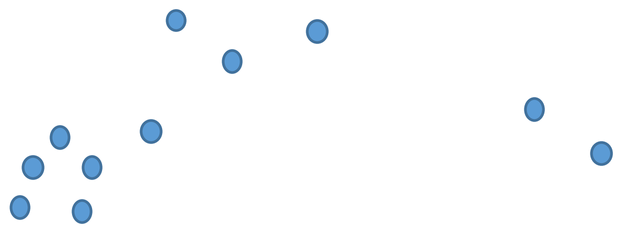
From Point Clouds to tessellated surfaces *explicit methods*

Paolo Cignoni,

Istituto di Scienza e Tecnologie dell'Informazione,
Consiglio Nazionale delle Ricerche

Problem Statement

Given a Point cloud $P = \{p_0, \dots, p_n\}, p_i \in \mathbb{R}^3$, find the mesh M that it *represents*

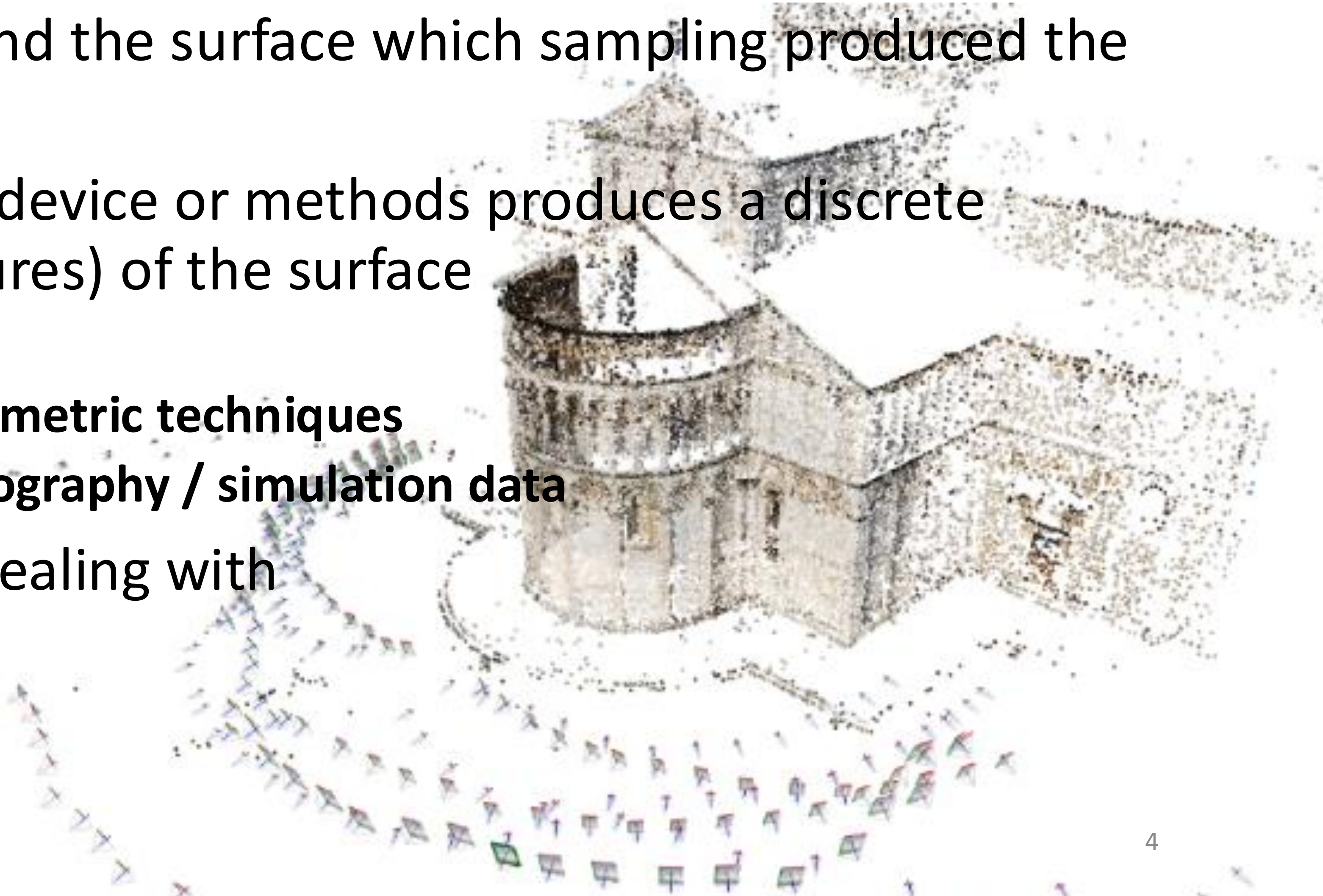


- Q1: It is a very ill posed problem, what does *represents* means?
- Q2: why do we care about this problem?

Motivations

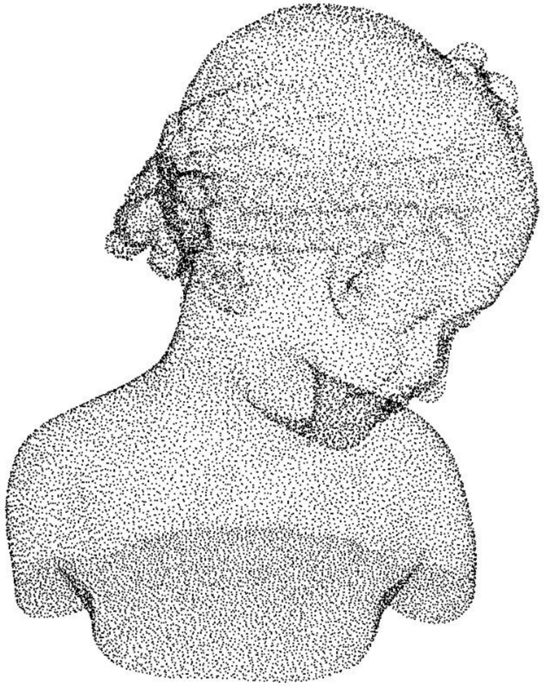
- A1: Ideally, we want to find the surface which sampling produced the input problem
- A2: Every 3D acquisition device or methods produces a discrete puntual sampling (measures) of the surface
  - **Laser scanning**
  - **Image based/photogrammetric techniques**
  - **Computerized Axial Tomography / simulation data**

... So that is what we are dealing with



Data sources

- Laser scanning with a turntable

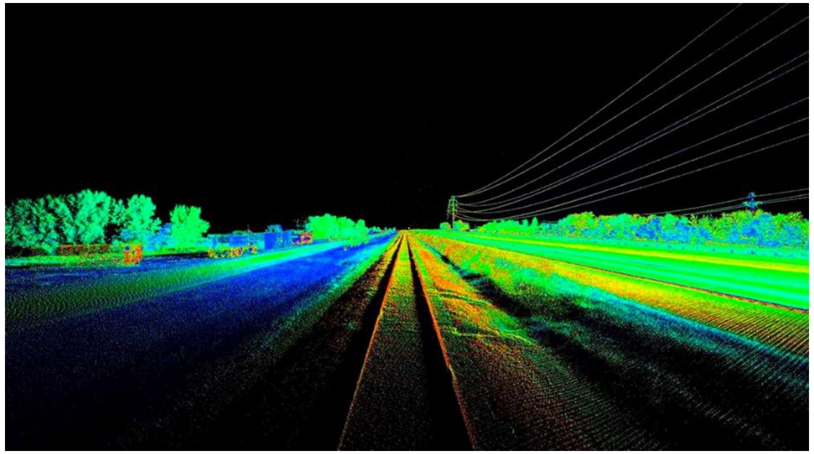


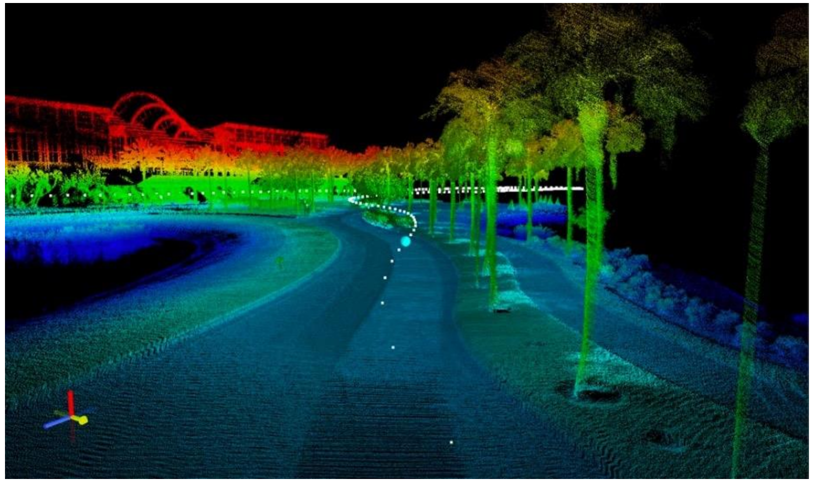
Data sources

- Laser scanning with static laser scanner (range of 100, 200... meters)

Data sources

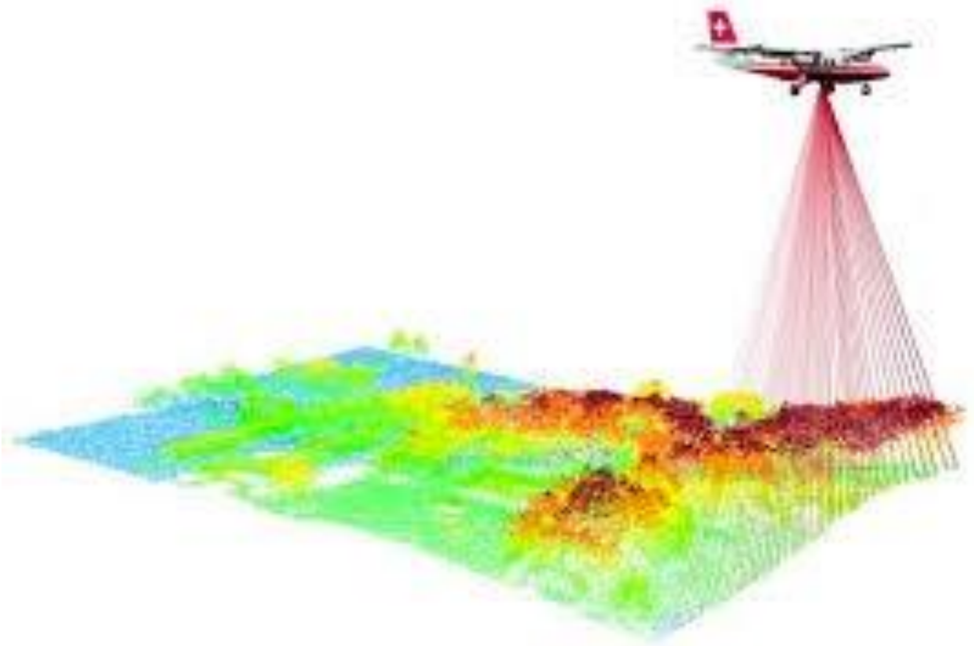
- Laser scanning – mobile scanners

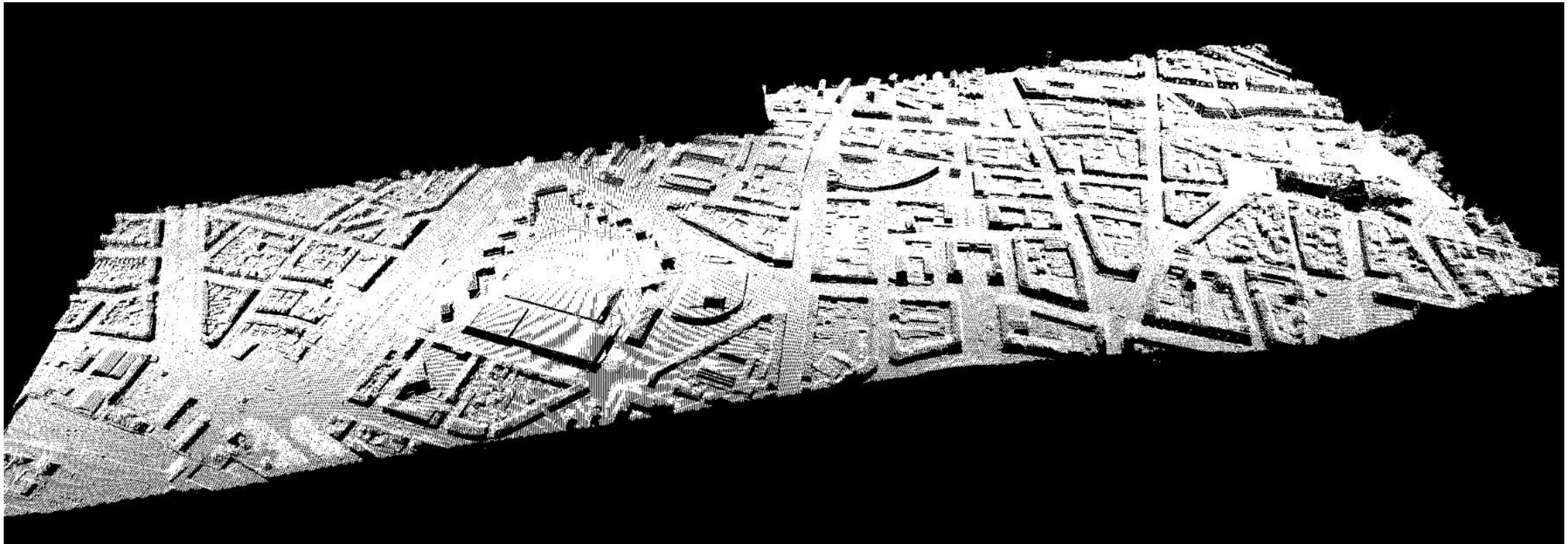




Data sources

- Laser scanning – airborne LiDAR





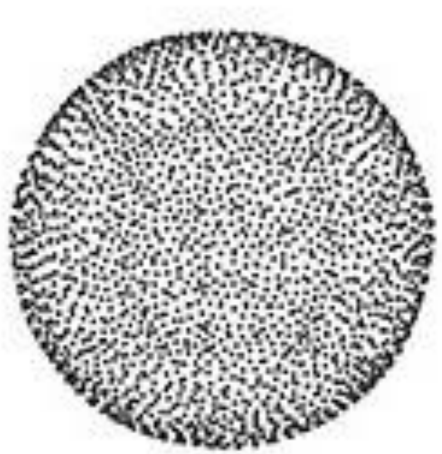
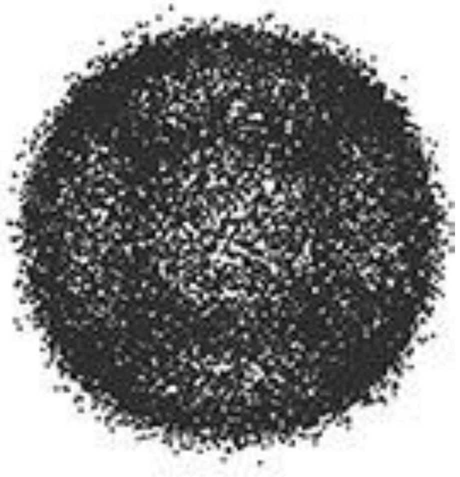
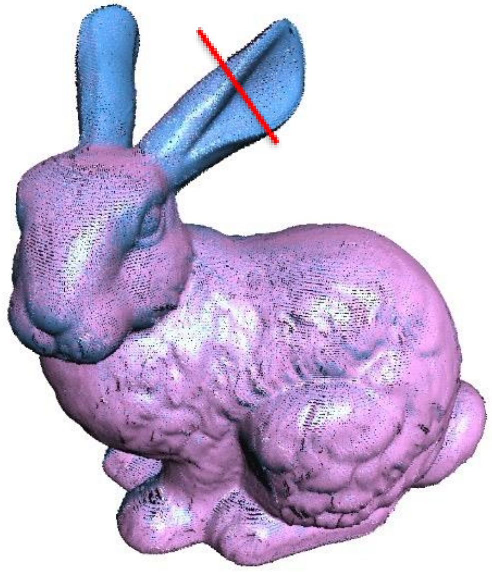
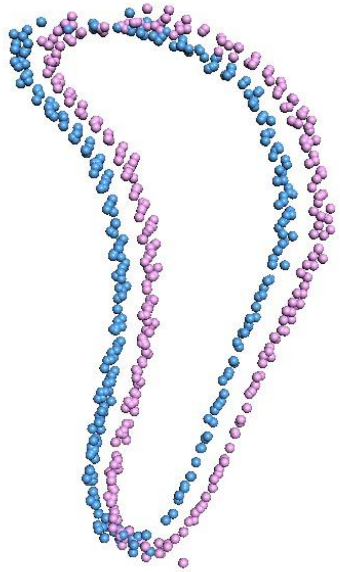
Data sources

- Structure from Motion (SfM) and Multi-view stereo (MVS)

Challenges

The positions and normals are generally noisy

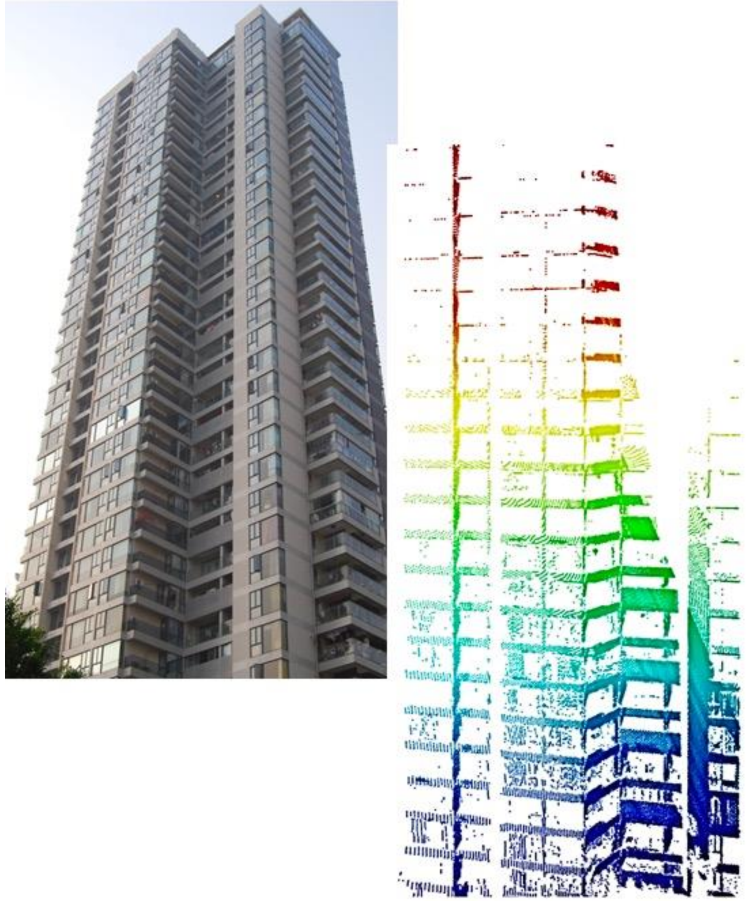
- Sampling inaccuracy
- Scan misregistration

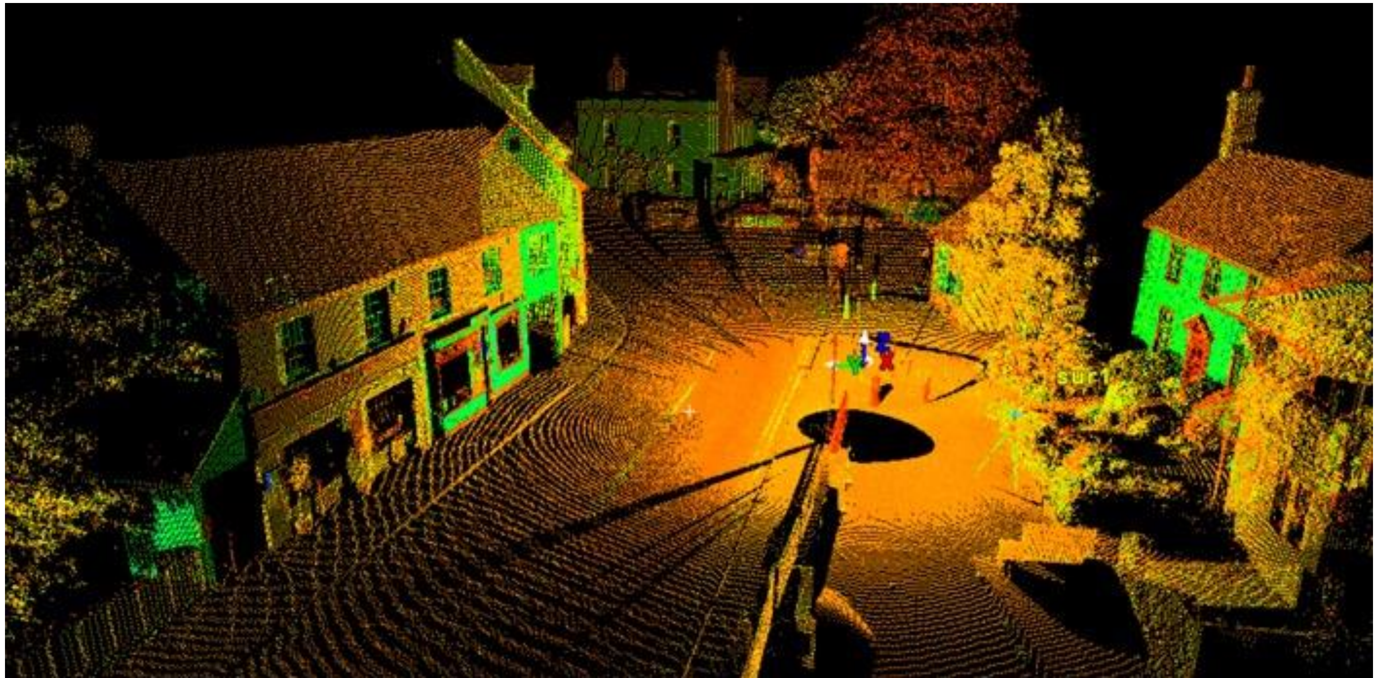






Challenges

The point samples may not be uniformly distributed

- Oblique scanning angles
- Laser energy attenuation





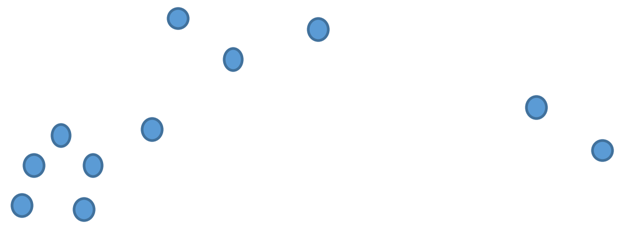
Challenges

Missing data

- Material properties, inaccessibility, occlusion, etc.

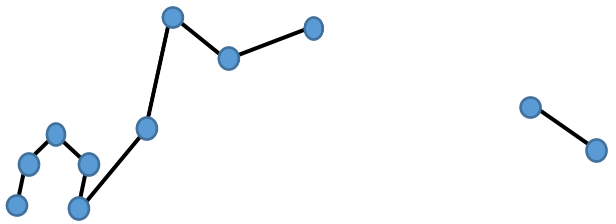
Explicit and Implicit Methods

$P =$



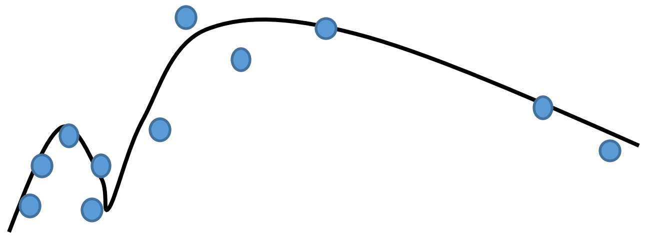
Explicit methods

Build a tessellation over the point cloud. The points become to vertices of the mesh



Implicit Methods

1. Define the surface implicitly, as the zeroes of a function $f_P\colon \mathbb{R}^3 \rightarrow \mathbb{R}^3$
2. Tessellate $\{f_P(x) = 0\}$



Explicit and Implicit Methods

Explicit methods

Build a triangulation over the point cloud. The points map to vertices of the mesh

- less robust to noise
- require a dense and even sampling
- Generally easier to implement

Implicit Methods

1. Define the surface implicitly, as the zeroes of a function $f_P : \mathbb{R}^3 \rightarrow \mathbb{R}^3$
2. Tessellate $\{f_P(x) = 0\}$

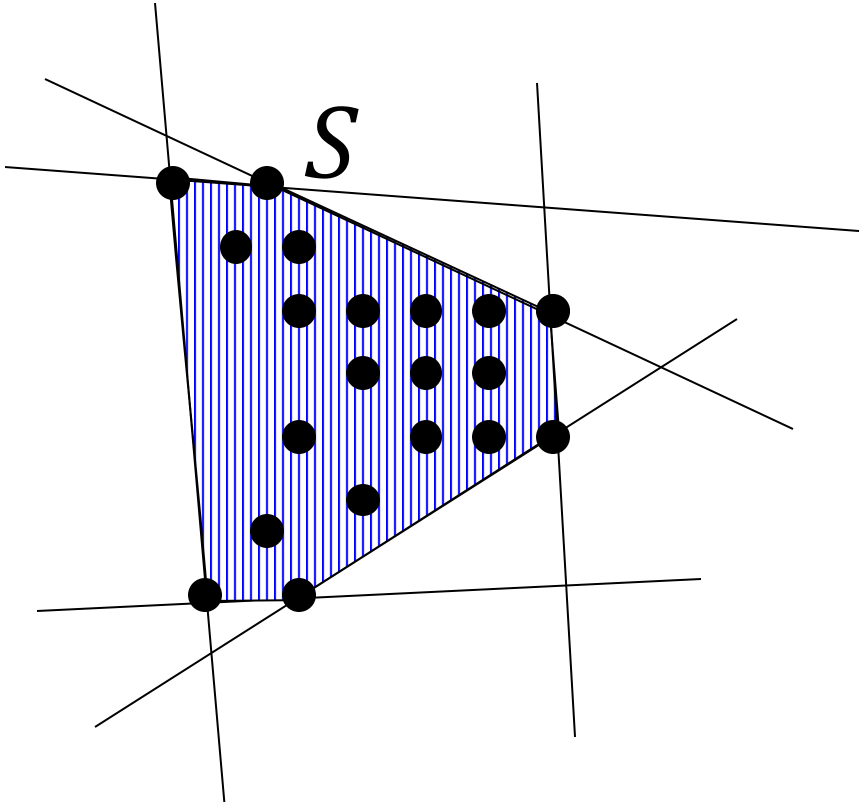
- more robust to noise
- more resilient to noise and uneven sampling

Alpha Shapes [Edelsbrunner83]

Convex Hull

$$CH(S) = \mathbb{R}^d \setminus \bigcup EH(S)$$

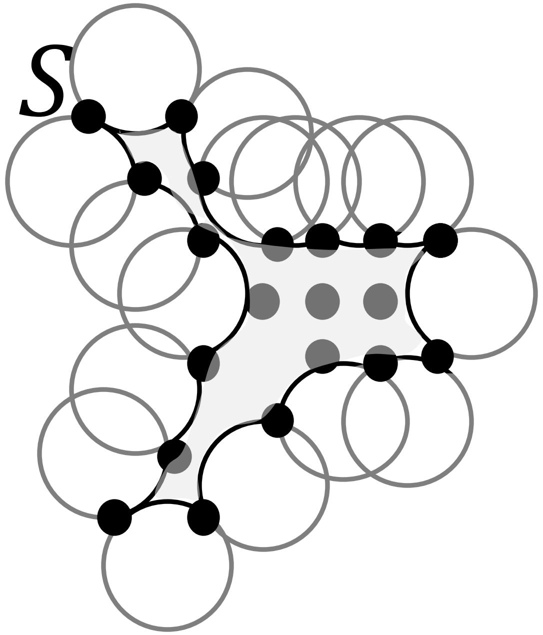
$EH(S)$: halfspace not containing any point in S

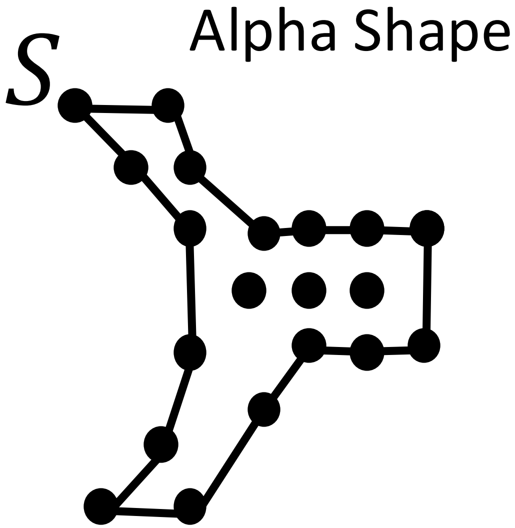


Alpha Hull

$$\alpha H(S) = \mathbb{R}^d \setminus \bigcup EB_\alpha(S)$$

EB_α(S): ball with radius α not containing any point in S



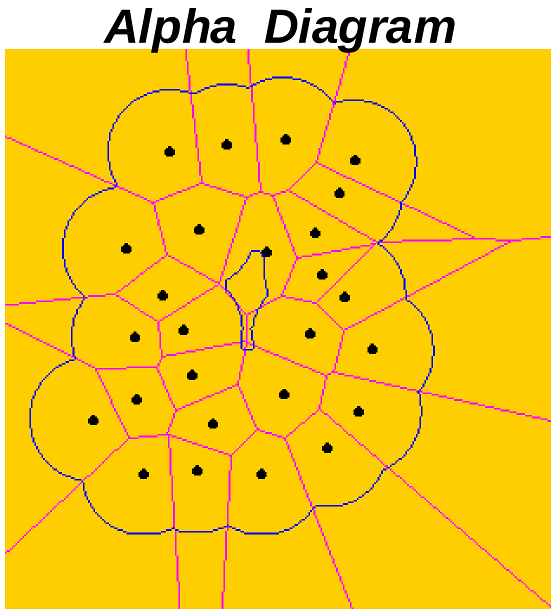


Computing Alpha Shapes

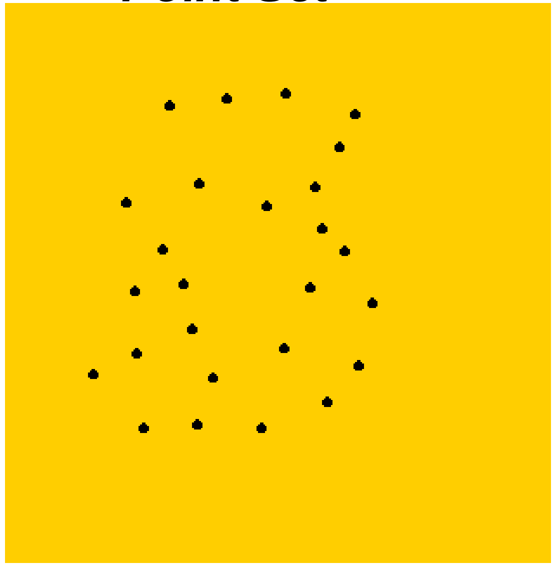
- **Alpha Diagram**: Voronoi Diagram restricted to space closest than α to one point in S
- **Alpha Complex**: Subset of Delaunay Triangulation computed as the dual of the alpha diagram

Computing Alpha Shapes

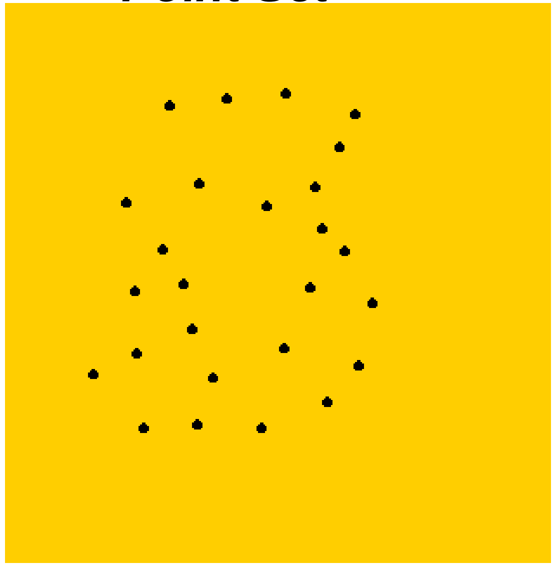
- **Alpha Diagram**: Voronoi Diagram restricted to space closest than α to one point in S
- **Alpha Complex**: Subset of Delaunay Triangulation computed as the dual of the alpha diagram



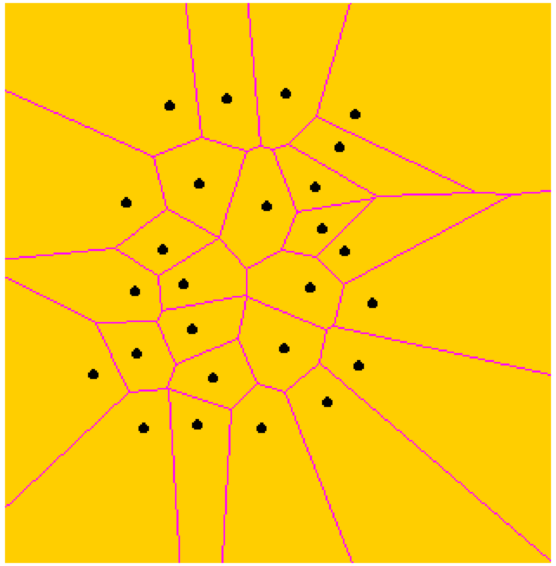
Point Set



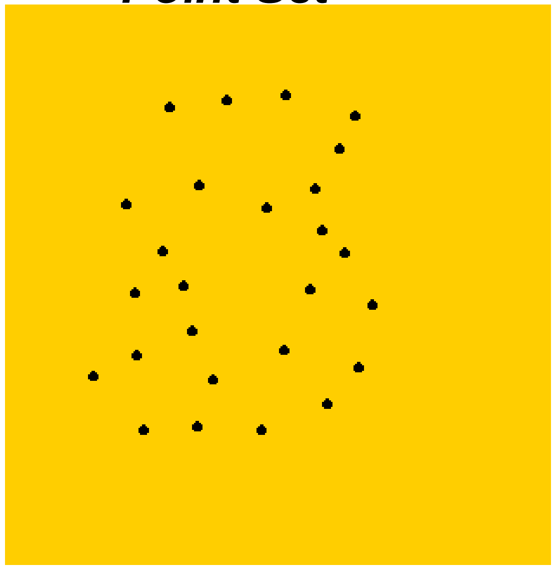
Point Set



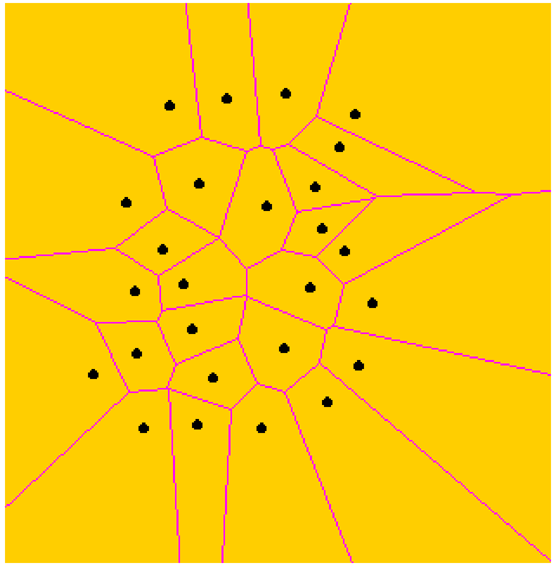
Voronoi Diagram



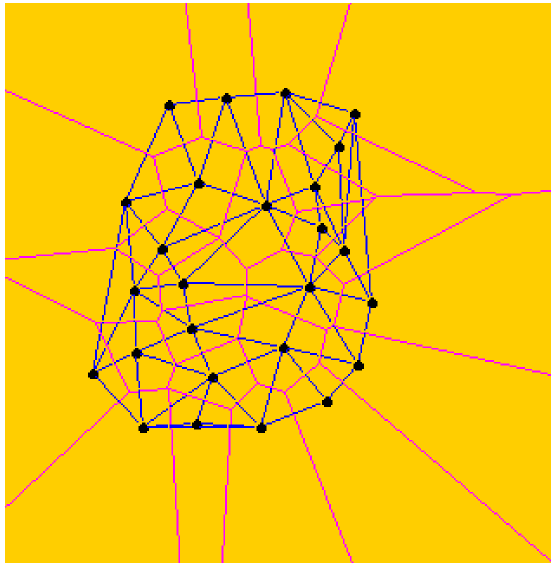
Point Set



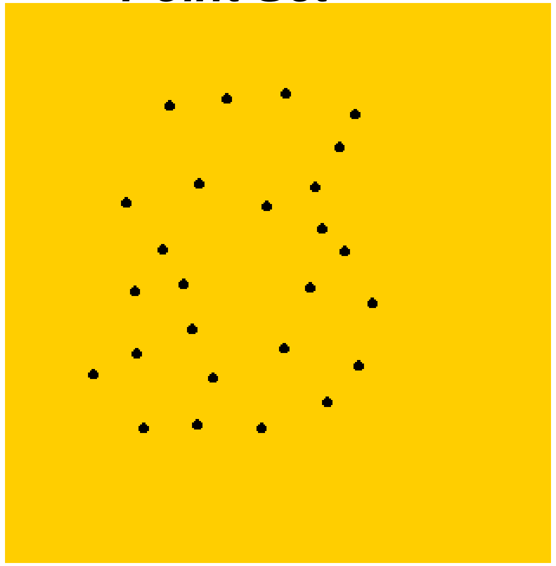
Voronoi Diagram



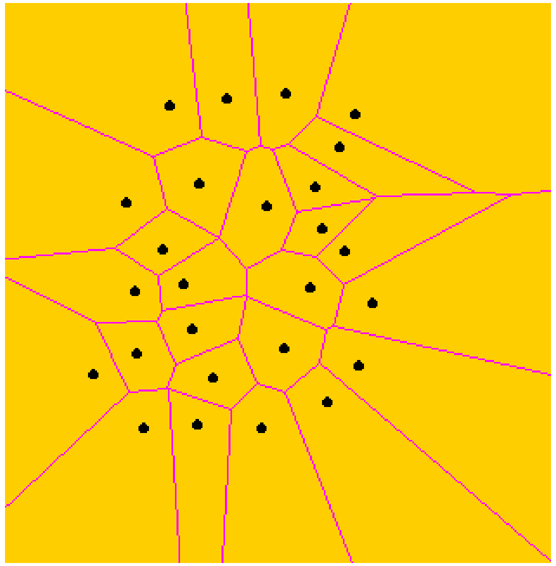
Delaunay Triangulation



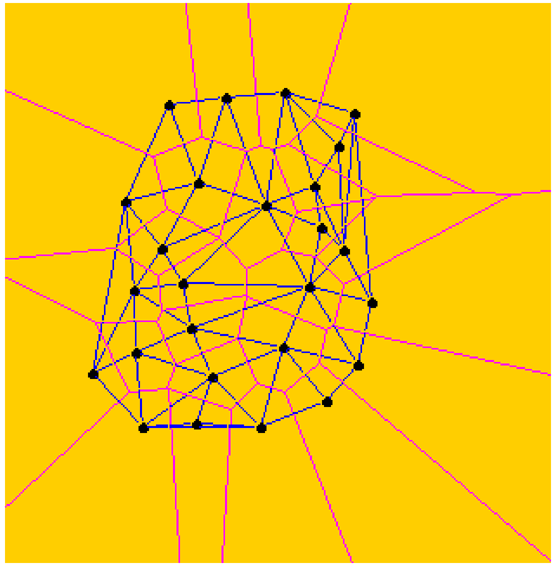
Point Set



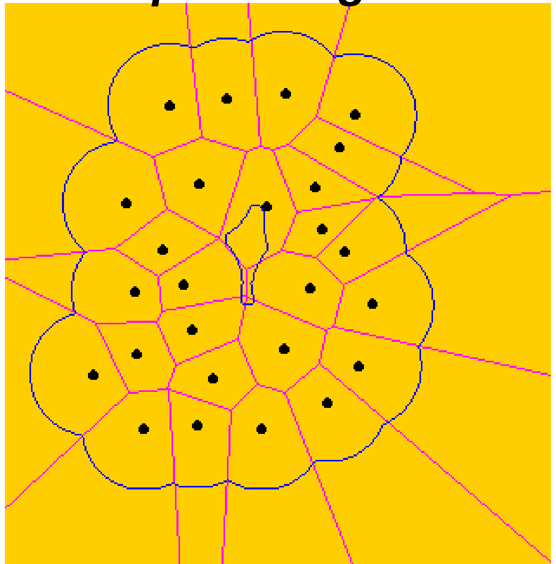
Voronoi Diagram



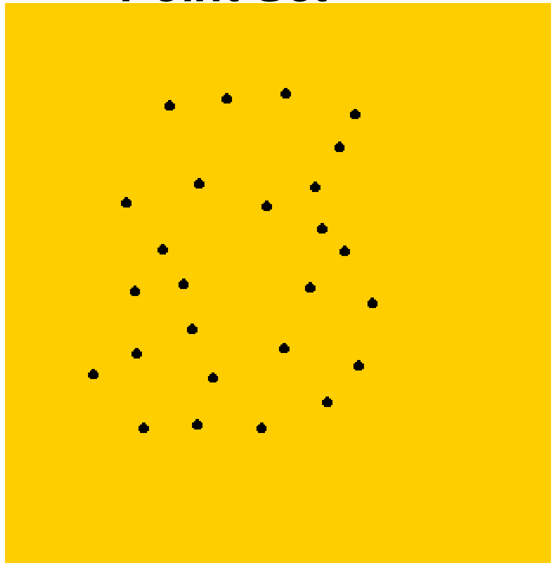
Delaunay Triangulation



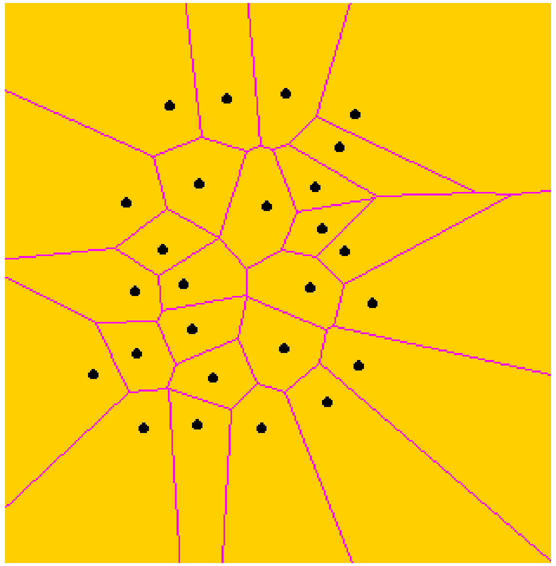
Alpha Diagram



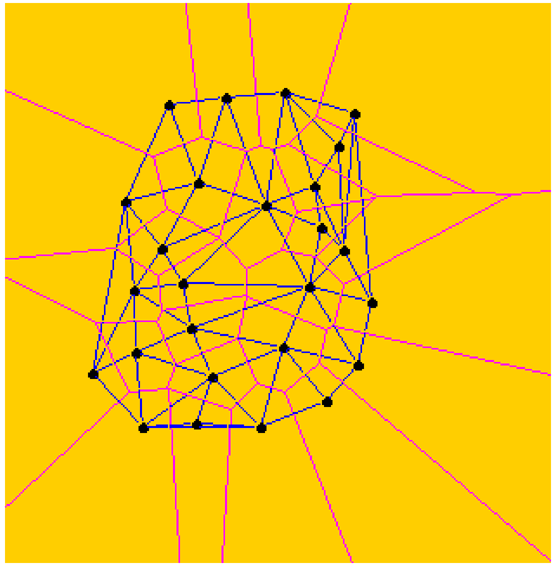
Point Set



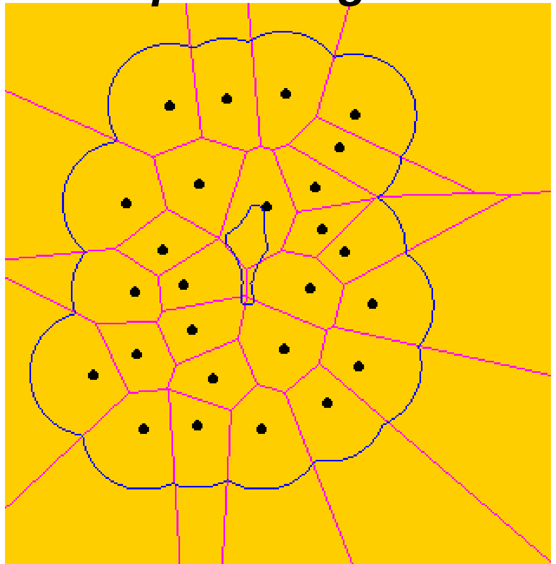
Voronoi Diagram



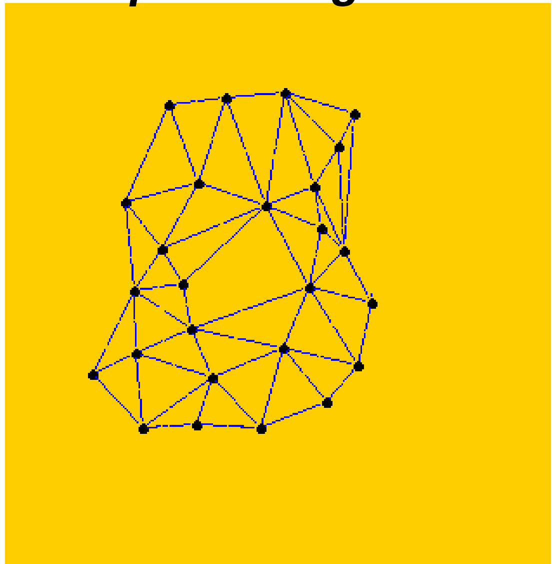
Delaunay Triangulation



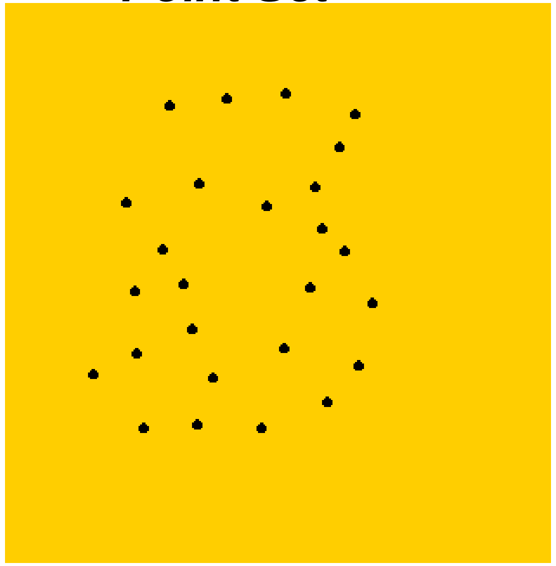
Alpha Diagram



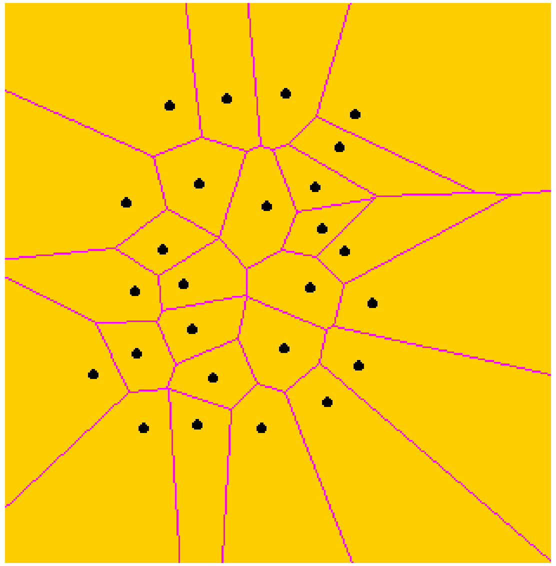
Alpha triangulation



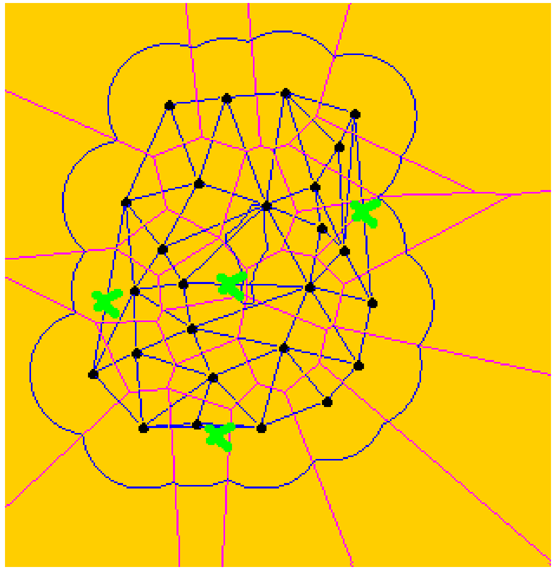
Point Set



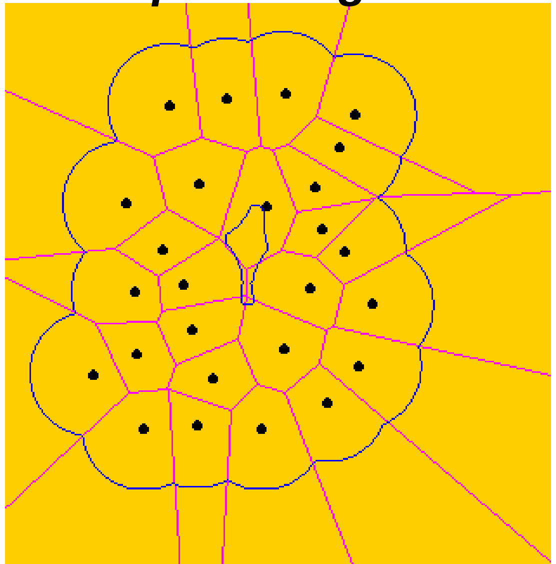
Voronoi Diagram



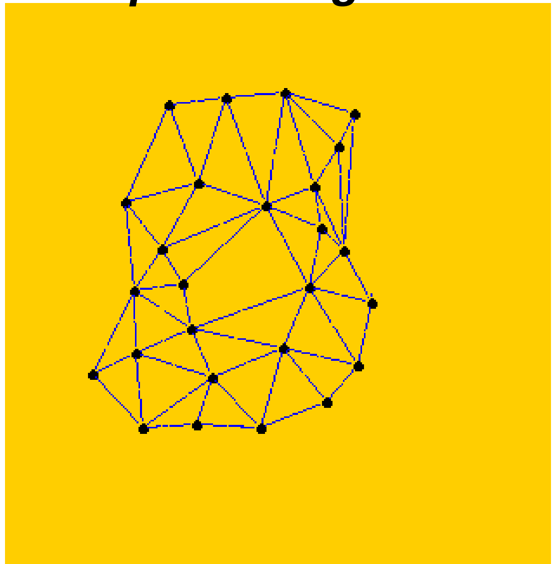
Delaunay Triangulation



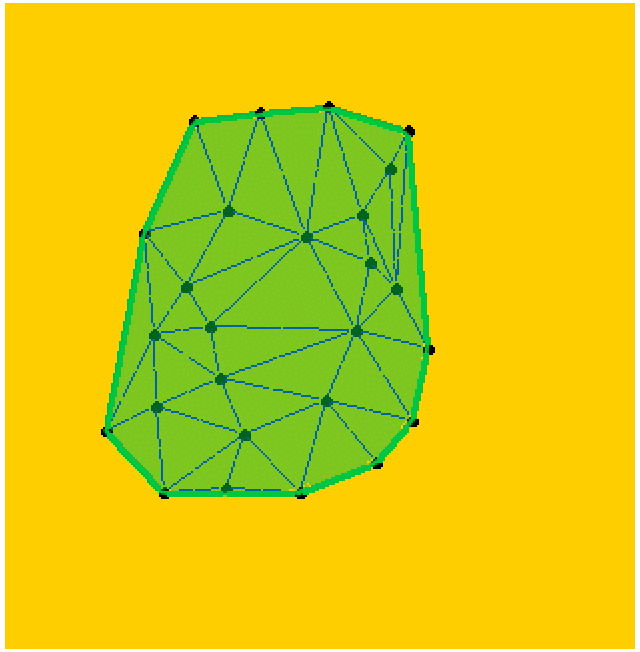
Alpha Diagram



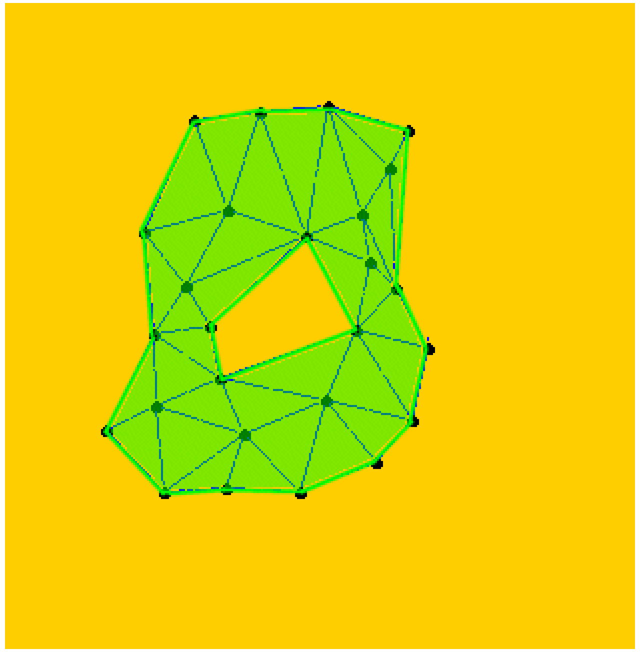
Alpha triangulation



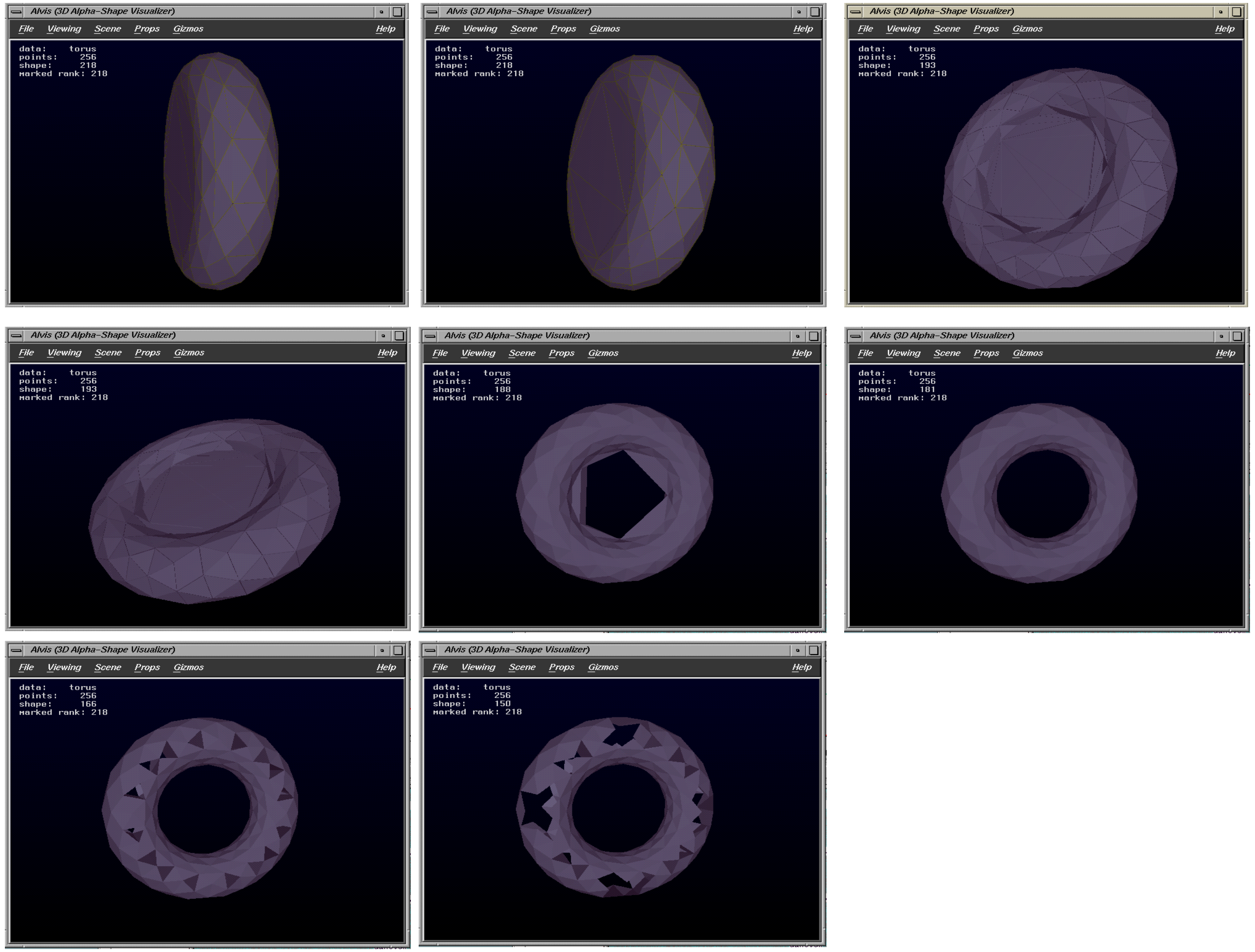
Delaunay triangulation



Alpha Complex



- α = 0 α-shape is the point set
- α → ∞ α-shape tends to the convex hull
- A finite number of thresholds $\alpha_0 < \alpha_1 < \ldots < \alpha_n$ defines all possible shapes (at most $2n^2 - 5n$)


Alvis (3D Alpha-Shape Visualizer)
File Viewing Scene Props Gizmos Help
data: torus
points: 256
shape: 218
marked rank: 218
Alvis (3D Alpha-Shape Visualizer)
File Viewing Scene Props Gizmos Help
data: torus
points: 256
shape: 218
marked rank: 218
Alvis (3D Alpha-Shape Visualizer)
File Viewing Scene Props Gizmos Help
data: torus
points: 256
shape: 193
marked rank: 218
Alvis (3D Alpha-Shape Visualizer)
File Viewing Scene Props Gizmos Help
data: torus
points: 256
shape: 193
marked rank: 218
Alvis (3D Alpha-Shape Visualizer)
File Viewing Scene Props Gizmos Help
data: torus
points: 256
shape: 188
marked rank: 218
Alvis (3D Alpha-Shape Visualizer)
File Viewing Scene Props Gizmos Help
data: torus
points: 256
shape: 181
marked rank: 218
Alvis (3D Alpha-Shape Visualizer)
File Viewing Scene Props Gizmos Help
data: torus
points: 256
shape: 166
marked rank: 218
Alvis (3D Alpha-Shape Visualizer)
File Viewing Scene Props Gizmos Help
data: torus
points: 256
shape: 150
marked rank: 218

Sampling Conditions for Alpha Shapes

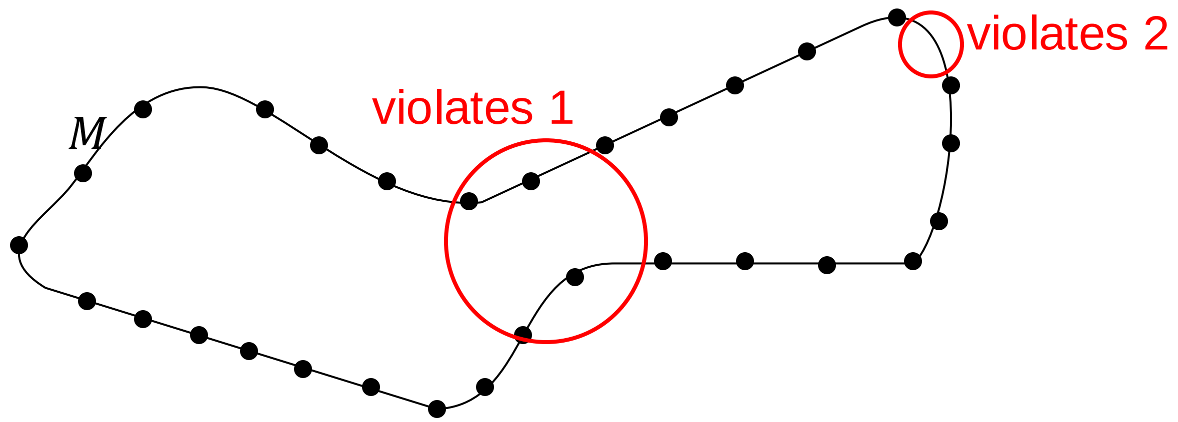
Proposition

Given a smooth manifold M and a sampling S,

if it holds that

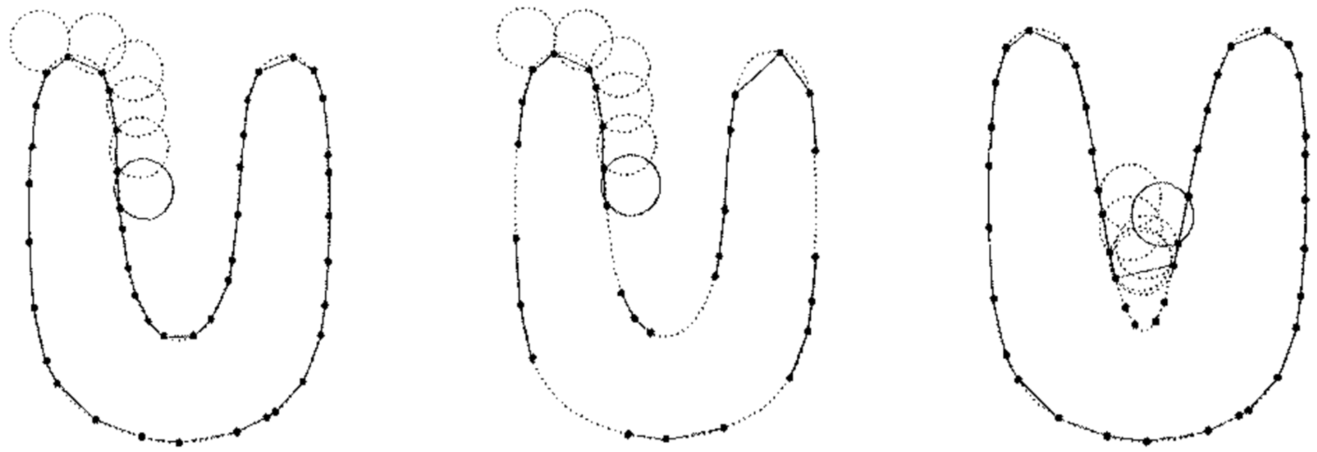
1. The intersection of any ball of radius α with M is homeomorphic to a disk
2. Any ball of radius α centered in the manifold contains at least one point of S

Then the α-shape of S is homeomorphic to M



Ball Pivoting [bernardini99]

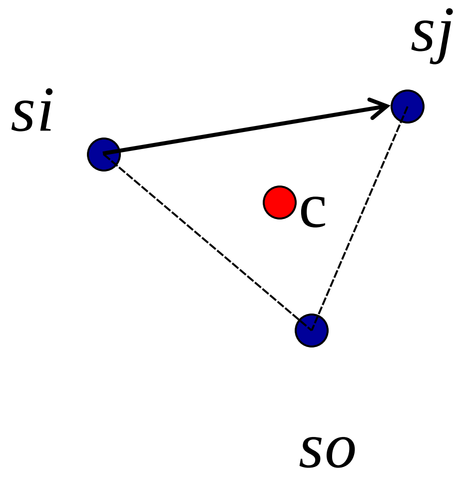
- Motivations
  - Alpha shapes computation is fairly cumbersome
  - May produce non manifold surfaces
- Core idea: approximate the alpha shapes just «rolling» a ball of radius α on the sampling S
- Same sampling conditions as α –shape holds



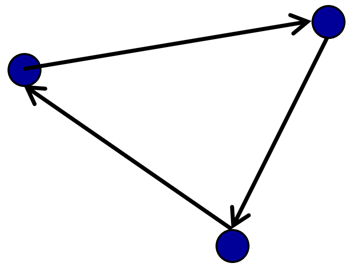
OK   Low sampling density   Curvature grater than $\frac{1}{\alpha}$

The algorithm

- Edge (s_i, s_j)
  - –Opposite point so, center of empty ball c
  - –Edge: “Active”, “Boundary”



Pivoting example

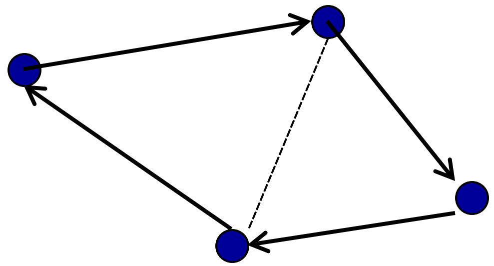


Initial seed triangle:
Empty ball of radius ρ passes through the three points

Active edge

Point on front

Pivoting example

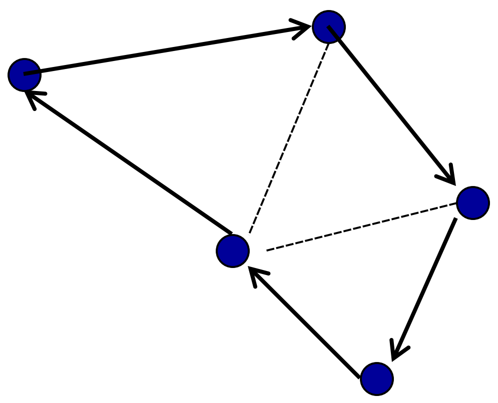


Ball pivoting around active edge

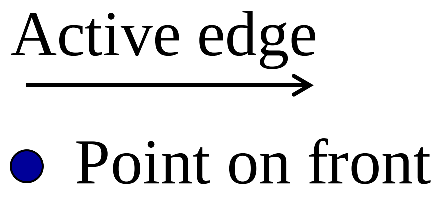
Active edge

Point on front

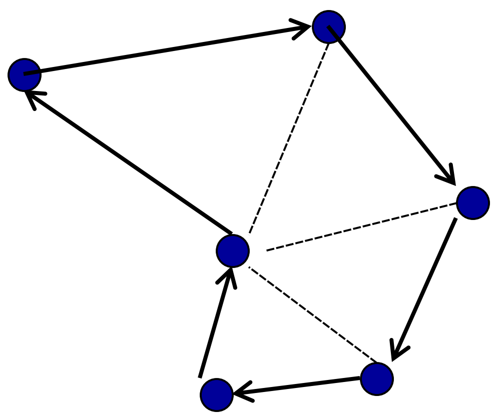
Pivoting example



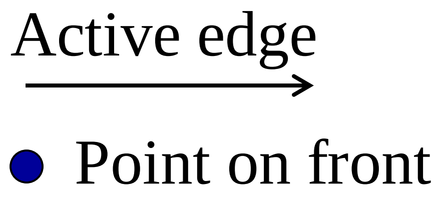
Ball pivoting around active edge



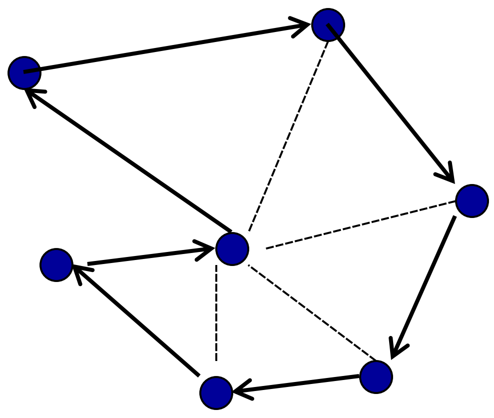
Pivoting example



Ball pivoting around active edge



Pivoting example

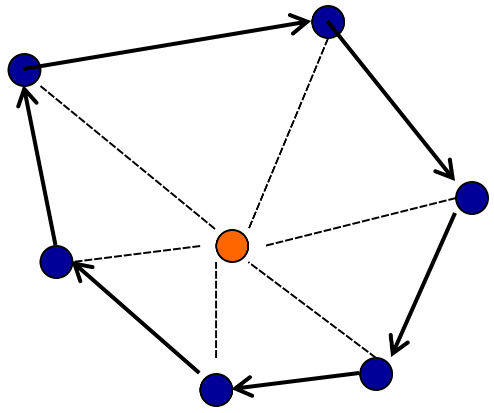


Ball pivoting around active edge

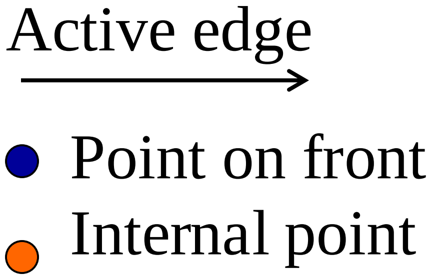
Active edge

Point on front

Pivoting example

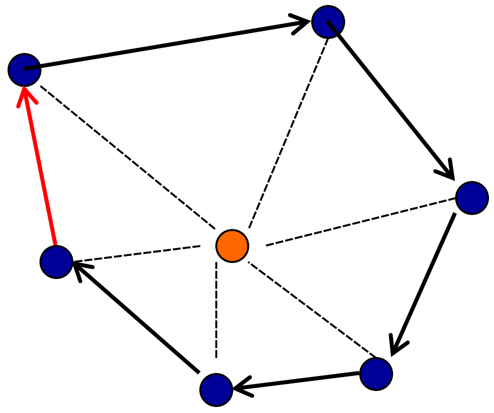


Ball pivoting around active edge



Pivoting example

Boundary edge



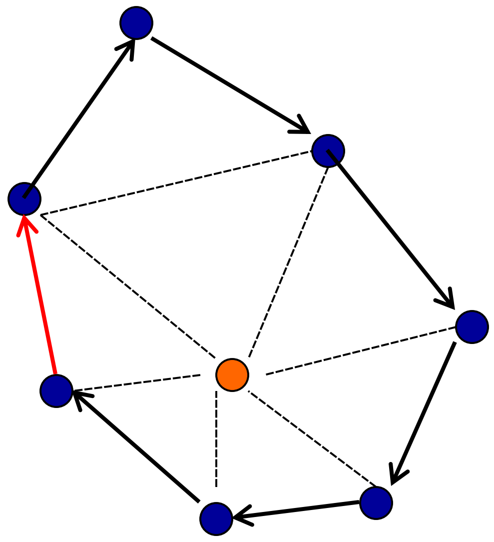
Ball pivoting around active edge
No pivot found

Active edge

Point on front
Internal point

Pivoting example

Boundary edge



Ball pivoting around active edge

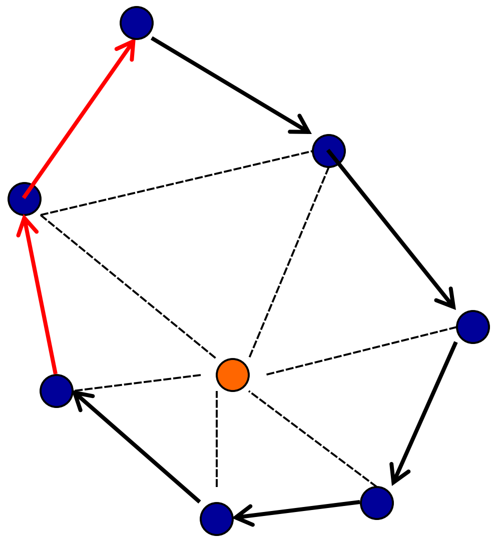
Active edge

Point on front

Internal point

Pivoting example

Boundary edge



Active edge

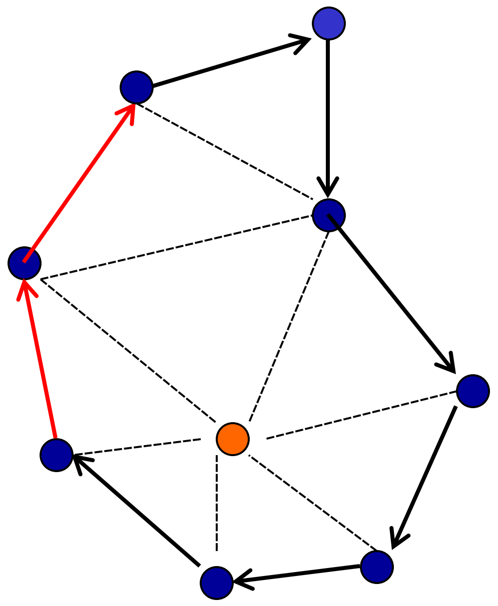
Point on front

Internal point

Ball pivoting around active edge
No pivot found

Pivoting example

Boundary edge



Ball pivoting around active edge

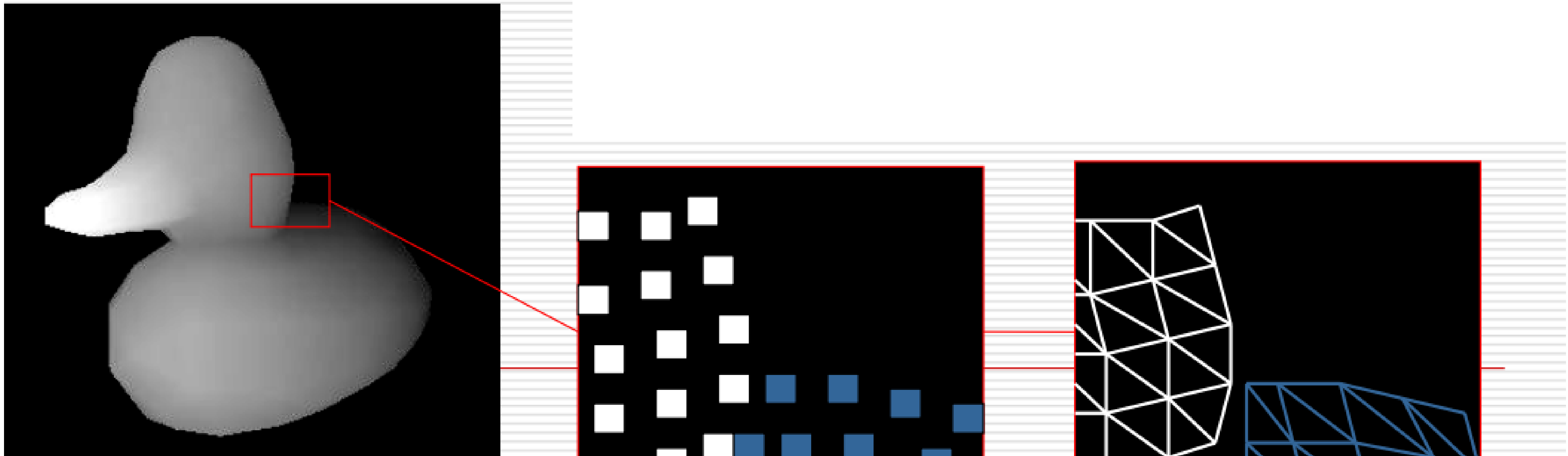
Active edge

Point on front

Internal point

Not only point clouds: the Range Maps

- 3D scanners produce a number of dense structured height fields, that is, a regular (X,Y) grid of points with a distance Z value. These are called **range maps**
- Trivial to triangulate but: How to merge different range maps?



Mesh Zippering [Turk94]

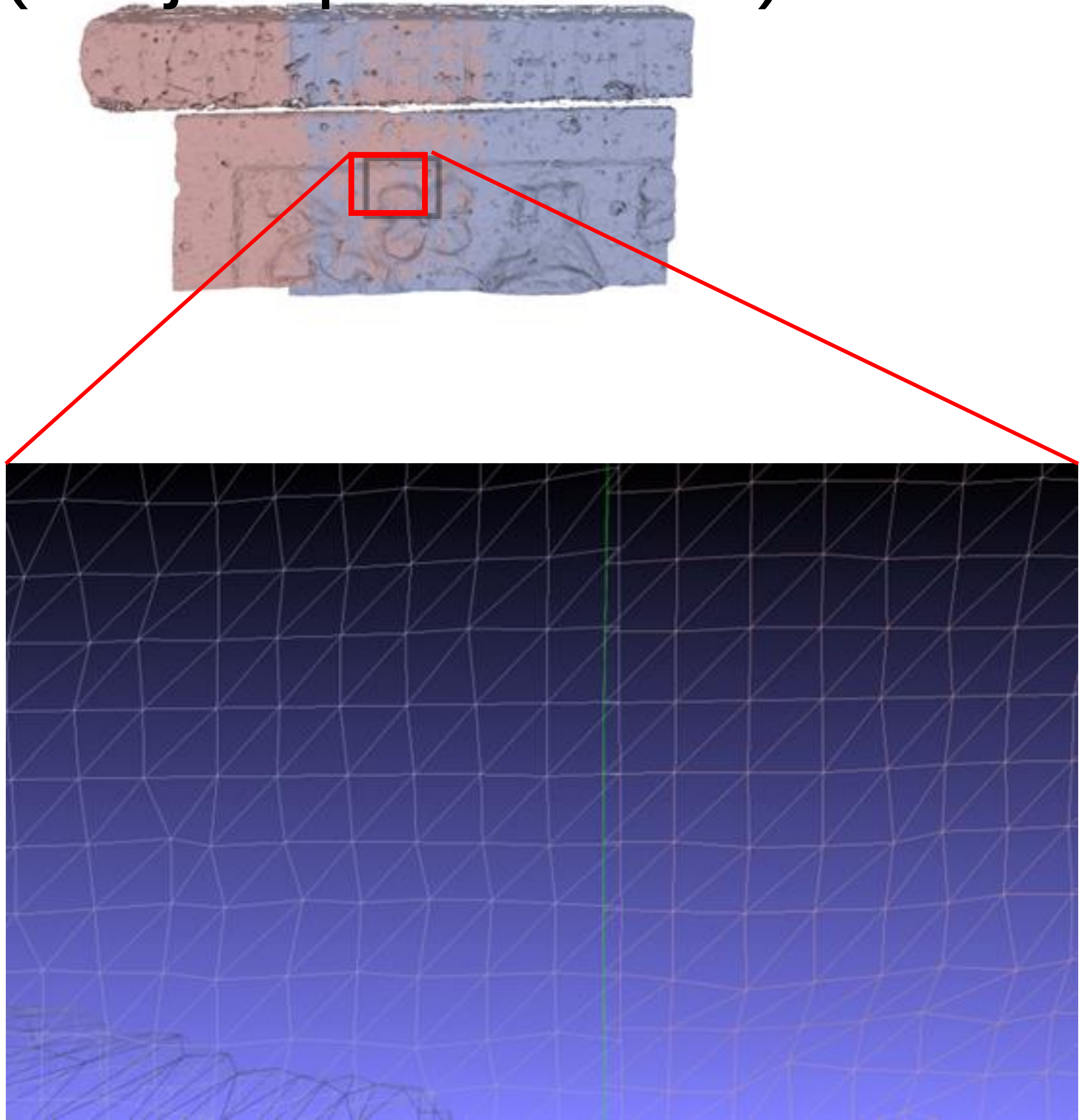
- Input: triangulated ranges maps (not just point clouds)
- Works in pairs:
  - **Remove overlapping portions**
  - Clip one RM against the other
  - Remove small triangles

Mesh Zippering

Input: triangulated ranges maps (not just point clouds)

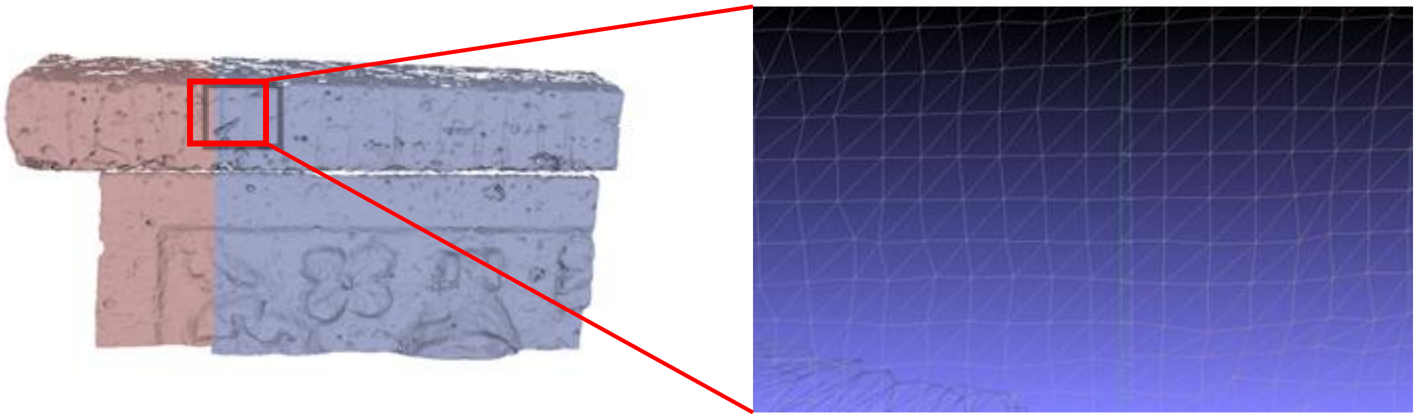
Works in pairs:

- **Remove overlapping portions**
- Clip one RM against the other
- Remove small triangles



Mesh Zippering

- Input: triangulated ranges maps (not just point clouds)
- Works in pairs:
  - **Remove overlapping portions**
  - Clip one RM against the other
  - Remove small triangles

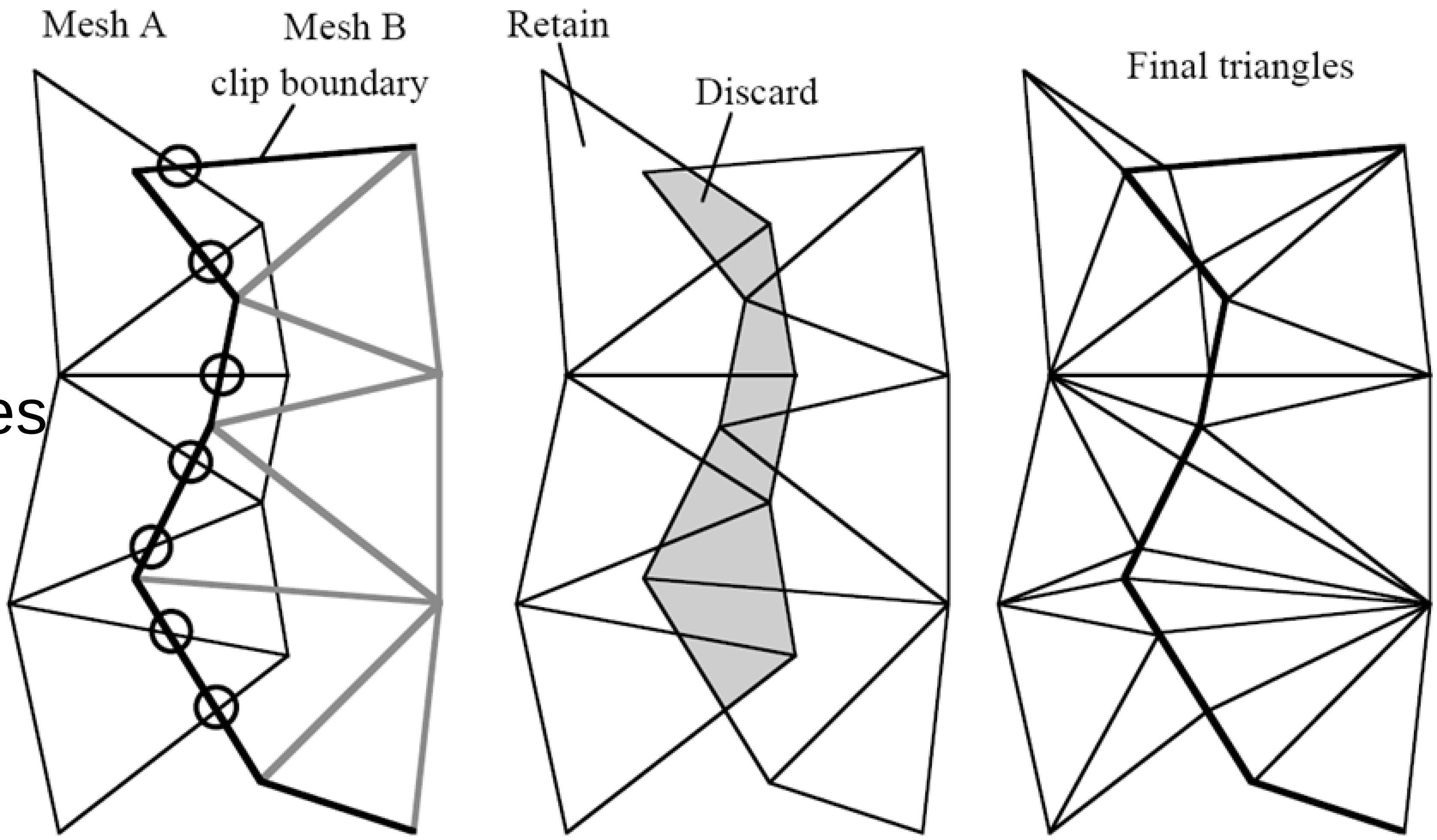


Mesh Zippering

Input: triangulated ranges maps (not just point clouds)

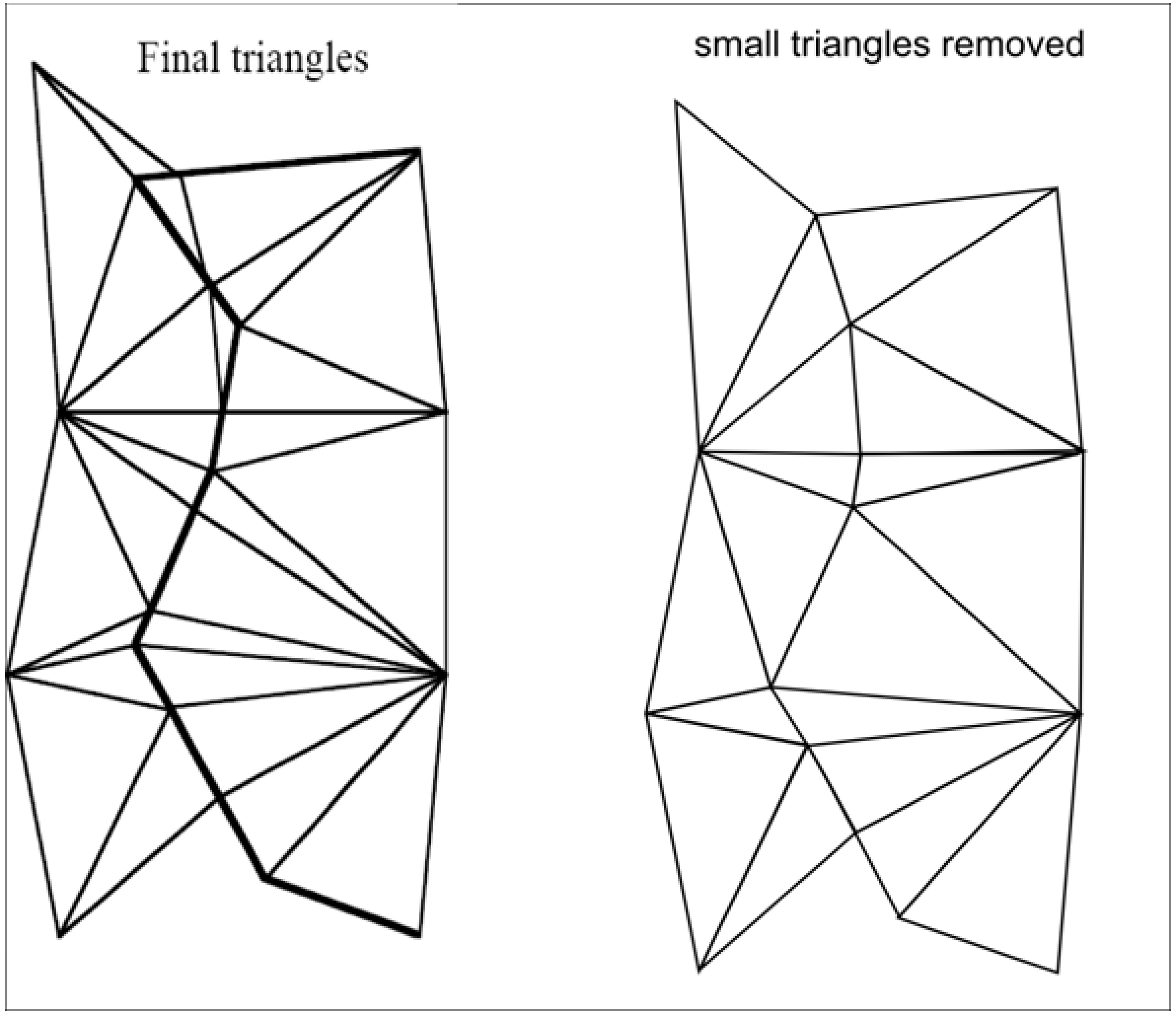
Works in pairs:

- Remove overlapping portions
- **Clip one RM against the other**
- Remove small triangles



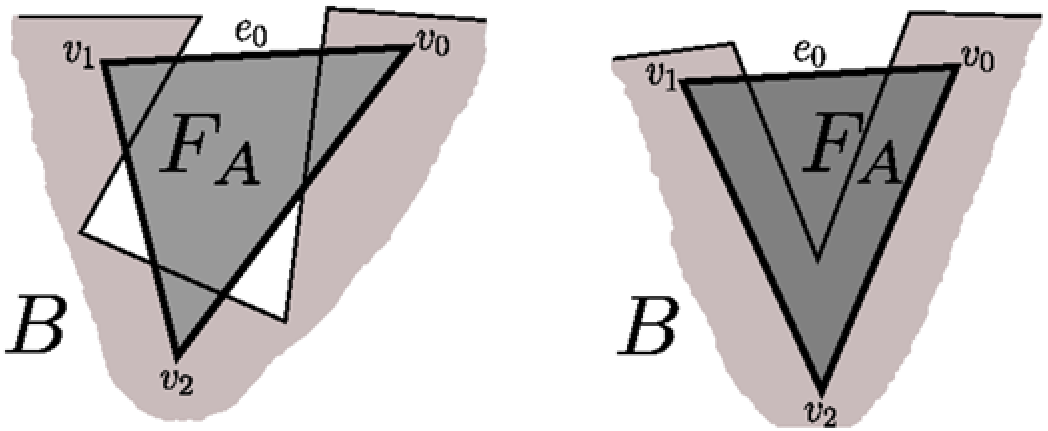
Mesh Zippering

- Input: triangulated ranges maps (not just point clouds)
- Works in pairs:
  - Remove overlapping portions
  - Clip one RM against the other
  - **Remove small triangles**



Mesh Zippering

- Not so trivial to implement…for example..
- **remove overlapping regions**: «a face of mesh A overlaps if its 3 vertices project on mesh B»
- Hole may appear, to be fixed later…

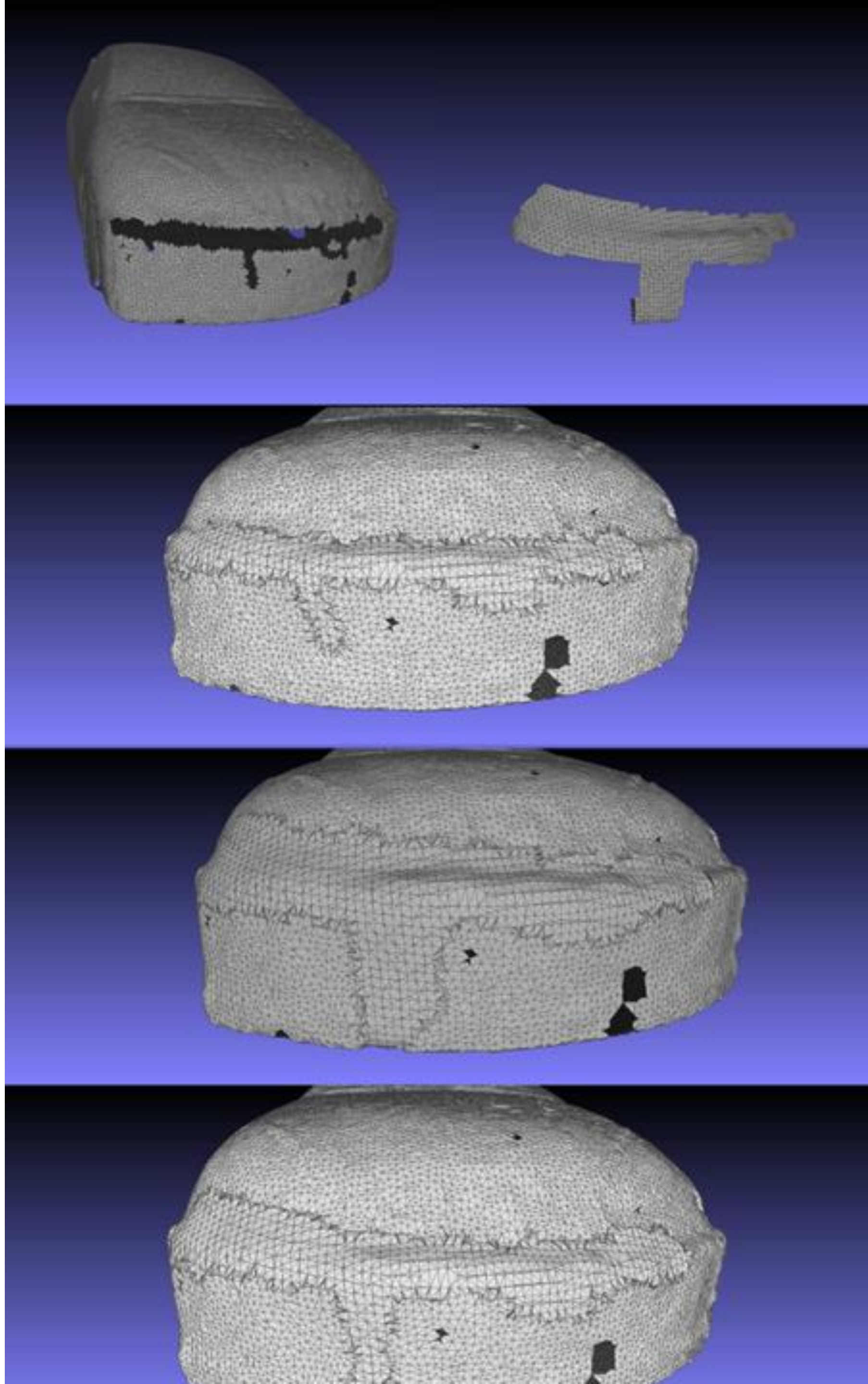


Mesh Zippering

- Not so trivial to implement…for example..
  - **remove overlapping regions**: criterion?

Mesh Zippering

- Not so trivial to implement…for example..
  - **remove overlapping regions**: criterion?



Preserve faces from left

Preserve faces from right

Halfway (distance from the border)