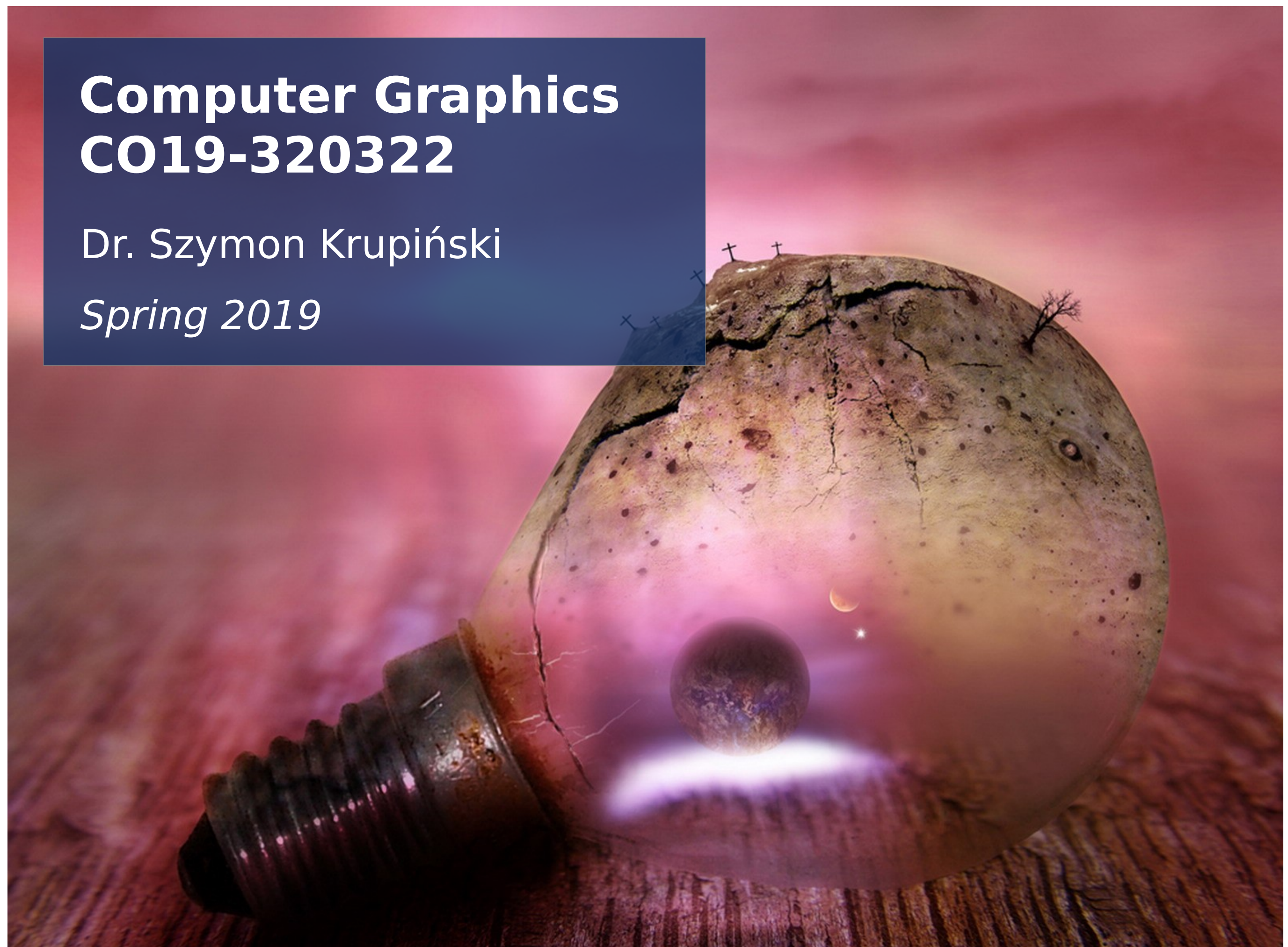
Lecture 20: Point clouds



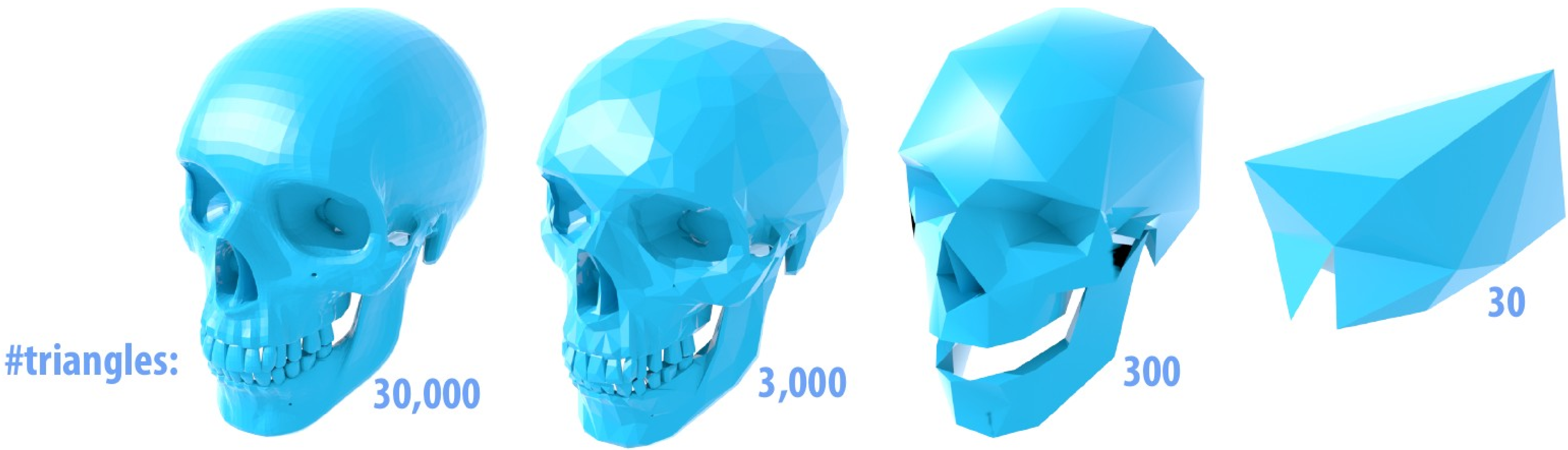
Computer Graphics
CO19-320322

Dr. Szymon Krupiński

Spring 2019

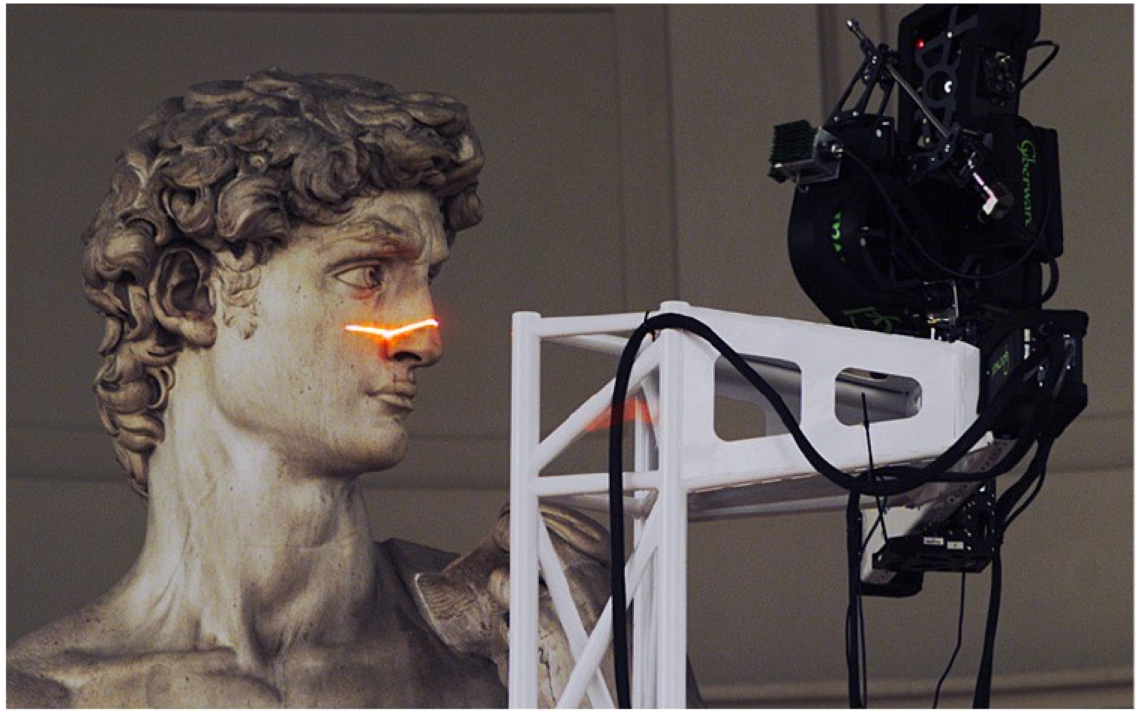
Recent subjects

- Explicit geometries: not only triangles!
  - Point clouds are the simples example of explicit geometry



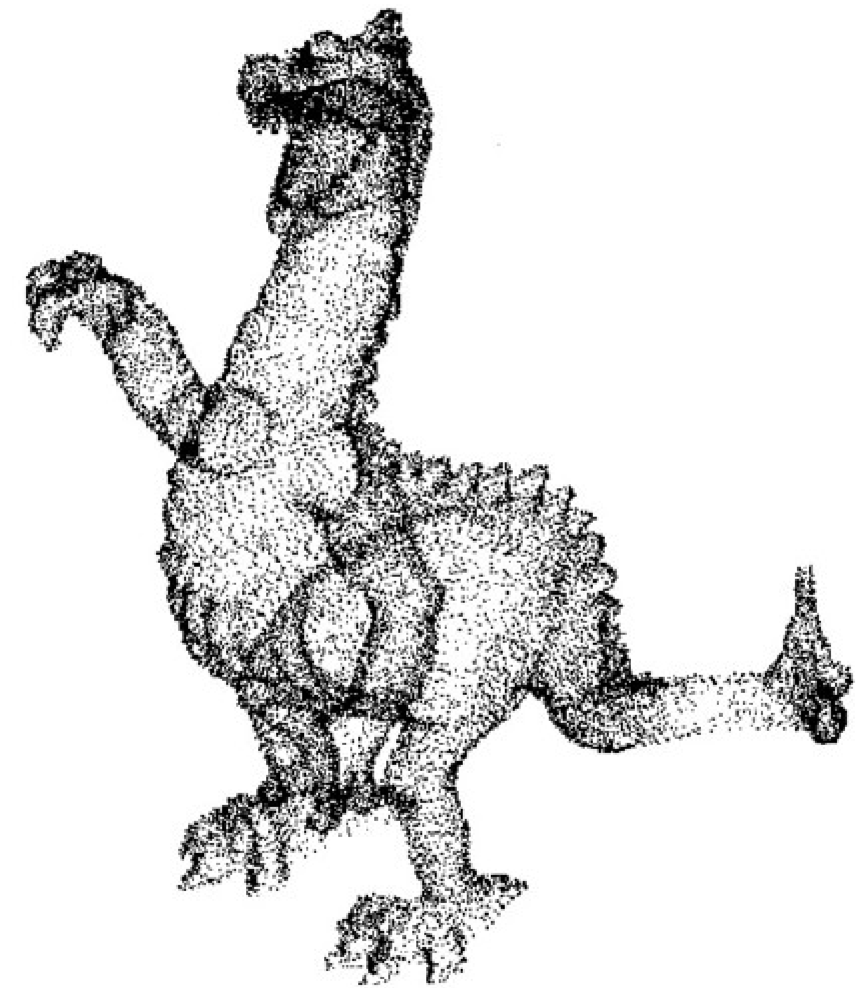
Object generation

- Modeling:
  - Interactive tools
  - Computer Aided (Geometric) Design
- Measured:
  - Scanning devices
  - Cameras
  - Sensors (Kinnect, etc)



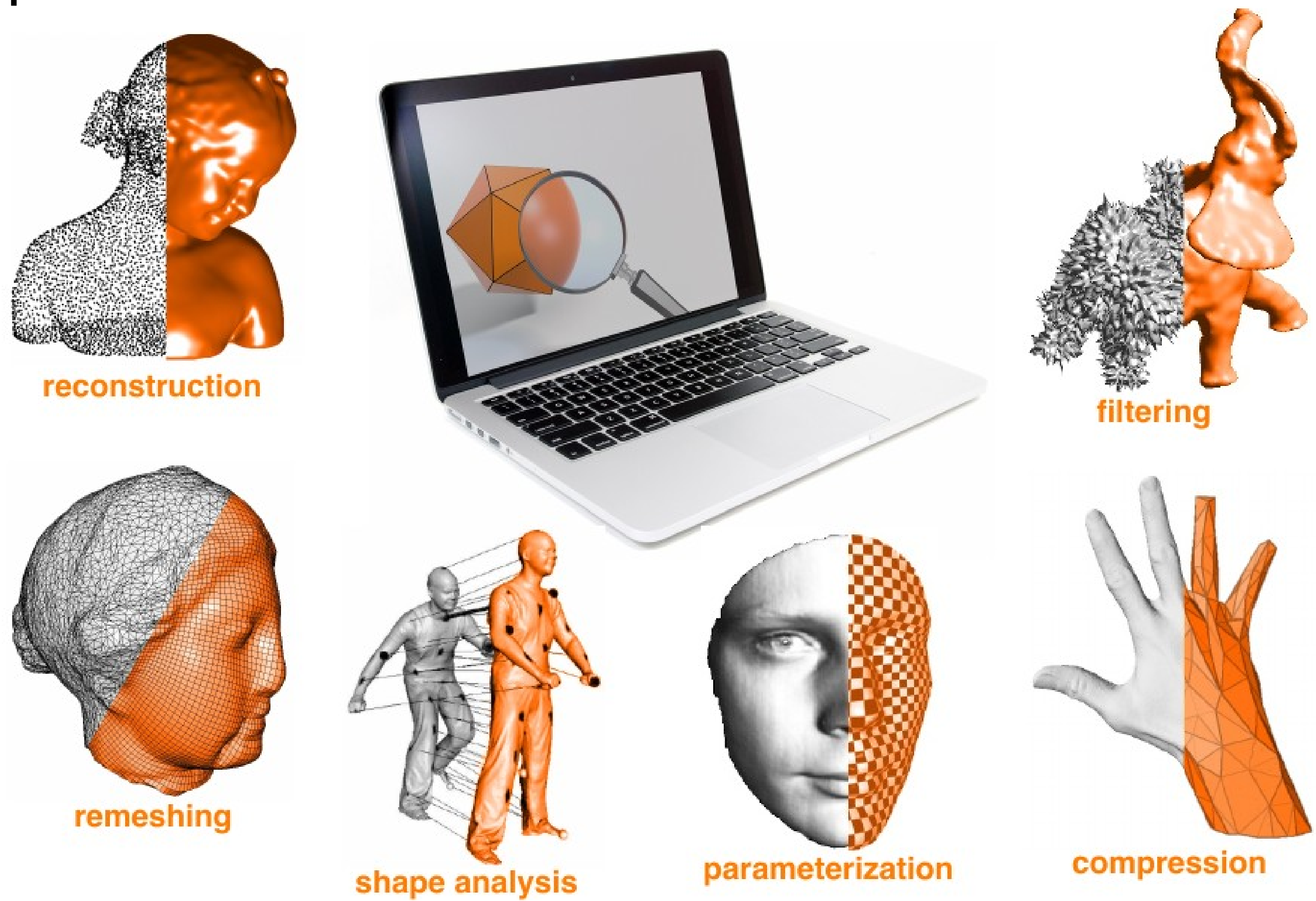
Result: point cloud

- Inherently unorganized set of points located on a surface / boundary
- Can appear as an recognisable shape to humans – but that's because of the developed "connect-the-dots" abilities of our brains
- From the CG point of view:
  - Inefficient storage
  - Difficult to display realistically
  - Very difficult to process efficiently



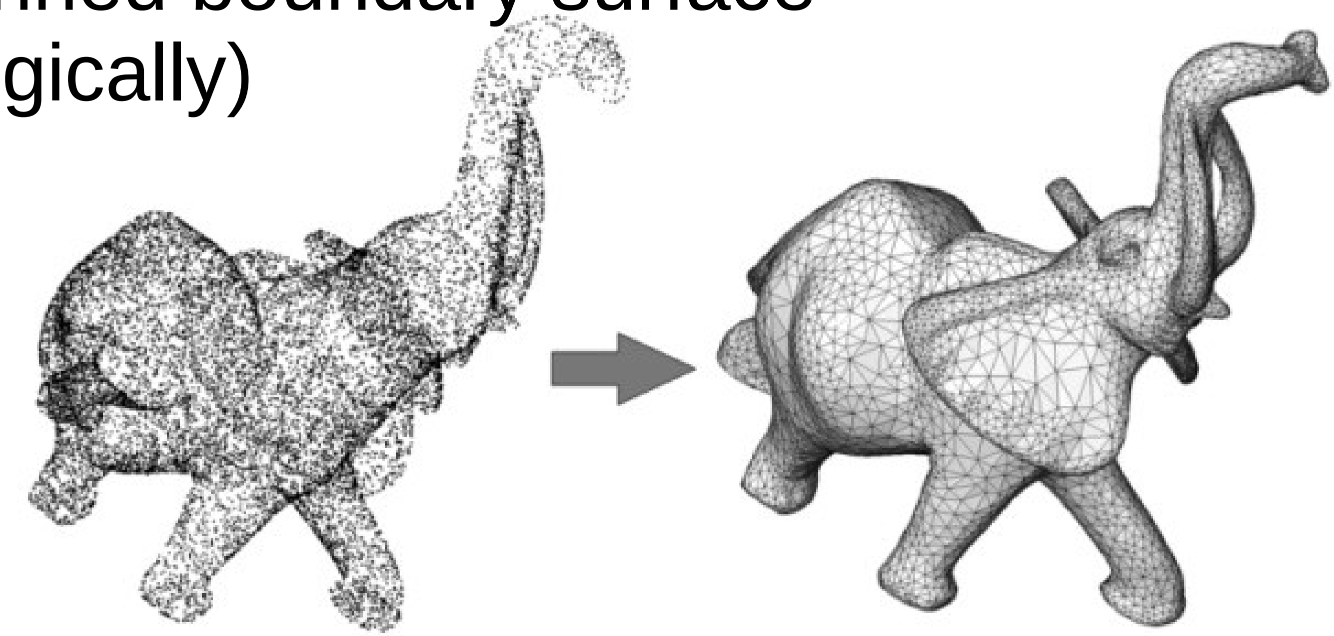
Geometry processing

- Collecting/producing a point cloud can be just a beginning of a whole process!



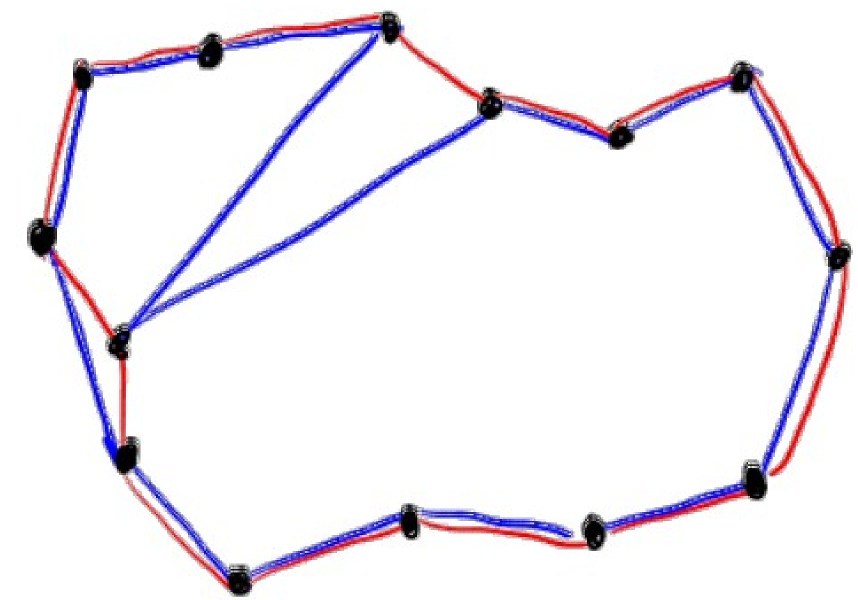
Surface reconstruction

- Goal:
  - Connect the points of a point cloud with triangles
  - ...such that the triangles form a closed triangular mesh
  - ...that represents the scanned boundary surface (geometrically and topologically)

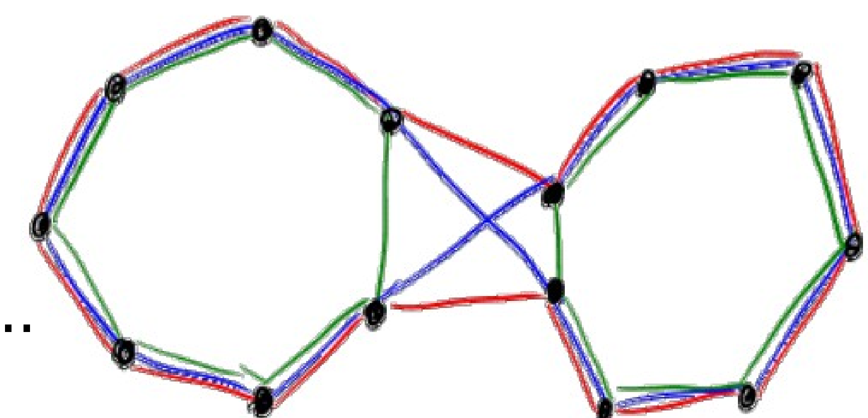


Connect the dots

- Creating arbitrary edges between dots does not yield good results
- We can derive conditions from our real life experience
  - Condition 1: the surface should be minimal
  - Condition 2: the resulting surface should be topologically equivalent to the scanned surface, e.g. it must have a well-defined interior
- Methods: *Voronoi diagrams, Poisson reconstruction, visual hull, marching cubes*, etc...

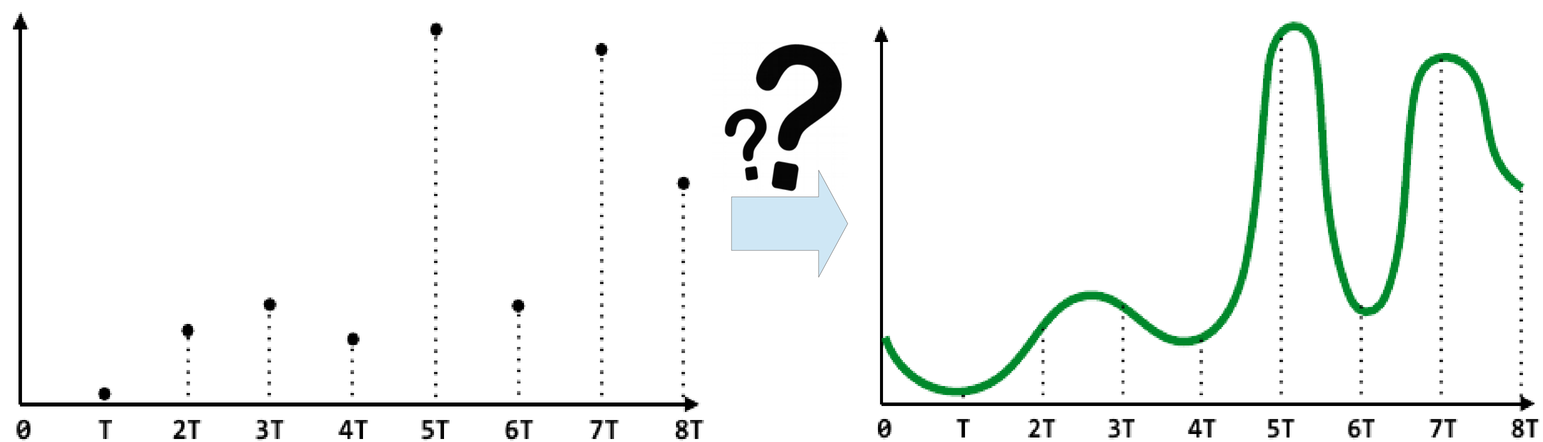


How to justify this? Think about the sampling theory and the Nyquist frequency!



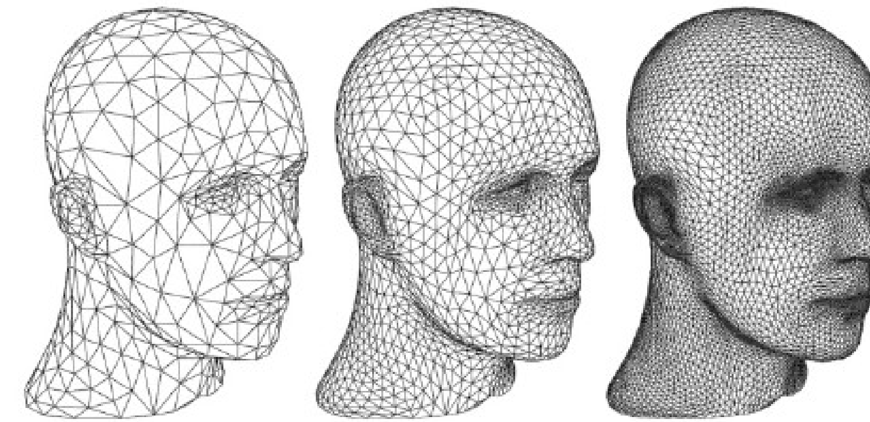
Point cloud generation as sampling

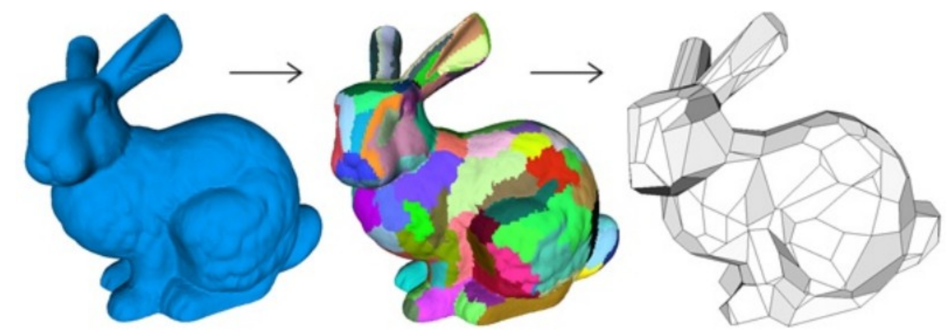
- The sampling rate of the scanning process needs to be sufficiently high to capture all topological information
- Shannon theorem: The sampling rate has to be, at least, twice the frequency of the highest frequency to be reconstructed from the signal (≡ the Nyquist frequency)

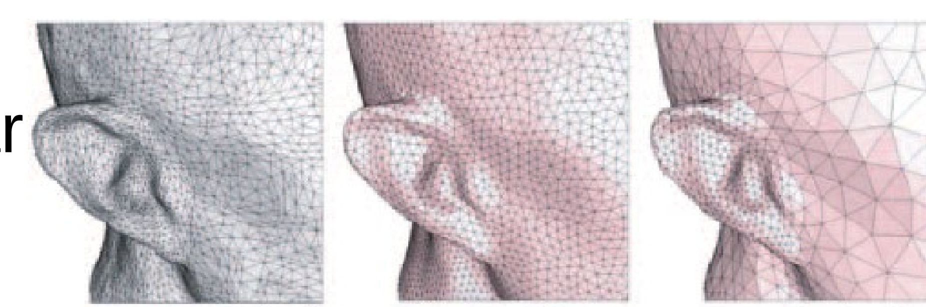


Sampling vs. surface

- Upsampling (e.g. interpolation, subdivision)
  - Increase resolution
  - Close-ups, shading, making “prettier” surfaces
- Downsampling (e.g. interpolation, decimation)
  - Efficient L.O.D.-based display, reducing noise
- Resampling (e.g. distribution analysis, grid fitting)
  - Storage efficiency, producing more regular mesh for calculations, removing outliers

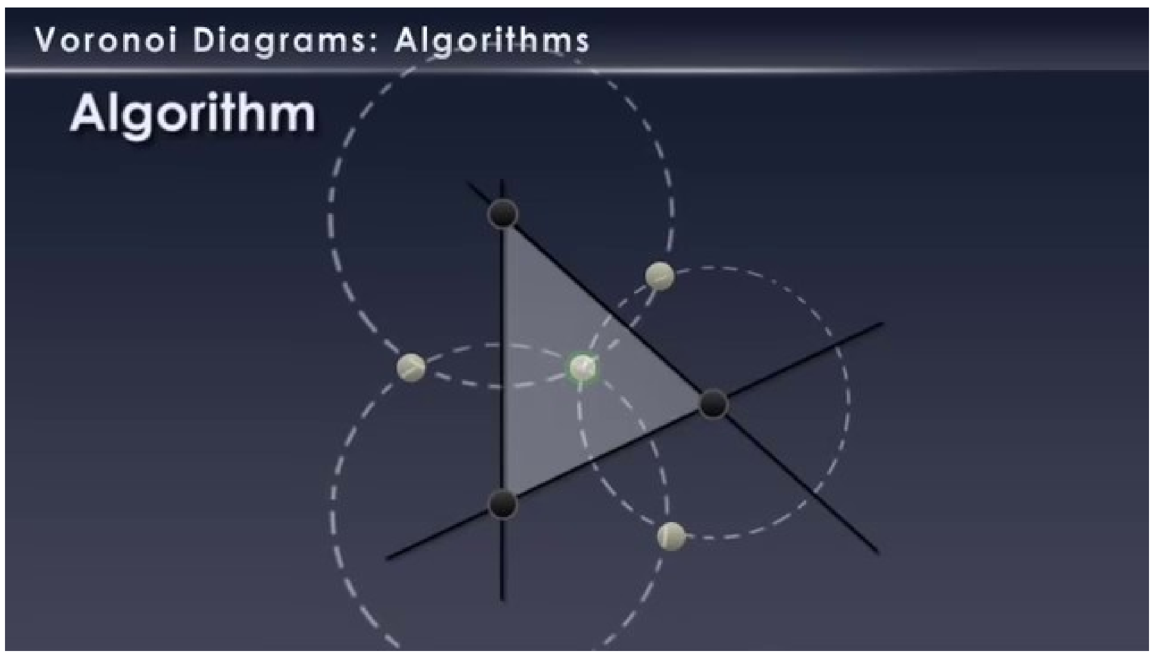




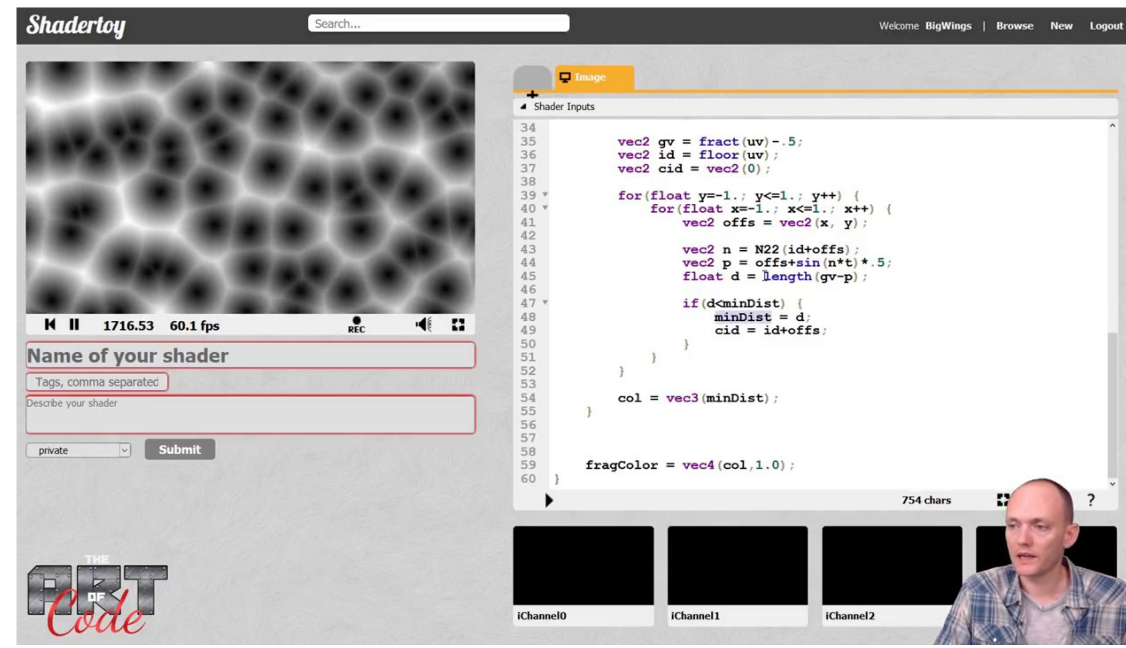


Voronoi diagrams

- Surface reconstruction can be based on Voronoi diagrams (Georgy Voronoi, 1868-1908)



Voronoi Diagram Intro Part 2 - Construction Algorithms
https://www.youtube.com/watch?v=Y5X1TvN9TpM



Voronoi Explained

https://www.youtube.com/watch?v=l-07BXzNdPw

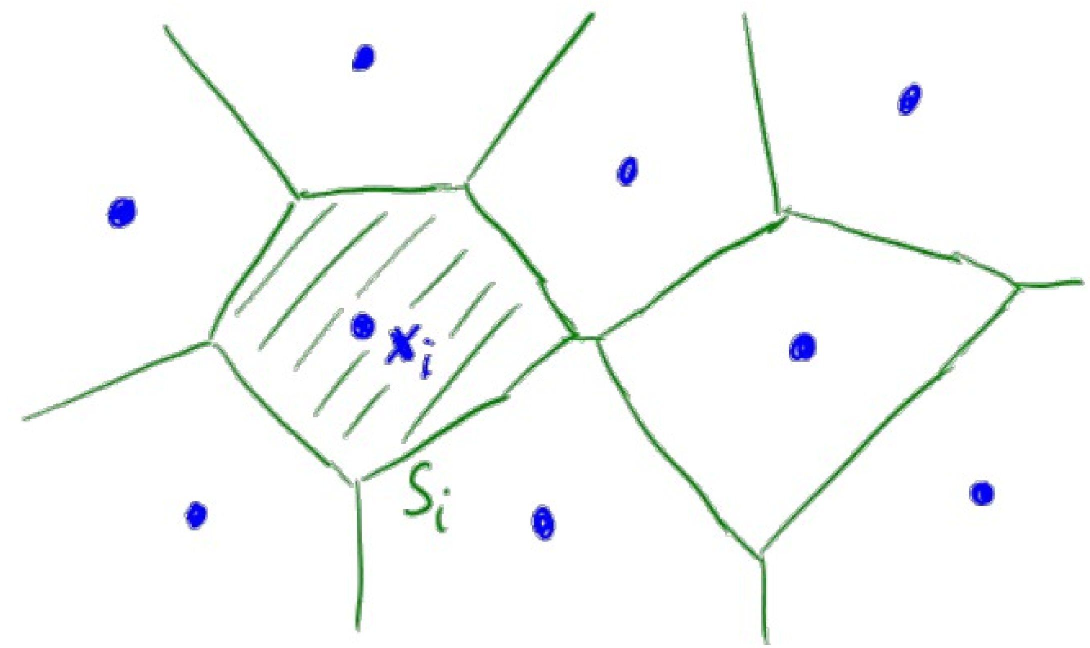
Voronoi diagrams

- Let's start with a set of points $\{x_j \in \mathbb{R}^3\}$
- A Voronoi region of a point x_j is defined as

$$S_i = \{y \in \mathbb{R}^3 : ||y - x_i||_2 < ||y - x_j||_2 \text{for all } j \neq i\}$$

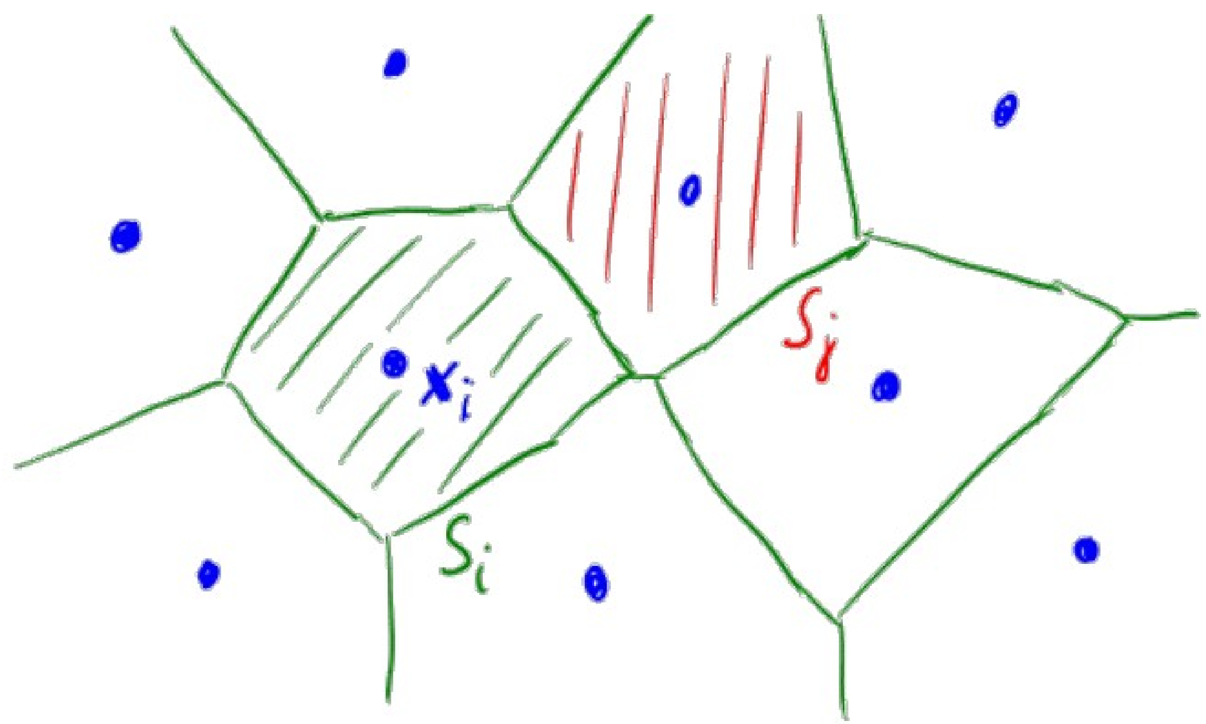
(= the set of points which are closer to point j than to any other point)

- The set of Voronoi regions for all points make up the Voronoi diagram
- Voronoi diagram induces a partition on a given space
- Voronoi regions are convex polytopes (*geometric objects with "flat" sides*)



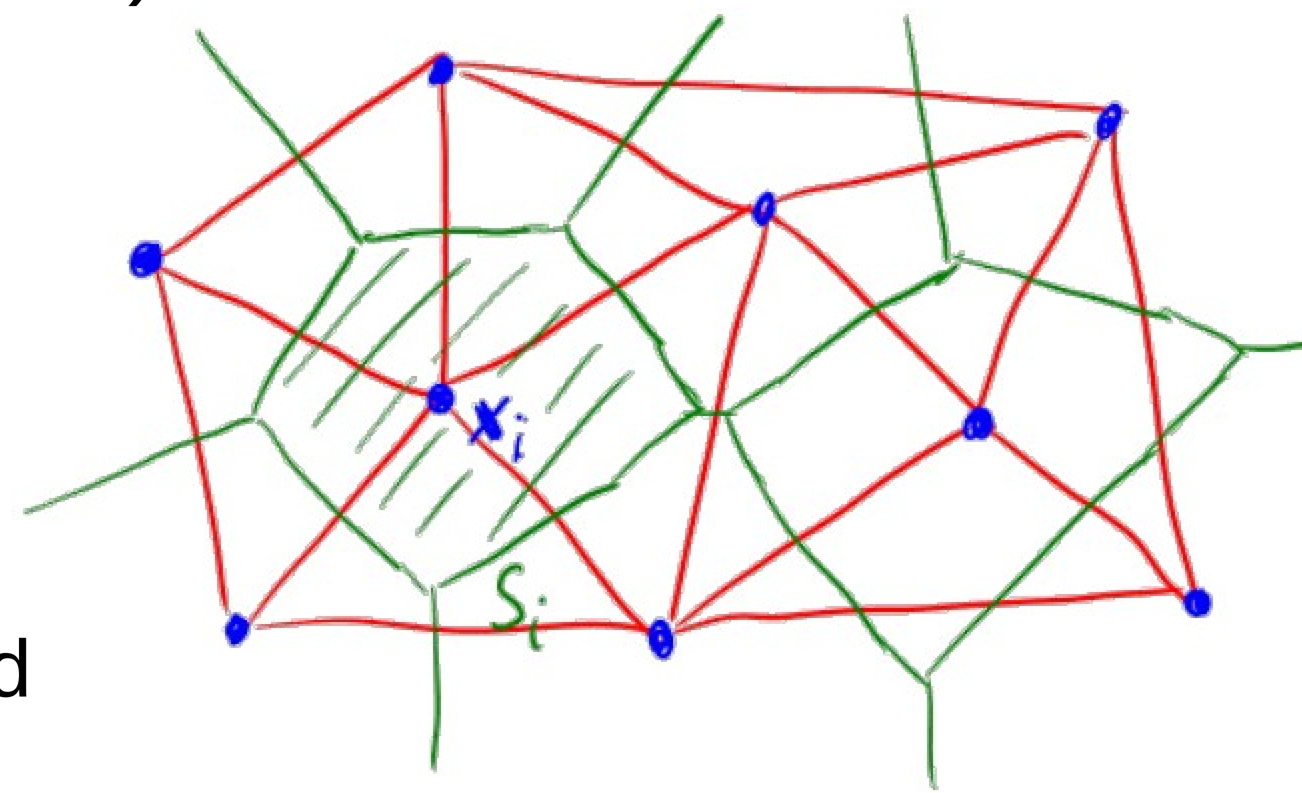
Voronoi diagrams

- If two Voronoi regions i and j share a common edge, the points x_i and x_j are called natural neighbours



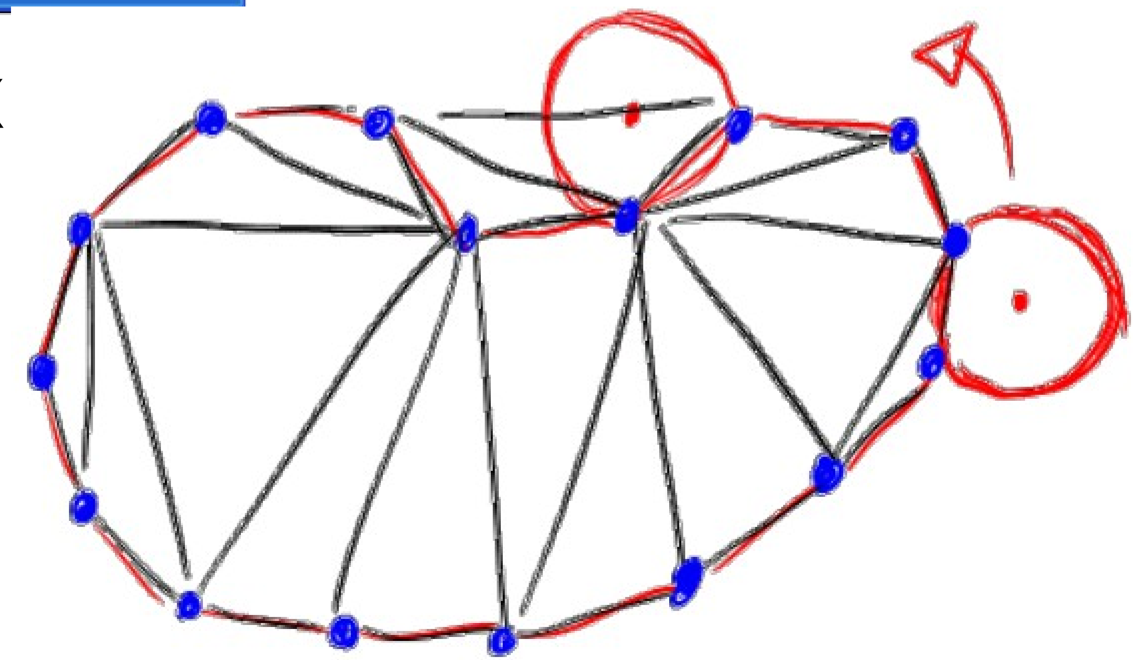
Delaunay tetrahedrization

- Connecting all natural neighbors in a Voronoi diagram leads to a Delaunay tetrahedrization (Boris Delaunay, 1890-1980)
  - Delaunay tetrahedrization partitions the convex hull of the point cloud
  - The convex hull is the smallest convex polytope that includes all points of the point cloud



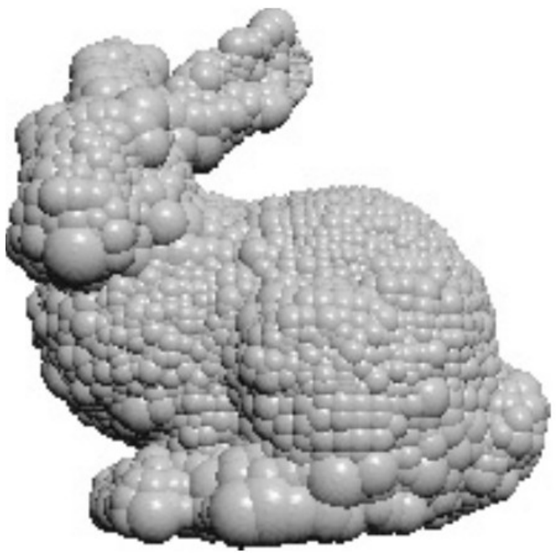
Delaunay tetrahedrization

- How to deal with non-convex shapes?
- To reconstruct surfaces of non-convex objects, we need to carve out concave regions off the convex hull
- Carving can be performed using a sphere with radius α, so-called α-shapes



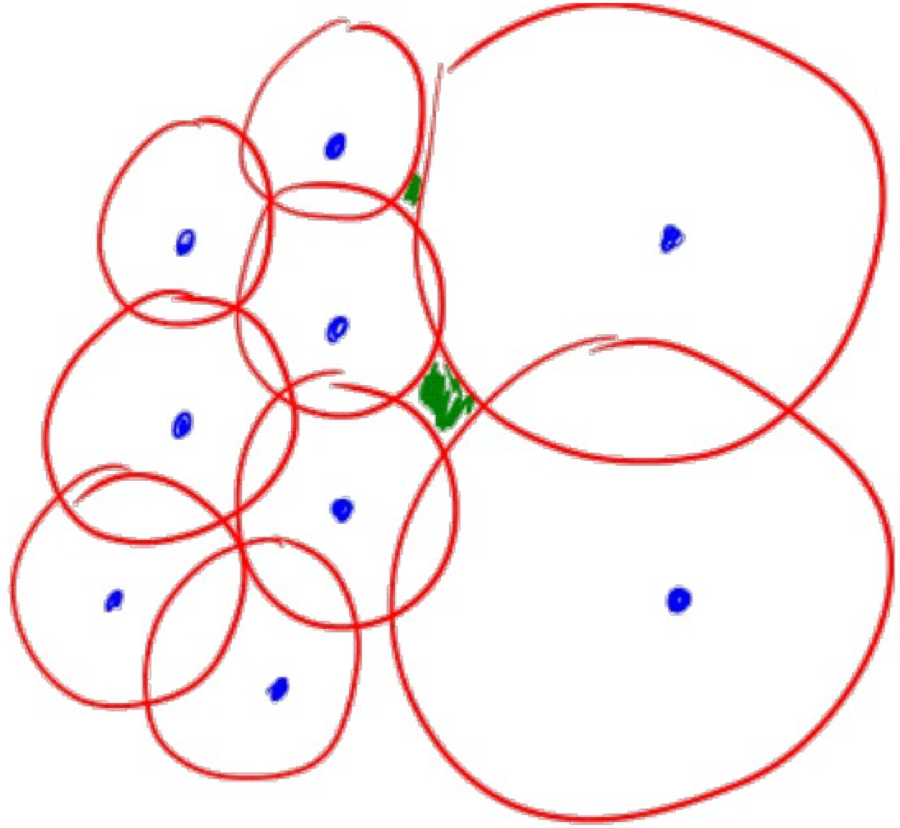
Rendering point clouds

- Rendering can be also done without surface reconstruction



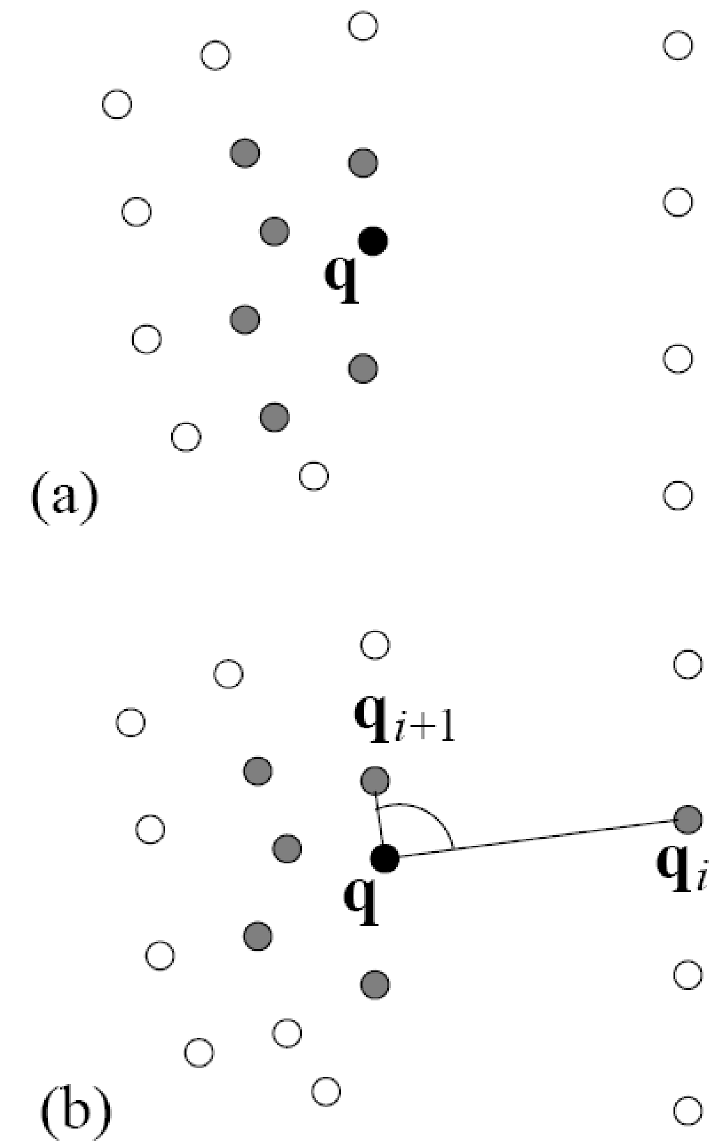
Rendering point clouds

- It is usually done by adding a volume around every point
  - Done in an isotropic fashion, it leads to spheres
  - The size of the spheres must be chosen appropriately such that the spheres overlap minimally but still do not exhibit holes between them
  - This is, again, driven by the sampling rate!

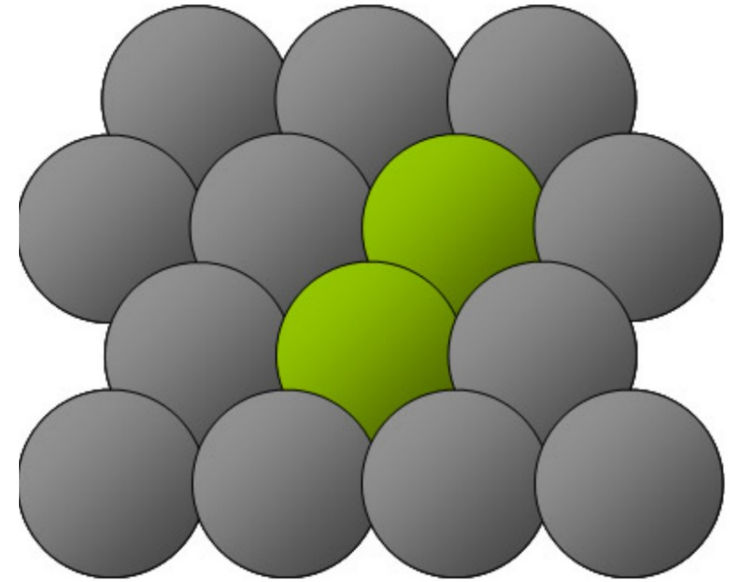


Rendering point clouds

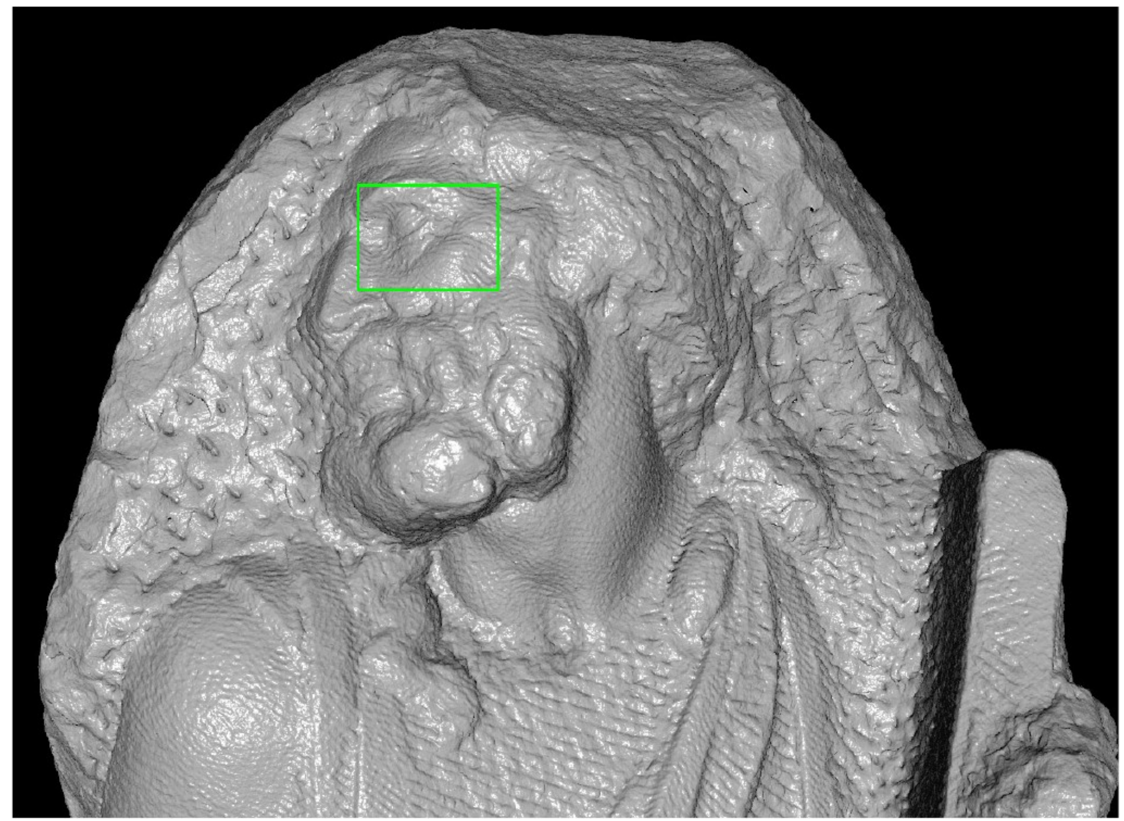
- How to estimate a good sphere radius?
  - The size of the spheres must be determined locally
  - It can be estimated by computing distances to neighboring points
    - natural neighbors
    - k nearest neighbors with fixed k
    - k nearest neighbors with variable k
    - k nearest neighbors with angle criterion



Splatting



- Rather than drawing isotropic extensions, one would like to draw extensions only in directions tangential to the surface
- Spheres are replaced by spherical disks
- The disks are called **splats**
- This approach is called **splatting**



Splatting

- Splats can be regarded as the intersection between the sphere and the tangential plane in the respective point
  - How to obtain the latter?
  - We can also express this problem in terms of normal vector

Splatting

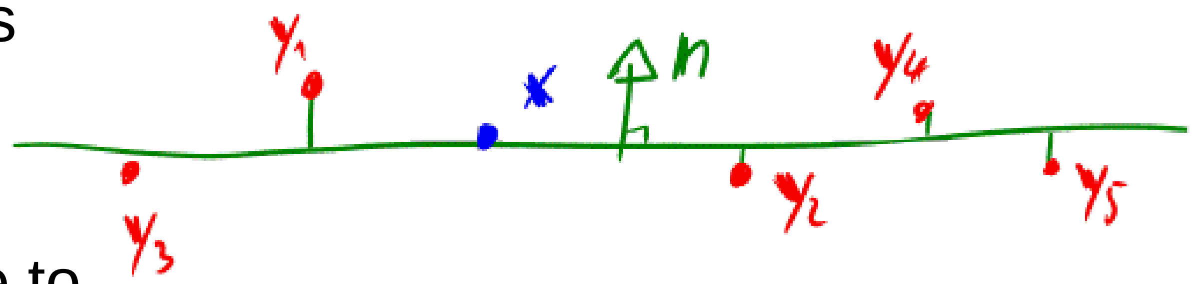
- Fitting a plane through the current point (x) and its neighbours ($y_{1..5}$) such that the distances from the points to the plane is minimal in the least squares sense
- Let $y_0 = x$ and the plane by given by $a_1x+a_2y+a_3z+a_4=0$. Ideally, we would like to have for $i = 1,\dots,n$ that

$$\mathbf{y}_i\mathbf{A} = 0$$

$$\mathbf{YA} = \mathbf{0} \qquad \mathbf{Y} = \begin{bmatrix} \mathbf{y}_0 \\ \vdots \\ \mathbf{y}_n \end{bmatrix}$$

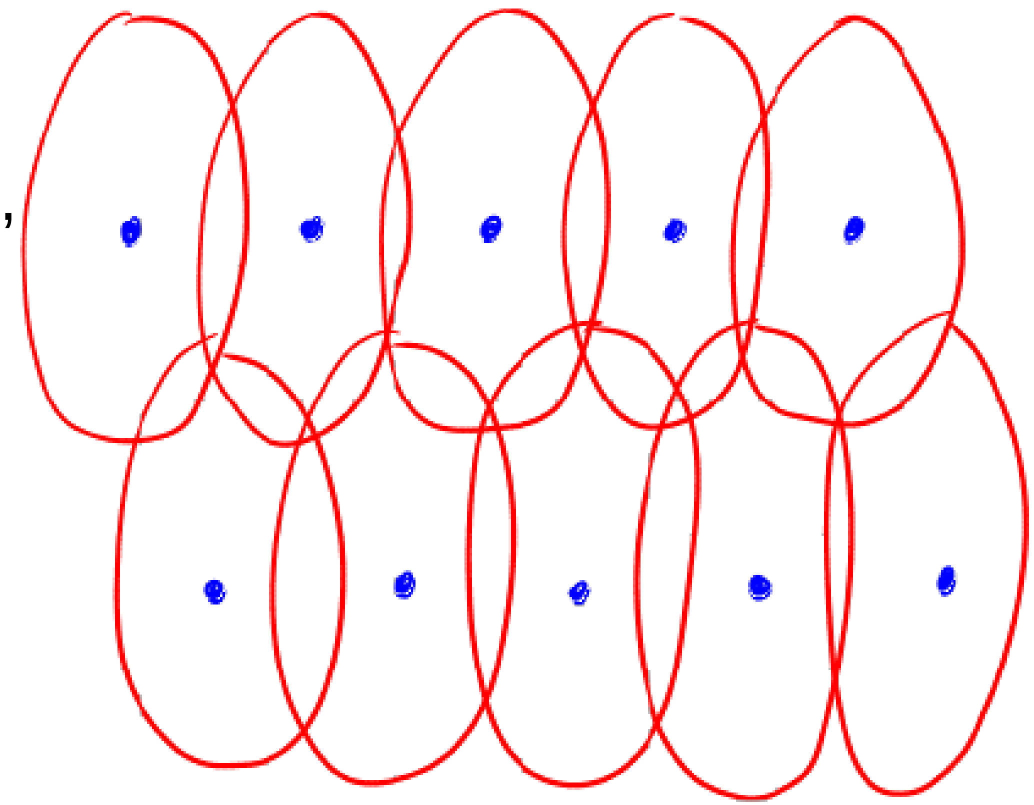
- We can use the least square approach to find the solution:

$$\mathbf{Y}^\top\mathbf{YA} = \mathbf{0}$$



Splatting

- In case of anisotropic sampling, circular splats can be replaced by elliptical splats
- To produce nicer transitions between splats, the splats are drawn with increasing transparency towards their border
- Transparency transition can be determined using Gaussian filters



Thank you!

- Questions?