valeo.ai

# Nuages de Points et Modélisation 3D

*3 - From local properties to surface reconstruction*

# Alexandre Boulch (that's me)

## CV

- Senior scientist at Valeo in the team valeo.ai.
- Researcher at ONERA
- Thesis at ENPC


## Research

- 3D understanding of scenes
  - From point clouds or images
- End-to-end driving
  - From sensor to decision

valeo.ai

**A Paris-based research team**
working on Valeo's AI applications, esp. perception for ADAS/AD

**Open research**

- Publications in Conferences / Journals
- Code publication under open source license

https://www.valeo.com/en/valeo-ai/

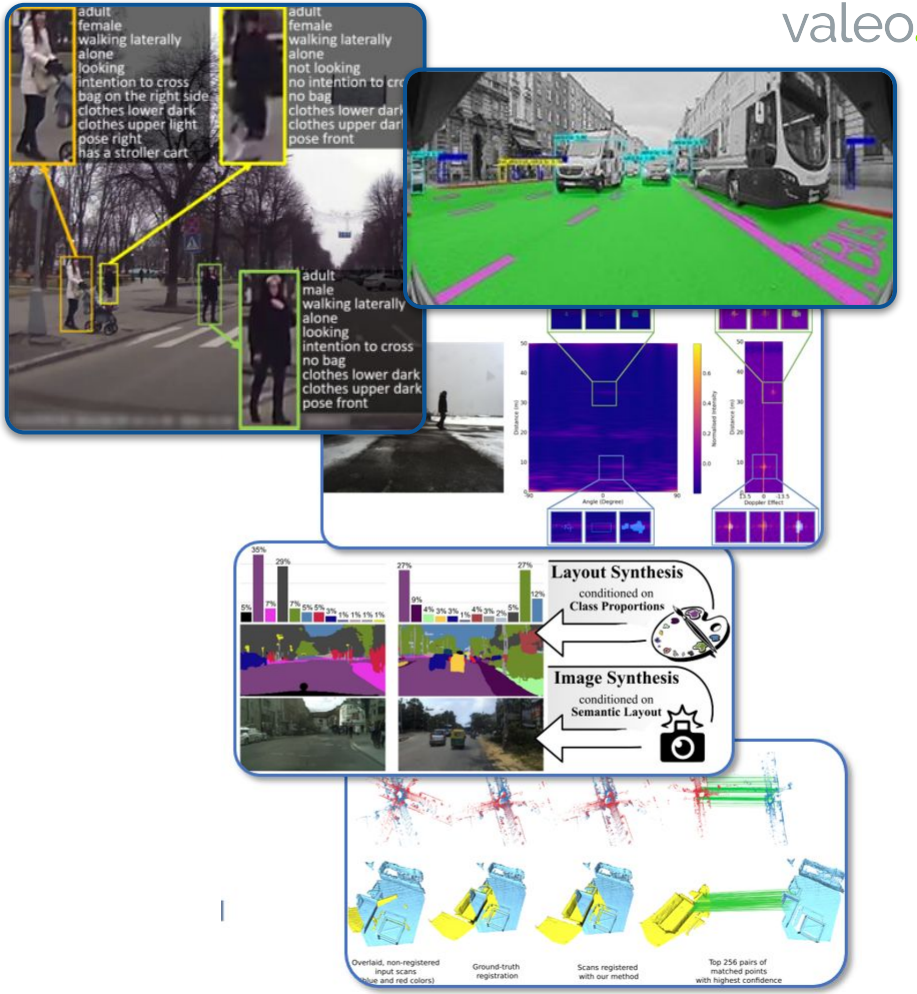
https://valeoai.github.io/blog/

https://github.com/valeoai/

https://twitter.com/valeoai

valeo.ai

Collaborations with Valeo's AI network (150+) and world-class academics.



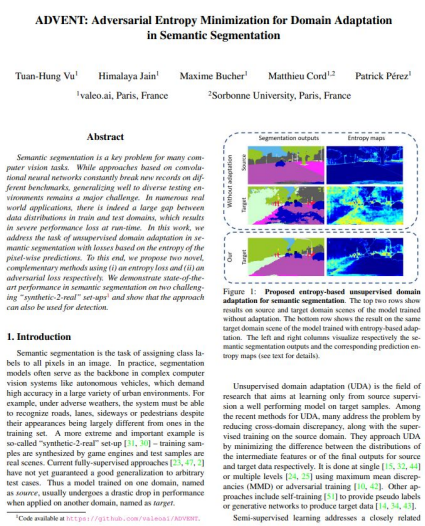
valeo.ai

## 100 publications

CVPR (25), ICCV+ECCV (22)

NeurIPS+ICLR+ICML (7)

11 Journals



**ADVENT: Adversarial Entropy Minimization for Domain Adaptation in Semantic Segmentation**

Tuan-Hung Vu[1] Himalaya Jain[1] Maxime Bucher[1] Matthieu Cord[1,2] Patrick Pérez[1]
[1]valeo.ai, Paris, France [2]Sorbonne University, Paris, France

**Abstract**

*Semantic segmentation is a key problem for many computer vision tasks. While approaches based on convolutional neural networks constantly break new records on different benchmarks, generalizing well to diverse testing environments remains a major challenge. In numerous real world applications, there is indeed a large gap between data distributions in train and test domains, which results in severe performance loss at run-time. In this work, we address the task of unsupervised domain adaptation in semantic segmentation with losses based on the entropy of the pixel-wise predictions. To this end, we propose two novel, complementary methods using (i) an entropy loss and (ii) an adversarial loss respectively. We demonstrate state-of-the-art performance in semantic segmentation on two challenging "synthetic-2-real" set-ups[1] and show that the approach can also be used for detection.*

**1. Introduction**

Semantic segmentation is the task of assigning class labels to all pixels in an image. In practice, segmentation models often serve as the backbone in complex computer vision systems like autonomous vehicles, which demand high accuracy in a large variety of urban environments. For example, under adverse weathers, the system must be able to recognize roads, lanes, sideways or pedestrians despite their appearances being largely different from ones in the training set. A more extreme and important example is so-called "synthetic-2-real" set-up [31, 30] – training samples are synthesized by game engines and test samples are real scenes. Current fully-supervised approaches [23, 47, 2] have not yet guaranteed a good generalization to arbitrary test cases. Thus a model trained on one domain, named as *source*, usually undergoes a drastic drop in performance when applied on another domain, named as *target*.

[1]Code available at https://github.com/valeoai/ADVENT.

Figure 1: **Proposed entropy-based unsupervised domain adaptation for semantic segmentation**. The top two rows show results on source and target domain scenes of the model trained without adaptation. The bottom row shows the result on the same target domain scene of the model trained with entropy-based adaptation. The left and right columns visualize respectively the semantic segmentation outputs and the corresponding prediction entropy maps (see text for details).

Unsupervised domain adaptation (UDA) is the field of research that aims at learning only from source supervision a well performing model on target samples. Among the recent methods for UDA, many address the problem by reducing cross-domain discrepancy, along with the supervised training on the source domain. They approach UDA by minimizing the difference between the distributions of the intermediate features or of the final outputs for source and target data respectively. It is done at single [15, 32, 44] or multiple levels [24, 28] using maximum mean discrepancies (MMD) or adversarial training [10, 42]. Other approaches include self-training [51] to provide pseudo labels or generative networks to produce target data [14, 34, 43].

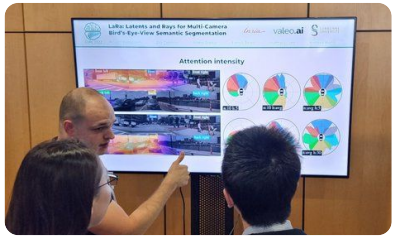
Semi-supervised learning addresses a closely related

## Talks & services

Talks, tutorials, panels

Reviewing and chairing

Juries and committees

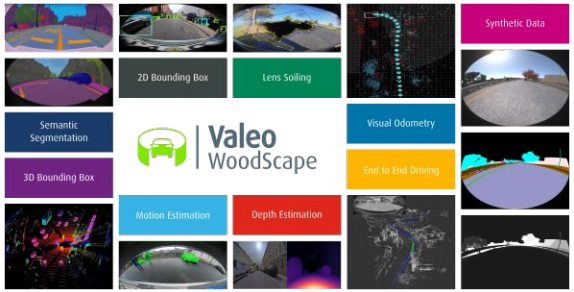


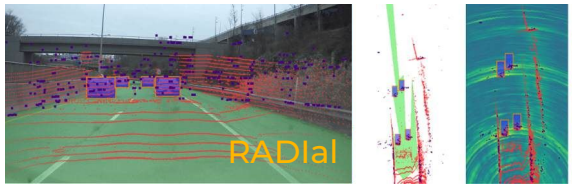


## Code & Datasets

38 research codes

3 datasets



Numpy

Optimization

**Points**

Neural networks

Segmentation

**Machine learning**

Pytorch

Geometry

Surfaces

Differentiable optimization

Scikit-learn

Random forests

Classification

Images

# Overview

valeo.ai

## Machine learning courses

- Surface reconstruction
- ML1
  - Descriptors and machine learning
  - Image based processing
- ML2
  - Geometric deep learning
- ML3
  - Convolutional and Transformer based architectures
- ML4
  - Tasks and corresponding architectures
- Opening session

# Evaluation

valeo.ai

Practical sessions

- ML2 and ML4 are evaluated

Final evaluation

- Scientific paper reading and analysis
- 10 minutes presentation with slides
- 10 questions and discussion
- Scheduled after ML4

# Ressources

valeo.ai

Web page: https://boulch.eu/teaching/

Github: https://github.com/aboulch/MSIA_points

Email: alexandre.boulch@valeo.com

# Overview

I. From point clouds to surfaces
II. Local features
    A. Normal estimation
    B. Normal orientation
III. Surface reconstruction
    A. Ball pivoting
    B. Delauney reconstruction
    C. Poisson reconstruction
    D. RANSAC

# Overview

I. **From point clouds to surfaces**
II. Local features
    A. Normal estimation
    B. Normal orientation
III. Surface reconstruction
    A. Ball pivoting
    B. Delauney reconstruction
    C. Poisson reconstruction
    D. RANSAC

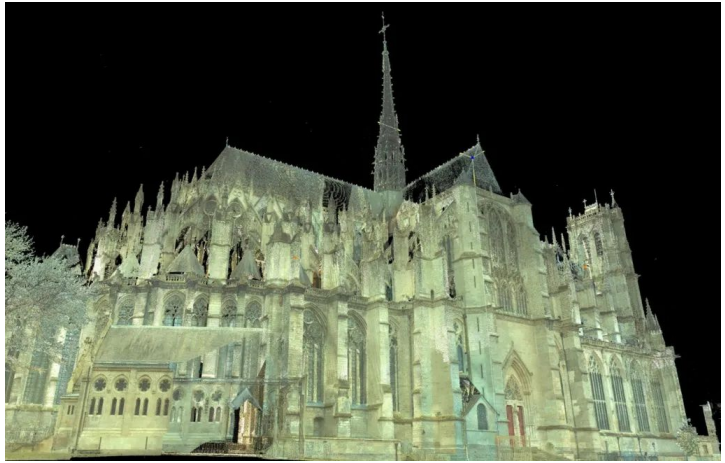
# 3D Rendering

valeo.ai

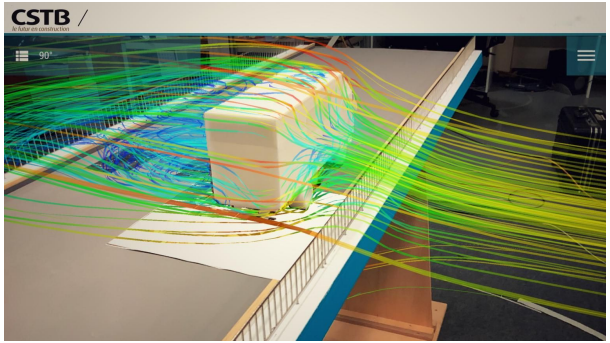
## Archeology

*3D modelling in archaeology: The application of Structure from Motion methods to the study of the megalithic necropolis of Panoria*

## Patrimony saving



Amiens cathedral



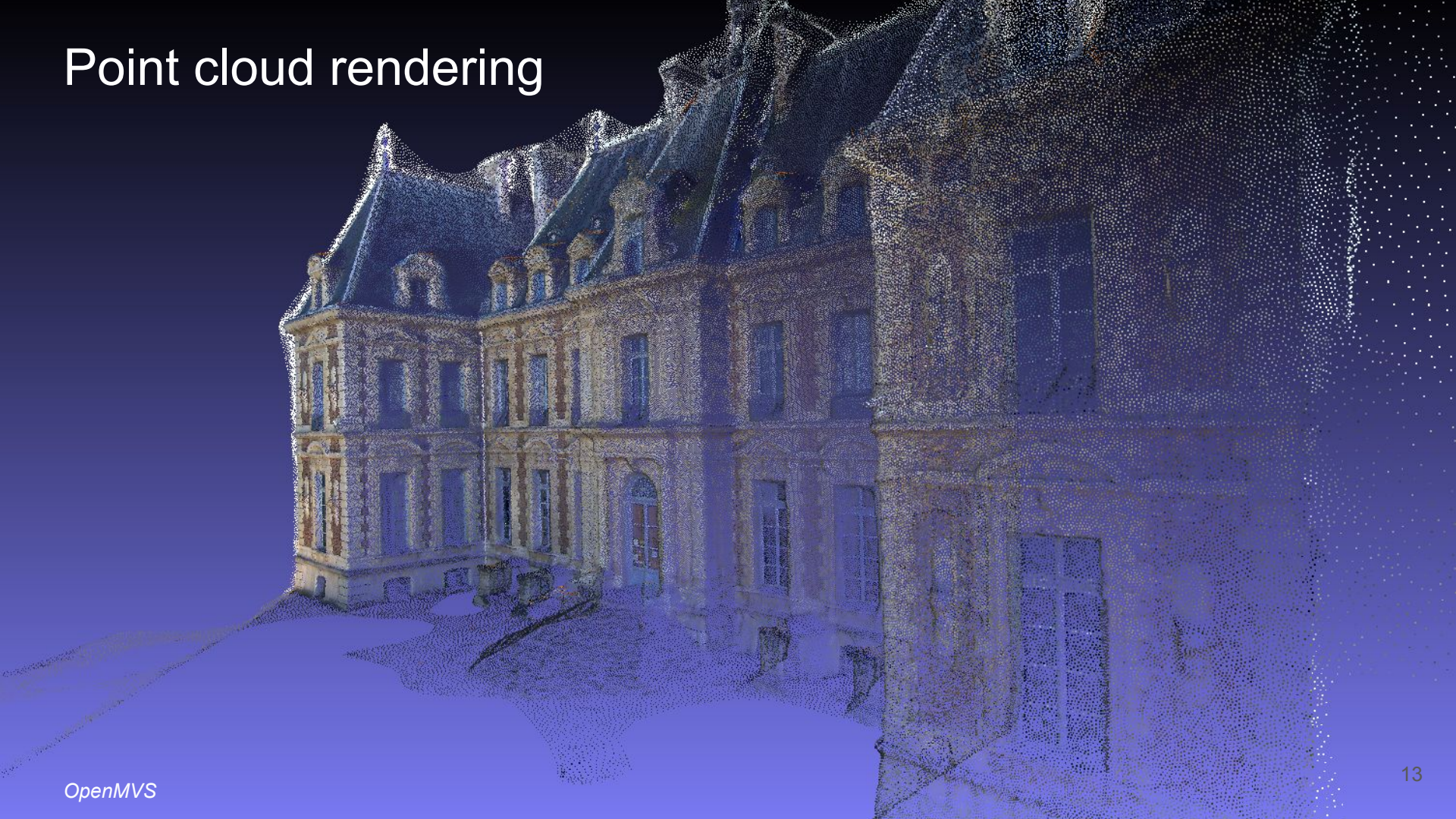

## Simulations

*CSTB*

## Games


Cyberpunk 2077 - Technology preview

# Point cloud rendering

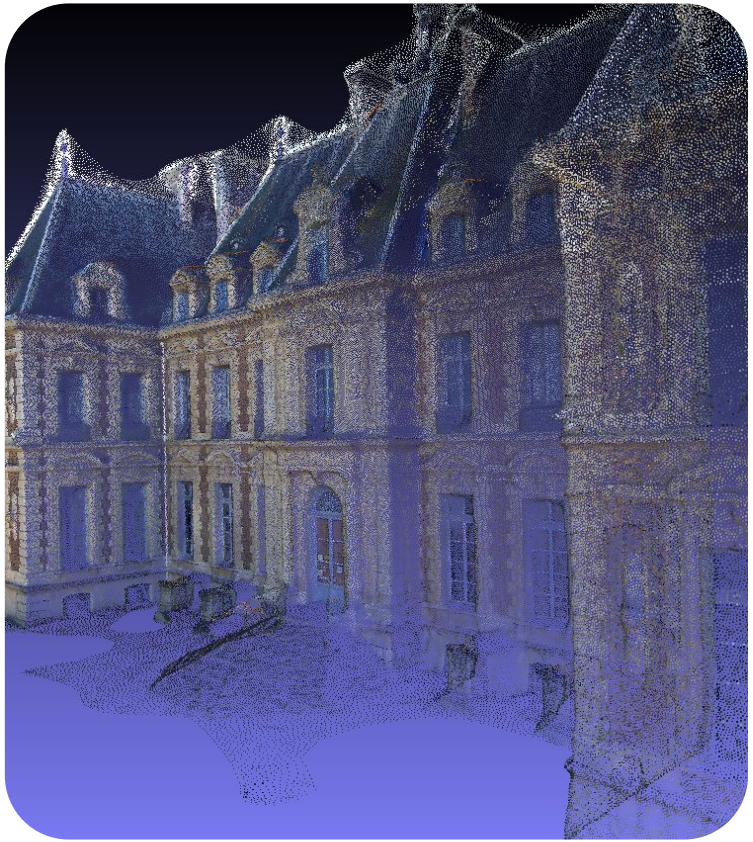


# Point clouds rendering

## I - From point cloud to surfaces

Point clouds are simple:

$$P = \{x \in \mathbb{R}^3\}$$



Note: most of the method we will discuss in the course can be applied in higher dimension

# Point clouds rendering

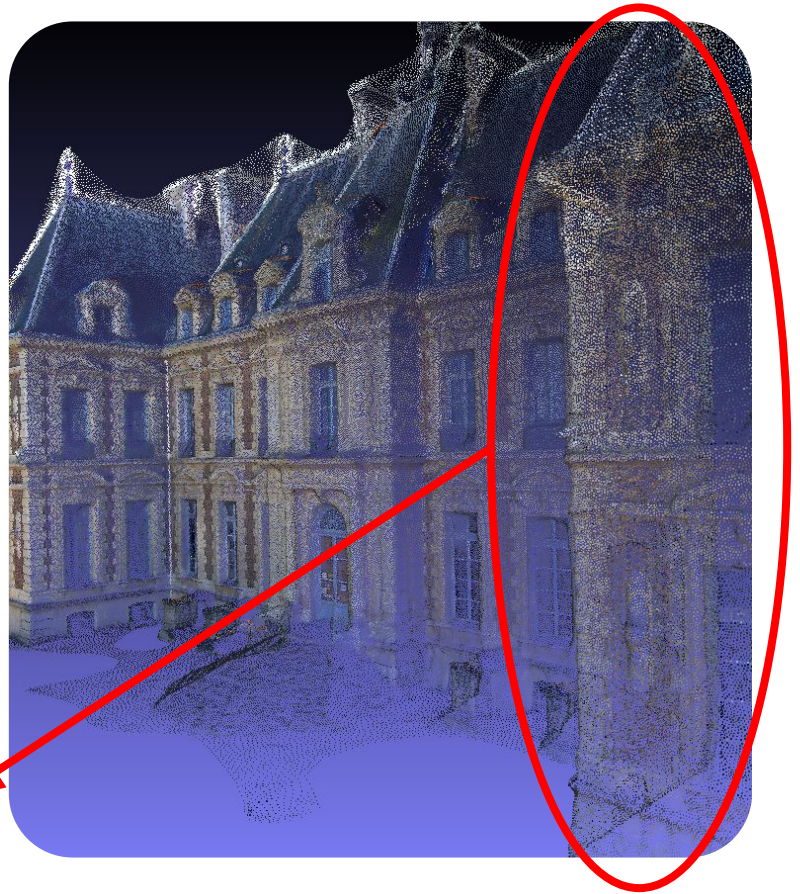


Point clouds are simple:

$$P = \{x \in \mathbb{R}^3\}$$

However:

- Points are independent
- Not easy to render



# Point clouds rendering



Point clouds are simple:

$$P = \{x \in \mathbb{R}^3\}$$

However:

- Points are independent
- Not easy to render

Higher point size
⇒ loss of details

# Point clouds

I - From point cloud to surfaces

Point clouds are simple:

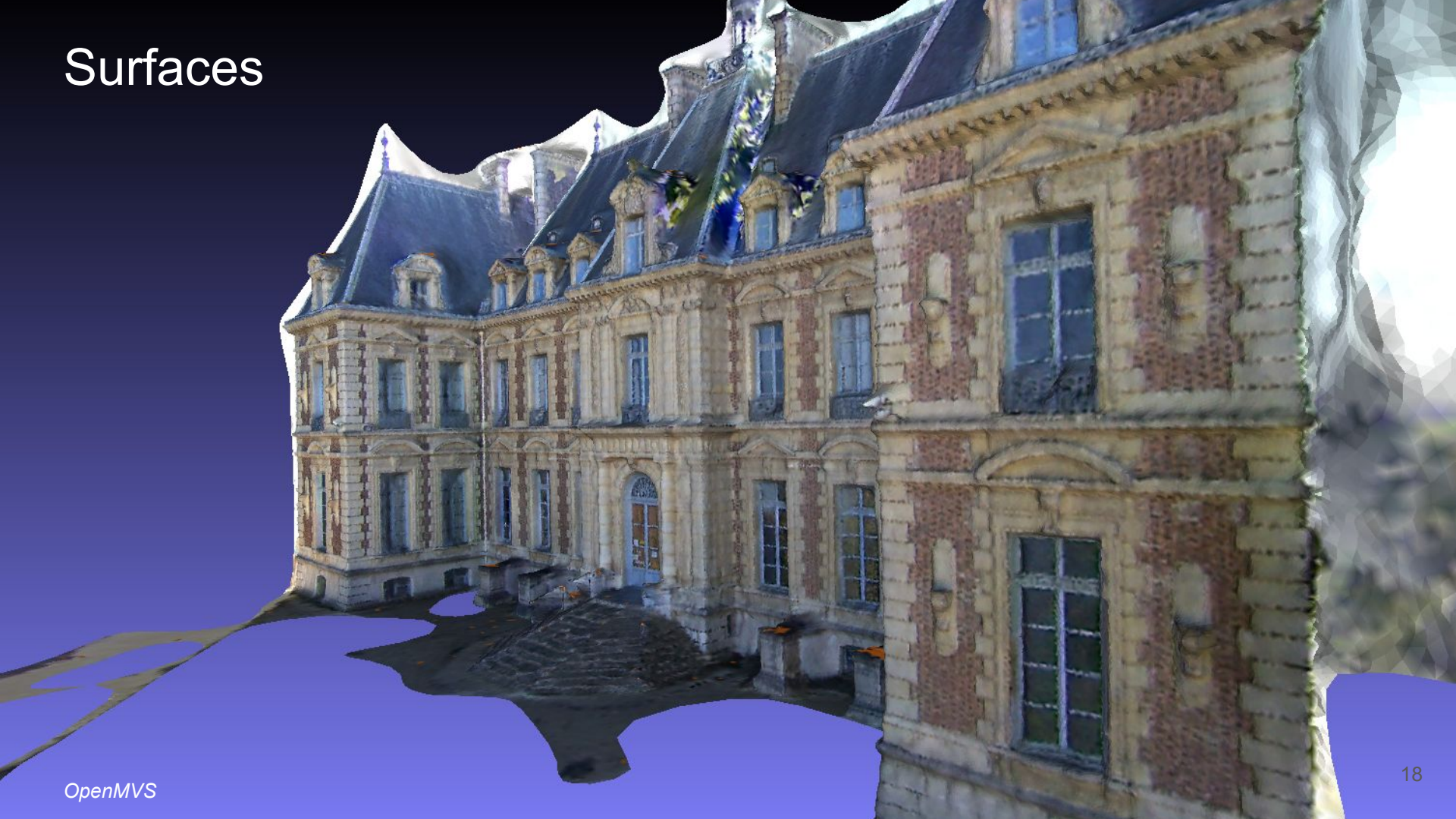
$$P = \{x \in \mathbb{R}^3\}$$

However:

- Points are independent
- Not easy to render

⇒ **Last course:** point rendering is not dead (but it requires a bit of work)

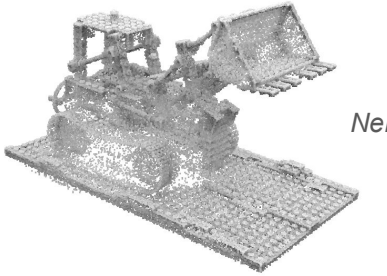
# Surfaces




unity

# What are point clouds?

I - From point cloud to surfaces



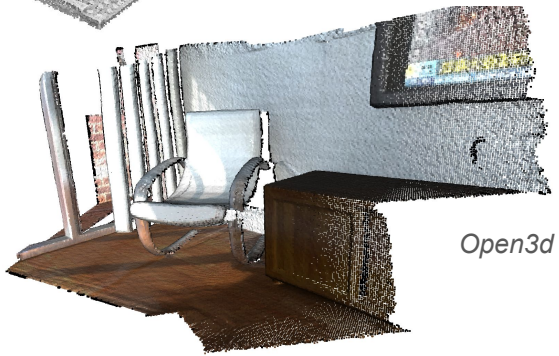
*NeRF*

A point cloud:

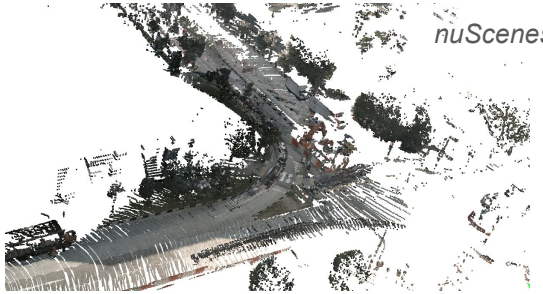
- A set of 3D coordinates

$$P = \{p \in \mathbb{R}^3\}$$

- No obvious order (at first sight)
- Sparse sample of surface
- Noisy / outliers
- Variation of size: several orders of magnitude



*Open3d*



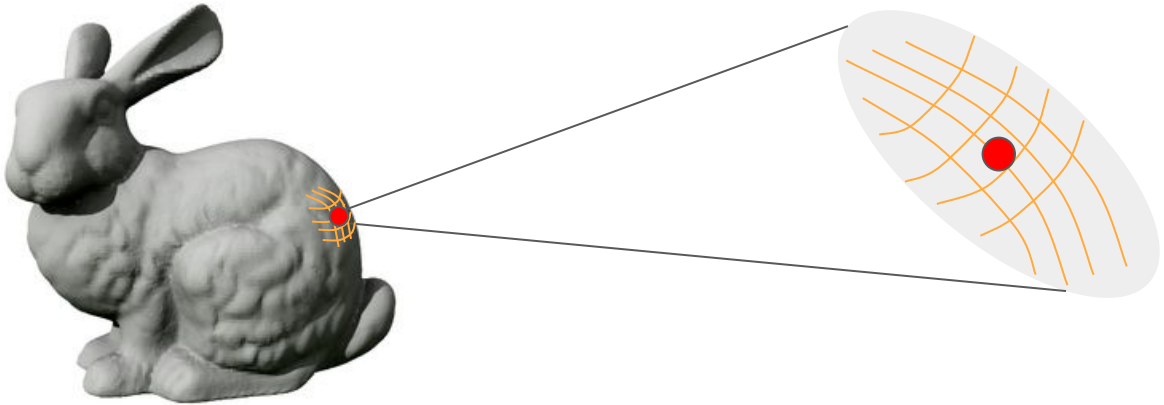
*nuScenes*

# What is a surface



A surface is a 2-manifold.

Locally, a manifold behave like the euclidean space, i.e, continuous.



*Stanford bunny*

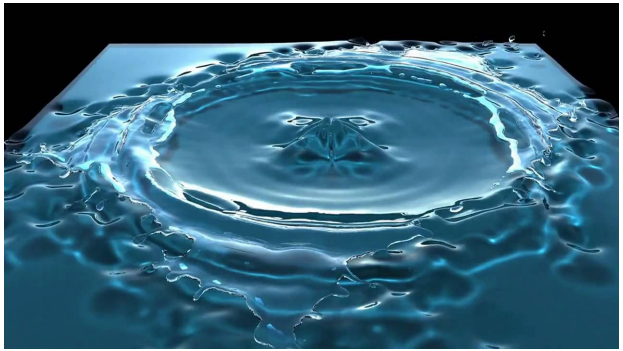
The neighborhood is homeomorphic to a 2D-euclidean space (open 2D ball)

# Why surfaces?

## I - From point cloud to surfaces

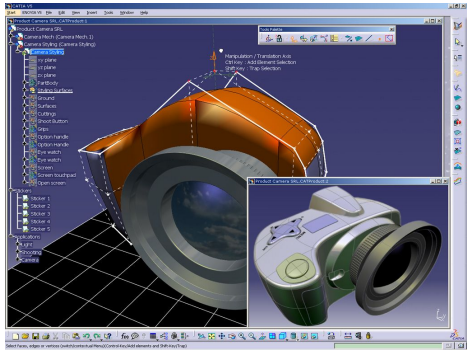
Surfaces for:

- Simulation
- Animation
- Design
- …



*A Multiscale Approach to Mesh-based Surface Tension Flows*

*Blender*



*Catia*

# How to represent a surface

valeo.ai

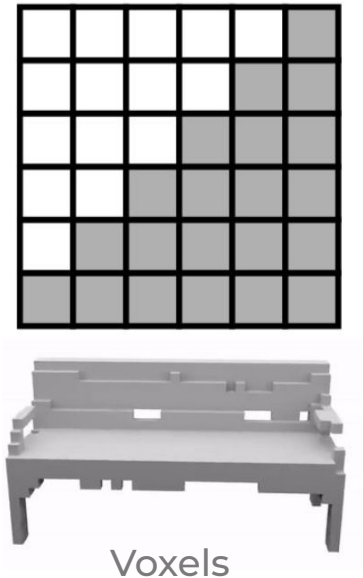
I - From point cloud to surfaces

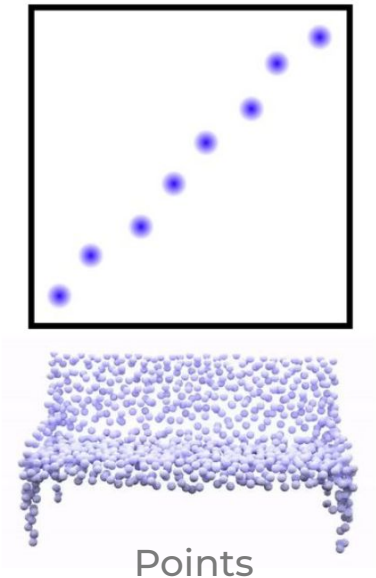
- Points
- Meshes
- Voxels
- Implicit representations
- Parametric shapes (planes, cylinder, spheres)
- Gaussians
- …

→ no unified / perfect representation

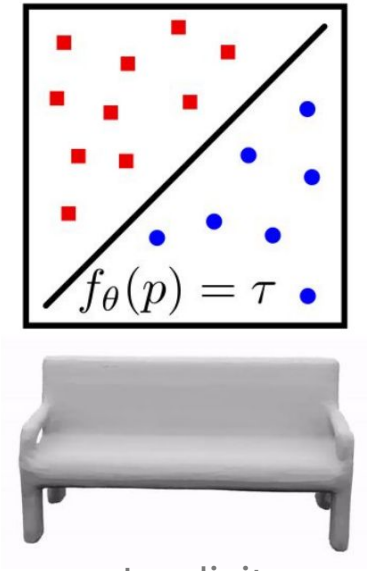
# How to represent a surface?

## I - From point cloud to surfaces



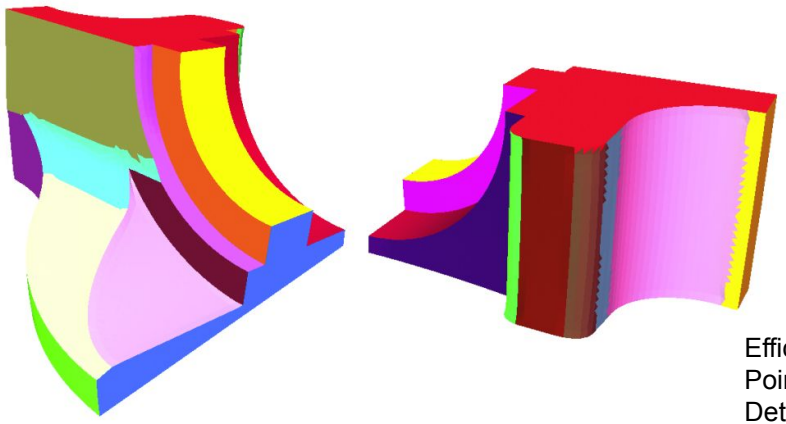





**Occupancy Networks: Learning 3D Reconstruction in Function Space**
*Mescheder, Lars and Oechsle, Michael and Niemeyer, Michael and Nowozin, Sebastian and Geiger, Andreas*
Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2019

# How to represent a surface?

## I - From point cloud to surfaces



Efficient RANSAC for Point-Cloud Shape Detection

# How to represent a surface?

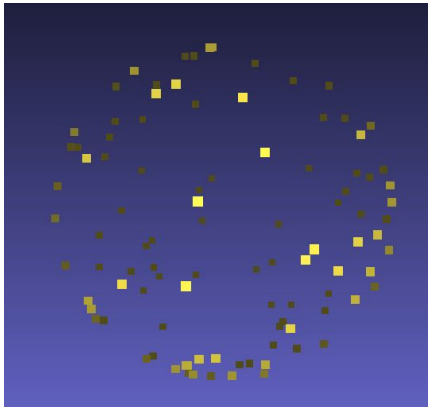
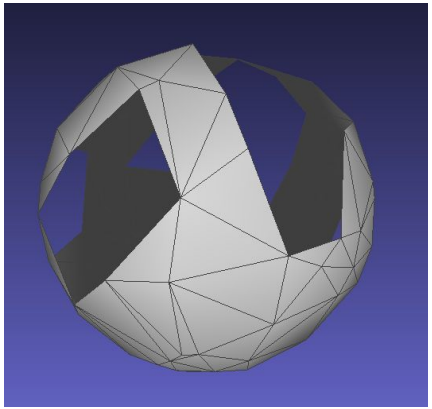
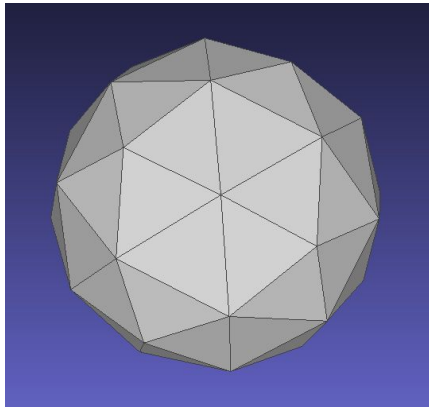
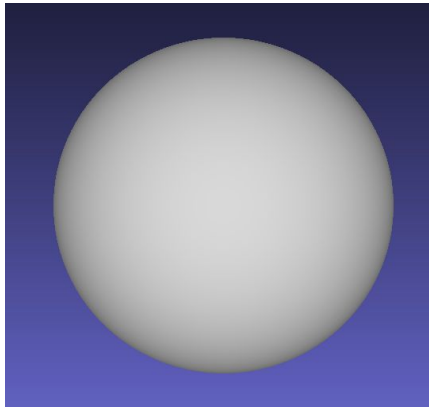
## I - From point cloud to surfaces


3DGS - Improving Gaussian splatting with Localized points management

# What are the properties we want?



- Sticking to the points?
- With holes?
- Abstract?
- Regular?
- Made of planes? ...

# Overview

I. From point clouds to surfaces

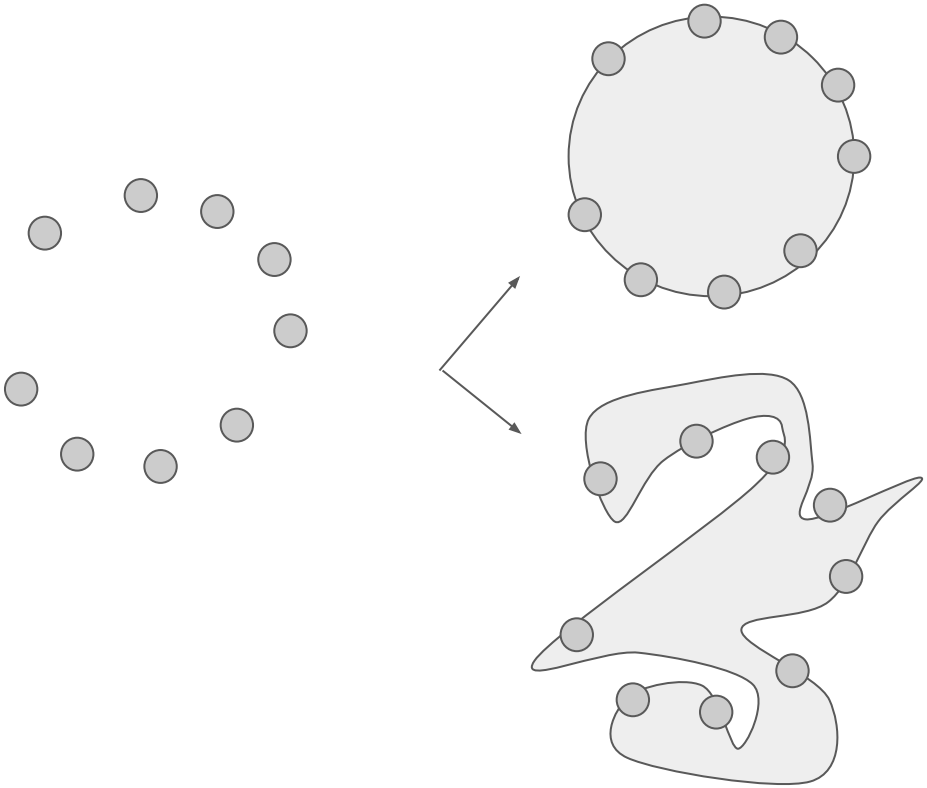
**II. Local features**

- A. Normal estimation
- B. Normal orientation

III. Surface reconstruction

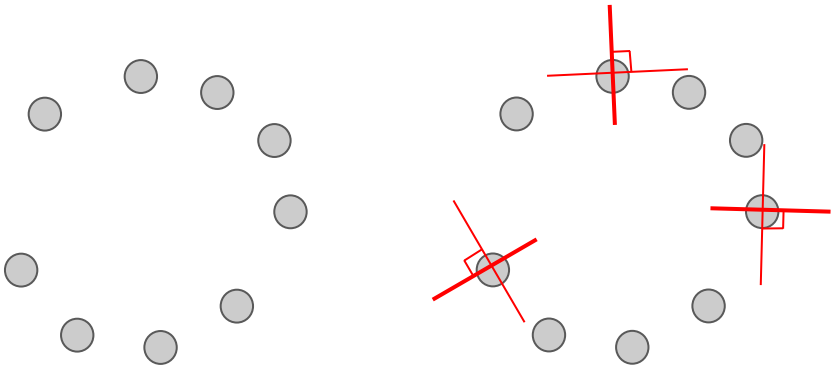
- A. Ball pivoting
- B. Delauney reconstruction
- C. Poisson reconstruction
- D. RANSAC

# Fitting a surface



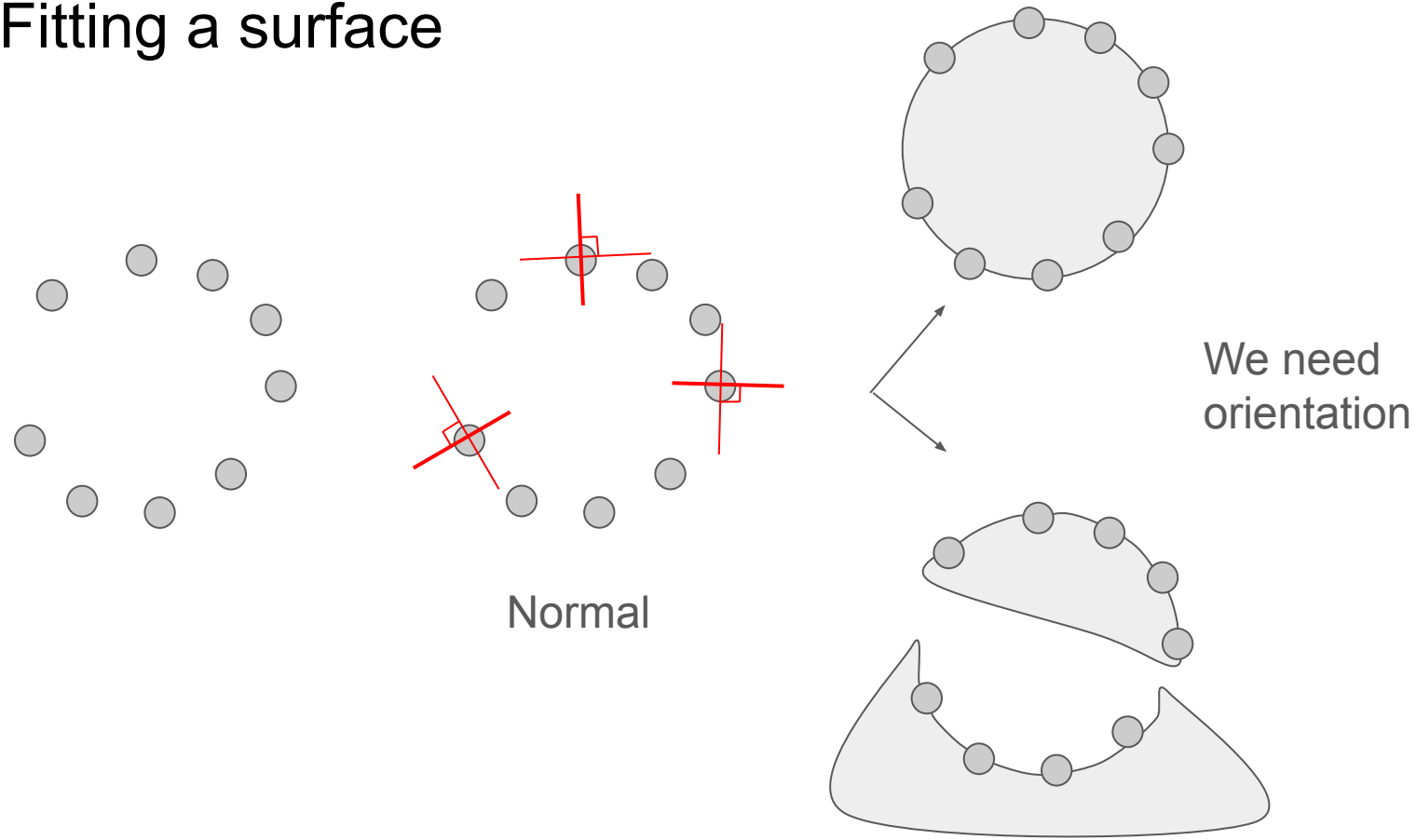
We need local tangent plane

# Fitting a surface

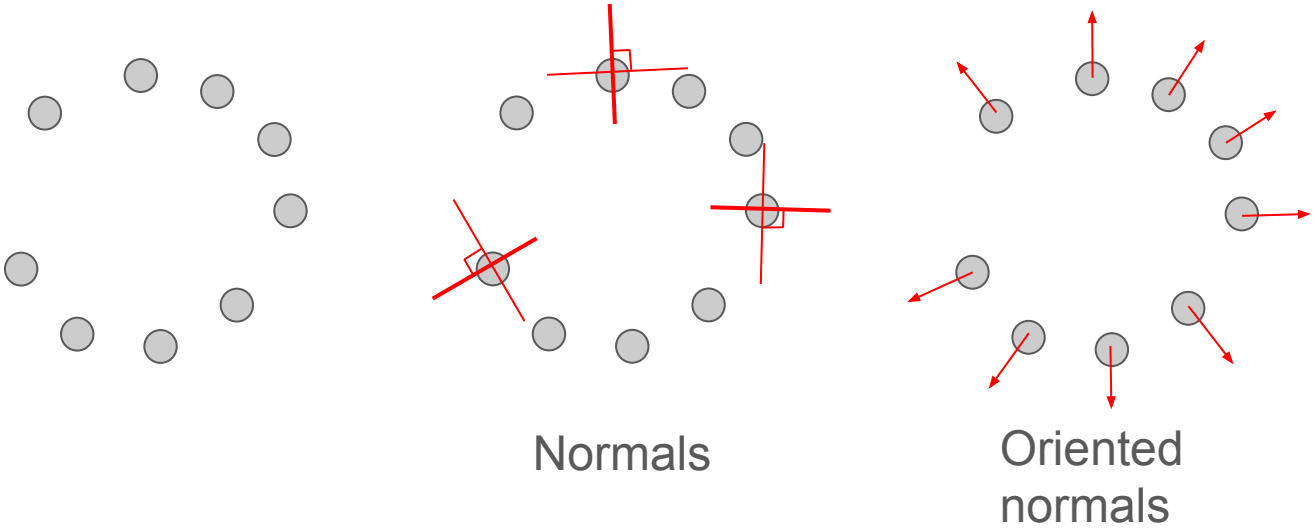


Normal

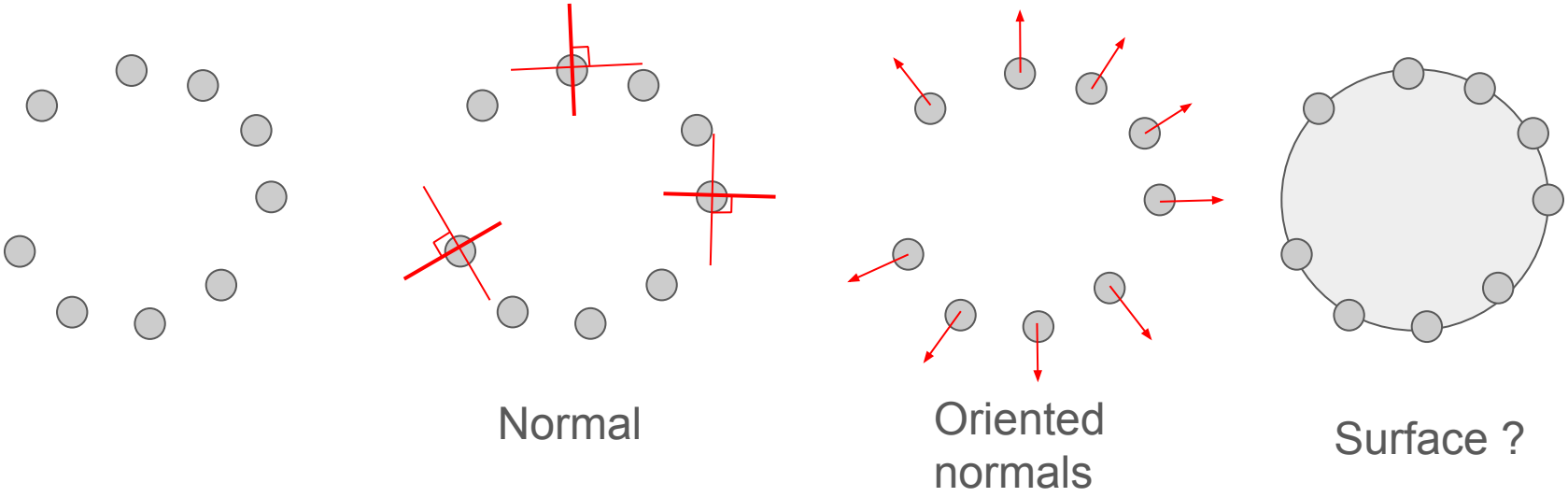
# Fitting a surface



# Fitting a surface



# Fitting a surface



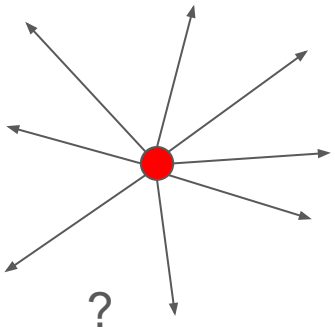
# A - Normal estimation

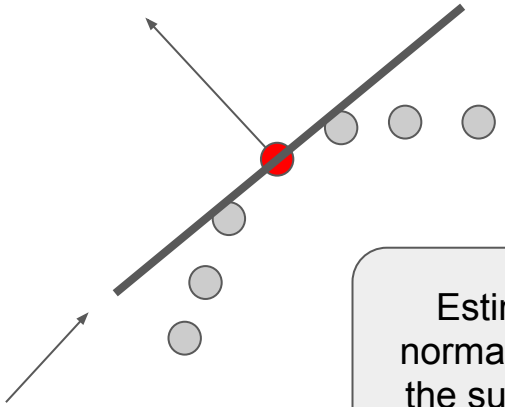


valeo.ai

A single point does not contain orientation information

Need to consider a neighborhood around a point




Estimating the normal ⇔ knowing the surface locally

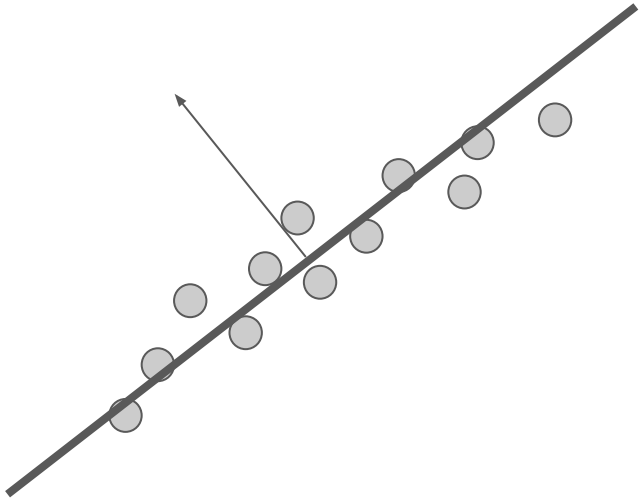
# A - Normal estimation



valeo.ai

## Plane fitting

- The plane is defined by the directions of the largest variance
- The normal corresponds to the direction with lowest variance
- **Principal Component Analysis**
- The normal is the eigenvector for the smallest eigenvalue.



# A - Normal estimation



valeo.ai

Plane fitting with PCA

- Compute covariance matrix

Average: $\bar{x} = \frac{1}{n} \sum_{x \in P} x$

Covariance: $Cov \in \mathbb{R}^3 \times \mathbb{R}^3$

$$Cov(i,j) = \frac{1}{n} \sum_{x \in P} (x_i - \bar{x}_i)(x_j - \bar{x}_j) = \frac{1}{n}^\top XX$$

# A - Normal estimation



valeo.ai

Plane fitting with PCA

- Compute covariance matrix

- Diagonalize the Matrix, with P orthonormal (Cov is positive, real, symmetric)

$$Cov = PDP^{-1}$$

$$D = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} \qquad P = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}$$

# A - Normal estimation



valeo.ai

Plane fitting with PCA

- Compute covariance matrix
- Diagonalize the Matrix
- Find the lowest eigenvalue and eigenvector $\lambda_1 \geq \lambda_2 \geq \lambda_3$

$$D = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} \qquad P = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}$$

# A - Normal estimation



valeo.ai

Plane fitting with PCA

- Compute covariance matrix
- Diagonalize the Matrix
- Find the lowest eigenvalue and eigenvector

$$\mathbf{n} = \overrightarrow{n} = \overrightarrow{e}_3$$

# Overview

I. From point clouds to surfaces

**II. Local features**

- A. Normal estimation
- B. Normal orientation

III. Surface reconstruction

- A. Ball pivoting
- B. Delauney reconstruction
- C. Poisson reconstruction
- D. RANSAC

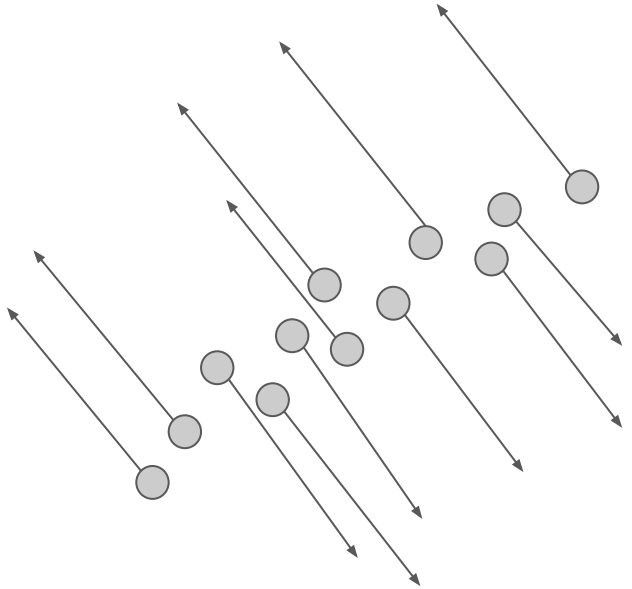
# B - Normal orientation



The plane fitting algorithm:

✓ direction

✗ orientation

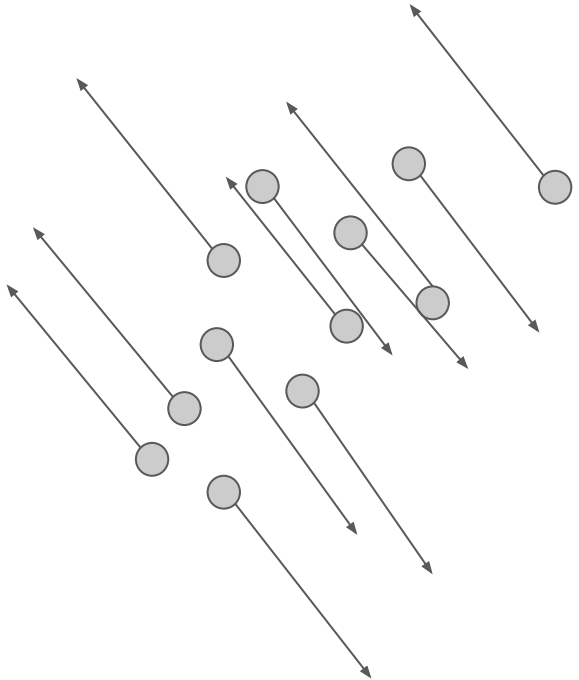


# B - Normal orientation



Normal orientation with minimal spanning tree

**Idea:** propagate the orientation from one seed



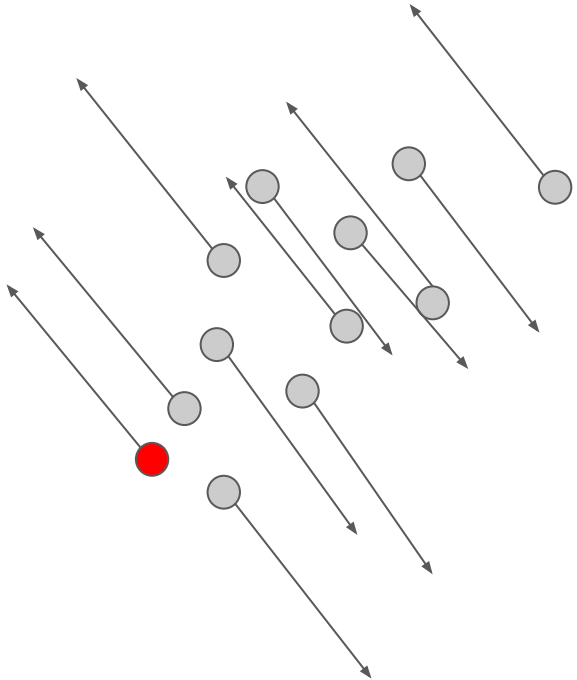
# B - Normal orientation



Normal orientation with minimal spanning tree

**Idea:** propagate the orientation from one seed

1. Select a seed



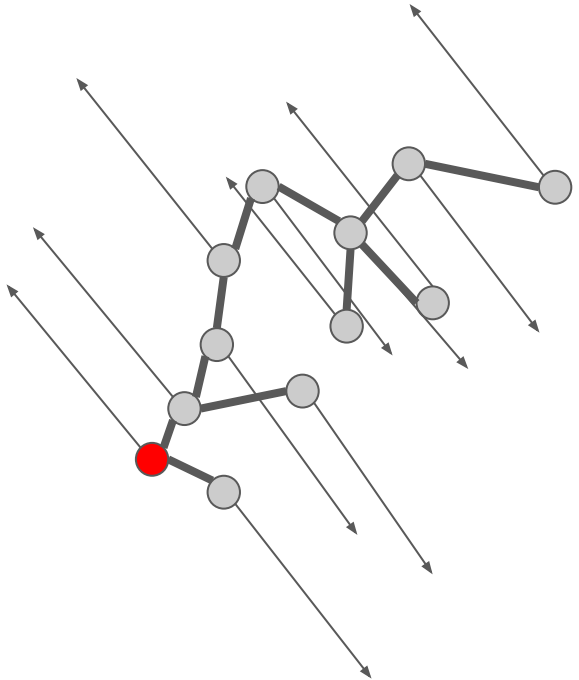
# B - Normal orientation



Normal orientation with minimal spanning tree

**Idea:** propagate the orientation from one seed

1. Select a seed
2. Build a spanning tree



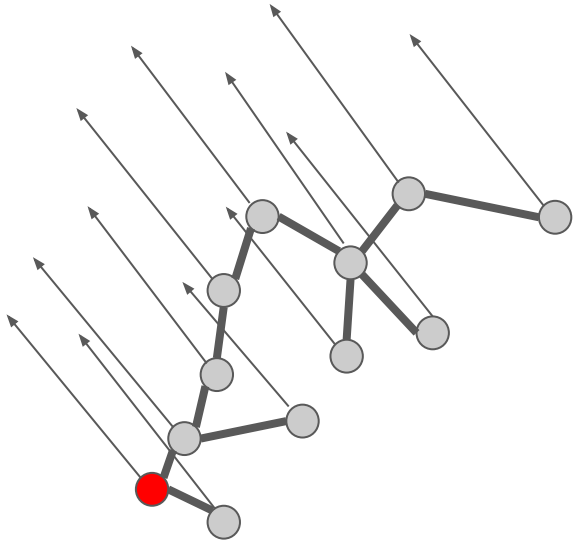
# B - Normal orientation



Normal orientation with minimal spanning tree

**Idea:** propagate the orientation from one seed

1. Select a seed
2. Build a spanning tree
3. Consistently orient the normals (inner product > 0) from parent to child

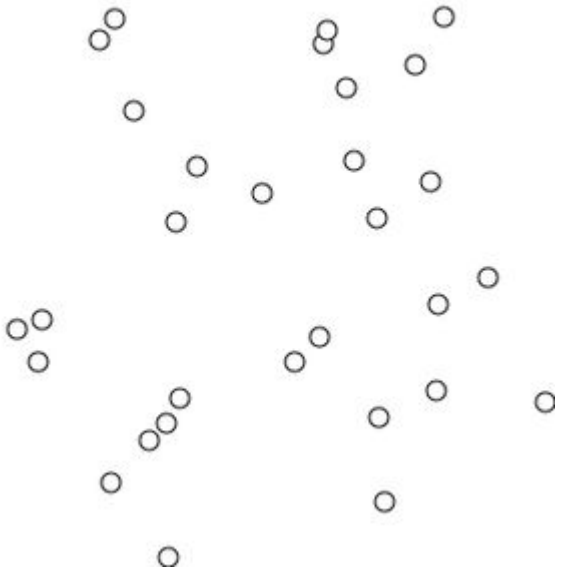


# B - Normal orientation



The Kruskal algorithm

- Compute the KNN graph (KDTree)
- Sort the edges of the graph (increasing distances)
- Loop on all edges
  - Add the edge if it does not create a loop



*Kruskal algorithm, Wikipedia*

# Overview

I. From point clouds to surfaces
II. Local features
   A. Normal estimation
   B. Normal orientation
**III. Surface reconstruction**
   A. Ball pivoting
   B. Delauney reconstruction
   C. Poisson reconstruction
   D. RANSAC

# Ball pivoting





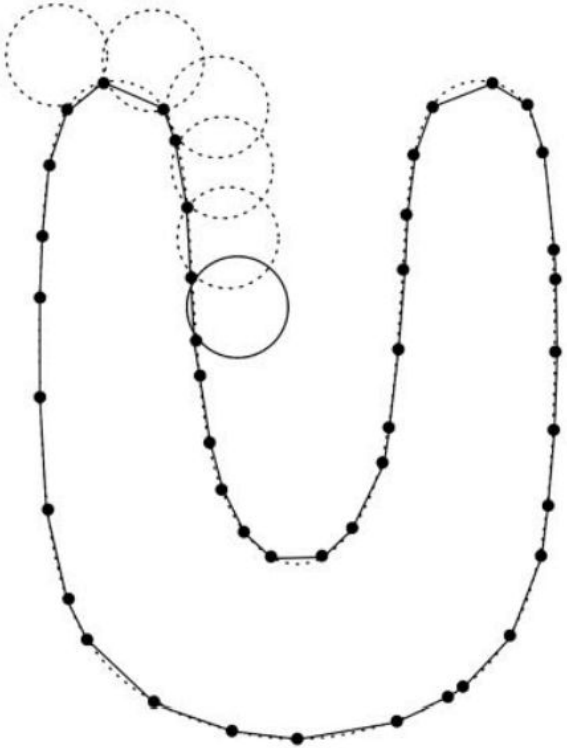
Principles:

Make a sphere “roll” on the points

2D: When the sphere touch 2 points, create an edge

3D: When the sphere touch 3 points, create a facet

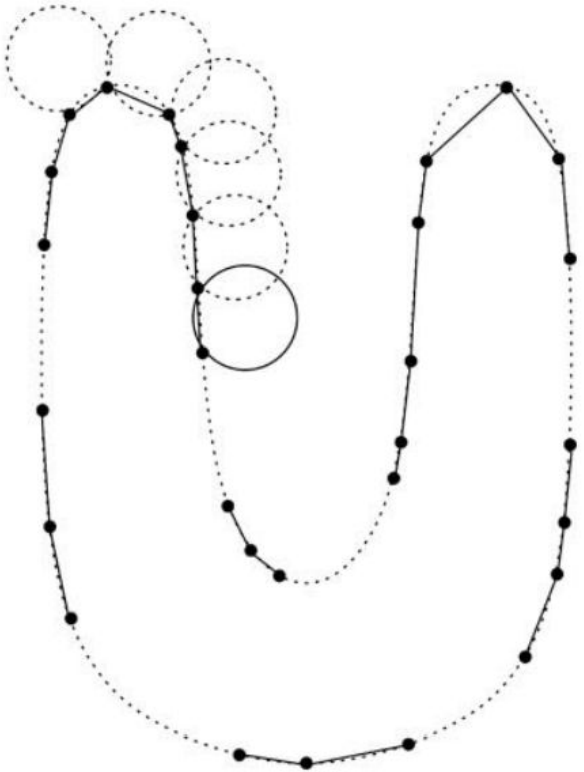
Iterate over the unexplored edges



# Ball pivoting

III - Surface reconstruction

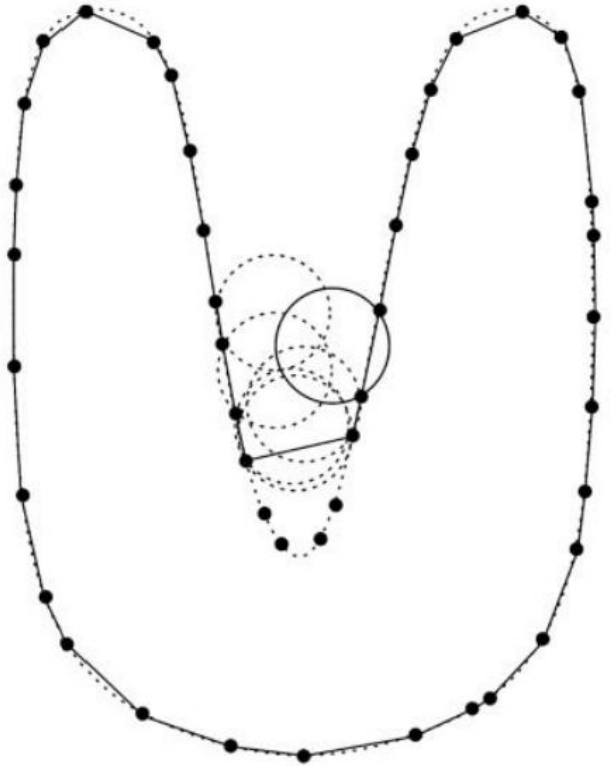
Space larger than sphere diameter creates a hole.



# Ball pivoting

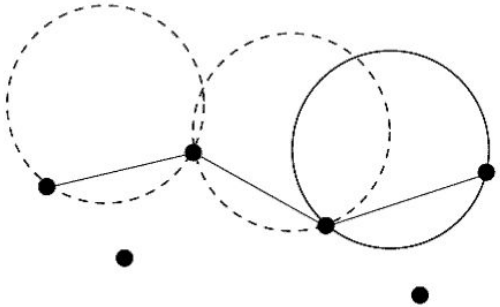


Sphere with to big radius leads to a loss of details

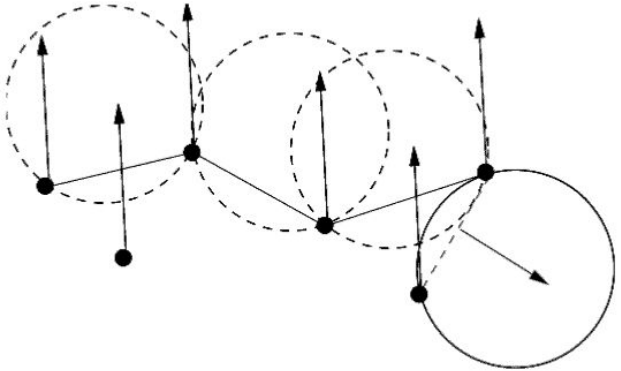


# Ball pivoting

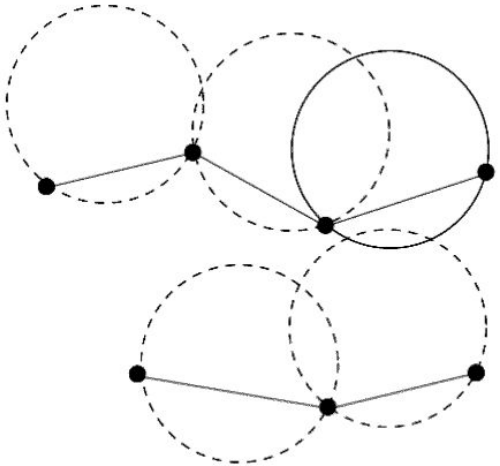




Presence of noise



Filter according to normals

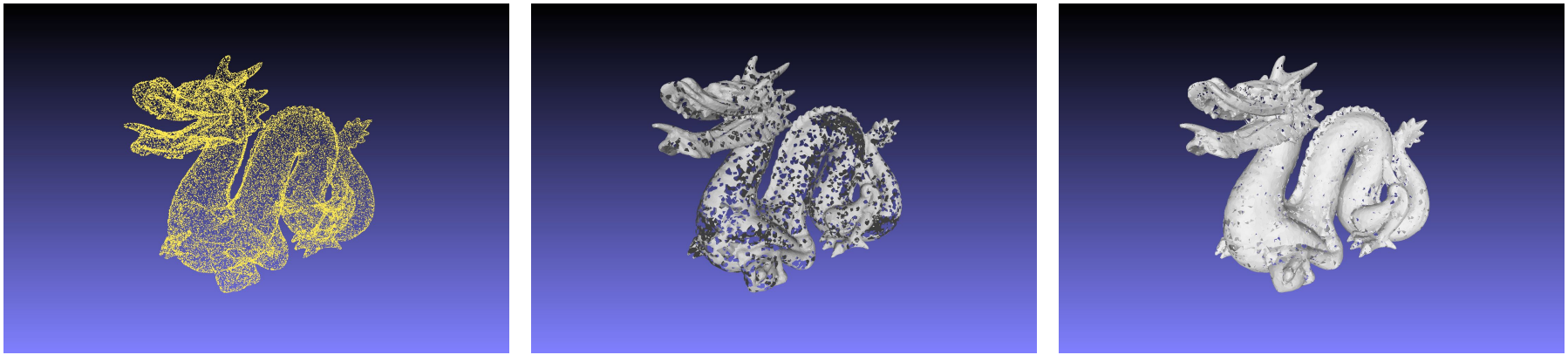


Double side surface

# Ball pivoting







Increasing sphere radius

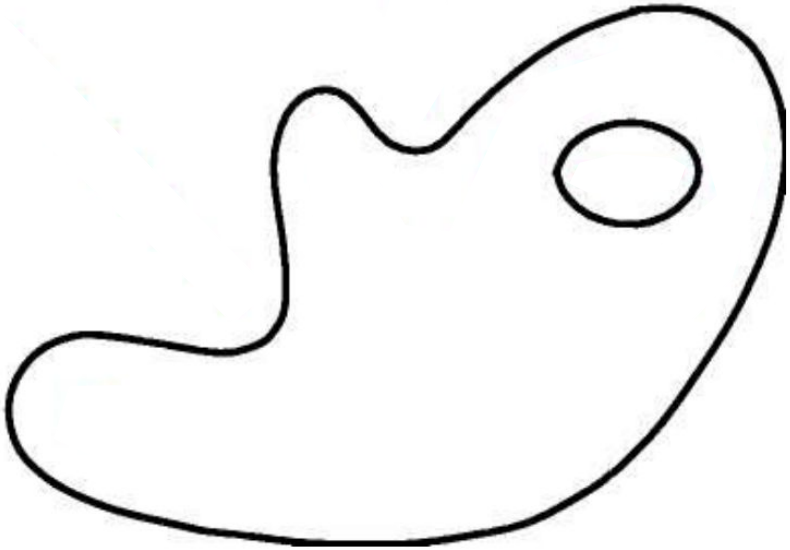
# Overview

I. From point clouds to surfaces
II. Local features
   A. Normal estimation
   B. Normal orientation
**III. Surface reconstruction**
   A. Ball pivoting
   B. Delauney reconstruction
   C. Poisson reconstruction
   D. RANSAC

# Surface

## B - Delauney reconstruction

Consider an abstract surface 2D surface (no hole).



A New Voronoi-Based Surface Reconstruction Algorithm, Amenta et al.

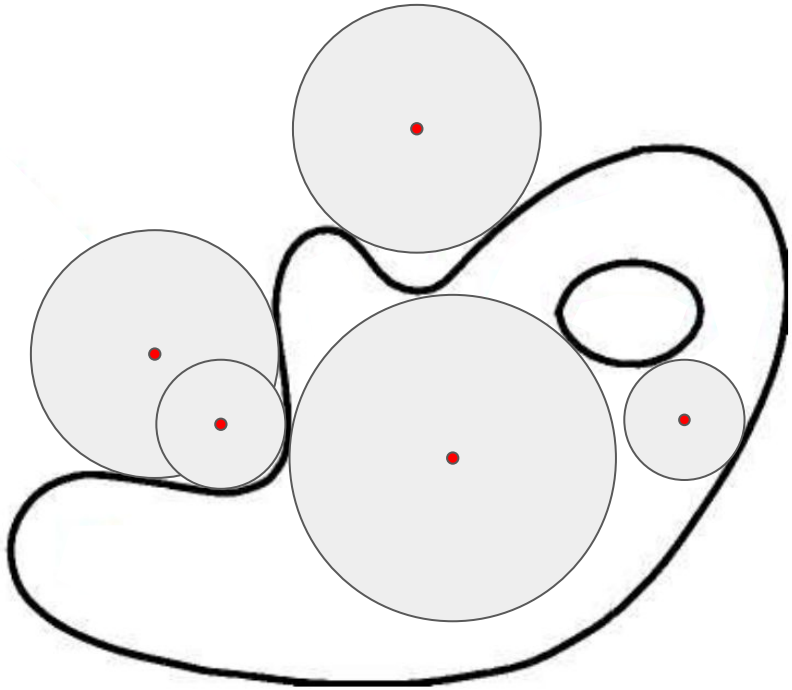
# Medial axis

B - Delauney reconstruction

Let’s put a sphere such that:

- It touches at least to points on the surface
- It does not include part of the surface

Let’s consider all the possible spheres



A New Voronoi-Based Surface Reconstruction Algorithm, Amenta et al.

# Medial axis

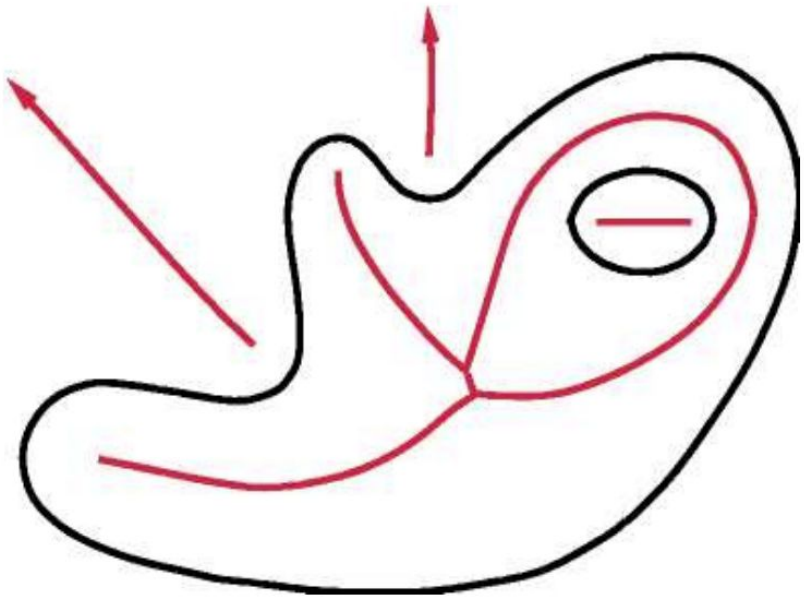
valeo.ai

B - Delauney reconstruction

The set of centers of these spheres is the **medial axis**

The medial axis is the dual of the surface.

⇒ from a medial axis, it is possible to find the surface



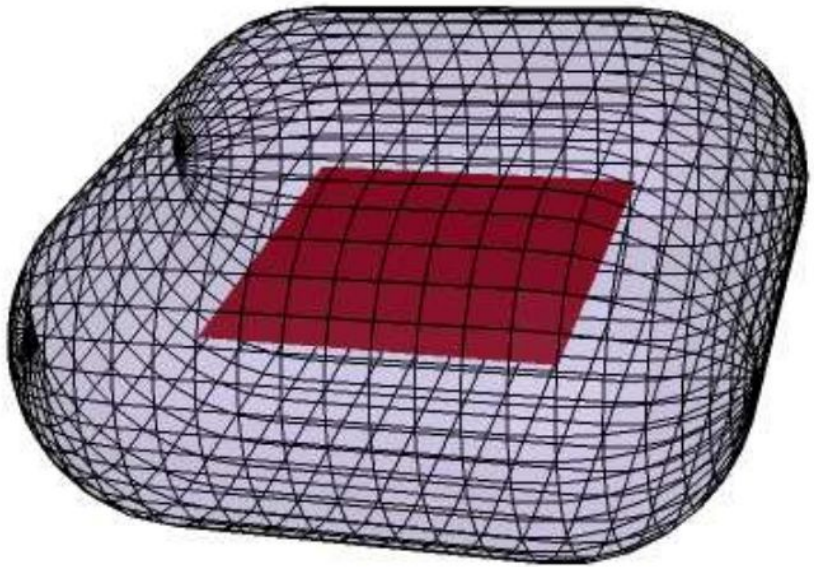
A New Voronoi-Based Surface Reconstruction Algorithm, Amenta et al.

# Medial axis

valeo.ai

## B - Delauney reconstruction

The medial axis generalizes to 3D



A New Voronoi-Based Surface Reconstruction Algorithm, Amenta et al.

# Voronoï diagram

valeo.ai

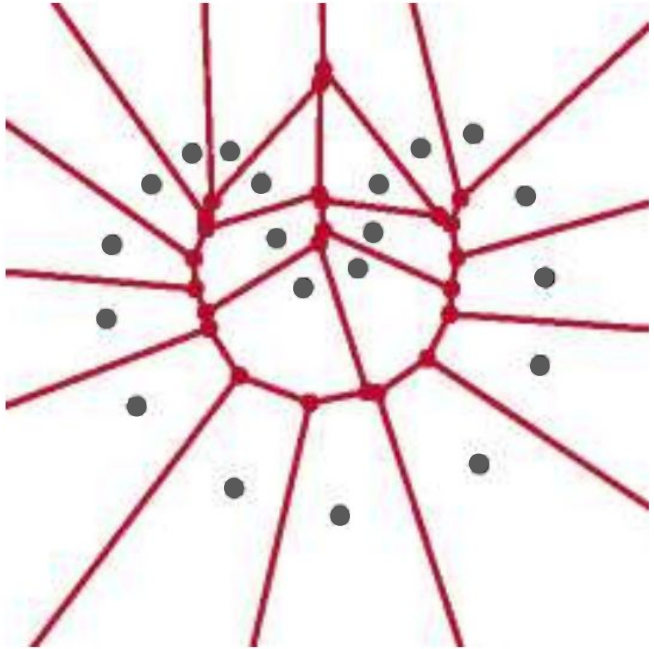
B - Delauney reconstruction

Is there an equivalent of the medial axis for point clouds?

⇒ yes this the Voronoï diagram

On an edge → equal distance to 2 points

On a node → equal distance to 3 points



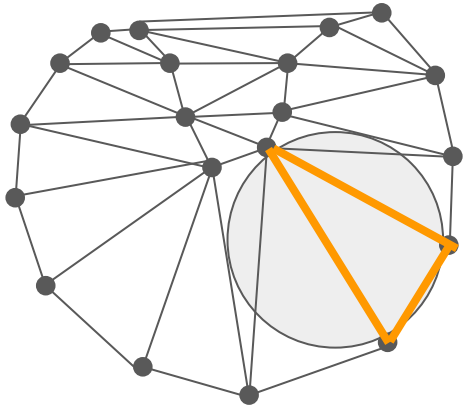
A New Voronoi-Based Surface Reconstruction Algorithm, Amenta et al.

# Voronoï diagram

## B - Delauney reconstruction

Given 3 points, create a triangle if the sphere encompassing the 3 points is empty.

Iterate over the triplets.



A New Voronoi-Based Surface Reconstruction Algorithm, Amenta et al.

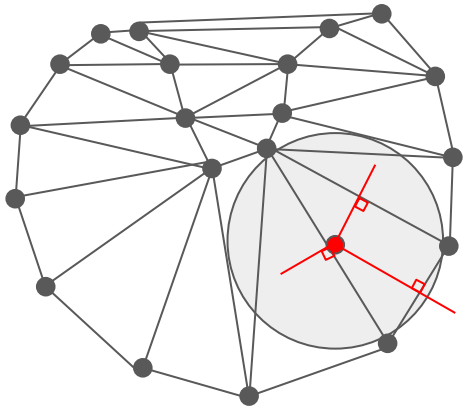
# Voronoï diagram

valeo.ai

## B - Delauney reconstruction

Create the graph.

- nodes → the center of the spheres
- Edges → mediator of the edges of the triangles



A New Voronoi-Based Surface Reconstruction Algorithm, Amenta et al.

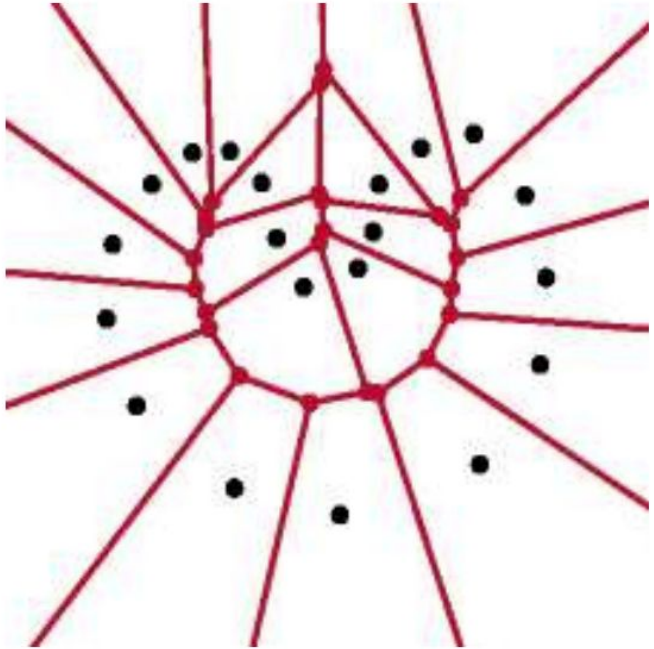
# Voronoï diagram

valeo.ai

B - Delauney reconstruction

Create the graph.

- nodes → the center of the spheres
- Edges → mediator of the edges of the triangles



# Augmented Delauney triangulation

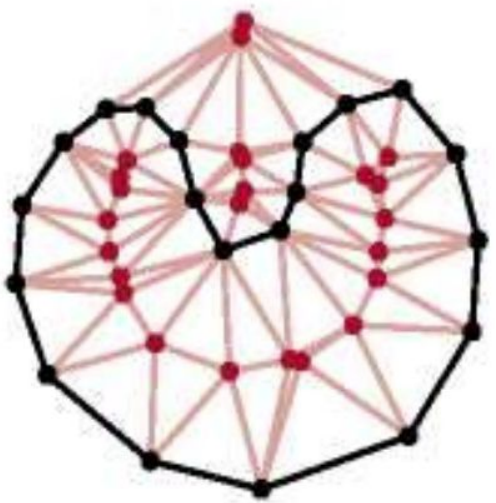
valeo.ai

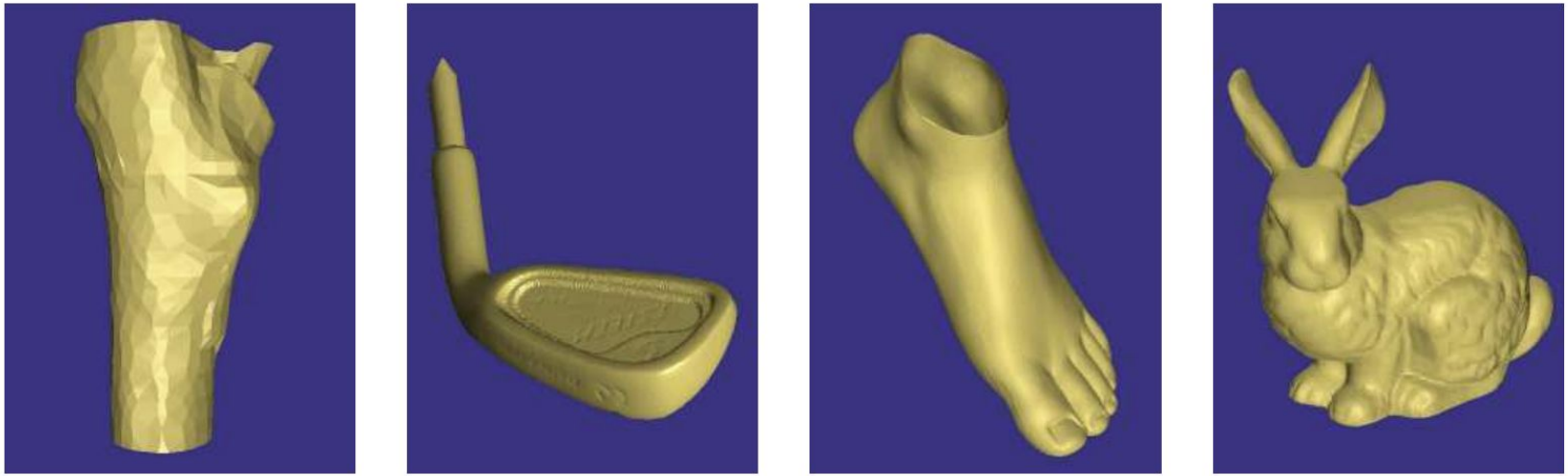
B - Delauney reconstruction

Recompute the Delaunay triangulation with the Voronoï nodes.

Remove edges linked to Voronoï nodes

The remaining is the **crust**.





Pros: theoretically grounded, very good results without noise

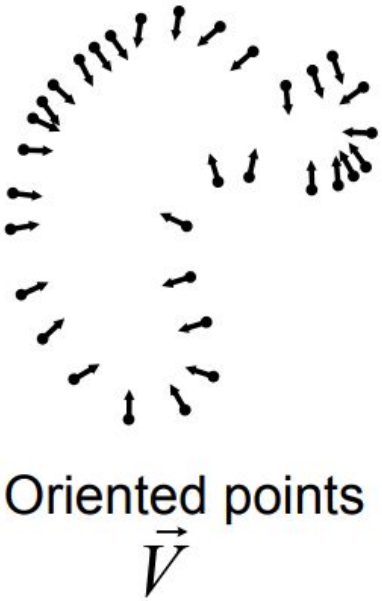
Cons: not smooth, cannot handle noise

# Overview

I. From point clouds to surfaces
II. Local features
    A. Normal estimation
    B. Normal orientation
**III. Surface reconstruction**
    A. Ball pivoting
    B. Delauney reconstruction
    C. Poisson reconstruction
    D. RANSAC

# Overview

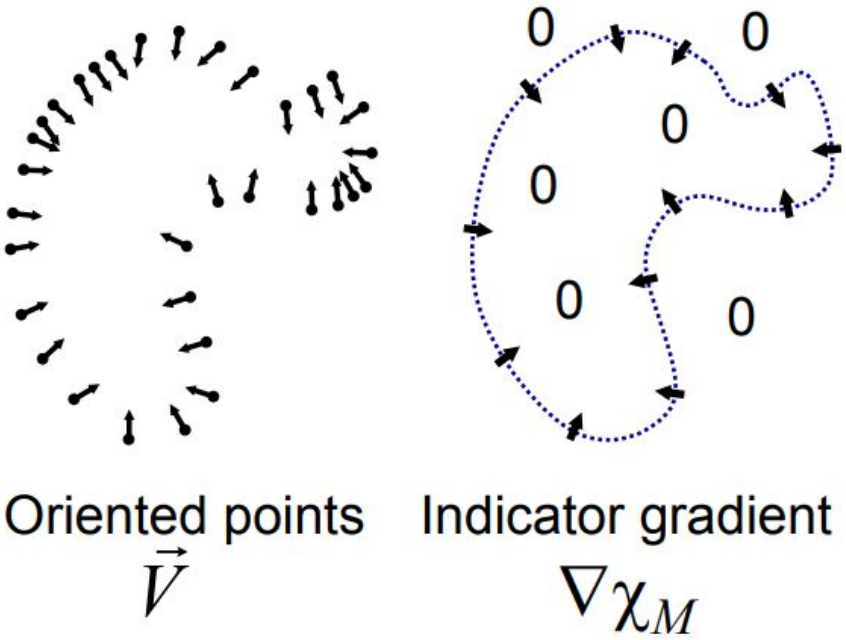
## C - The Poisson reconstruction pipeline




*Kazhdan et al. "Poisson surface reconstruction." SGP, 2006.*

# Overview

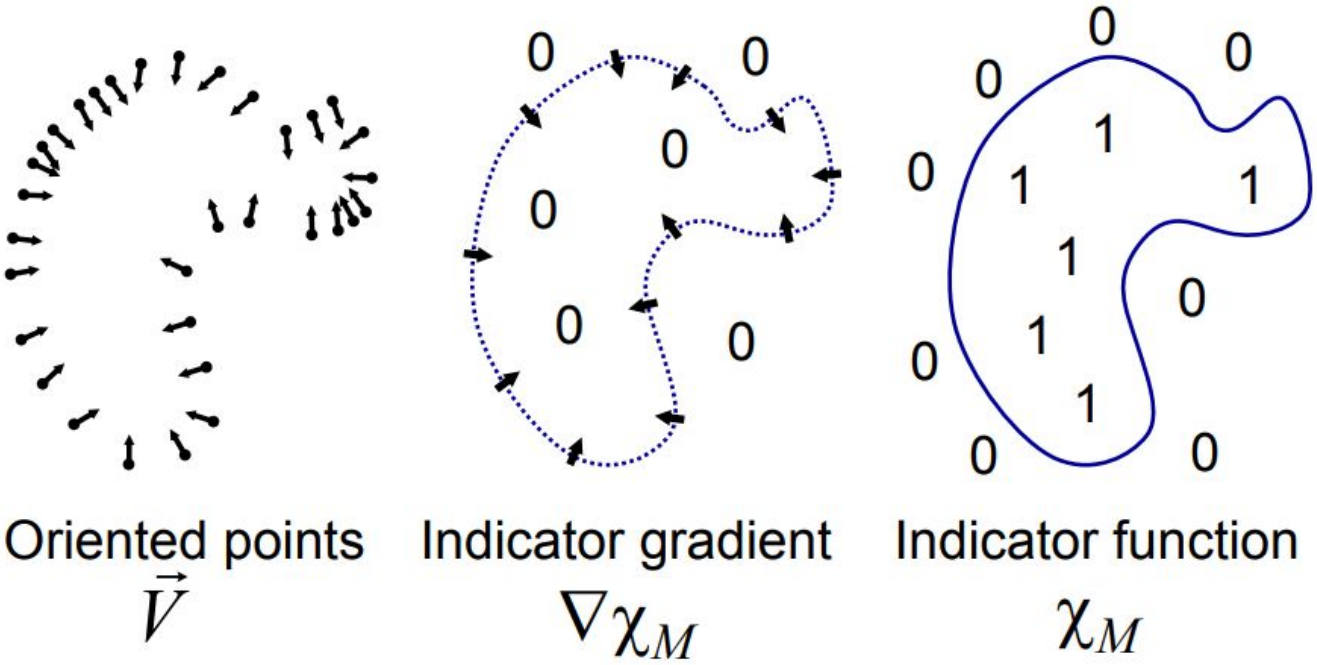
## C - The Poisson reconstruction pipeline




*Kazhdan et al. "Poisson surface reconstruction." SGP, 2006.*

# Overview

## C - The Poisson reconstruction pipeline



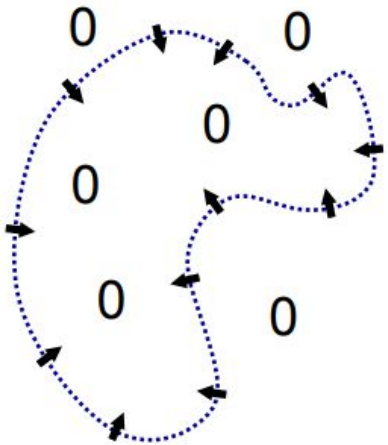

*Kazhdan et al. "Poisson surface reconstruction." SGP, 2006.*

# Overview

## C - The Poisson reconstruction pipeline

Oriented points

$\vec{V}$




Indicator gradient

$\nabla \chi_M$

splatted normals

$$\min_{\chi} \int \|\nabla \chi(\mathbf{x}) - \mathcal{N}(\mathbf{x})\|_2^2 \, dx$$

variational calculus

$$\Delta \chi = \nabla \cdot \mathcal{N}$$

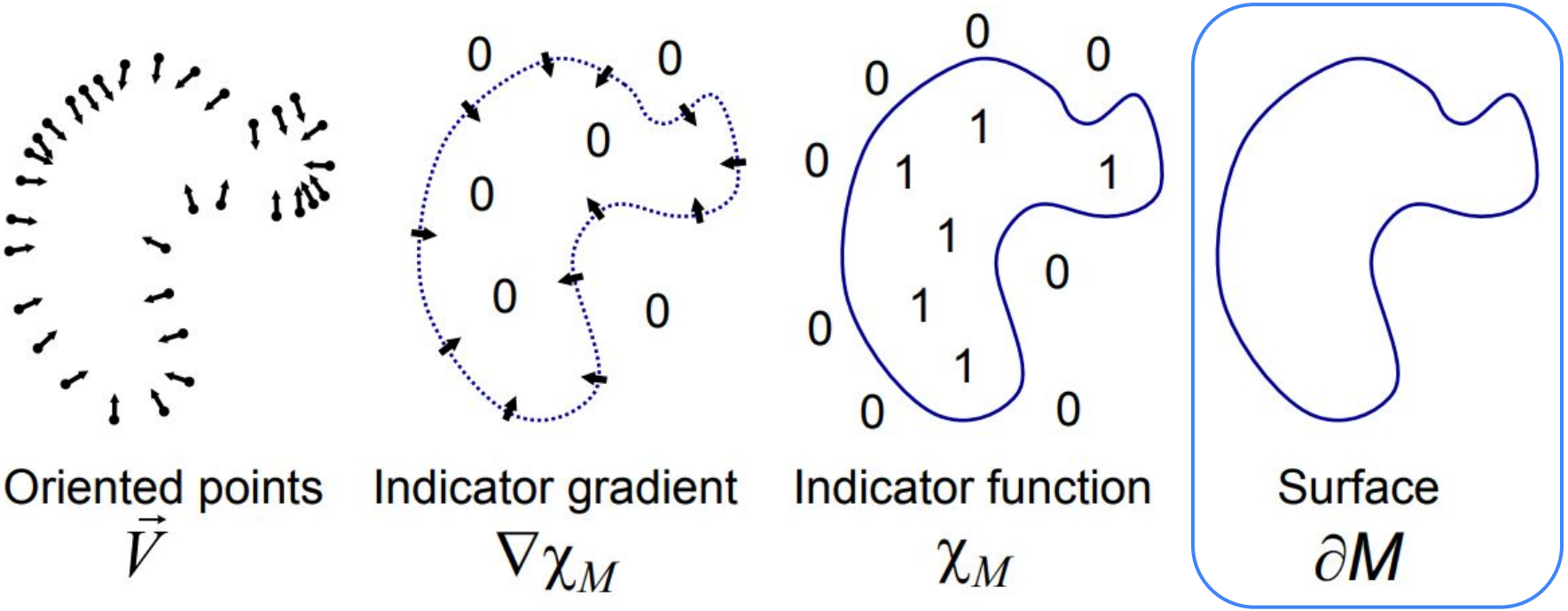
sparse linear system

*Kazhdan et al. "Poisson surface reconstruction." SGP, 2006.*

# Overview

valeo.ai

## C - The Poisson reconstruction pipeline





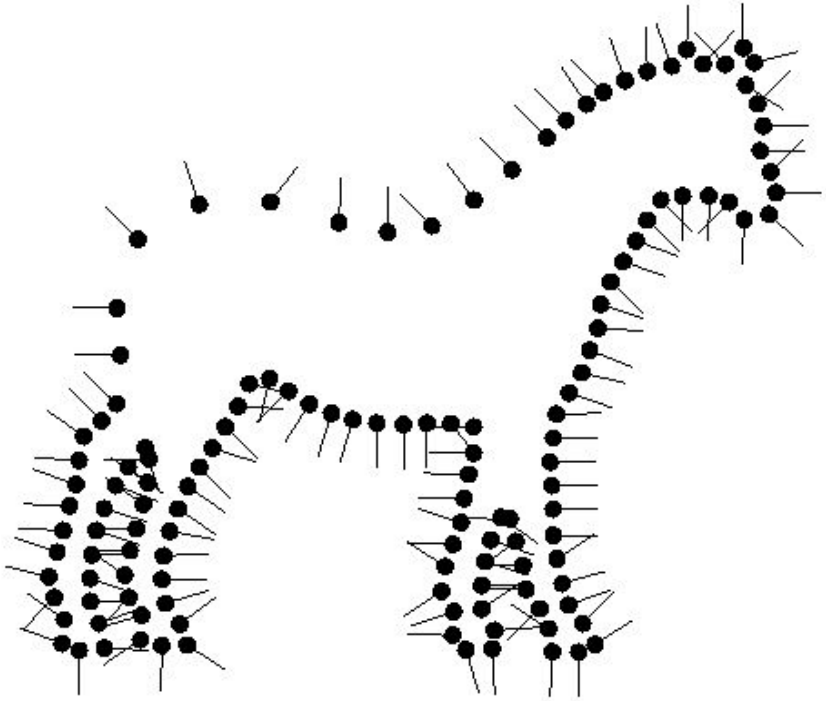

*Kazhdan et al. "Poisson surface reconstruction." SGP, 2006.*

# In practice

C - The Poisson reconstruction pipeline

**Given the Points:**

- Set octree
- Compute vector field
- Compute indicator function
- Extract iso-surface

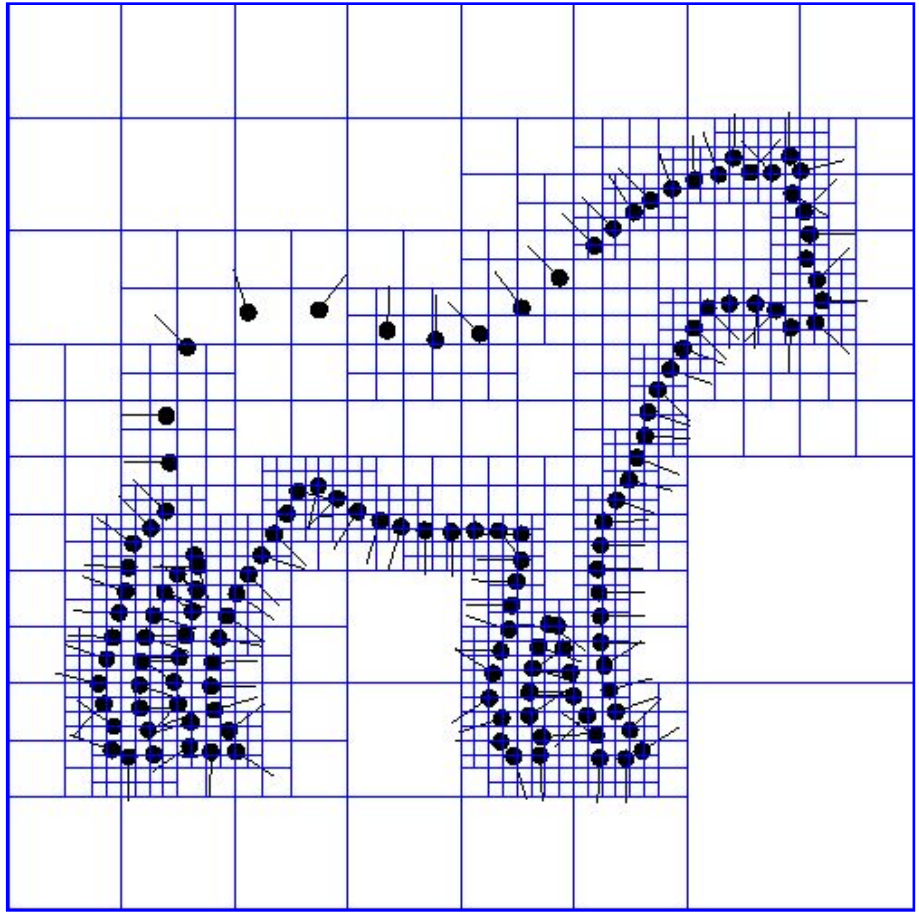


# In practice

C - The Poisson reconstruction pipeline

Given the Points:

- **Set octree**
- Compute vector field
- Compute indicator function
- Extract iso-surface

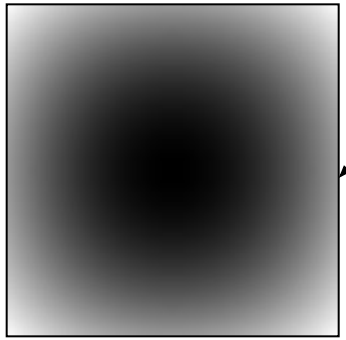


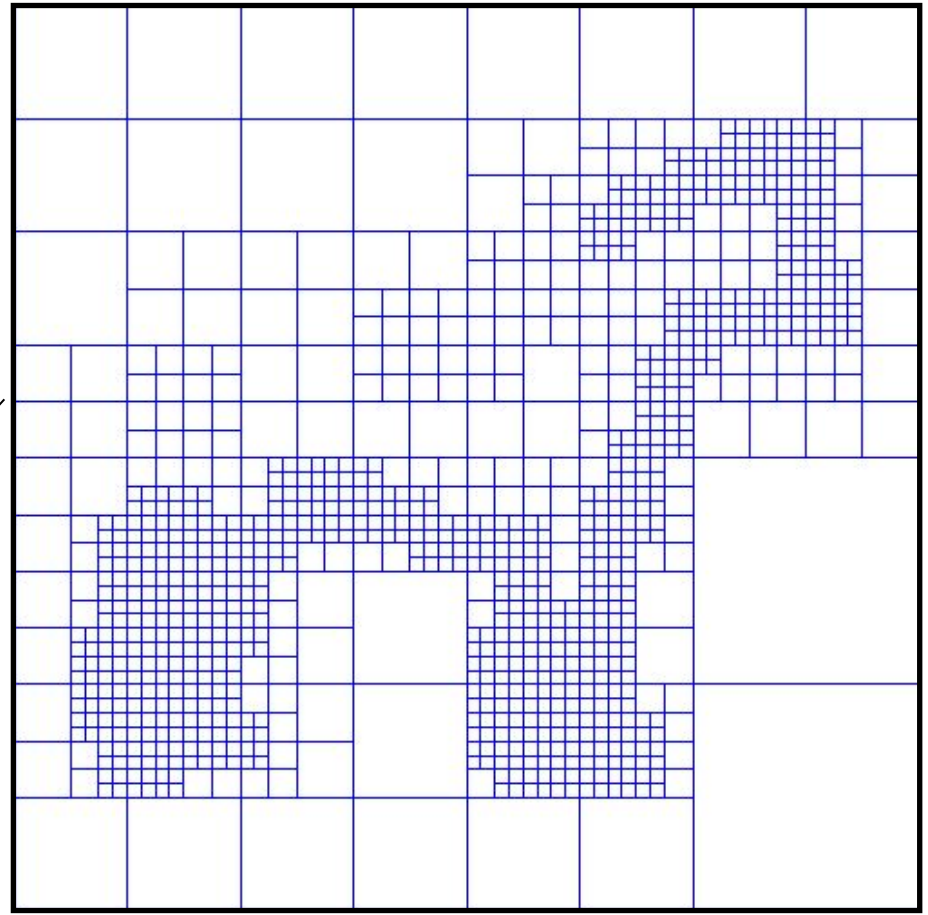
# In practice

C - The Poisson reconstruction pipeline

Given the Points:

- Set octree
- **Compute vector field**
  - **Define a function space**
  - Splat the samples
- Compute indicator function
- Extract iso-surface



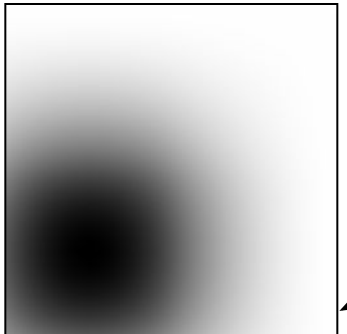


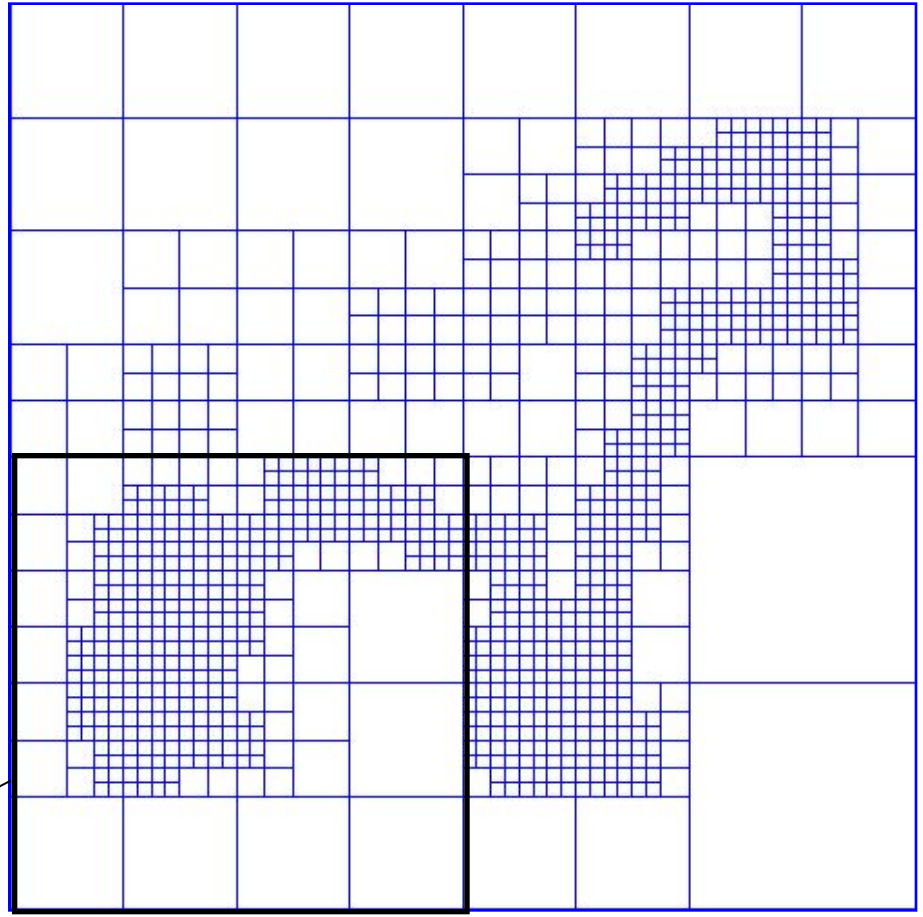
# In practice

C - The Poisson reconstruction pipeline

Given the Points:

- Set octree
- **Compute vector field**
  - **Define a function space**
  - Splat the samples
- Compute indicator function
- Extract iso-surface



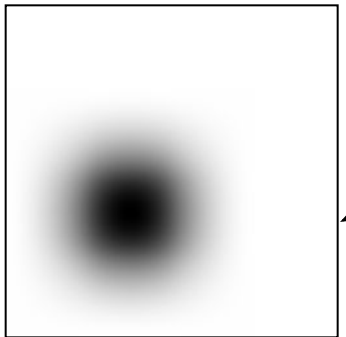


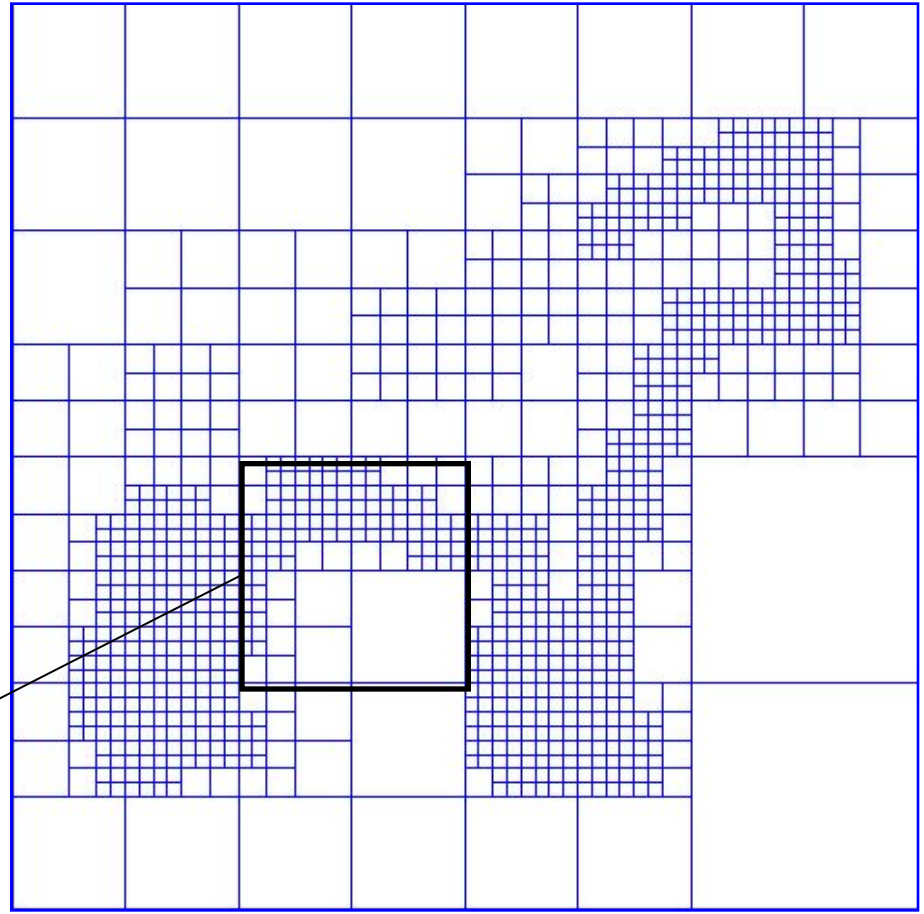
# In practice

C - The Poisson reconstruction pipeline

Given the Points:

- Set octree
- **Compute vector field**
  - **Define a function space**
  - Splat the samples
- Compute indicator function
- Extract iso-surface



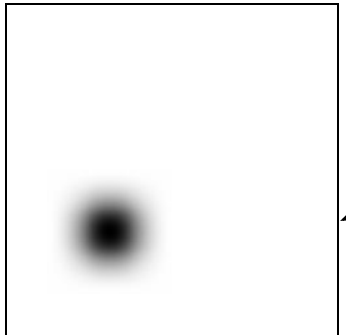


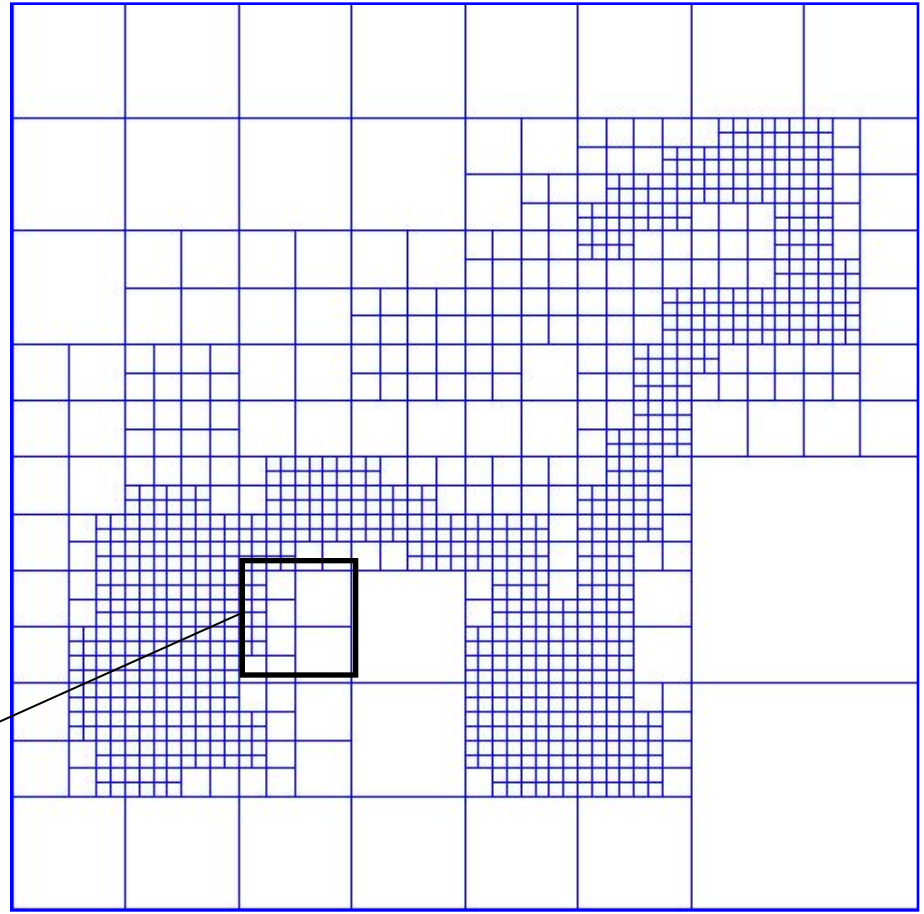
# In practice

C - The Poisson reconstruction pipeline

Given the Points:

- Set octree
- **Compute vector field**
  - **Define a function space**
  - Splat the samples
- Compute indicator function
- Extract iso-surface



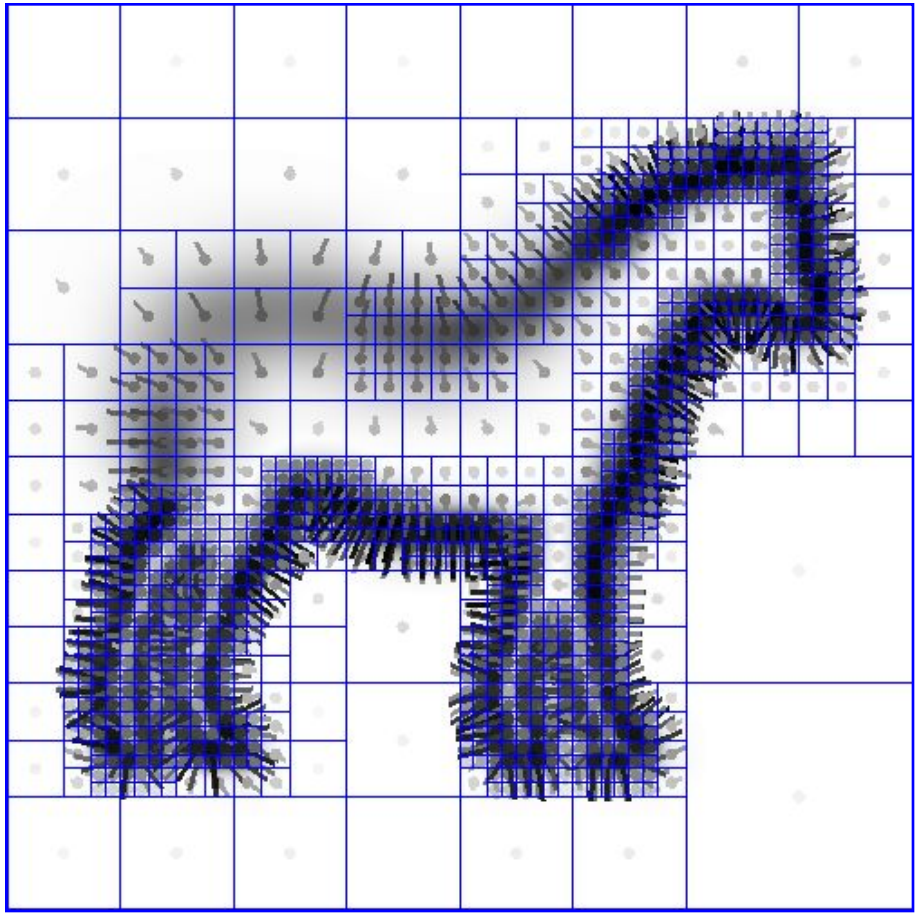


# In practice

## C - The Poisson reconstruction pipeline

Given the Points:

- Set octree
- **Compute vector field**
  - Define a function space
  - **Splat the samples**
- Compute indicator function
- Extract iso-surface



# In practice

## C - The Poisson reconstruction pipeline

Given the Points:

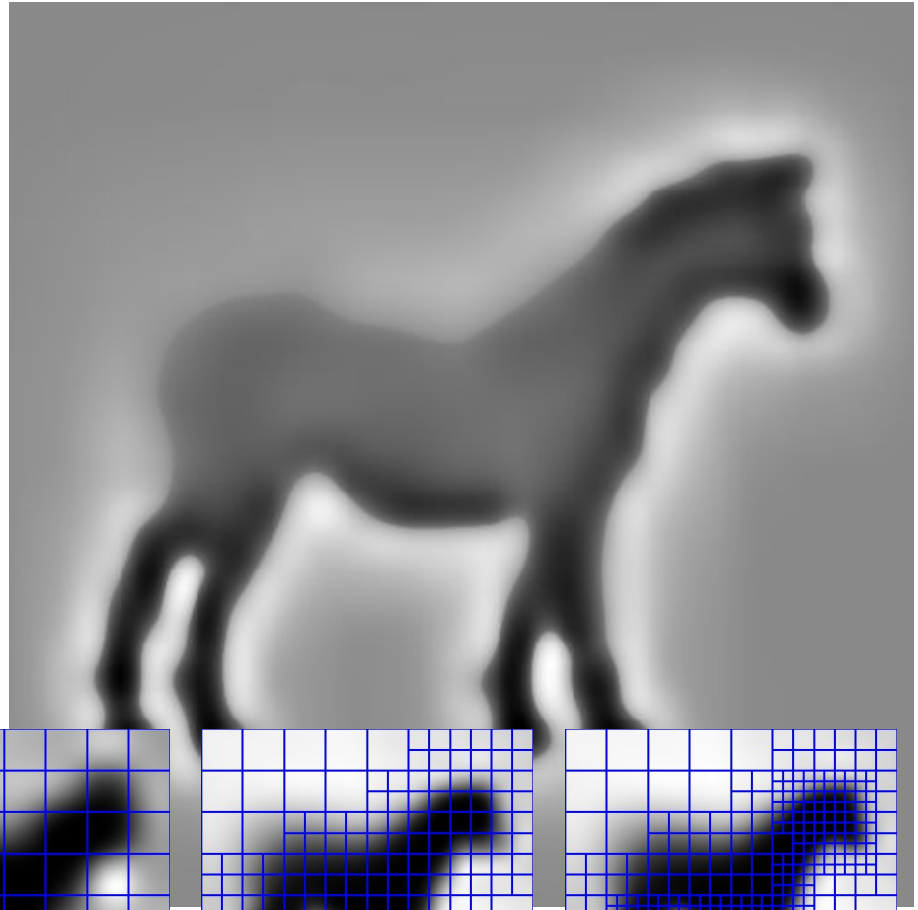
- Set octree
- Compute vector field
- **Compute indicator function**
  - **Compute divergence**
  - Solve Poisson equation
- Extract iso-surface

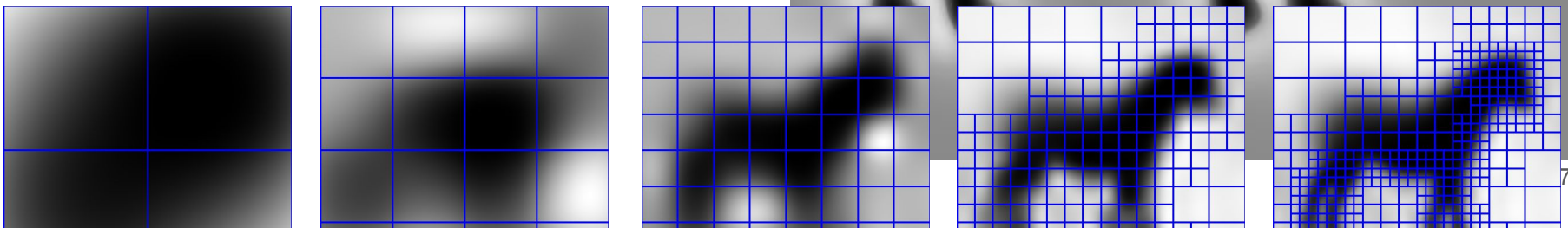
# In practice

## C - The Poisson reconstruction pipeline

Given the Points:

- Set octree
- Compute vector field
- **Compute indicator function**
  - Compute divergence
  - **Solve Poisson equation**
- Extract iso-surface





# In practice

## C - The Poisson reconstruction pipeline

Given the Points:

- Set octree
- Compute vector field
- **Compute indicator function**
  - Compute divergence
  - **Solve Poisson equation**
- Extract iso-surface

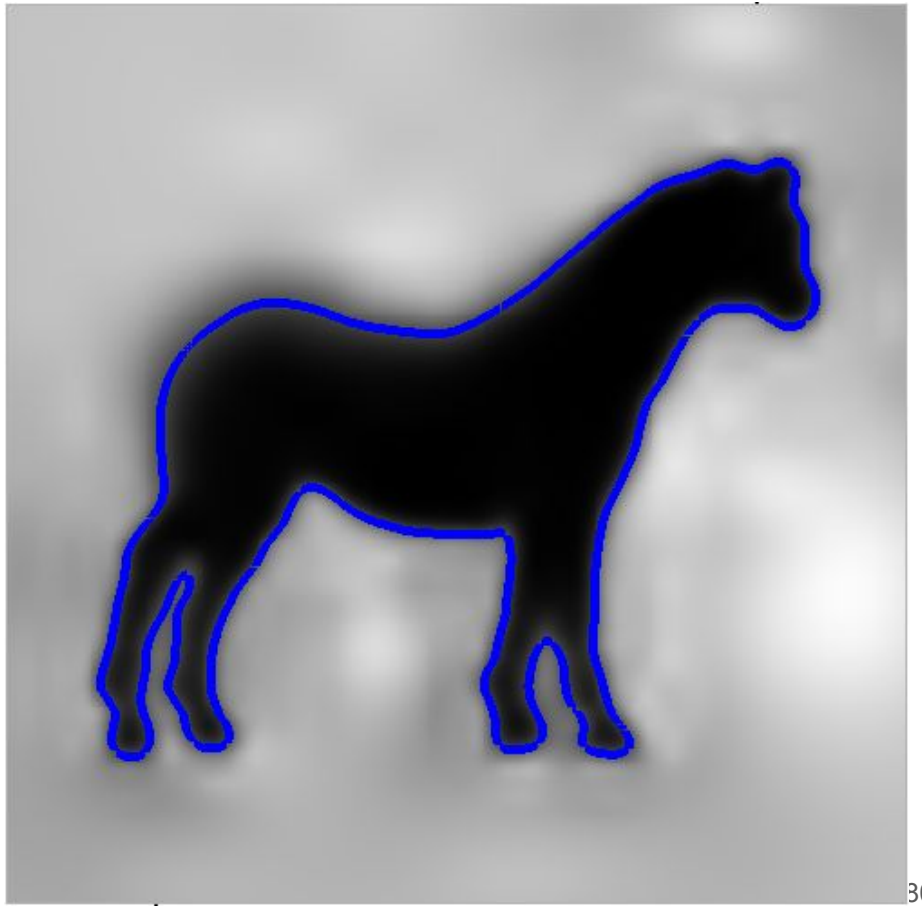
# In practice

valeo.ai

C - The Poisson reconstruction pipeline

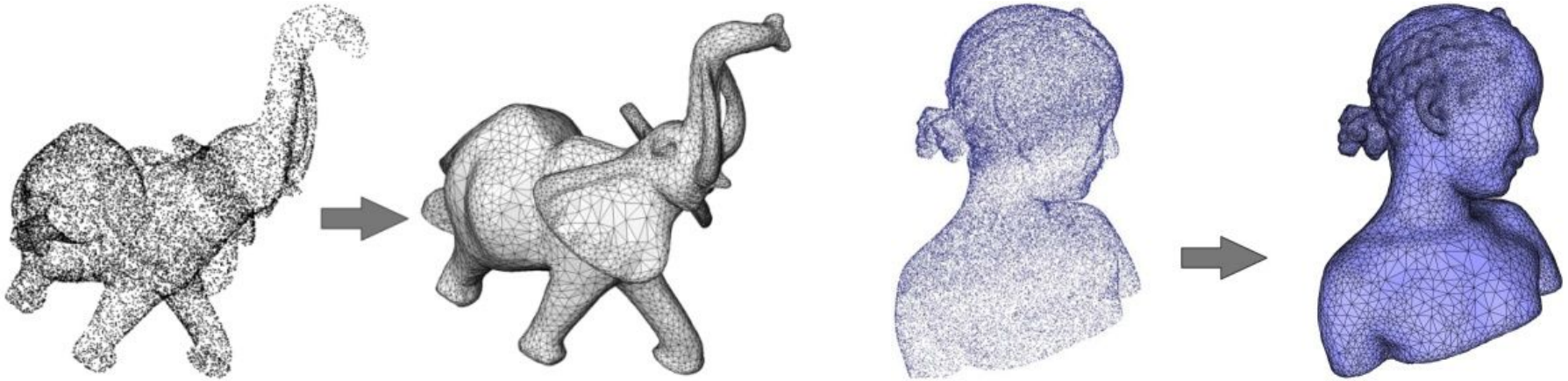
Given the Points:

- Set octree
- Compute vector field
- Compute indicator function
- **Extract iso-surface**



# Examples

## C - The Poisson reconstruction pipeline



# Overview

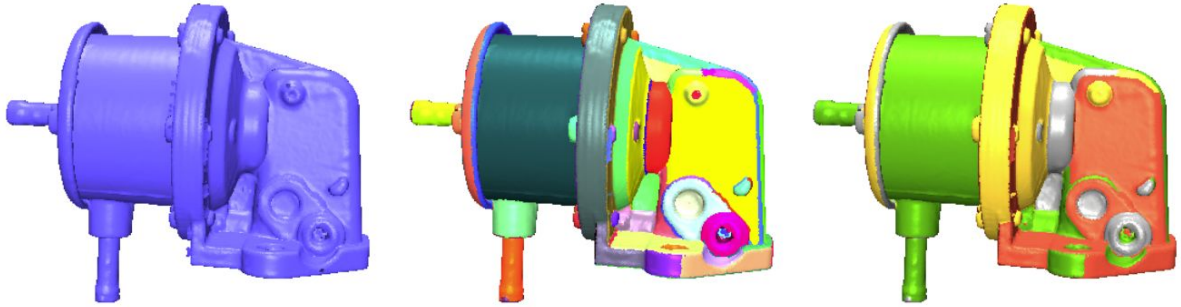
I. From point clouds to surfaces
II. Local features
    A. Normal estimation
    B. Normal orientation
**III. Surface reconstruction**
    A. Ball pivoting
    B. Delauney reconstruction
    C. Poisson reconstruction
    D. RANSAC

# Model-based approaches

valeo.ai

Fit a model

- Abstraction
- Simplification
- CAD
- …



(a) Original (b) Random colors (c) Colored by type

(a) Original (b) Approximation

*Efficient RANSAC for Point-Cloud Shape Detection, Schabel et al.*

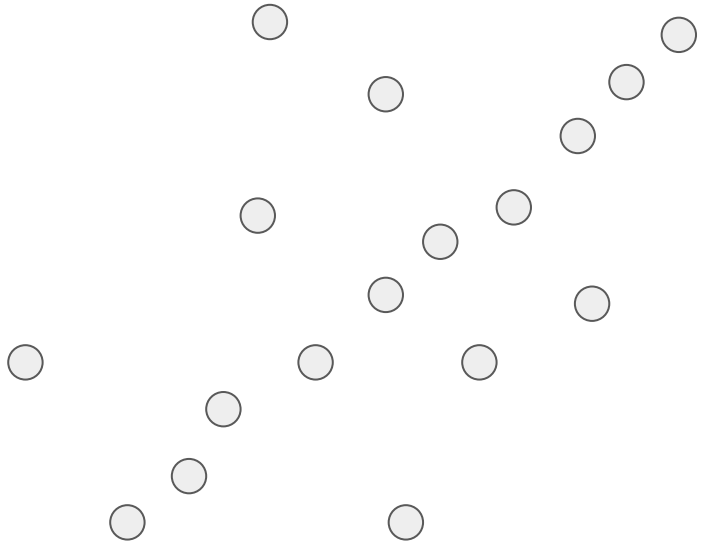
# Random Sample Consensus

D - RANSAC

valeo.ai

**Ran**dom **Sa**mple **C**oncensus

- Defining a model
  - Line
  - Defined by to (different) points



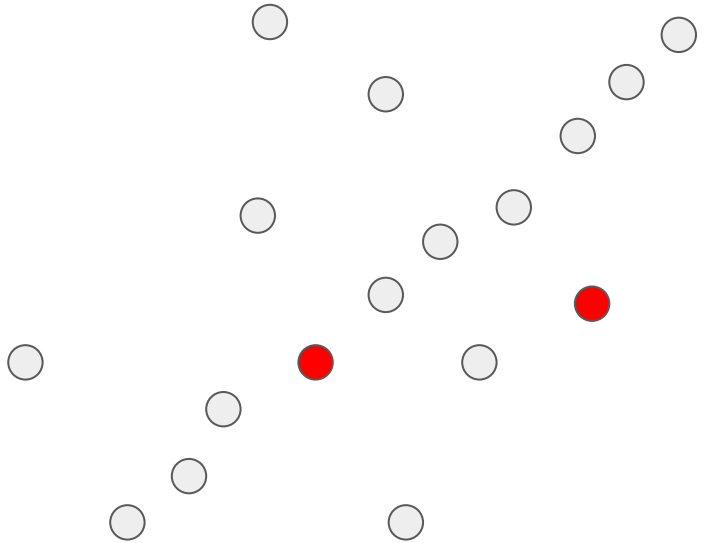
# Random Sample Consensus

D - RANSAC

valeo.ai

**Ran**dom **Sa**mple **C**oncensus

- Defining a model
- Hypothesis generation
  - Pick subset of the data



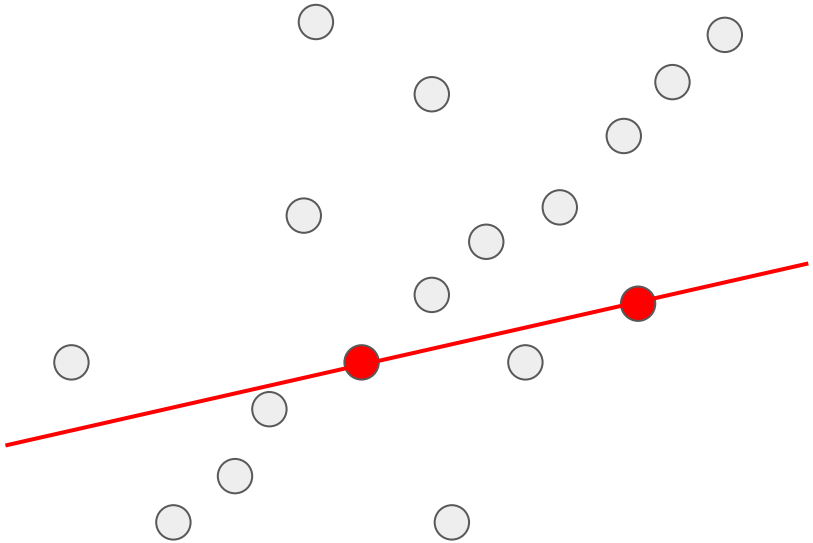
# Random Sample Consensus

D - RANSAC

valeo.ai

**Ran**dom **Sa**mple **C**oncensus

- Defining a model
- Hypothesis generation
  - Pick subset of the data
  - Estimate the corresponding model



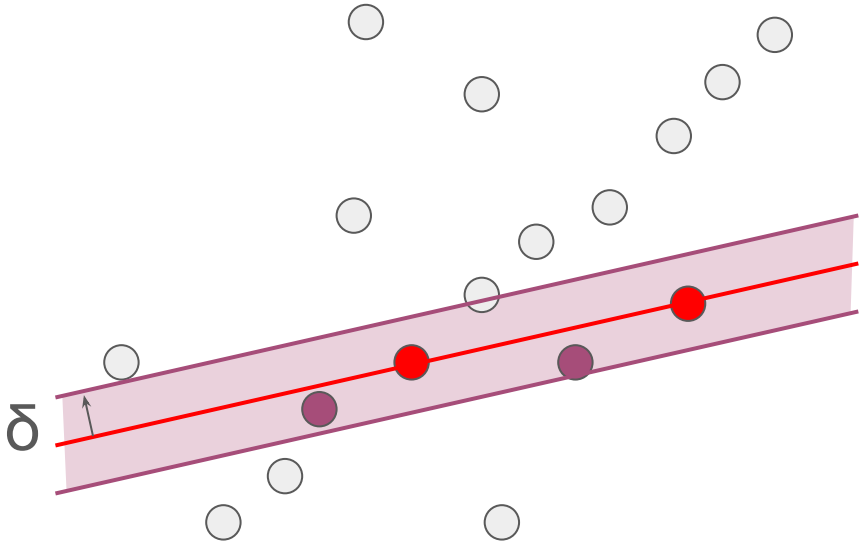
# Random Sample Consensus

valeo.ai

D - RANSAC

**Ran**dom **Sa**mple **C**oncensus

- Defining a model
- Hypothesis generation
  - Pick subset of the data
  - Estimate the corresponding model
  - Compute inliers



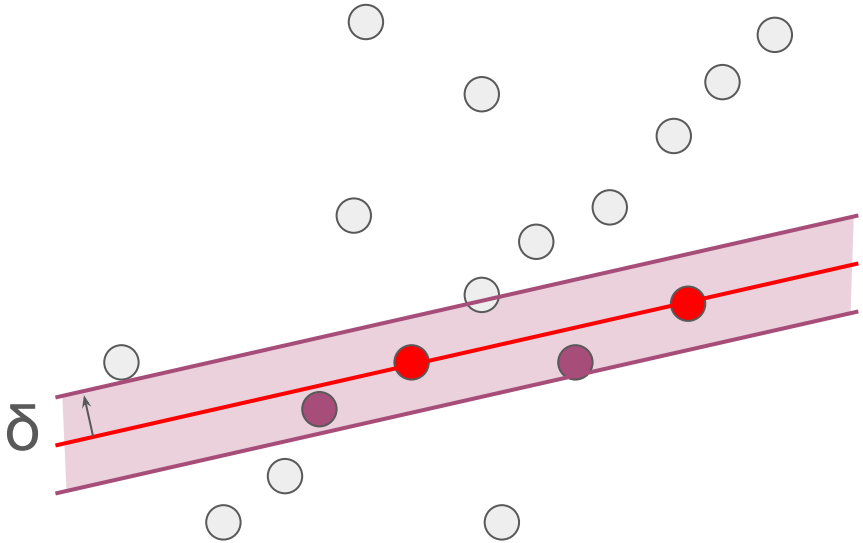
# Random Sample Consensus

valeo.ai

D - RANSAC

**Ran**dom **Sa**mple **C**oncensus

- Defining a model
- Hypothesis generation
  - Pick subset of the data
  - Estimate the corresponding model
  - Compute inliers
- Compare to best candidate so far



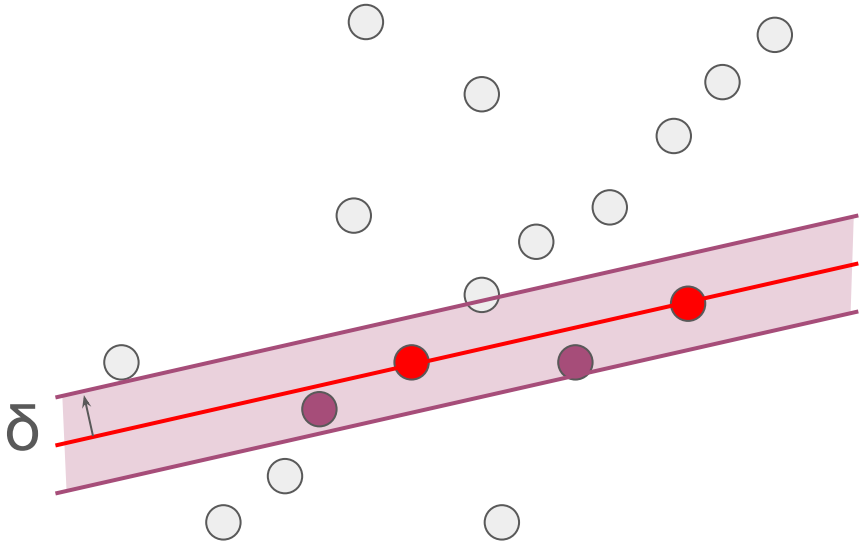
# Random Sample Consensus

valeo.ai

D - RANSAC

**Ran**dom **Sa**mple **C**oncensus

- Defining a model
- Hypothesis generation
  - Pick subset of the data
  - Estimate the corresponding model
  - Compute inliers
- Compare to best candidate so far
- Iterate

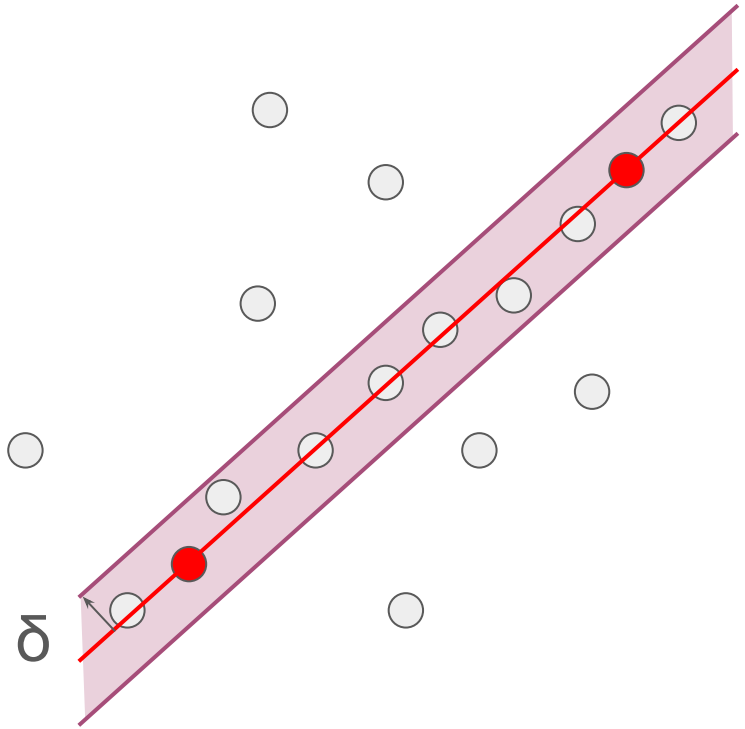


# Random Sample Consensus

D - RANSAC

**Ran**dom **Sa**mple **C**oncensus

- Defining a model
- Hypothesis generation
  - Pick subset of the data
  - Estimate the corresponding model
  - Compute inliers
- Compare to best candidate so far
- Iterate



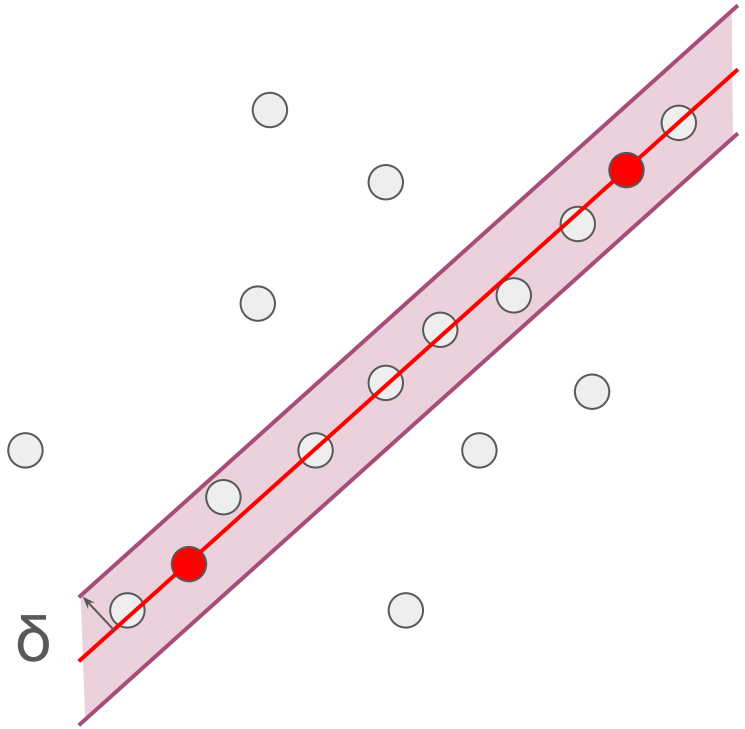
# Random Sample Consensus

D - RANSAC

valeo.ai

**Parameters**

- N: size of the point cloud
- k: number of points to generate an hypothesis
- δ: inlier tolerance
- T: number of models to pick in the loop



# Random Sample Consensus

valeo.ai

D - RANSAC

**Estimating T**

Probability of picking one inlier point :

$$\frac{n = \text{num. inliers}}{N = \text{p.c. size}}$$

Probability of picking one inlier hypothesis (k=3 inlier points):

$$P(n) = \binom{n}{k} / \binom{N}{k} \sim \left(\frac{n}{N}\right)^k$$

Probability that the hypothesis is an outlier:

$$1 - P(n)$$

# Random Sample Consensus

valeo.ai

D - RANSAC

**Estimating T**

Probability that after $s$ hypotheses picks none of $s$ hypothesis is an inlier

$$(1 - \left(\frac{n}{N}\right)^k)^s$$

Probability that at least one hypothesis in an inlier

$$P(n, s) = 1 - (1 - \left(\frac{n}{N}\right)^k)^s$$

We want $T$ such that the probability of picking an inlier to be more than $p_t$

$$P(n, T) = 1 - (1 - \left(\frac{n}{N}\right)^k)^T > p_t$$

# Random Sample Consensus

valeo.ai

D - RANSAC

**Estimating T**

We want $T$ such that the probability of picking an inlier to be more than $p_t$

$$P(n, T) = 1 - (1 - \left(\frac{n}{N}\right)^k)^T > p_t$$

Then:

$$T = \frac{\ln(1 - p_t)}{\ln(1 - \left(\frac{n}{N}\right)^k)}$$

$n$ and $p_t$ are parameters to be set.

# Conclusion and practical session

# Surface reconstruction

valeo.ai

Conclusion and practical session

Many methods have been developed with various characteristics:

Simple, Smooth, Model-based, Optimal (for given criteria), Robust to noise…

Non-learning methods are usable off-the-shelf.

Learning-based methods… see the following courses

# Practical session



valeo.ai

Implement a RANSAC plane extractor

https://github.com/aboulch/MSIA_points/blob/main/03_surfaces/MSIA_Points_3_surfaces.ipynb

https://www.college-de-france.fr/media/jean-daniel-boissonnat/UPL2159668480191960308_alliez_reconstruction.pdf

https://www.cs.jhu.edu/~misha/MyPapers/SGP06.ppt

https://courses.grainger.illinois.edu/cs598dwh/fa2021/lectures/Lecture%2011%20-%203D%20Registration%20and%20Shape%20Fitting%20-%203DVision.pdf