the “co-cone” method [Amenta, Bern, Dey]

1. Reconstruction

Validity of "co-cone" and "crust": if X is an ε-sampling of S, with $\varepsilon < 0.1$

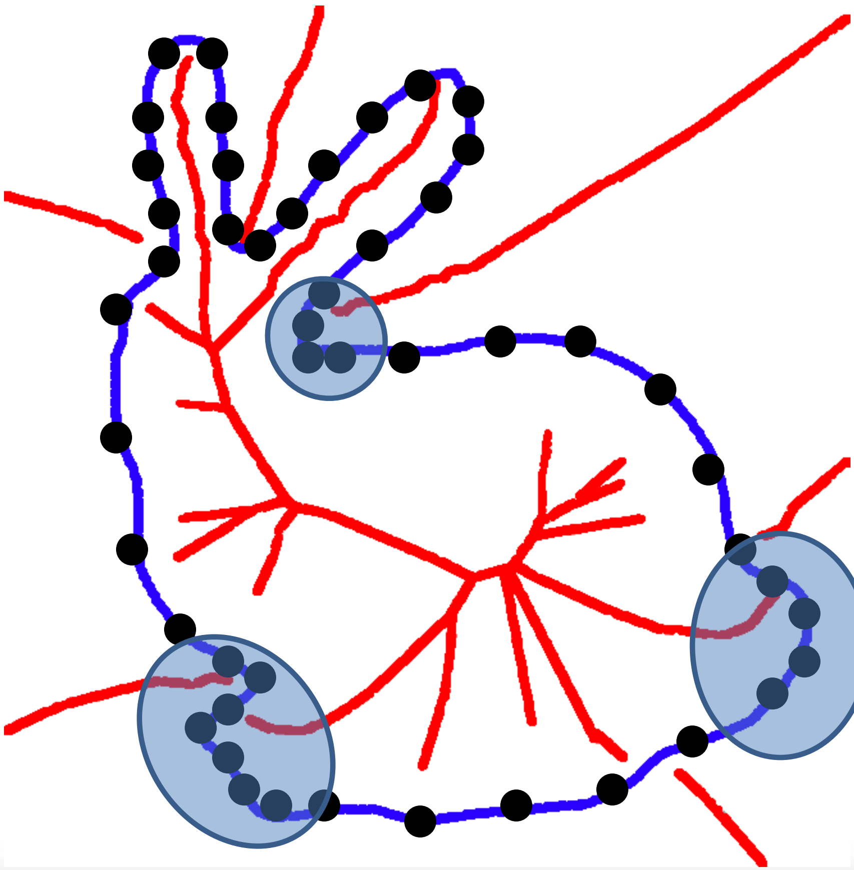


For all point p of S, there is a point x in X nearer to p than ε lfs(p)

lfs(p) = d(p, medial axis)

1. Reconstruction

Validity of “co-cone” and “crust”: if X is an ε-sampling of S, with ε < 0.1

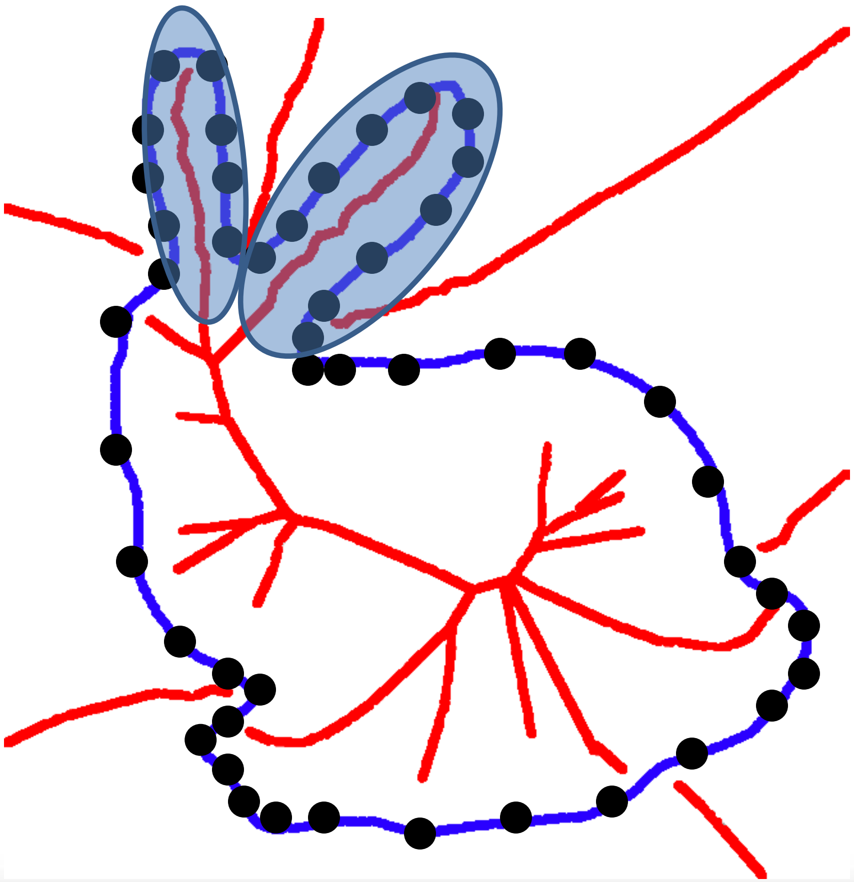


Sampling density proportional to 1/lfs

lfs measures curvature and thickness

1. Reconstruction

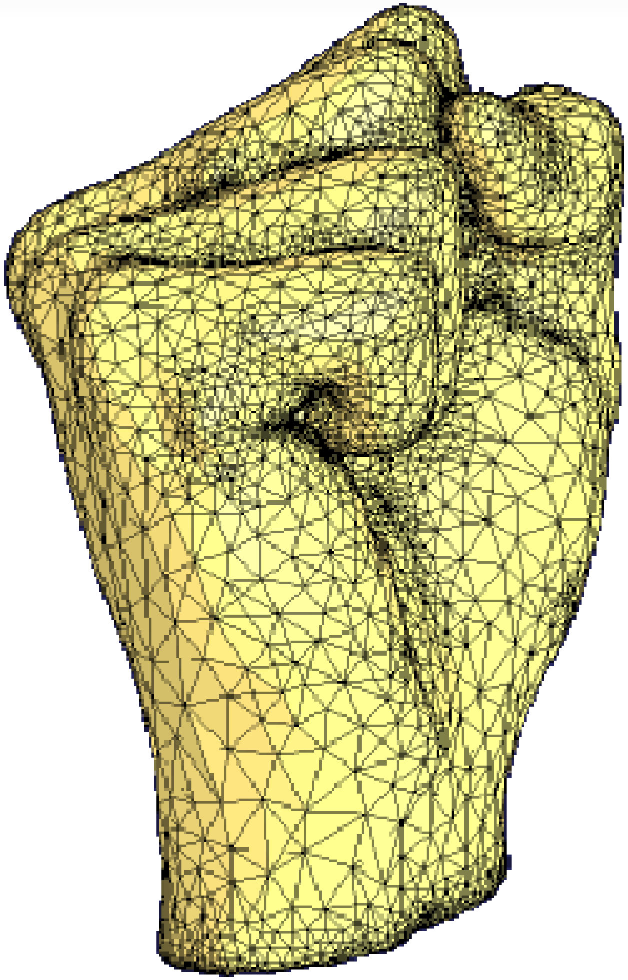
Validity of "co-cone" and "crust": if X is an ε-sampling of S, with ε < 0.1



Sampling density proportional to 1/lfs

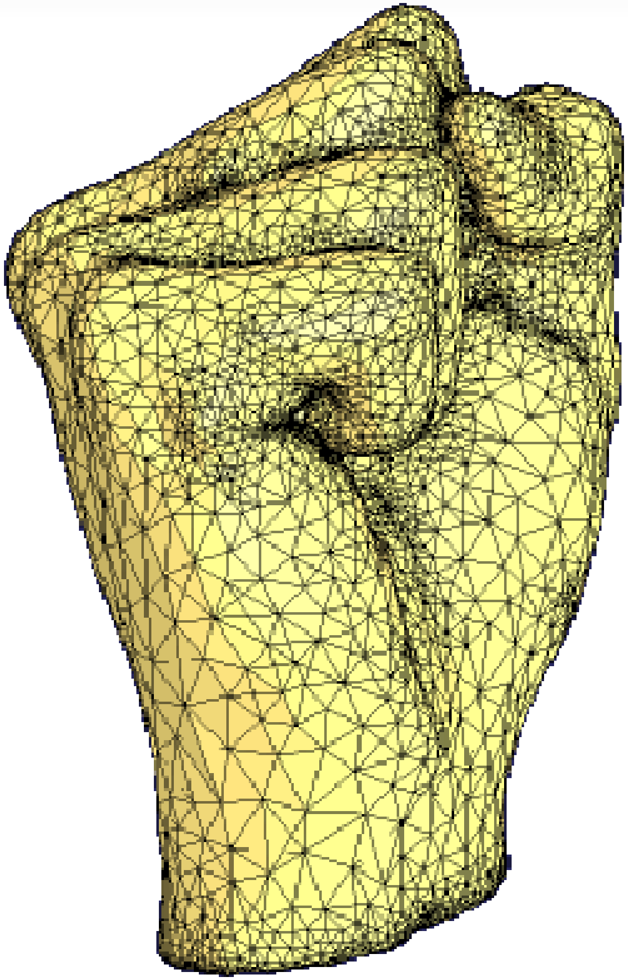
lfs measures curvature and thickness

1. Reconstruction



The “Crust” [Amenta, Bern, Kamvysselis 98]

1. Reconstruction

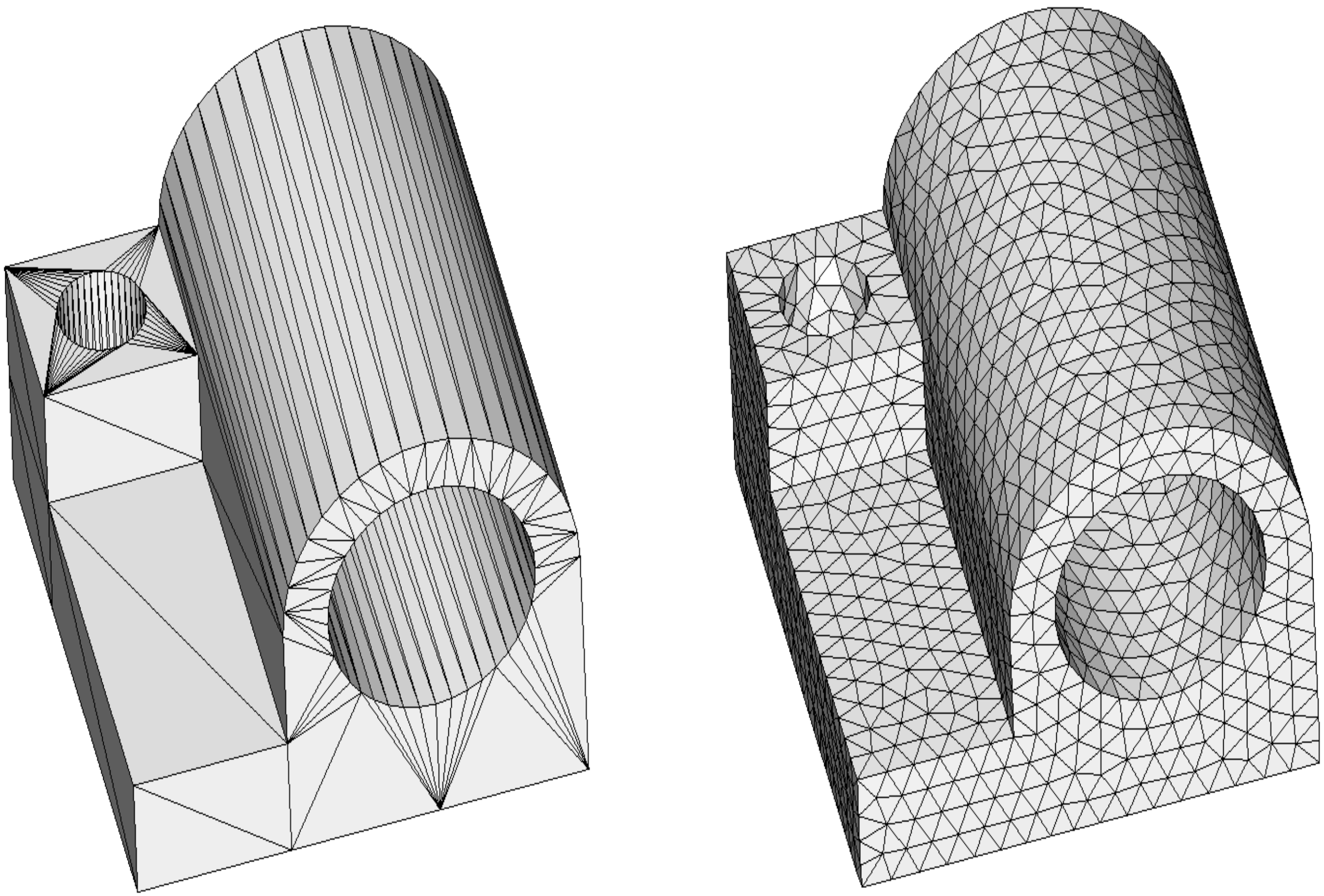


The “Crust” [Amenta, Bern, Kamvysselis 98]

Further reading:

- The “co-cone” family of methods
  (See Nina Amenta, Tamal Dey’s website)
- The “eigencrust” [Kolluri and Shewchuk]
- Boissonnat and Yvinec, Computational Geometry
- Polygon Mesh Processing

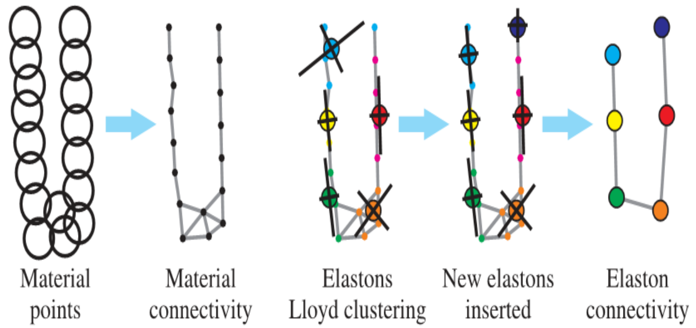
2. Remeshing…



(Re)-meshing [Du et.al], [Alliez et.al - SIGGRAPH], [Yan et.al - SGP]

2. Remeshing…

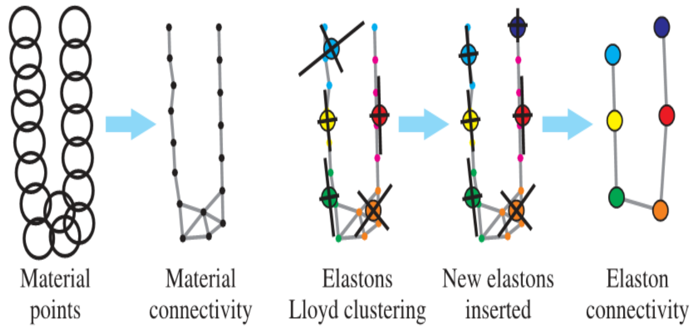
… and more



Deformations – Elastons sampling
[Martin et.al, SIGGRAPH2010]

2. Remeshing…

… and more

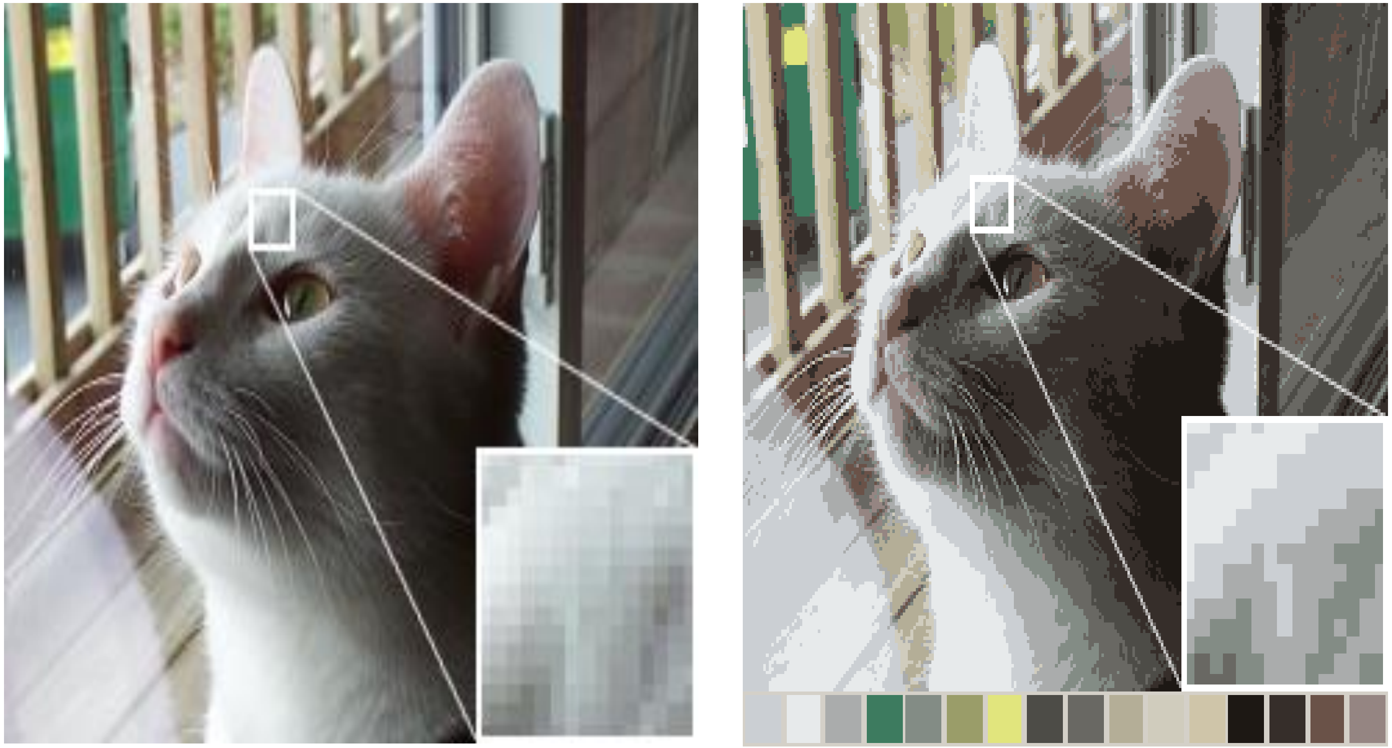


Deformations – Elastons sampling
[Martin et.al, SIGGRAPH2010]

Fluids – Free surface sampling
[Bridson et.al, SIGGRAPH2010]
[Wotjan et.al, SIGGRAPH2010]

2. Remeshing…

… and more

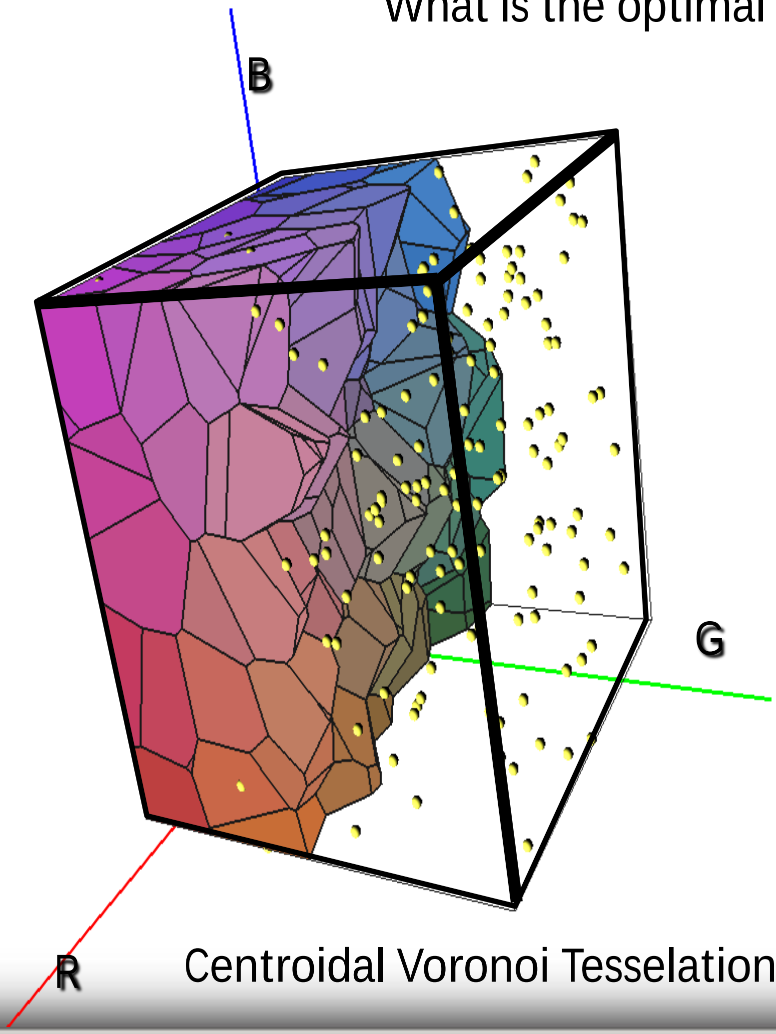


Color quantization
[Leung et.al, GPU Pro, AK Peters, 2010]

2. Remeshing…

Centroidal Voronoi Tesselation

What is the optimal colormap ?

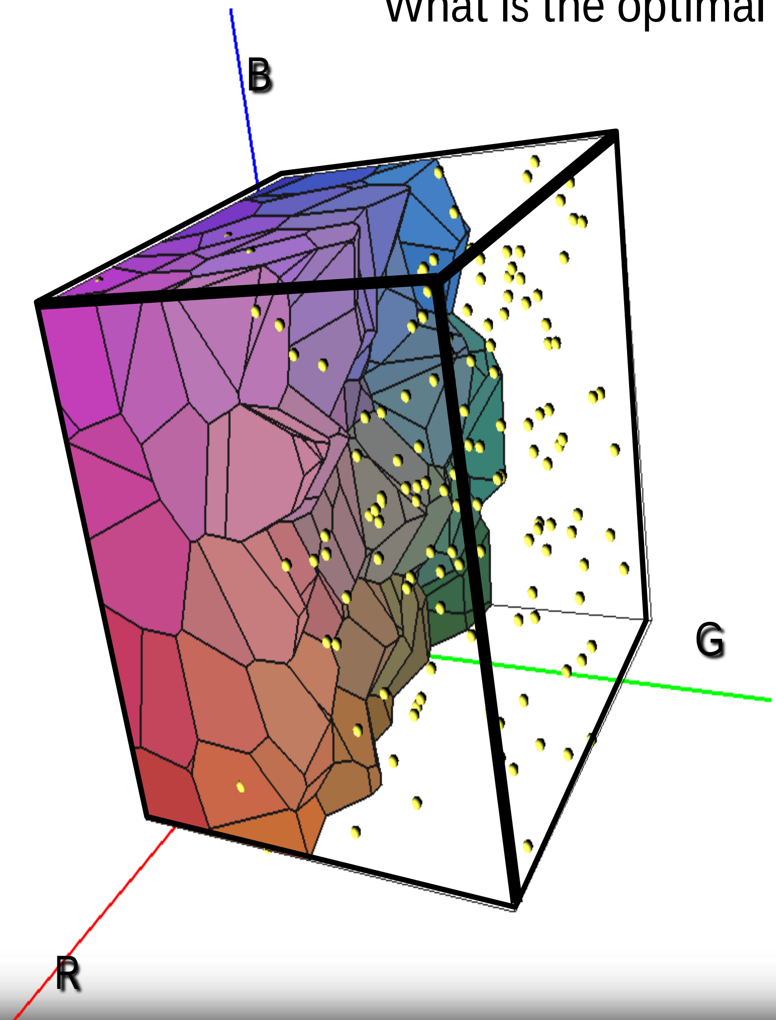


Centroidal Voronoi Tesselation from the information theory perspective…

2. Remeshing…

Centroidal Voronoi Tesselation

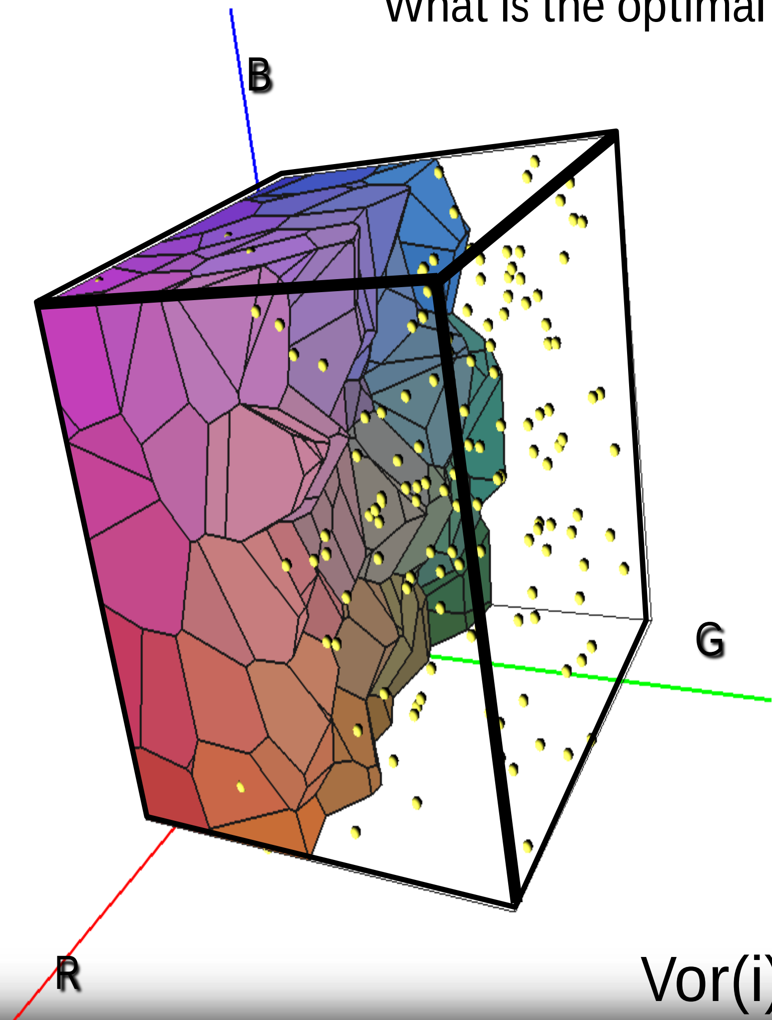
What is the optimal colormap ?



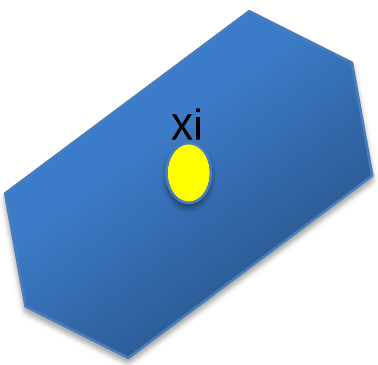
xi = (ri,gi,bi) Colormap entry

2. Remeshing...
Centroidal Voronoi Tesselation

What is the optimal colormap ?



$x_i = (r_i, g_i, b_i)$ Colormap entry

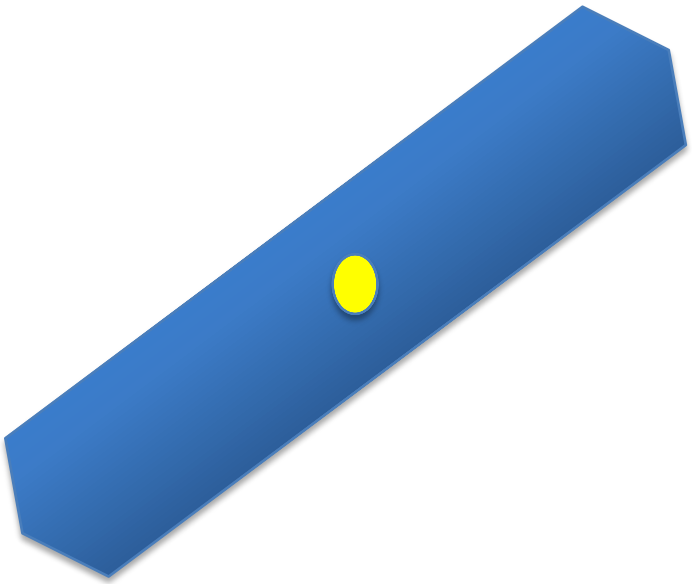


$Vor(i) = \{ x \mid d(x, x_i) < d(x, x_j) \} \; \forall i \neq j$

2. Remeshing…

Centroidal Voronoi Tesselation

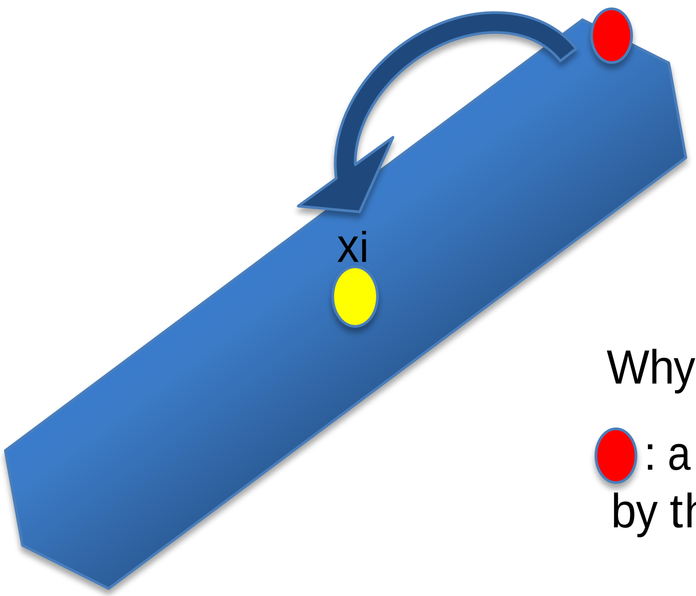
What is the optimal colormap ?



A « bad » colormap entry / Voronoi cell

2. Remeshing…
Centroidal Voronoi Tesselation

What is the optimal colormap ?



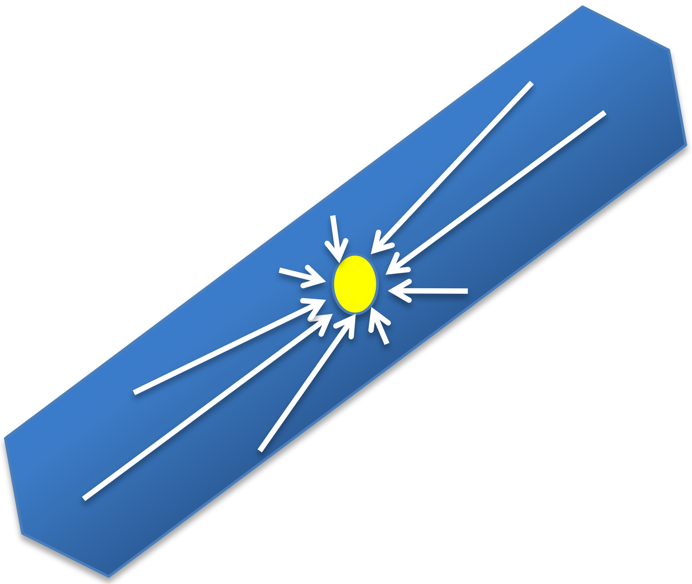
Why bad ? Because Vor(xi) contains

: a color poorly approximated by the colormap entry xi

A « bad » colormap entry / Voronoi cell

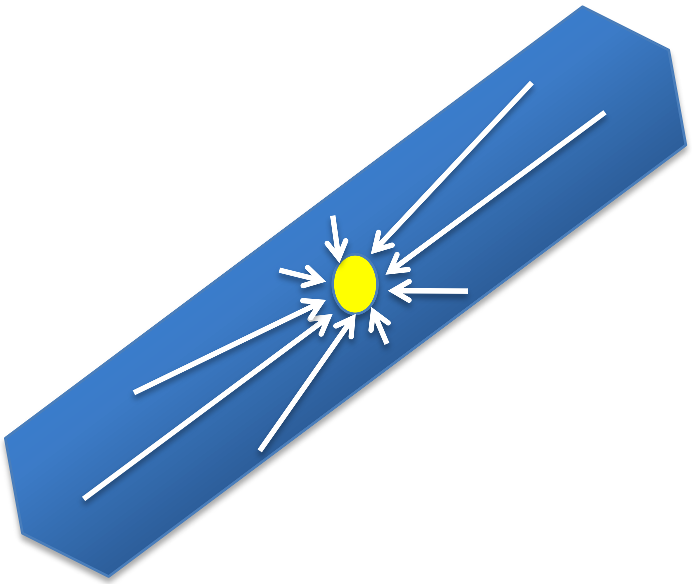
2. Remeshing…
Centroidal Voronoi Tesselation

What is the optimal colormap ?



2. Remeshing…
Centroidal Voronoi Tesselation

What is the optimal colormap ?

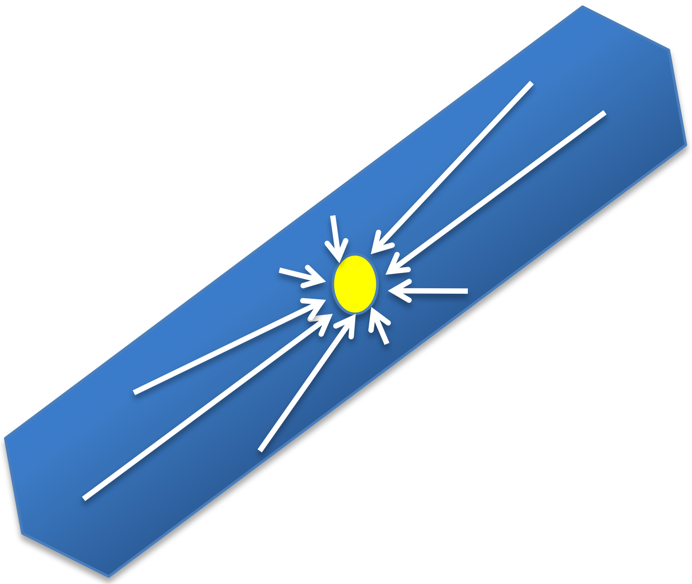


$$F = \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$

2. Remeshing…

Centroidal Voronoi Tesselation

What is the optimal colormap ?



Quantization noise power

$$F = \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$

2. Remeshing…

Centroidal Voronoi Tesselation

What is the optimal colormap ?

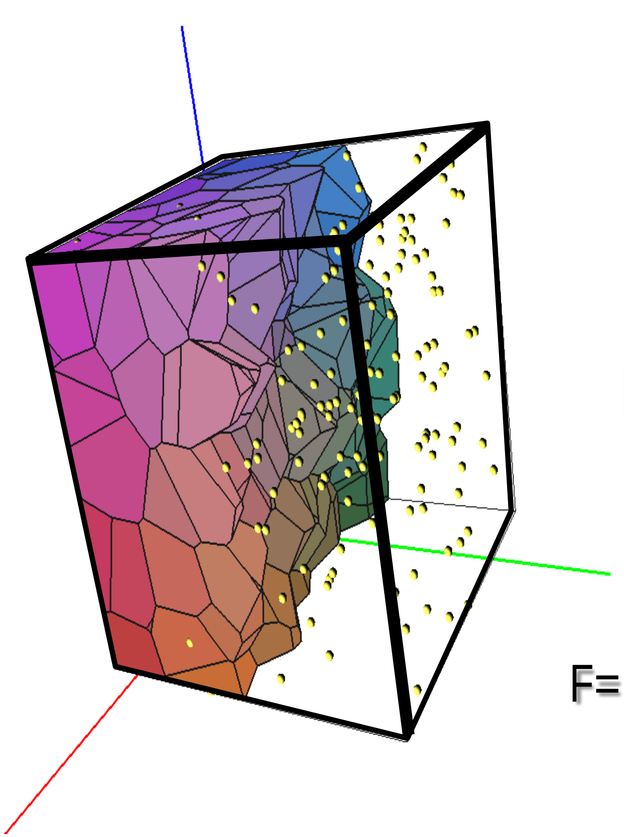
Quantization noise power

$$F = \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$

2. Remeshing…

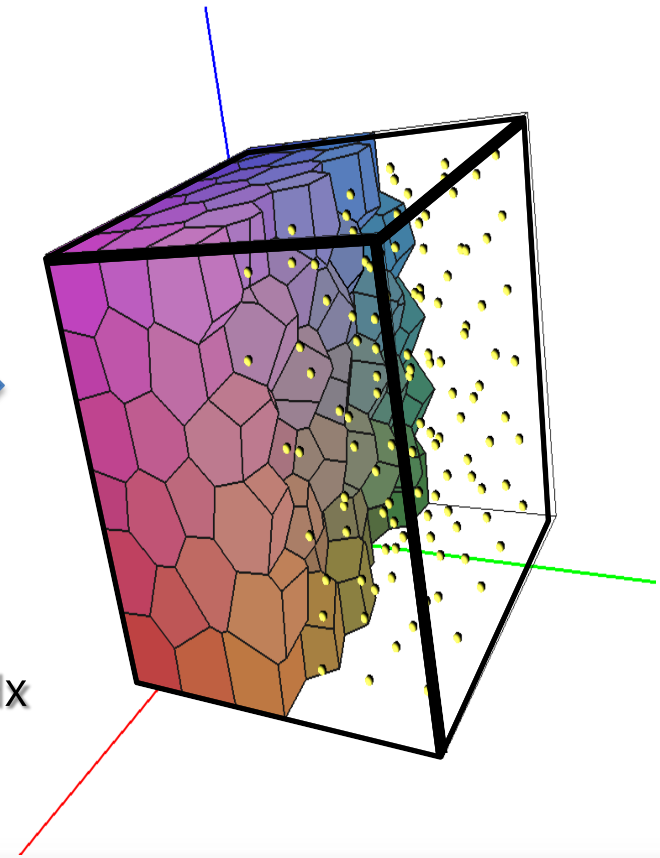
Centroidal Voronoi Tesselation

What is the optimal colormap ?



Minimize

$$F = \sum_i \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$



2. Remeshing…

Centroidal Voronoi Tesselation

The classical method:

Lloyd’s algorithm = gradient descent

$$F = \sum_i \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$

Centroidal Voronoi Tesselation

The classical method:

Lloyd's algorithm = gradient descent

$$F = \sum_i \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$

$$\nabla F|_{xi} = 2\, mi\, (xi - gi)$$

[Iri et.al], [Du et.al]

Volume of Vor(i)   Centroid of Vor(i)

2. Remeshing…

Centroidal Voronoi Tesselation

The classical method:

Lloyd's algorithm = gradient descent

$$F = \sum_i \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$

$$\nabla F|_{xi} = 2\, \underbrace{mi}_{\text{Volume of Vor(i)}}\, (xi - \underbrace{gi}_{\text{Centroid of Vor(i)}})$$

[Iri et.al], [Du et.al]

If xi coincides with the centroid of Vor(i), we got a stationary point of F (therefore a « good sampling »)

2. Remeshing…

Centroidal Voronoi Tesselation

The classical method:

Lloyd's algorithm = gradient descent

$$F = \sum_i \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$

$$\nabla F|_{xi} = 2\, \underbrace{mi}_{\text{Volume of Vor(i)}}\, (xi - \underbrace{gi}_{\text{Centroid of Vor(i)}})$$

[Iri et.al], [Du et.al]

If xi coincides with the centroid of Vor(i), we got a stationary point of F (therefore a « good sampling ») → Vor(X): Centroidal Voronoi Tesselation

2. Remeshing…

Disclaimer:

This presentations contains live demos.
Crashes may occur.
The presenter assumes no liability.

2. Remeshing…

2. Remeshing…

Lloyd's Relaxation:

(Geometric point of view)

Loop

Move the x_i's to the g_i's

Re-triangulate

End loop

+ Provably decreases F [Du et.al]
+ Reasonably easy to implement
- Slow (linear) convergence

2. Remeshing…

Theorem: F is C^2 almost everywhere

[Liu, Wang, L, Sun, Yan, Lu and Yang 09]

2. Remeshing…

<u>Theorem:</u> F is C^2 almost everywhere

[Liu, Wang, L, Sun, Yan, Lu and Yang 09]

What's the point ?

2. Remeshing…

Theorem: F is C^2 almost everywhere

[Liu, Wang, L, Sun, Yan, Lu and Yang 09]

What's the point ?

- Faster CVT algorithm (Newton)
- More general (quads)

2. Remeshing…

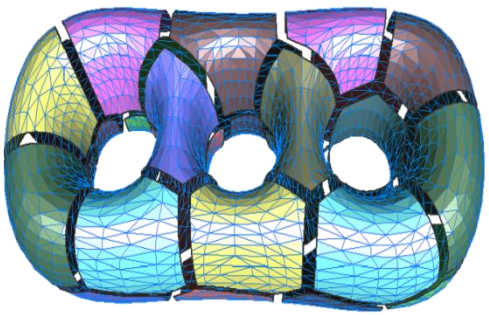
CVT in 2D

2. Remeshing…

2. Remeshing…

CVT in 2D

CVT on surfaces



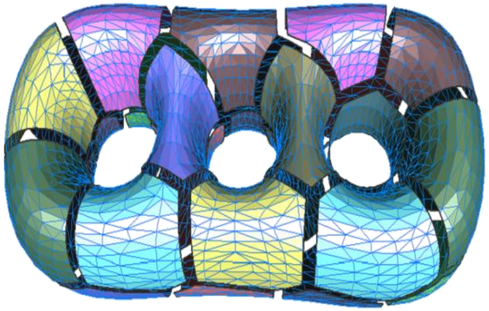
Constrained CVD [Qiang Du et.al 1999]
[Yan, L, Liu, Sun and Wang SGP 2009]

2. Remeshing…

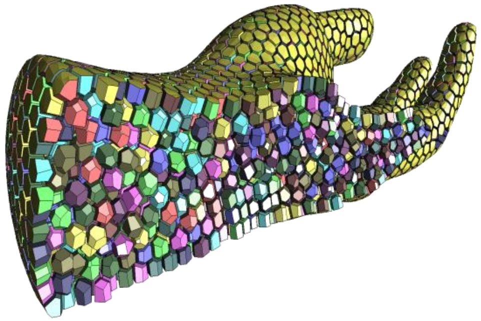
2. Remeshing…

CVT in 2D

CVT on surfaces



CVT in volumes



[Yan, Wang, L, Liu]

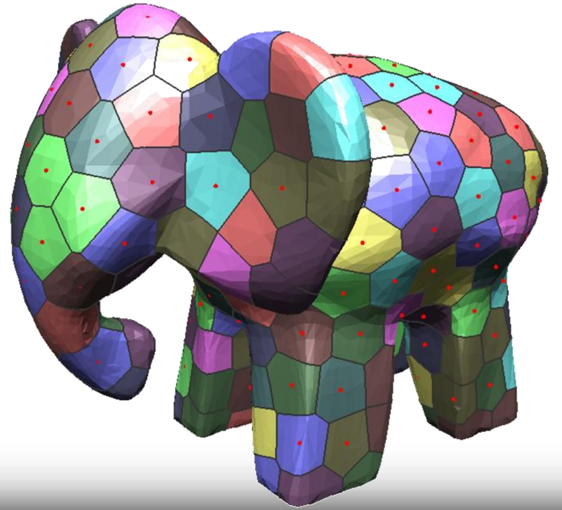
2. Remeshing…

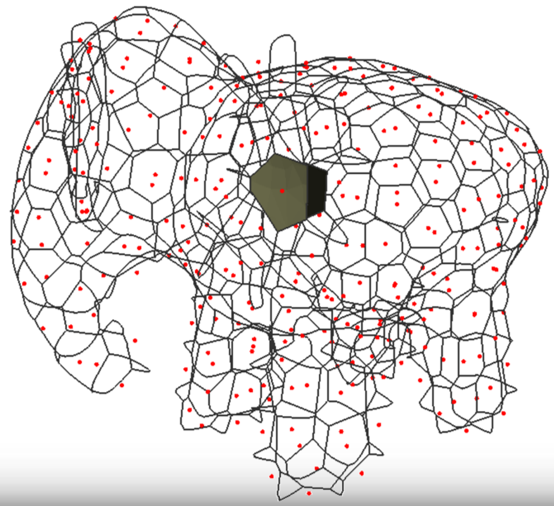
2. Remeshing…

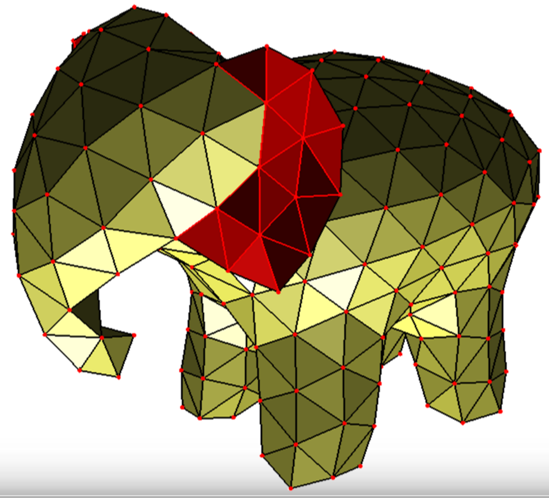
Validity of Restricted Voronoi Diagram / Restricted Delaunay Triangulation

Theorem : [Edelsbrunner & Shah 1997]

Topological Ball Property: if each face of the Restricted Voronoi Diagram is homeomorphic to a disc, then the Restricted Delaunay Triangulation is homeomorphic to the surface S







2. Remeshing…

2. Remeshing…

Validity of Restricted Voronoi Diagram / Restricted Delaunay Triangulation

Theorem : [Edelsbrunner & Shah 1997]

Topological Ball Property: if each k-face of the Restricted Voronoi Diagram is homeomorphic to a disc, then the Restricted Delaunay Triangulation is homeomorphic to the surface S

Theorem: [Amenta & Bern 1999]

if X is an ε-sampling of S, with $\varepsilon < 0.1$, then the Restricted Delaunay triangulation is homeomorphic to the surface S

2. Remeshing…

"take home" message

Def. 1: CVT is what you obtain when moving the points to the centroids of their Voronoi cells

Tells you "how to compute it"

"take home" message

Def. 1: CVT is what you obtain when moving the points to the centroids of their Voronoi cells

Tells you "how to compute it"

Def. 2: CVT is what you obtain by minimizing the quantization noise power

$$F = \sum_i \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$

Tells you "what it is" !

2. Remeshing…

"take home" message

Def. 1: CVT is what you obtain when moving the points to the centroids of their Voronoi cells

Tells you "how to compute it"

Def. 2: CVT is what you obtain by minimizing the quantization noise power

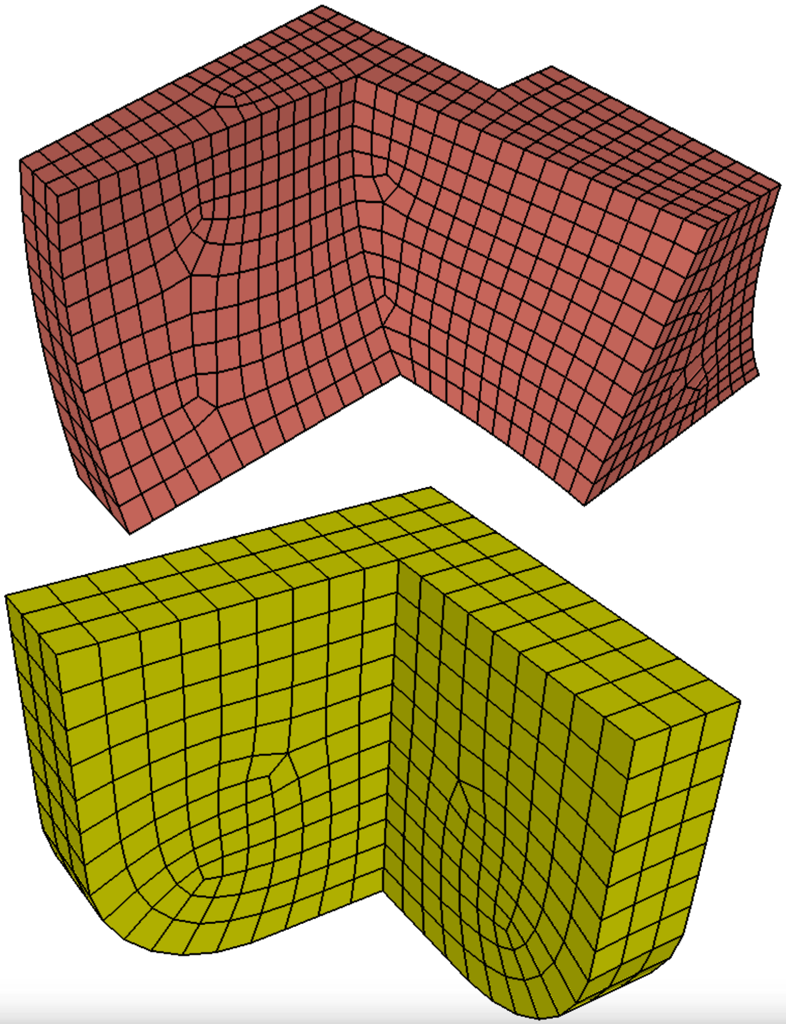
$$F = \sum_i \int_{Vor(i)} \left\| x_i - x \right\|^2 dx$$

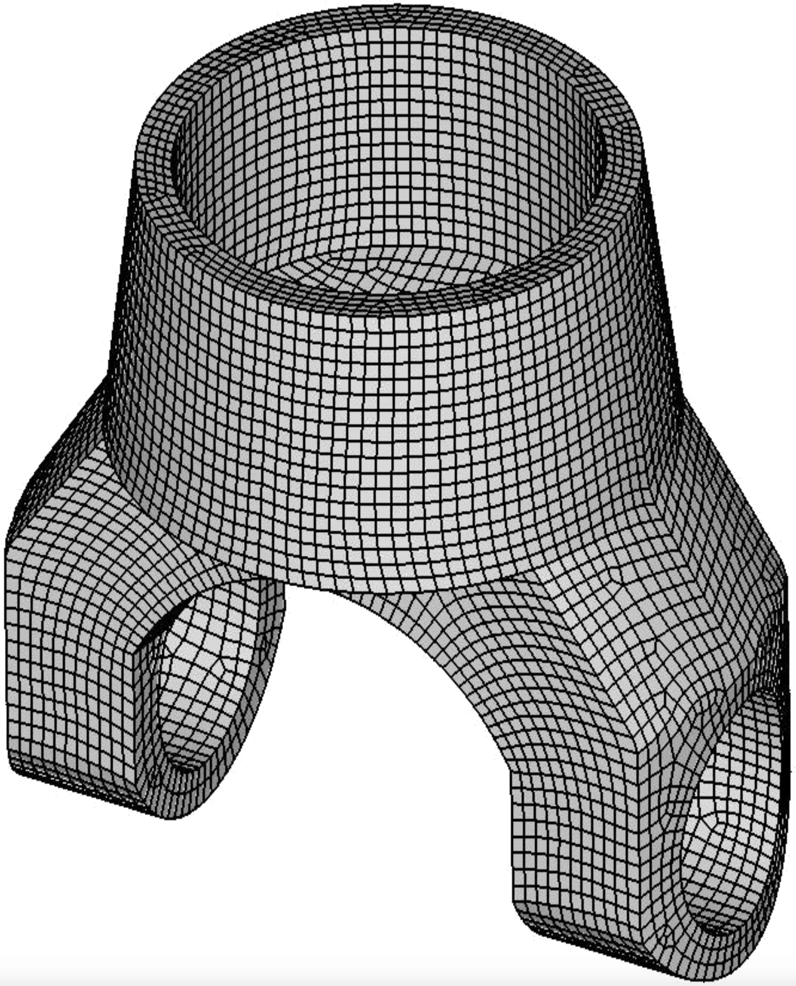
Tells you "what it is" !

- Faster solution mechanism
- Generalizations

2. Remeshing…

Quad and Hex meshing

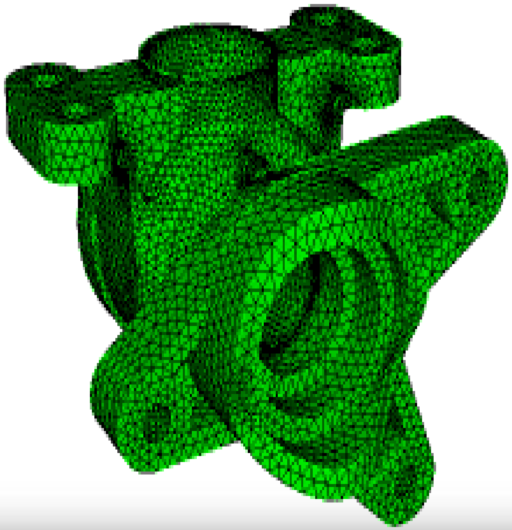


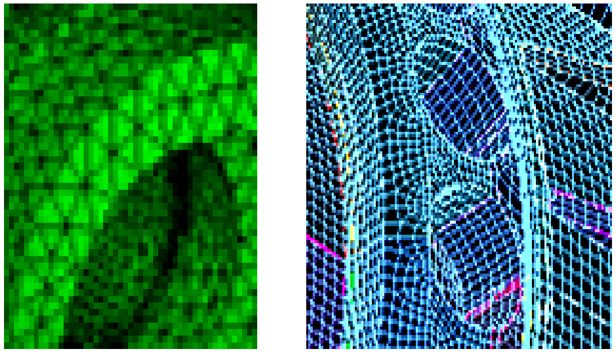


Motivations – Why Hexes ?

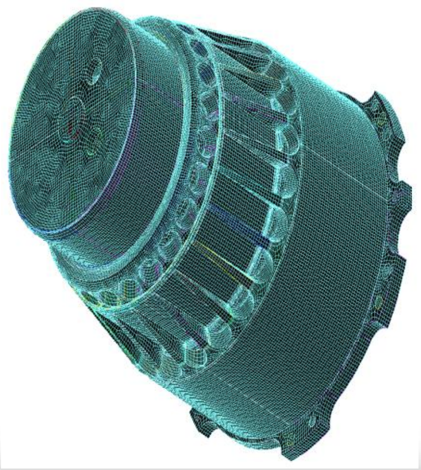
Tet Meshing
1. Fully Automated
2. Millions of elements in minutes/seconds
3. Adequate for some analysis
4. Inaccurate for other Analysis

Hex Meshing
1. Partially Automated, some Manual
2. Millions of elements in days/weeks/months
3. Preferred by some analysts for solution quality



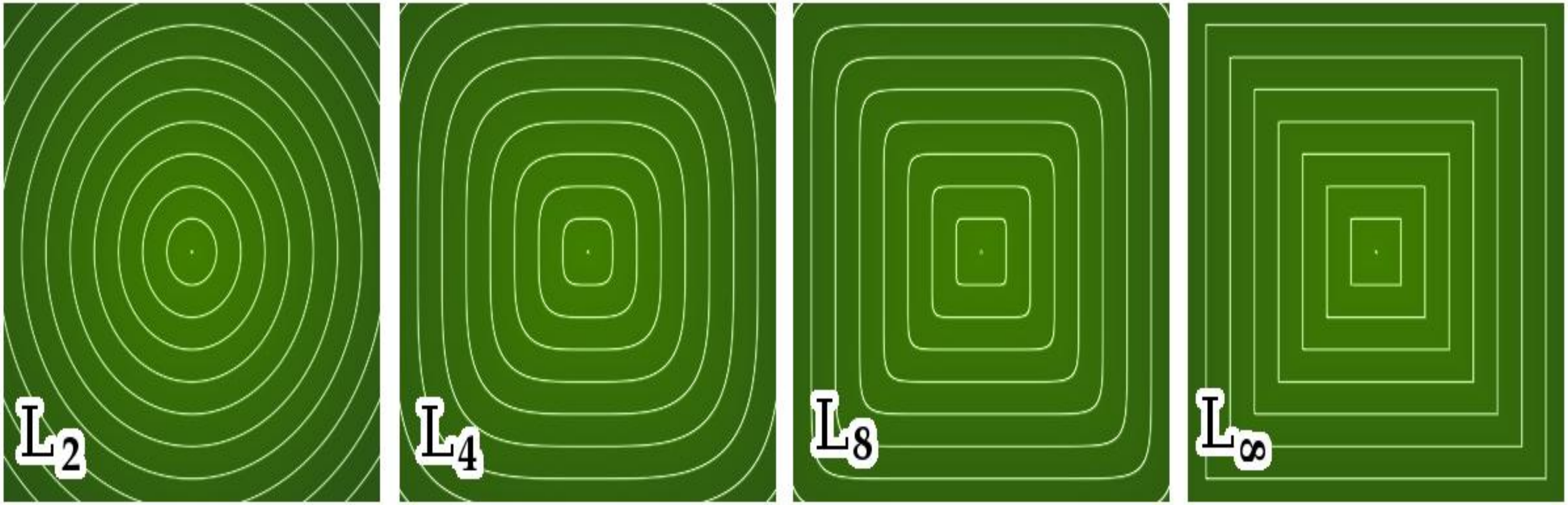


[Matt Staten] (Sandial Labs)



Blowing Square Bubbles

p=2 p=4 p=8 ....



2. Remeshing…

2. Remeshing…

Standard CVT: $F = \sum_i \int_{Vor(i)} \left\| (x_i - x) \right\|^2 dx$

2. Remeshing…

Standard CVT: $$F = \sum_i \int_{Vor(i)} \left\| (x_i - x) \right\|^2 dx$$

Lp CVT: $$F = \sum_i \int_{Vor(i)} \left\| M(x)\,(x_i - x) \right\|_p^p dx$$

[L and Liu 2010]

2. Remeshing…

Lp CVT: $$F = \sum_i \int_{Vor(i)} \left\| M(x)\,(x_i - x) \right\|_p^p \, dx$$

Anisotropy, encodes desired orientation
Riemannian metric $G = M^t M$

2. Remeshing…

Lp CVT:
$$F = \sum_i \int_{Vor(i)} \| M(x)\,(x_i - x) \|_p^p \, dx$$

Lp norm: $|| x ||_p = \sqrt[p]{|x|^p + |y|^p + |z|^p}$

2. Remeshing...

Lp CVT: $$F = \sum_i \int_{Vor(i)} \left\| M(x)\,(x_i - x) \right\|_p^p dx$$

Lp norm: $$\| x \|_p = \sqrt[p]{|x|^p + |y|^p + |z|^p}$$

If p is even: $$\| x \|_p^p = x^p + y^p + z^p$$

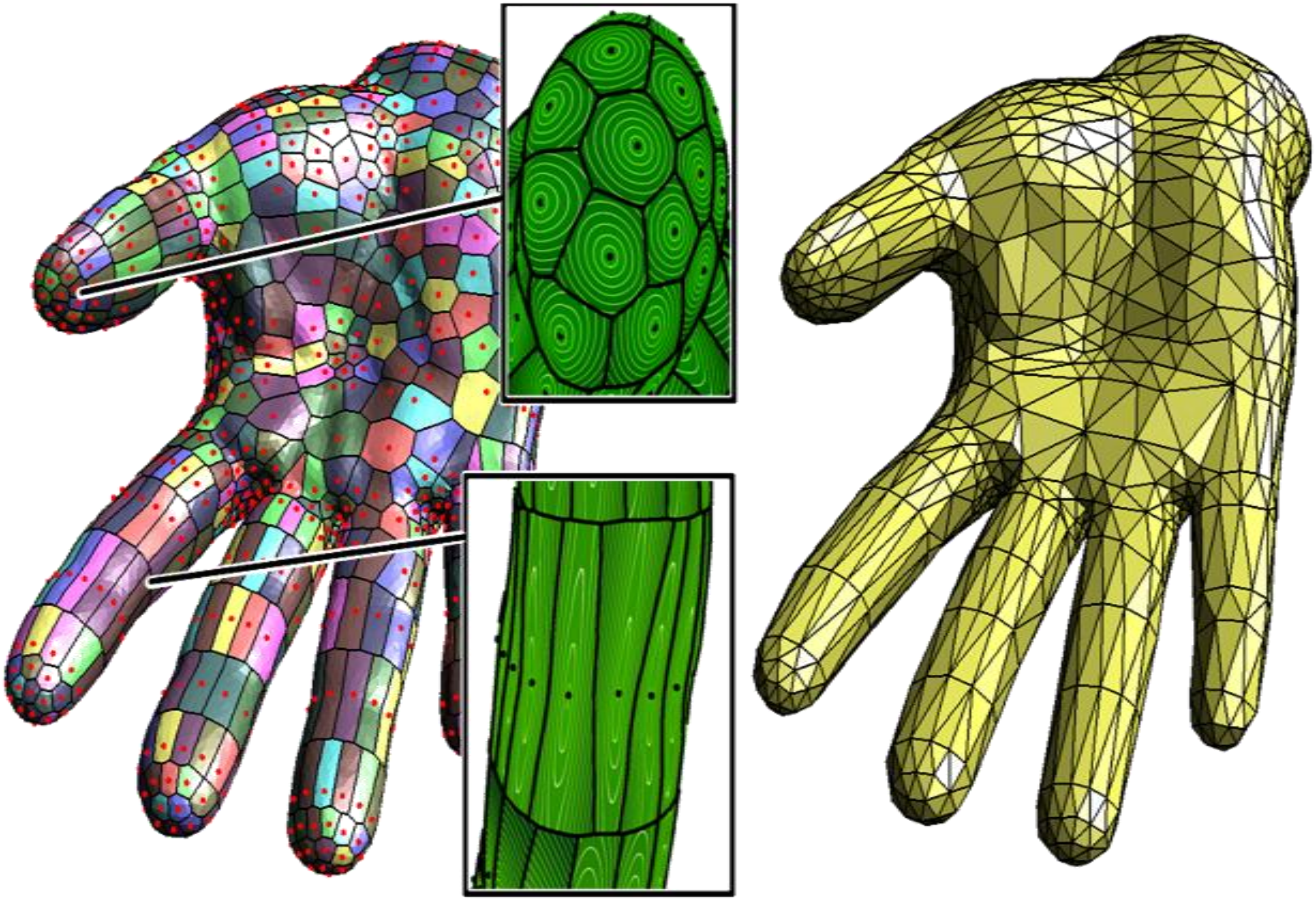
2. Remeshing…

Lp CVT: $$F = \sum_i \int_{Vor(i)} \left\| M(x)\,(x_i - x) \right\|_p^p \, dx$$

<u>Optimization with LBFGS (quasi-Newton)</u>

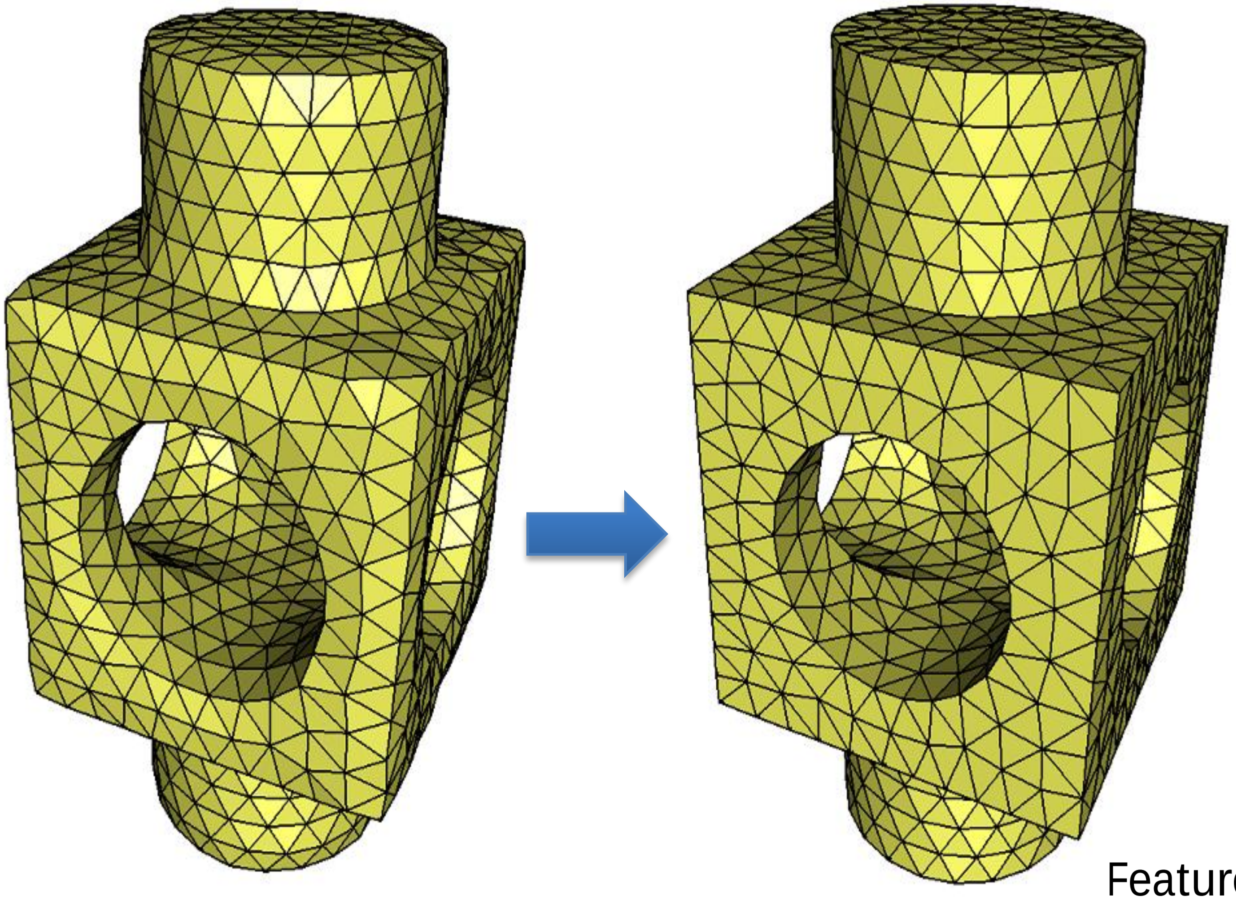
For each iterate $X^{(k)}$:
 Compute $F(X^{(k)})$ and $\nabla F(X^{(k)})$

2. Remeshing…

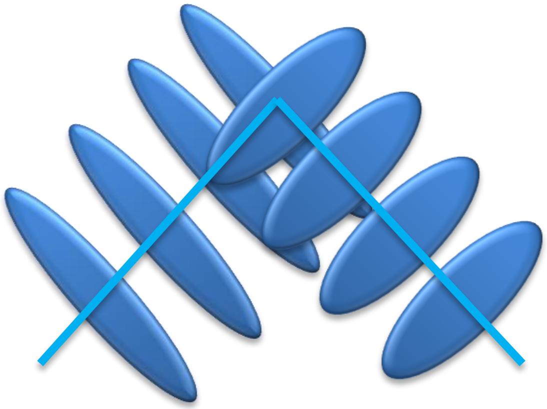


p = 2
M(x) = ppal dir.
of curvature.

2. Remeshing…

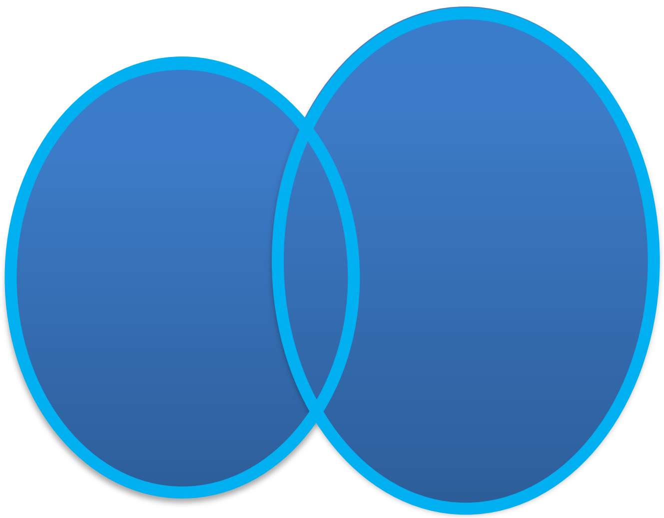


p = 2
M(x) = Normal
anisotropy.



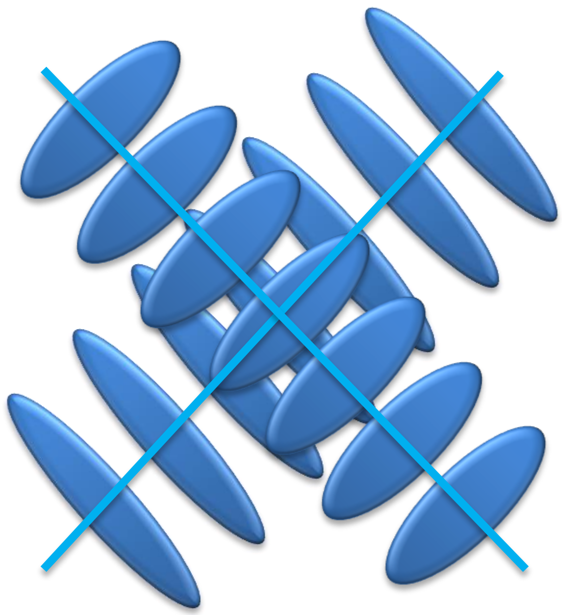
Feature-sensitive meshing

2. Remeshing…



CSG-Remeshing

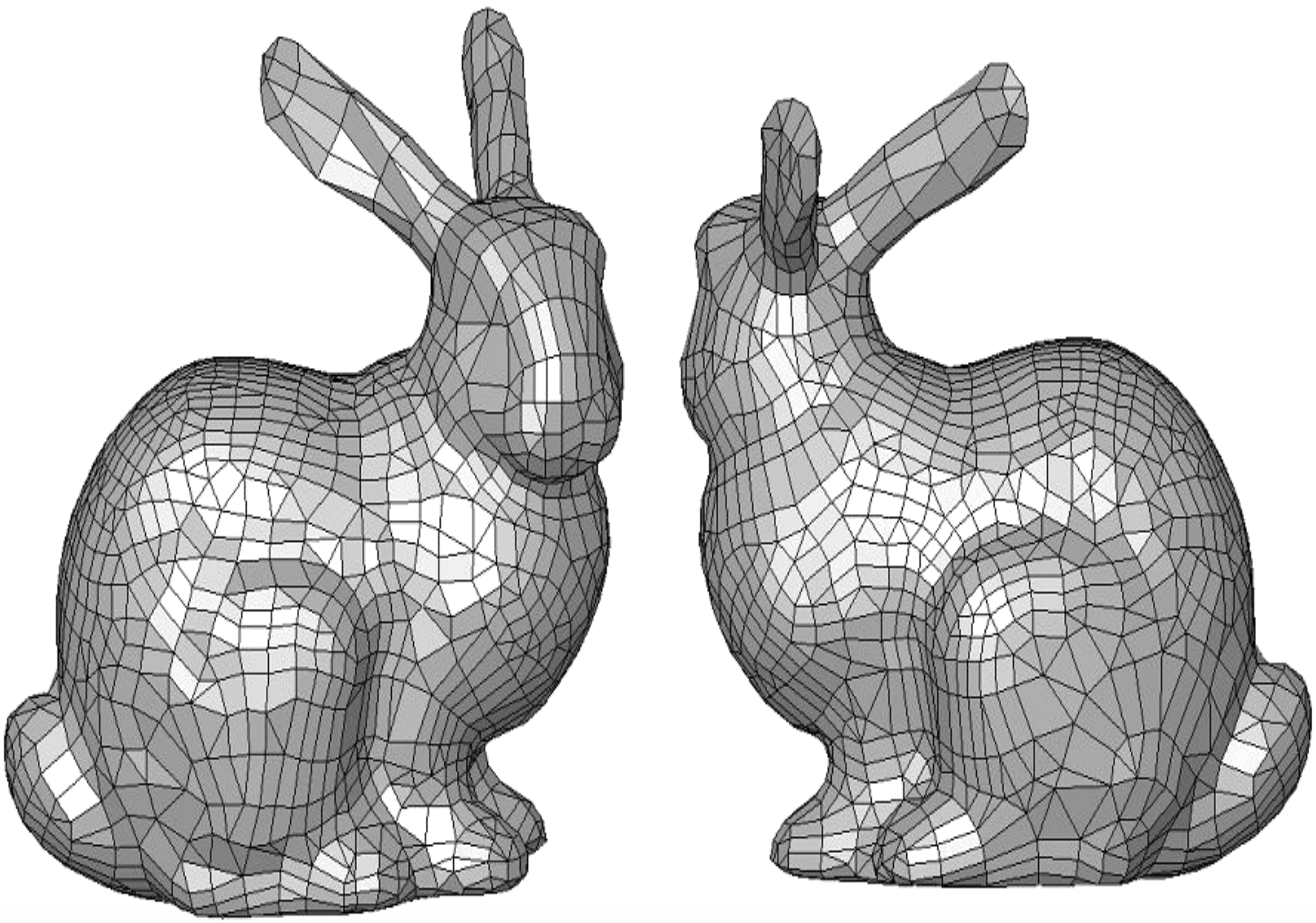
p = 2
M(x) = Normal anisotropy.



Feature-sensitive meshing

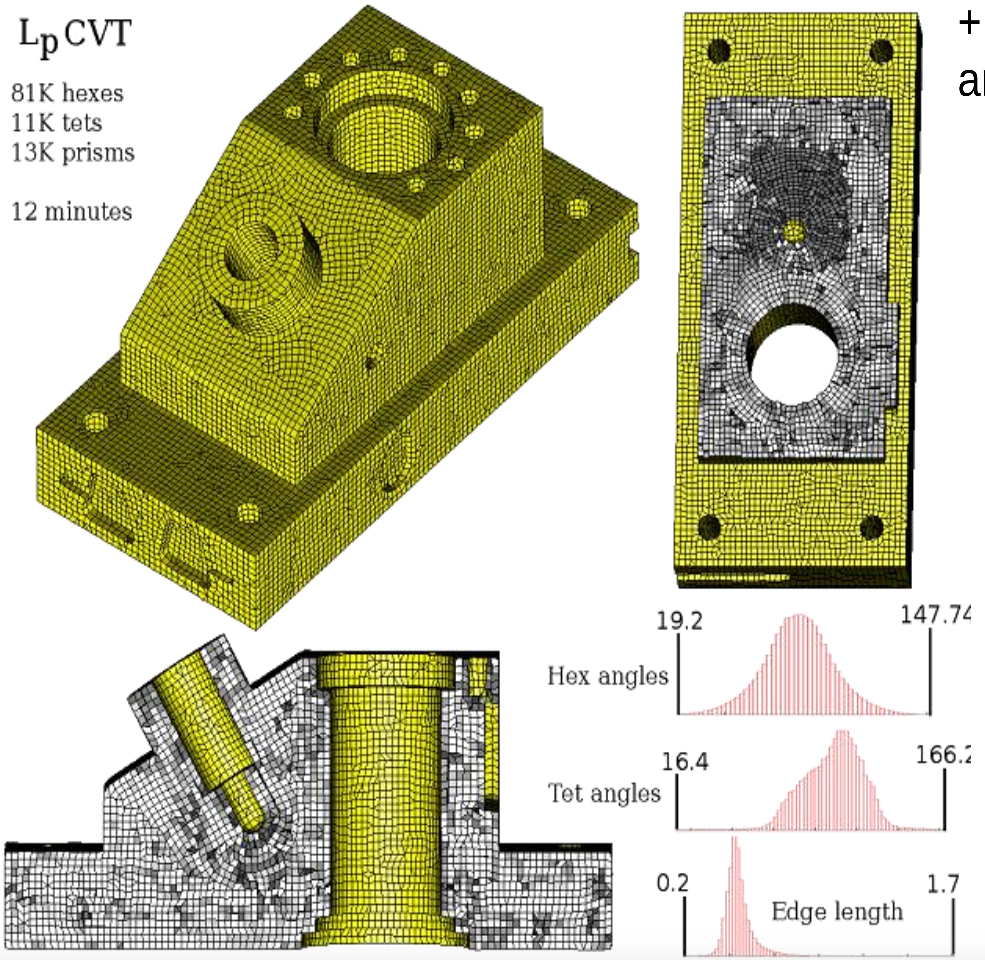
2. Remeshing…

2. Remeshing…

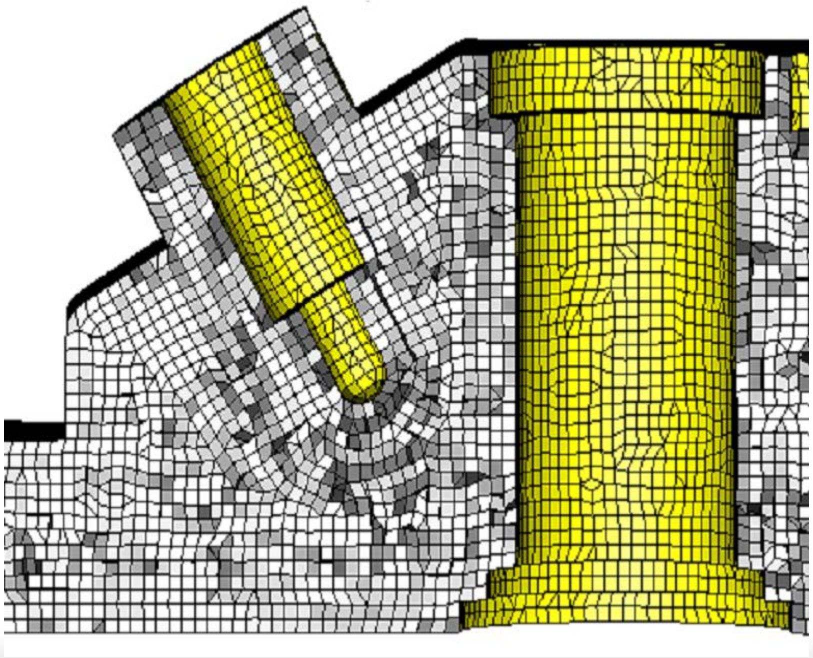


p = 8
M(x) = ppal dir. of curvature.

2. Remeshing…



+ many other examples in paper and supplemental material.



2. Remeshing…

2. Remeshing…

Further reading on (re)meshing

VD: Papers by Nina Amenta, Tamal Dey, Herbert Edelsbrunner
CVT: Papers by Qiang Du, Maria Emelianenko, Max Gunzberger

Space Tesselations, Okabe
Meshing ("Le maillage *facile")*, Paul-Louis George and Pascal Frey
Computational Geometry, Jean-Daniel Boissonnat and M. Yvinec
Polygon Mesh Processing Botsch, Kobbelt, Pauly, Alliez, L Chapter 6