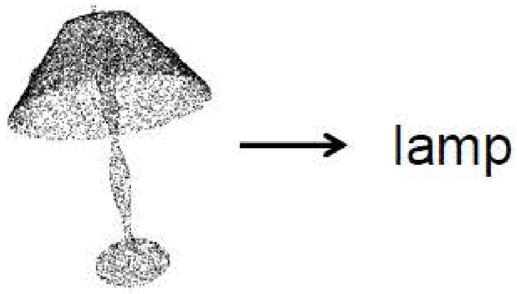
A mesh from a 3D point cloud

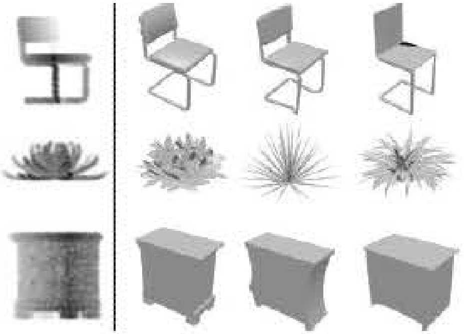
Václav Hlaváč
Czech Technical University in Prague
Czech Institute of Informatics, Robotics and Cybernetics
160 00 Prague 6, Jugoslávských partyzánů 1580/3, Czech Republic
http://people.ciirc.cvut.cz/hlavac; vaclav.hlavac@cvut.cz

Courtesy: Maks Ovsjanikov, Helmut Pottmann

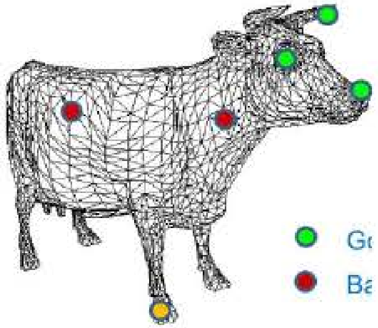
Motivation, application tasks



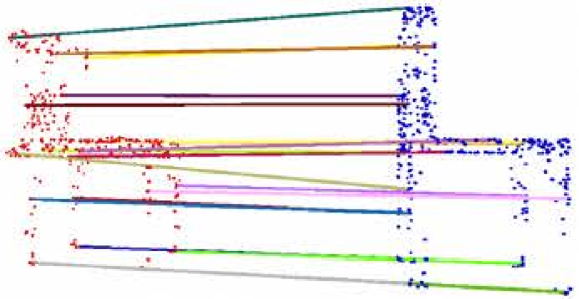
shape classification



shape retrieval

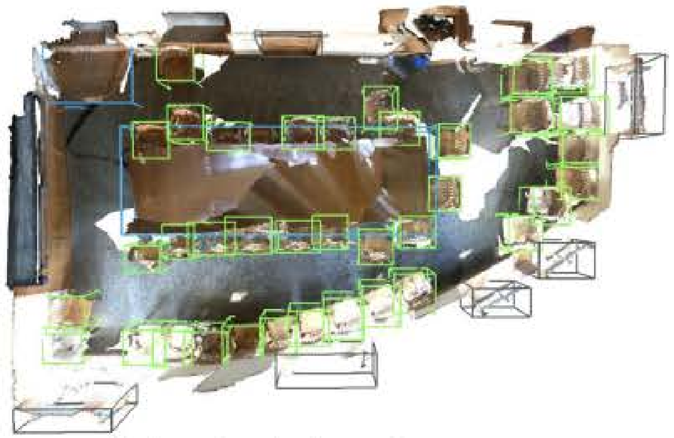


keypoint detection

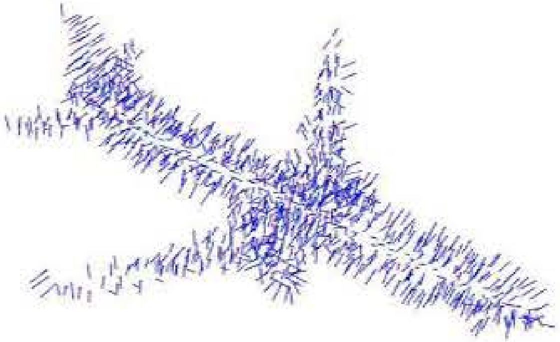


shape correspondence

semantic segmentation



object detection

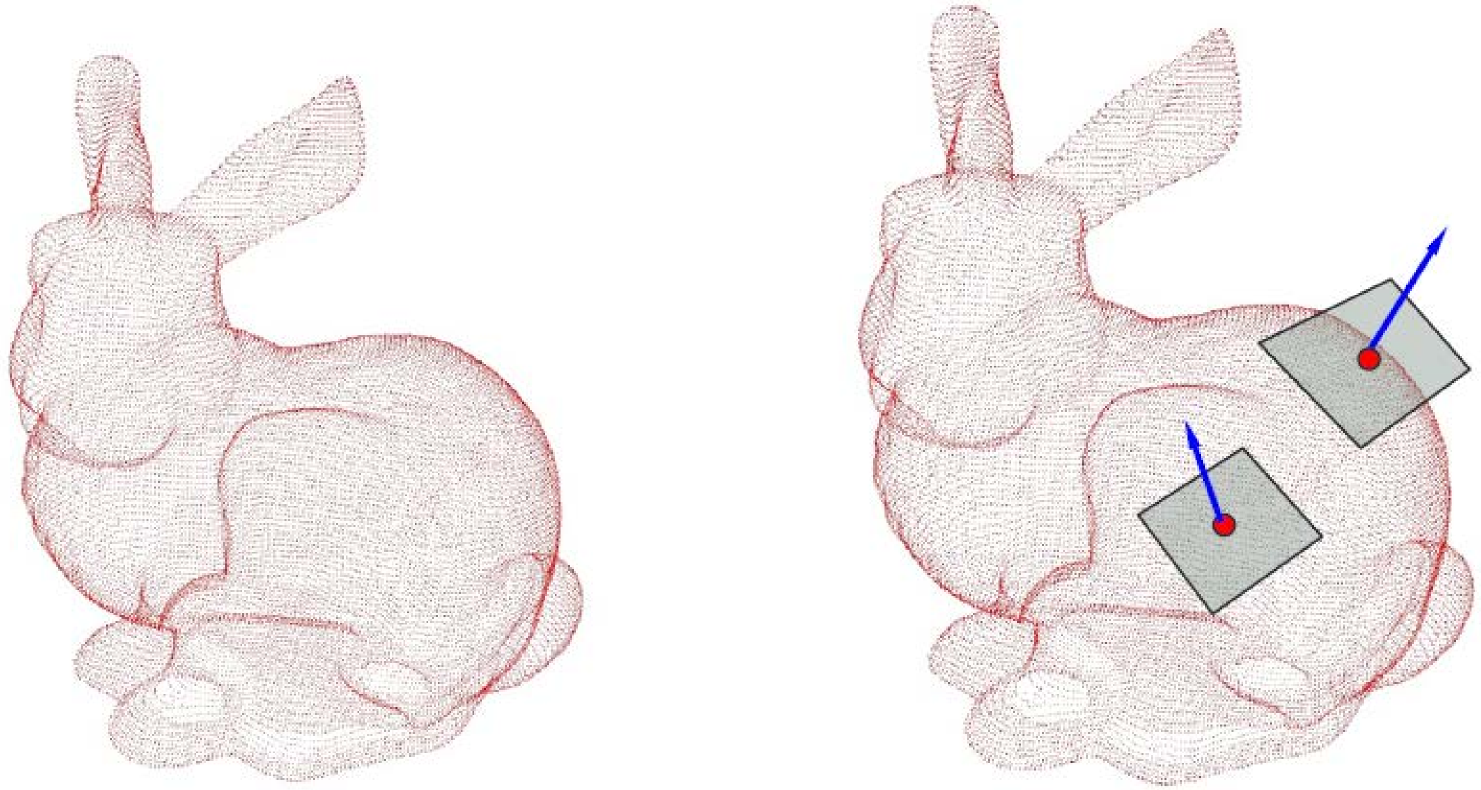


normal estimation

......

3D point cloud

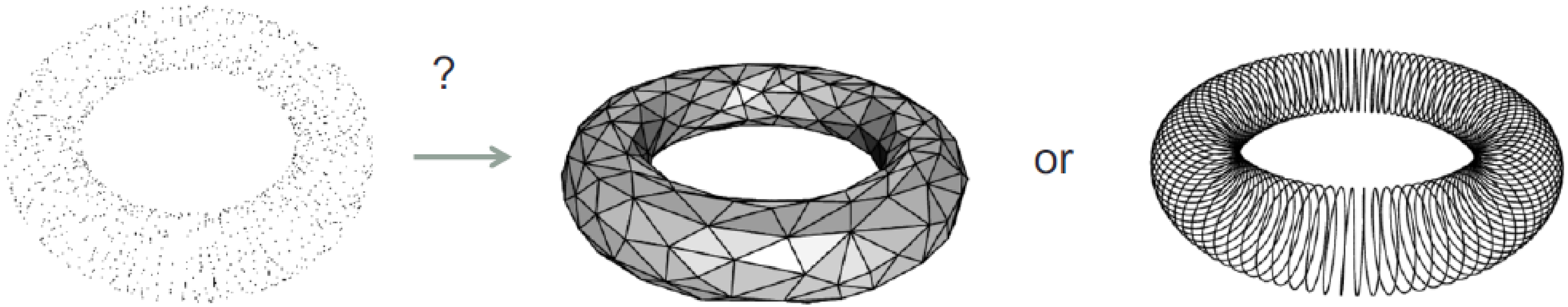
- A point cloud is a set of data points in (3D) space.
- Simplest representation: **only points**, no connectivity.
- Collection of (x,y,z) coordinates, possibly with normals.
- Severe limitations
  - No simplification or subdivision
  - No direct smooth rendering
  - No topological information



Courtesy: Maks Ovsjanikov

3D point cloud, severe limitations

- No simplification or subdivision
- No direct smooth rendering
- No topological information



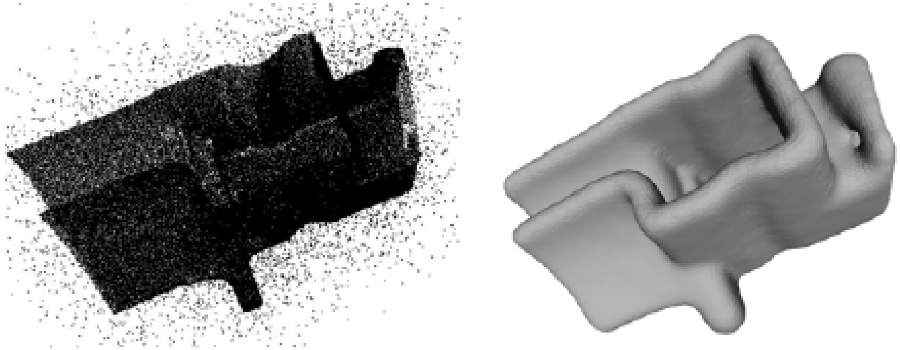
Why point clouds?

- Typically, it is the only measured data which is available
- Most of 3D scanning devices produce point clouds.

Typical scanning and reconstruction pipeline

Use of 3D point cloud

- Point set registration
  - Point clouds are often aligned with 3D models or with other point clouds.
- Conversion of a 3D point cloud to a 3D surface to
  - polygon mesh or triangle mesh models
  - NURBS surface models
  - CAD models



Range image vs. point cloud, a terminology note

Range image

- It is really 2.5D.
- x, y coordinates often constitute a matrix grid of evenly spaced samples (points). A rather strong neighborhood (topological) constraint is available.
- z value corresponds to depth (range) from the observer.
- Presumption: no overlapping surfaces (points) within the image, i.e. only one z value per x, y coordinates is allowed.
- Multi-return LiDARs produce such multi depth z data.

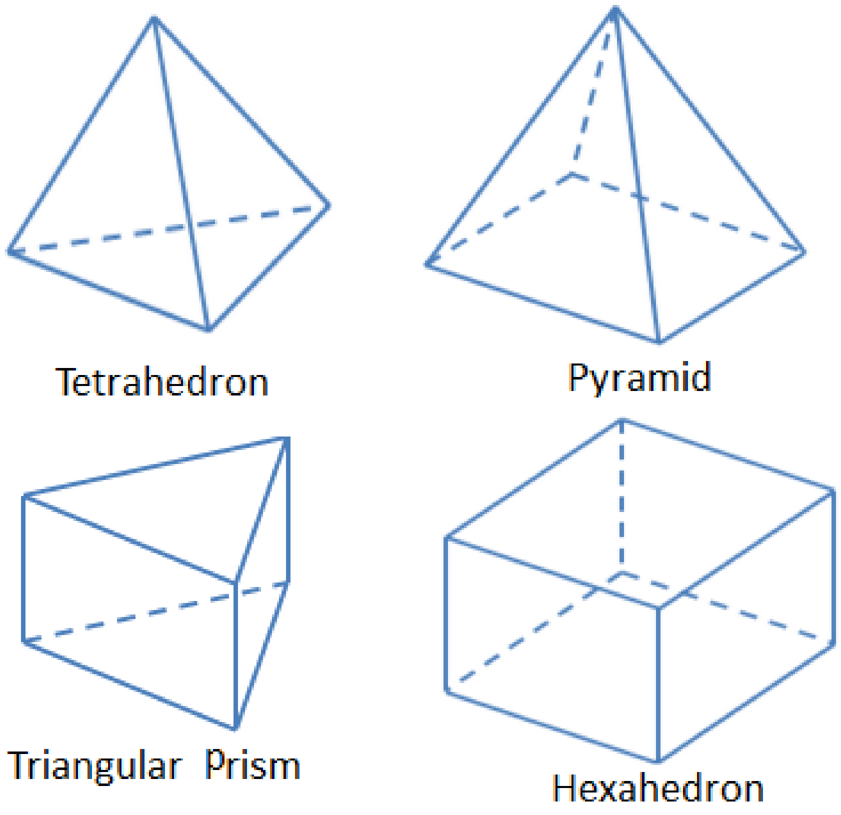
Point cloud

- It is a collection of unorganized 3D points in general.
- The point x, y, z does not assume an underlying matrix grid on x, y. A point cloud, if it is produced from multiple scans cannot be defined by a 2.5D image as there are overlapping points. This makes it fully 3D.

Note: Knowledge about the neighborhood relation depends on a particular sensor or merging method used to fuse individual observations (either multi-view or multi-modality).

Polygon mesh

- A polygon mesh is a representation of a larger geometric domain by smaller discrete cells.
- Meshes are commonly used to
  - compute solutions of partial differential equations (e.g. in Finite elements method),
  - render in computer graphics,
  - analyze geographical and cartographic data.
- 2-dimensional: triangle, quadrilateral
- 3-dimensional (our interest): tetrahedron, pyramid, triangular prism, hexahedron, and a general polyhedron.



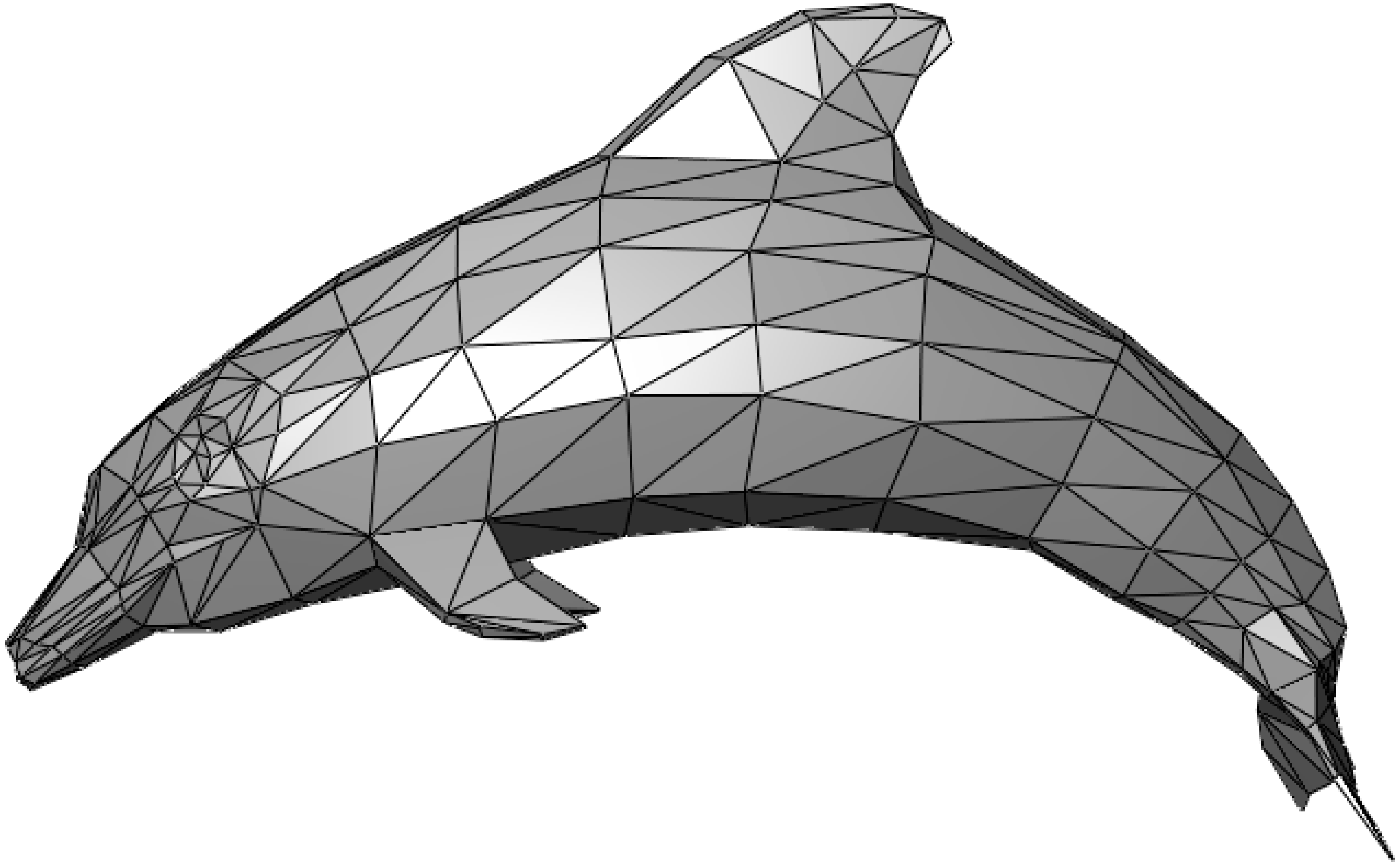
2D / 3D triangulation, definition

A 2d- (3d-) triangulation is a set of triangles (tetrahedra) such that:

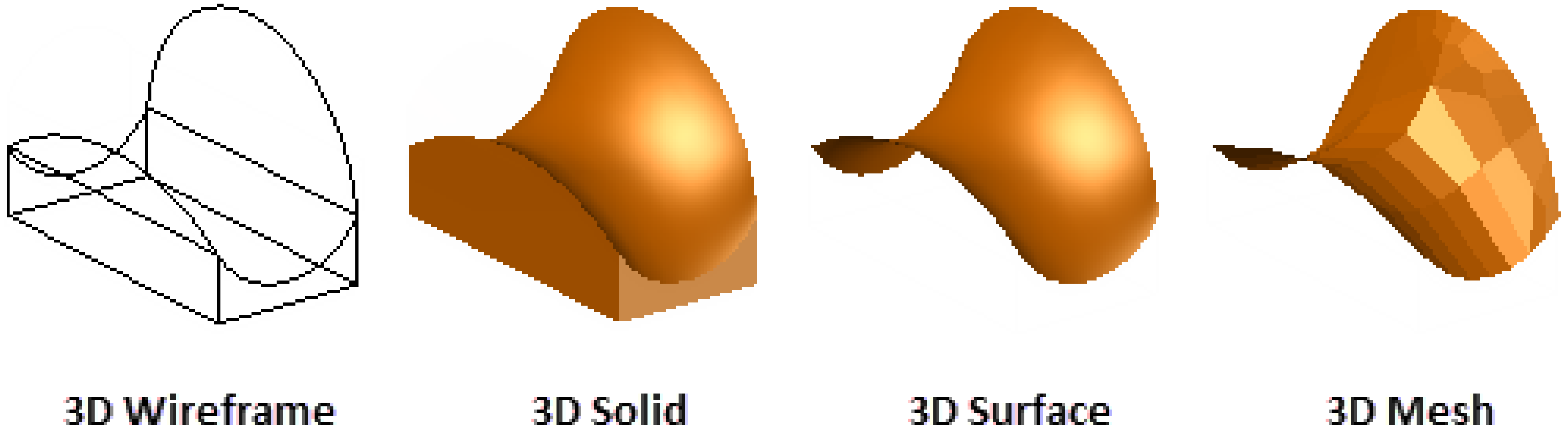
- the set is edge- (facet-) connected
- two triangles (tetrahedra) are either disjoint or share (a facet or) an edge or a vertex

An example, triangle mesh

- A triangle mesh is a type of polygon mesh which is common, e. g. in computer graphics.
- It comprises a set of triangles (typically in 3 dimensions) that are connected by their common edges or corners.



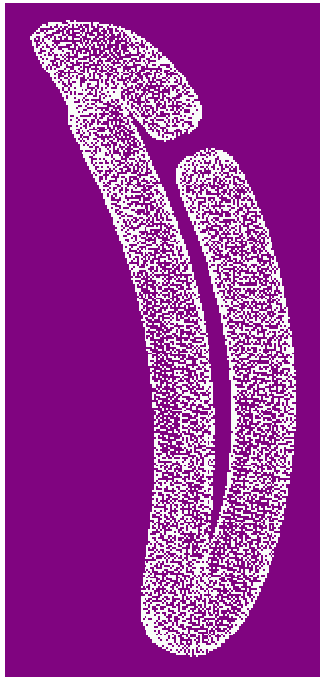
Solid modeling vs. geometric modeling

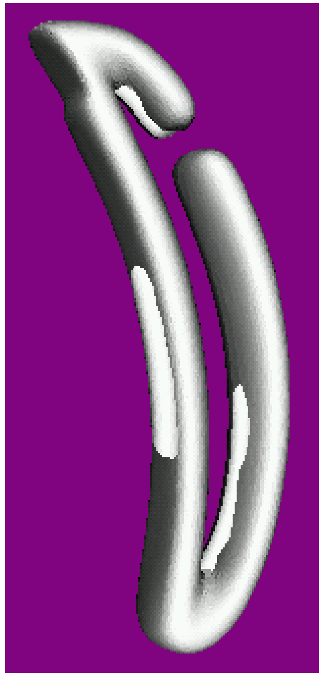


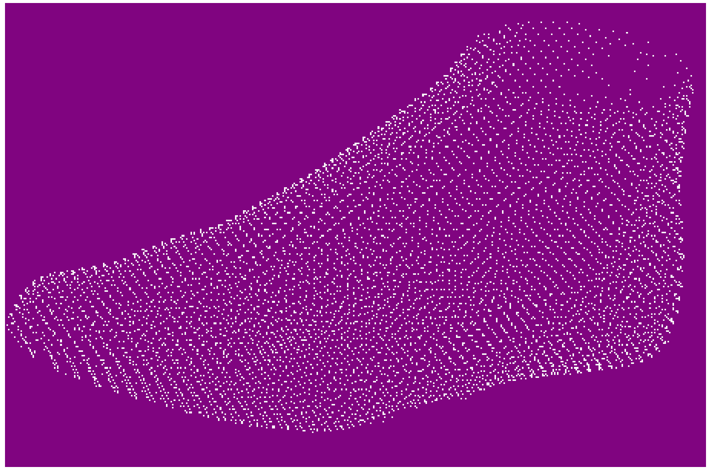
- Solid modeling (emphasizes physics view) https://en.wikipedia.org/wiki/Solid_modeling
- Geometric modeling (emphasizes computer graphics view) https://en.wikipedia.org/wiki/Geometric_modeling

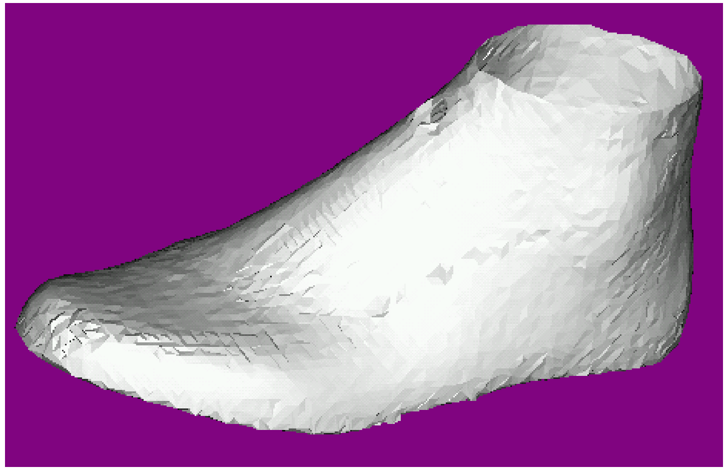
Problem description and two examples

- Unorganised set of points (point cloud) in the 3D space
- Assumed to be on a 2D surface.
- Reconstruct surface by creating a triangulation
  - Interpolation: Use only given points as vertices
  - Approximation: Allowed to use artificial vertices
- Related problem: reconstructing curves in 2D or 3D



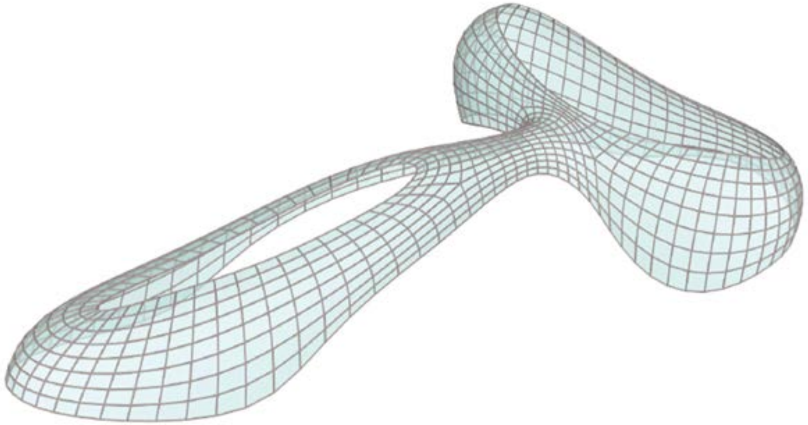


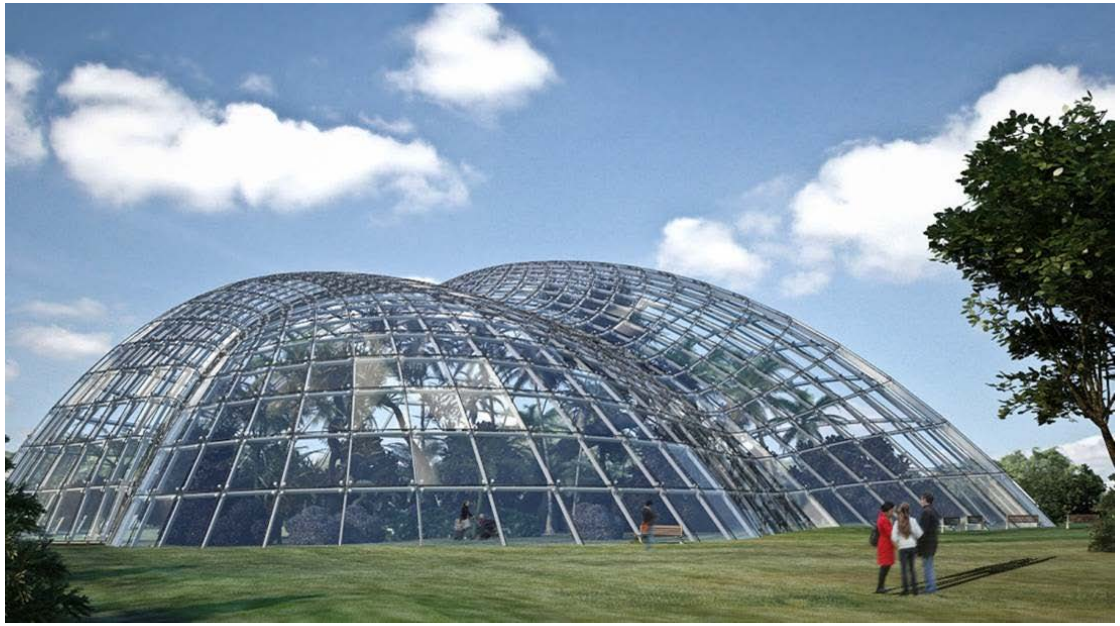




Topological model of the solid surface

- A surface can be represented as 2D manifold embedded in 3D Euclidean space.
- I was motivated by Helmut Pottmann's research in computer graphics.
- I sought a representation of a free-folded piece of cloth, e.g. a towel, including a rough representation of the 'invisible'.





3D Point cloud processing

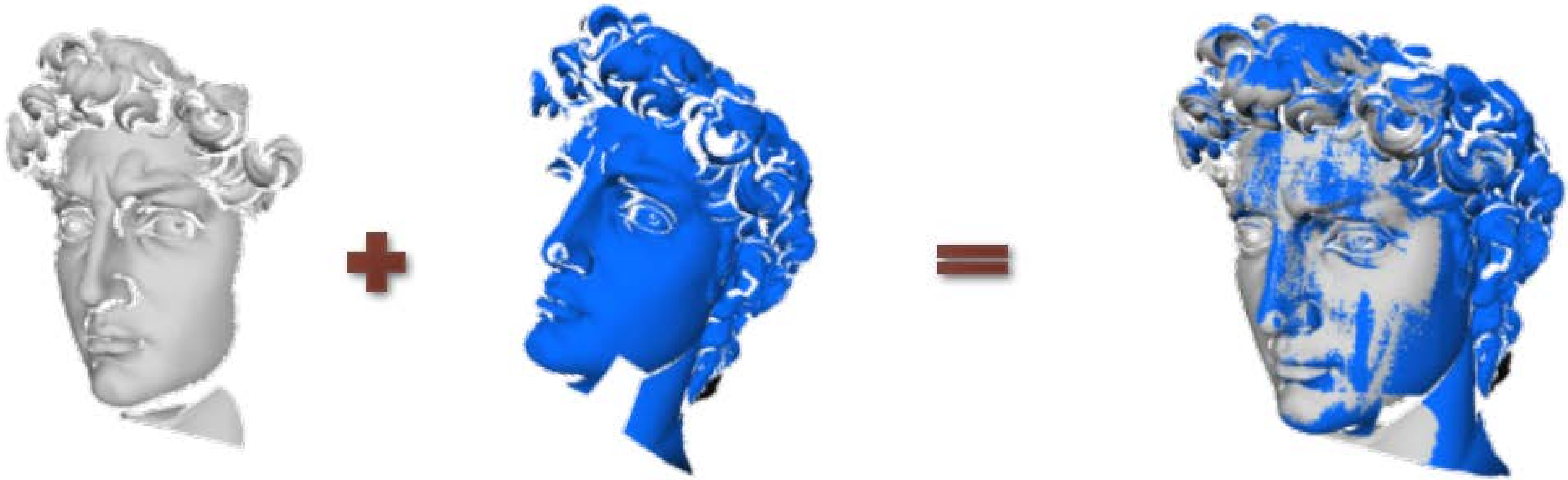
Typically point cloud sampling of a shape is insufficient for most applications.

Main processing stages stages:

1. Shape scanning (Acquisition).
2. If you have multiple scans, align them. (Registration)
3. Smoothing – remove local noise.
4. Estimate surface normals.
5. Surface reconstruction (aka 3D point cloud reconstruction)

Fundamental registration problem

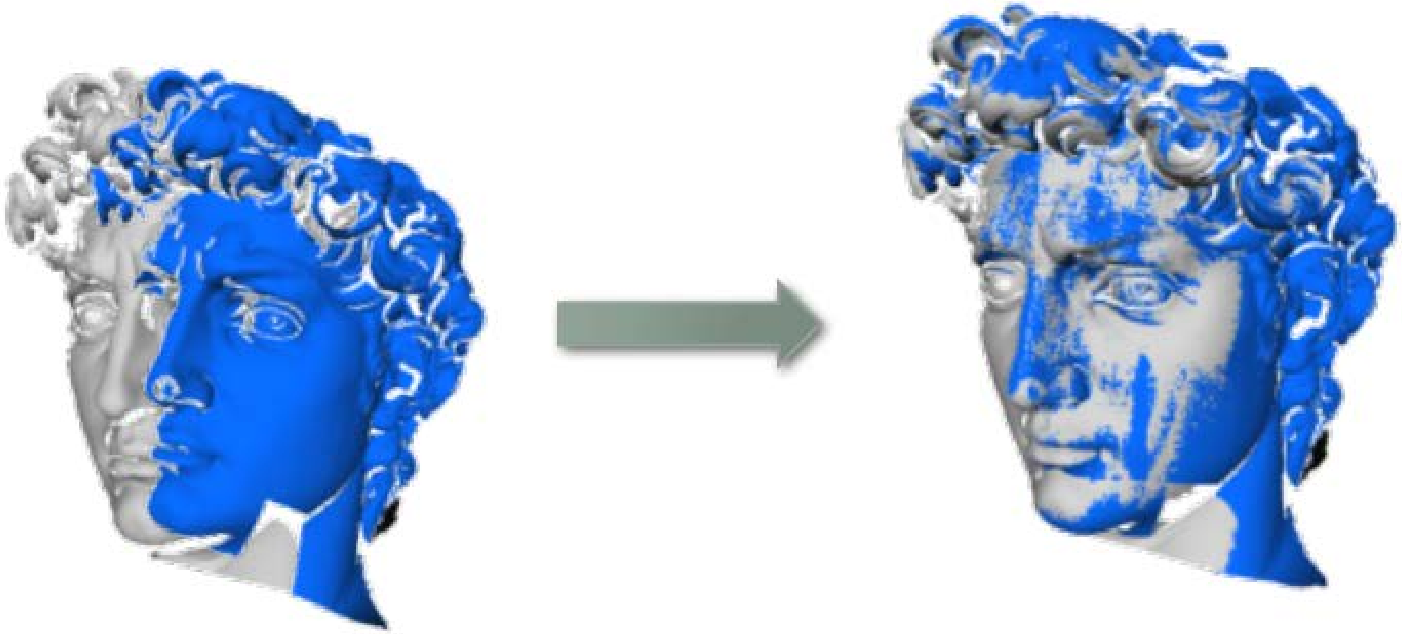
Given (at least) two shapes with partially overlapping geometry, find an alignment between them.



Iterative Closest Points (ICP) algorithm is the base solution.

Local 3D point cloud alignment

- The simplest instance of the registration problem.
- Given two shapes that are approximately aligned (e.g. by a human) we want to find the optimal transformation.
- Intuition: want corresponding points to be close after transformation.
- We do not know:
  - What points correspond;
  - What is the optimal alignment.



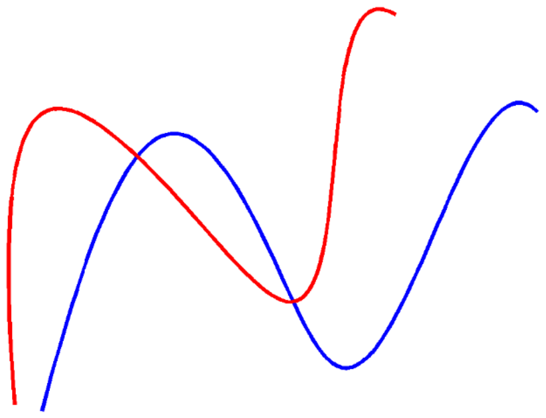
Iterative Closest Point (ICP)

- Approach: iterate between finding correspondences and finding the rigid transformation (rotation $\mathbf{R}$ and translation t).

- Given a pair of shapes, X and Y, iterate:
  1. For each $x_i \in X$ find the **nearest** neighbor $y_i \in Y$
  2. Find deformation minimizing: $\sum_{i=1}^{N} ||\mathbf{R}x_i + t - yi||^2$

Iterative Closest Point (ICP)

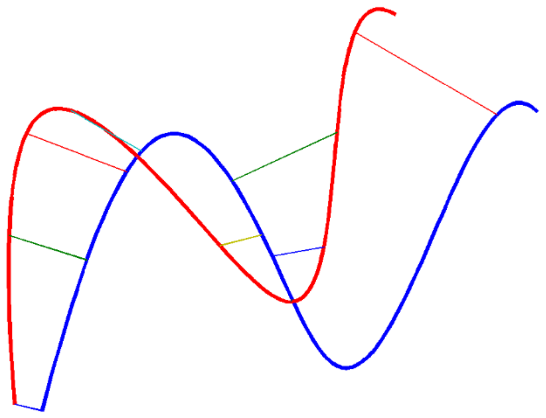
- Approach: iterate between finding correspondences and finding the rigid transformation (rotation $\mathbf{R}$ and translation t).



- Given a pair of shapes, X and Y, iterate:
  1. For each $x_i \in X$ find the **nearest** neighbor $y_i \in Y$
  2. Find deformation minimizing: $\sum_{i=1}^{N} ||\mathbf{R}x_i + t - yi||^2$

Iterative Closest Point (ICP)

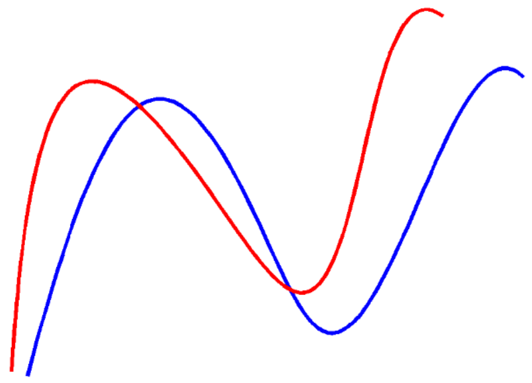
- Approach: iterate between finding correspondences and finding the rigid transformation (rotation $\mathbf{R}$ and translation t).



- Given a pair of shapes, X and Y, iterate:
  1. For each $x_i \in X$ find the **nearest** neighbor $y_i \in Y$
  2. Find deformation minimizing: $\sum_{i=1}^{N} ||\mathbf{R}x_i + t - yi||^2$

Iterative Closest Point (ICP)

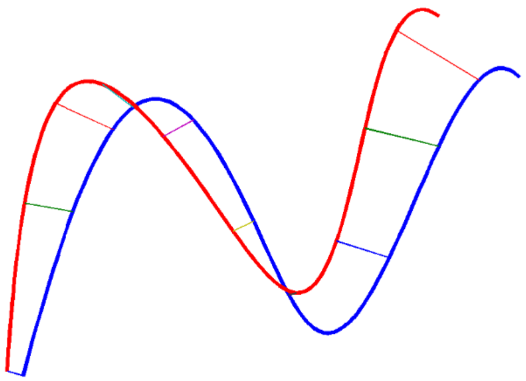
- Approach: iterate between finding correspondences and finding the rigid transformation (rotation $\mathbf{R}$ and translation t).



- Given a pair of shapes, X and Y, iterate:
  1. For each $x_i \in X$ find the **nearest** neighbor $y_i \in Y$
  2. Find deformation minimizing: $\sum_{i=1}^{N} ||\mathbf{R}x_i + t - yi||^2$

Iterative Closest Point (ICP)

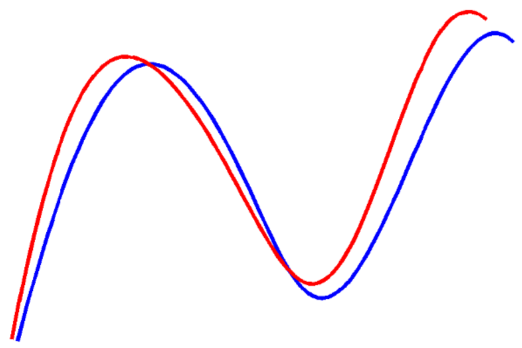
- Approach: iterate between finding correspondences and finding the rigid transformation (rotation $\mathbf{R}$ and translation t).



- Given a pair of shapes, X and Y, iterate:
  1. For each $x_i \in X$ find the **nearest** neighbor $y_i \in Y$
  2. Find deformation minimizing: $\sum_{i=1}^{N} ||\mathbf{R}x_i + t - yi||^2$

Iterative Closest Point (ICP)

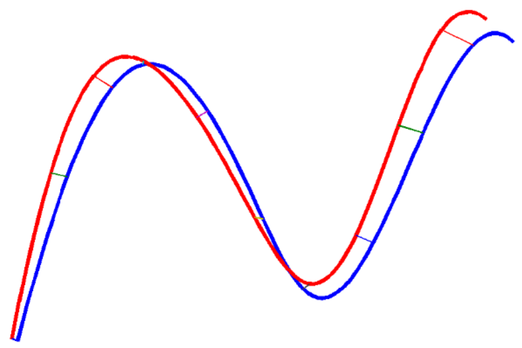
- Approach: iterate between finding correspondences and finding the rigid transformation (rotation $\mathbf{R}$ and translation t).



- Given a pair of shapes, X and Y, iterate:
  1. For each $x_i \in X$ find the **nearest** neighbor $y_i \in Y$
  2. Find deformation minimizing: $\sum_{i=1}^{N} ||\mathbf{R}x_i + t - yi||^2$

Iterative Closest Point (ICP)

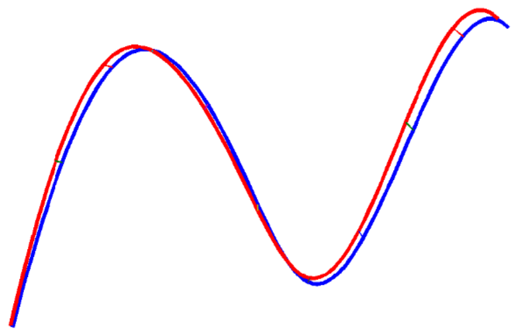
- Approach: iterate between finding correspondences and finding the rigid transformation (rotation $\mathbf{R}$ and translation t).



- Given a pair of shapes, X and Y, iterate:
  1. For each $x_i \in X$ find the **nearest** neighbor $y_i \in Y$
  2. Find deformation minimizing: $\sum_{i=1}^{N} ||\mathbf{R}x_i + t - yi||^2$

Iterative Closest Point (ICP)

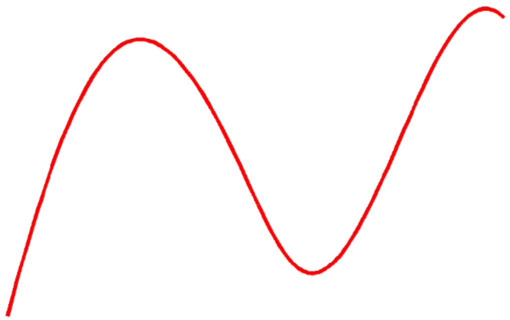
- Approach: iterate between finding correspondences and finding the rigid transformation (rotation $\mathbf{R}$ and translation t).



- Given a pair of shapes, X and Y, iterate:
  1. For each $x_i \in X$ find the **nearest** neighbor $y_i \in Y$
  2. Find deformation minimizing: $\sum_{i=1}^{N} ||\mathbf{R}x_i + t - yi||^2$

Iterative Closest Point (ICP)

- Approach: iterate between finding correspondences and finding the rigid transformation (rotation $\mathbf{R}$ and translation t).



- Given a pair of shapes, X and Y, iterate:
  1. For each $x_i \in X$ find the **nearest** neighbor $y_i \in Y$
  2. Find deformation minimizing: $\sum_{i=1}^{N} ||\mathbf{R}x_i + t - yi||^2$

3D Point cloud processing

Typically point cloud sampling of a shape is insufficient for most applications.

Main processing stages stages:

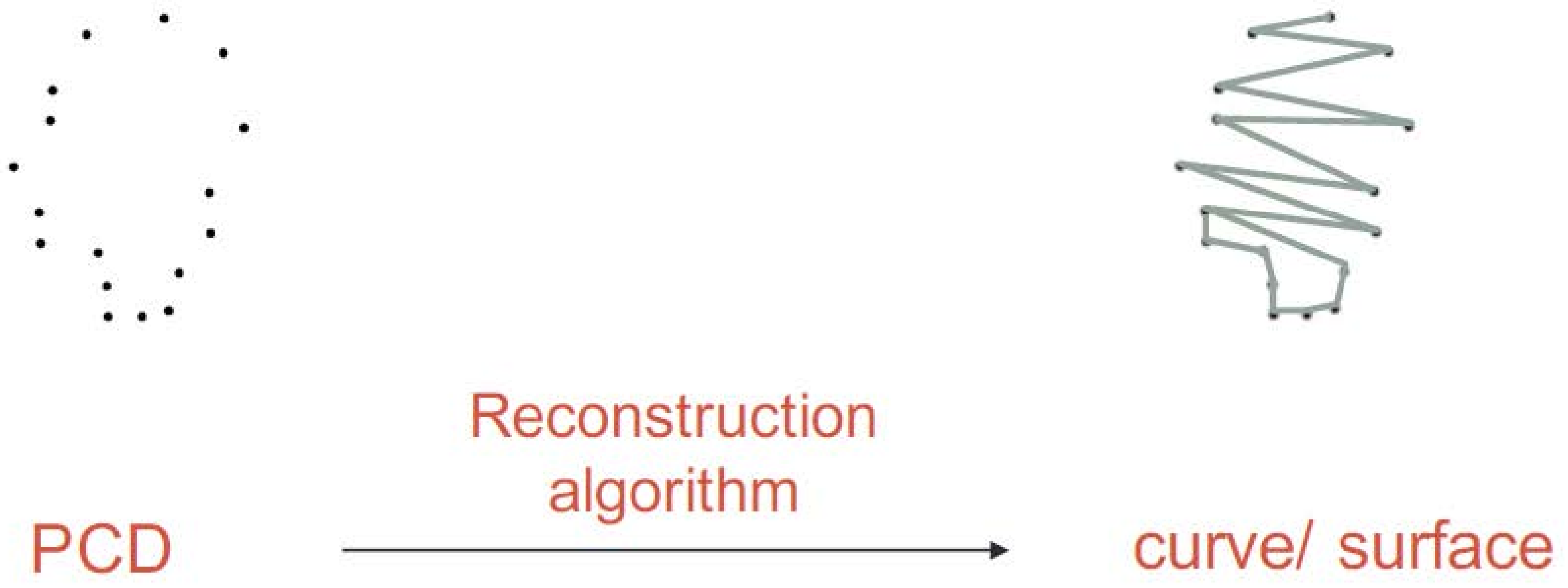
1. Shape scanning (Acquisition).
2. If you have multiple scans, align them. (Registration)
3. Smoothing – remove local noise.
4. Estimate surface normals.
5. Surface reconstruction (aka 3D point cloud reconstruction)

3D point cloud reconstruction

Main trouble:

Unstructured data. Data points are not ordered. The problem is inherently ill-posed, i.e. difficult.

2D example



3D point cloud reconstruction