

Surface Reconstruction from Point Clouds: An Introduction

Jean-François Remacle

LMECA2170

What is the problem? *and why there is no single solution*

- ▶ **Input:** an unorganized 3D point set $P = \{p_i\}$, optionally with normals; affected by noise, outliers, non-uniform densities, and occlusions.
- ▶ **Goal:** recover a surface $\mathcal{S}$ (implicit field or triangular/quad mesh) that approximates the unknown ground-truth geometry and topology.
- ▶ **No silver bullet:** methods trade off assumptions (sampling, noise), feature preservation, watertightness, open boundaries, and parameter sensitivity.
- ▶ **Typical scales:**
 - ▶ Small: 10^3 – 10^5 points (classroom demos, tabletop scans)
 - ▶ Medium: 10^6 – 10^7 points (single-object laser/photogrammetry)
 - ▶ Large: 10^8 + points (full rooms, heritage sites, LiDAR/TLS)
- ▶ **Practical limits:** memory (octrees/DTs), normal estimation quality, scale selection (bandwidth r , α , octree depth), and robustness to *holes*, *thin parts*, and *sharp features*.
- ▶ **Outputs vary:** watertight vs. open; manifold vs. non-manifold; smoothed vs. feature-aware; explicit mesh vs. implicit field.

Why a *point cloud*? Sources and artifacts

- ▶ **Photogrammetry (SfM/MVS):** overlapping calibrated images $\Rightarrow$ sparse then dense points.
Pros: low-cost sensors, high detail on textured areas.
Cons: textureless/specular regions, scale selection, drift, occlusions.
- ▶ **Active scanning:** LiDAR/TLS, structured light, time-of-flight.
Pros: accurate ranges, large dynamic scenes.
Cons: multi-path, mixed pixels, transparent/specular failure modes, registration seams.
- ▶ **"Triangle soups":** depth-map meshing or CAD fragments produce disconnected, non-manifold triangles. Often converted back to points (sampling) before robust reconstruction/repair.
- ▶ **Other domains:** robotics/SLAM, AR/VR mapping, industrial metrology, medical imaging (isosurface points), cultural heritage.
- ▶ **Implication:** the acquisition modality fixes the *noise model* and *sampling pattern* $\Rightarrow$ different reconstruction methods and parameters.

Classical methods at a glance

- ▶ **Tight Cocone / Crust family:** Delaunay-based restricted surfaces under sampling conditions; good feature retention; sensitive to noise/outliers.
- ▶ **Poisson Surface Reconstruction:** solves a global Poisson problem from oriented normals; watertight by design; octree depth and regularization control detail vs. smoothness.
- ▶ **Alpha Shapes:** α -complex substructure of Delaunay; topology evolves with α ; simple and fast, but parameter- and noise-sensitive.
- ▶ **Alpha Wrapping:** wraps the data with an α -complex to produce a closed, manifold surface; more stable than raw α -shapes for non-uniform sampling.
- ▶ **Ball Pivoting (BPA):** rolls a ball of radius r on the cloud to stitch triangles; effective on dense, fairly uniform scans.
- ▶ **MLS / Implicit fits:** Moving Least Squares, signed-distance/RBF fits; smooth implicit fields with optional meshing; may blur sharp features if not feature-aware.

Tip: always check prerequisites (normals? orientation?), scale parameters (α , r , octree depth), and whether you need watertight or boundary-preserving outputs.

Poisson Surface Reconstruction

Jean-François Remacle

LMECA2170

Normal estimation from a raw point cloud $\{\mathbf{p}_i\}$

The normal vectors $\{\mathbf{n}_i\}$ must first be estimated.

The standard approach is to compute a local tangent plane around each point.

For each sample $\mathbf{p}_i$, consider its k nearest neighbors $\mathcal{N}_i$.

The goal is to approximate the local surface by a plane that best fits these neighboring points in the least-squares sense. The plane can be written as

$$(\mathbf{x} - \mathbf{c}_i) \cdot \mathbf{n}_i = 0,$$

where $\mathbf{c}_i$ is the centroid of the neighborhood,

$$\mathbf{c}_i = \frac{1}{|\mathcal{N}_i|} \sum_{\mathbf{p}_j \in \mathcal{N}_i} \mathbf{p}_j,$$

and $\mathbf{n}_i$ is the unit normal to the plane.

Normal estimation from a raw point cloud $\{\mathbf{p}_i\}$

The optimal normal $\mathbf{n}_i$ is the eigenvector associated with the *smallest eigenvalue* of the covariance matrix

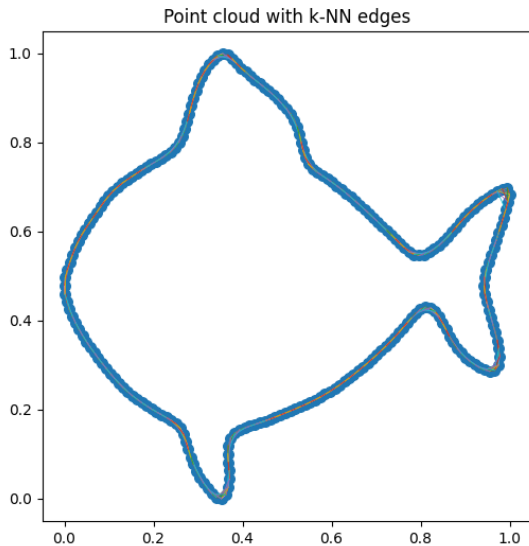
$$\mathbf{C}_i = \sum_{\mathbf{p}_j \in \mathcal{N}_i} (\mathbf{p}_j - \mathbf{c}_i)(\mathbf{p}_j - \mathbf{c}_i)^T.$$

This eigenvector represents the direction of least variation of the local point distribution.

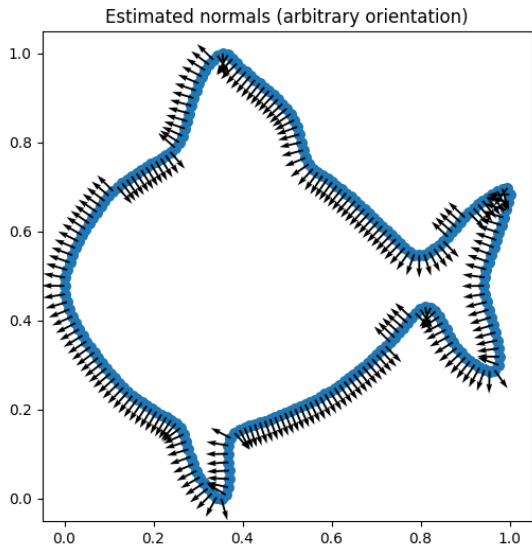
To ensure a globally consistent orientation of normals (all pointing either inward or outward), one may propagate orientations along a graph connecting neighboring points, flipping normals that are inconsistent with already oriented neighbors.

A minimum spanning tree traversal is typically used for this propagation.

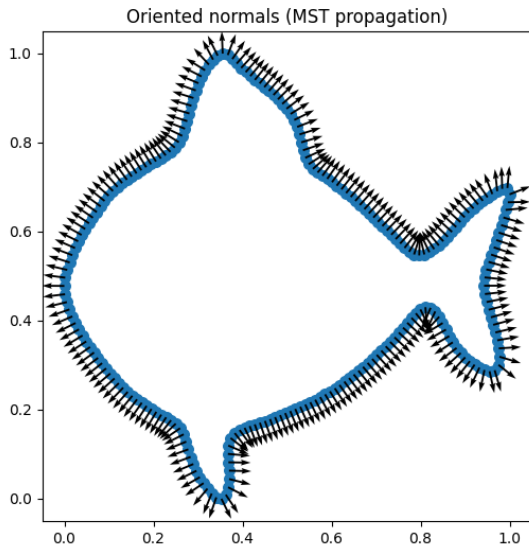
The "Poisson"



The "Poisson"



The "Poisson"



Poisson Surface Reconstruction

The goal of Poisson surface reconstruction is to recover a smooth scalar function $\chi(\mathbf{x})$ whose zero (or isovalue) level set represents the reconstructed surface.

Given a set of oriented samples $(\mathbf{p}_i, \mathbf{n}_i)$, where $\mathbf{p}_i \in \mathbb{R}^2$ (or $\mathbb{R}^3$) are points on the unknown surface and $\mathbf{n}_i$ are their corresponding outward unit normals, we seek a scalar potential χ whose gradient best approximates the vector field of normals.

The key observation is that the gradient of the indicator function of the object (equal to 1 inside and 0 outside) is proportional to the surface normal field:

$$\nabla \chi \approx \mathbf{n}.$$

This relation can be expressed in weak form by integrating over the domain Ω :

$$\int_{\Omega} \nabla \chi \cdot \nabla \psi \, d\mathbf{x} = \int_{\Omega} \mathbf{V} \cdot \nabla \psi \, d\mathbf{x}, \quad \forall \psi \in H_0^1(\Omega),$$

where $\mathbf{V}$ is a smoothed vector field constructed from the oriented samples

Poisson Surface Reconstruction

The last equation is the weak form of a Poisson equation

$$\Delta\chi = \nabla \cdot \mathbf{V},$$

with homogeneous Dirichlet boundary conditions $\chi = 0$ on $\partial\Omega$.

The equation could be discretized on a regular grid using a finite-difference approximation of the Laplacian:

$$(\Delta\chi)_{i,j} \approx \frac{\chi_{i+1,j} - 2\chi_{i,j} + \chi_{i-1,j}}{h_x^2} + \frac{\chi_{i,j+1} - 2\chi_{i,j} + \chi_{i,j-1}}{h_y^2},$$

which gives rise to a sparse linear system

$$A\chi = f, \quad \text{where} \quad f = \nabla \cdot \mathbf{V}.$$

The system is symmetric and positive-definite, and can be solved efficiently using the conjugate gradient (CG) method.

Poisson Surface Reconstruction

Once the scalar field χ has been computed, the reconstructed surface is obtained as an iso-contour (in 2D) or iso-surface (in 3D) of χ :

$$S = \{\mathbf{x} \mid \chi(\mathbf{x}) = \chi_0\},$$

where χ_0 is typically chosen as a mid-range value between the minimum and maximum of χ in the region influenced by the samples.

Poisson reconstruction has several attractive properties:

- ▶ It produces a globally smooth and watertight surface even when the samples are noisy or non-uniformly distributed.
- ▶ The reconstruction is formulated as a single linear PDE, which can be efficiently solved using standard numerical methods.
- ▶ The divergence operator naturally integrates local normal information into a coherent global surface.

Gaussian Splatting of Oriented Normals – computing $\mathbf{V}$ on the grid

Given a set of oriented samples $\{(\mathbf{p}_i, \mathbf{n}_i)\}_{i=1}^N$ with points $\mathbf{p}_i \in \mathbb{R}^2$ and outward unit normals $\mathbf{n}_i$, we first need to construct a continuous vector field $\mathbf{V}(\mathbf{x})$ defined over a regular grid. This field will later serve as input to the Poisson equation

$$\Delta \chi = \nabla \cdot \mathbf{V}.$$

Because the samples are sparse and irregularly distributed, we must interpolate them in a smooth way. To that end, each normal vector is “splatted” onto the grid using a Gaussian weighting kernel.

The vector field is defined as the sum of all local normal contributions:

$$\mathbf{V}(\mathbf{x}) = \sum_{i=1}^N \mathbf{n}_i w(\|\mathbf{x} - \mathbf{p}_i\|), \quad w(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right),$$

where σ controls the spatial extent of the Gaussian kernel.

Poisson Surface Reconstruction

Each sample therefore influences nearby grid points according to a radially symmetric Gaussian weight that decays smoothly with distance.

For efficiency, the summation is restricted to the neighboring grid nodes lying within a few standard deviations, typically 3σ :

$$V(x_{jk}) = \sum_{i \in \text{neighbors}} n_i \exp\left(-\frac{\|x_{jk} - p_i\|^2}{2\sigma^2}\right),$$

where x_{jk} denotes the center of the grid cell (j, k) .

The resulting field V is smooth, continuous, and well suited for finite-difference computation of its divergence:

$$f(\mathbf{x}) = \nabla \cdot \mathbf{V}(\mathbf{x}).$$

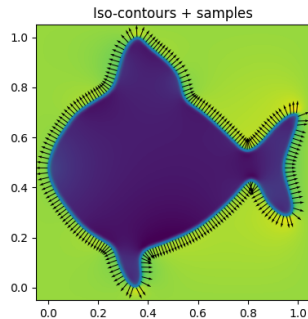
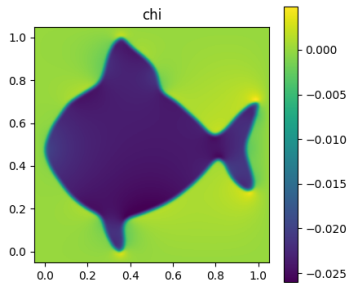
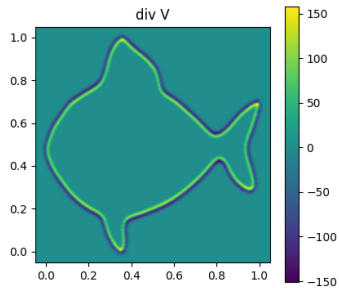
Gaussian Splatting of Oriented Normals – computing $\mathbf{V}$ on the grid

Gaussian splatting provides a simple and robust way to turn discrete, noisy samples into a continuous vector field:

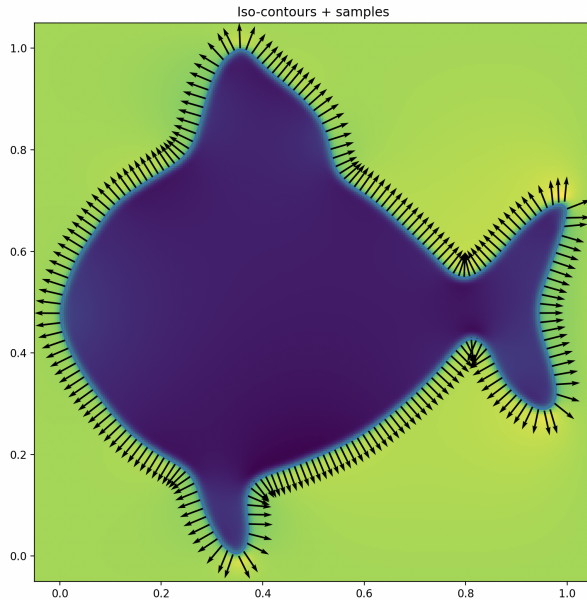
- ▶ It distributes the influence of each sample over several grid nodes, thus reducing aliasing and numerical artifacts.
- ▶ The Gaussian kernel ensures C^∞ smoothness, which leads to stable finite-difference derivatives.
- ▶ The parameter σ acts as a smoothing factor: small σ preserves details but increases noise, while large σ produces a smoother field.

In essence, Gaussian splatting converts a set of irregularly sampled normals into a dense and differentiable vector field that encodes local surface orientation information, preparing the data for solving the Poisson reconstruction equation.

The "Poisson"



The "Poisson"



Issues

- ▶ Normals – it is ok to use nearest neighbors and use covariance matrix analysis.
- ▶ Oriented normals – not that easy...